Departamento de Sistemas Informáticos y Computación

Towards a Universal Test of Social Intelligence

Javier Insa Cabrera

Supervisor: José Hernández Orallo

A thesis presented for the degree of Doctor of Philosophy
at the Universitat Politècnica de València

Valencia, Spain
May 2016

Abstract

Under the view of artificial intelligence, an intelligent agent is an autonomous entity which interacts in an environment through observations and actions, trying to achieve one or more goals with the aid of several signals called rewards. The creation of intelligent agents is proliferating during the last decades, and the evaluation of their intelligence is a fundamental issue for their understanding, construction and improvement.

Social intelligence is recently obtaining special attention in the creation of intelligent agents due to the current view of human intelligence as highly social. Social intelligence in natural and artificial systems is usually measured by the evaluation of associated traits or tasks that are deemed to represent some facets of social behaviour. The amalgamation of these traits or tasks is then used to configure an operative notion of social intelligence. However, this operative notion does not truly represent what social intelligence is and a definition following this principle will not be precise. Instead, in this thesis we investigate the evaluation of social intelligence in a more formal and general way, by actually considering the evaluee's interaction with other agents.

In this thesis we analyse the implications of evaluating social intelligence using a test that evaluates general intelligence. For this purpose, we include other agents into an initially single-agent environment to figure out the issues that appear when evaluating an agent in the context of other agents. From this analysis we obtain useful information for the evaluation of social intelligence.

From the lessons learned, we identify the components that should be considered in order to measure social intelligence, and we provide a *formal* and *parametrised* definition of social intelligence. This definition calculates an agent's social intelligence as its expected performance in a set of environments with a set of other agents arranged in teams and participating in line-ups, with rewards being re-understood appropriately. This is conceived as a tool to define social intelligence testbeds where we can generate several degrees of competitive and cooperative behaviours. We test this definition by experimentally analysing the influence of teams and agent line-ups for several multi-agent systems with variants of Q-learning agents.

However, not all testbeds are appropriate for the evaluation of social intelligence. To facilitate the analysis of a social intelligence testbed, we provide some formal property models about social intelligence in order to characterise the testbed and thus assess its suitability. Finally, we use the presented properties to characterise some social games and multi-agent environments, we make a comparison between them and discuss their strengths and weaknesses in order to evaluate social intelligence.

Resumen

Bajo la visión de la inteligencia artificial, un agente inteligente es una entidad autónoma la cual interactúa en un entorno a través de observaciones y acciones, tratando de lograr uno o más objetivos con la ayuda de varias señales llamadas recompensas. La creación de agentes inteligentes está proliferando durante las últimas décadas, y la evaluación de su inteligencia es un asunto fundamental para su entendimiento, construcción y mejora.

Recientemente la inteligencia social está obteniendo especial atención en la creación de agentes inteligentes debido a la visión actual de la inteligencia humana como altamente social. Normalmente la inteligencia social en sistemas naturales y artificiales se mide mediante la evaluación de rasgos asociados o tareas que se consideran que representan algunas facetas del comportamiento social. La agrupación de estos rasgos o tareas se utiliza entonces para configurar una noción operacional de inteligencia social. Sin embargo, esta noción operacional no representa fielmente a la inteligencia social y no sería posible una definición siguiendo este principio. En su lugar, en esta tesis investigamos la evaluación de la inteligencia social de un modo más formal y general, considerando la interacción del agente a evaluar con otros agentes.

En esta tesis analizamos las implicaciones de evaluar la inteligencia social utilizando un test que evalúe la inteligencia general. Con este objetivo incluimos otros agentes en un entorno inicialmente diseñado para un único agente con el fin de averiguar qué cuestiones aparecen cuando evaluamos a un agente en un contexto con otros agentes. A partir de este análisis obtenemos información útil para la evaluación de la inteligencia social.

A partir de las lecciones aprendidas identificamos los componentes que deberían considerarse al medir la inteligencia social y proporcionamos una definición *formal* y *parametrizada* de esta inteligencia social. Esta definición calcula la inteligencia social de un agente como su rendimiento esperado en un conjunto de entornos y con un conjunto de otros agentes organizados en equipos y distribuidos en alineaciones, reinterpretando apropiadamente las recompensas. Esto se concibe como una herramienta para definir bancos de prueba de inteligencia social donde podamos generar varios grados de comportamientos competitivos y cooperativos. Probamos esta definición analizando experimentalmente la influencia de los equipos y las alineaciones de agentes en varios sistemas multiagente con variantes de agentes Q-learning.

Sin embargo, no todos los bancos de prueba son apropiados para la evaluación de la inteligencia social. Para facilitar el análisis de un banco de pruebas de inteligencia social, proporcionamos algunos modelos de propiedades formales sobre la inteligencia social con el objetivo de caracterizar el banco de pruebas y así valorar su idoneidad. Finalmente, usamos las propiedades presentadas para caracterizar algunos juegos sociales y entornos multiagente, hacemos una comparación entre ellos y discutimos sus puntos fuertes y débiles para ser usados en la evaluación de la inteligencia social.

Resum

Davall la visió de la intel·ligència artificial, un agent intel·ligent és una entitat autònoma la qual interactua en un entorn a través d'observacions i accions, tractant d'aconseguir un o més objectius amb l'ajuda de diverses senyals anomenades recompenses. La creació d'agents intel·ligents està proliferant durant les últimes dècades, i l'avaluació de la seua intel·ligència és un assumpte fonamental per al seu enteniment, construcció i millora.

Recentment la intel·ligència social està obtenint especial atenció en la creació d'agents intel·ligents a causa de la visió actual de la intel·ligència humana com altament social. Normalment la intel·ligència social en sistemes naturals i artificials es mesura per mitjà de l'avaluació de trets associats o tasques que es consideren que representen algunes facetes del comportament social. L'agrupació d'aquests trets o tasques s'utilitza llavors per a configurar una noció operacional d'intel·ligència social. No obstant això, aquesta noció operacional no representa fidelment a la intel·ligència social i no seria possible una definició seguint aquest principi. En el seu lloc, en aquesta tesi investiguem l'avaluació de la intel·ligència social d'una manera més formal i general, considerant la interacció de l'agent a avaluar amb altres agents.

En aquesta tesi analitzem les implicacions d'avaluar la intel·ligència social utilitzant un test que avalue la intel·ligència general. Amb aquest objectiu incloem altres agents en un entorn inicialment dissenyat per a un únic agent amb la finalitat d'esbrinar quines qüestions apareixen quan avaluem un agent en un context amb altres agents. A partir d'aquesta anàlisi obtenim informació útil per a l'avaluació de la intel·ligència social.

A partir de les lliçons apreses identifiquem els components que haurien de considerar-se al mesurar la intel·ligència social i proporcionem una definició *formal* i *parametrizada* d'aquesta intel·ligència social. Aquesta definició calcula la intel·ligència social d'un agent com el seu rendiment esperat en un conjunt d'entorns i amb un conjunt d'altres agents organitzats en equips i distribuïts en alineacions, reinterpretant apropiadament les recompenses. Açò es concep com una ferramenta per a definir bancs de prova d'intel·ligència social on podem generar diversos graus de comportaments competitius i cooperatius. Provem aquesta definició analitzant experimentalment la influència dels equips i les alineacions d'agents en diversos sistemes multiagent amb variants d'agents Q-learning.

No obstant això, no tots els bancs de prova són apropiats per a l'avaluació de la intel·ligència social. Per a facilitar l'anàlisi d'un banc de proves d'intel·ligència social, proporcionem alguns models de propietats formals sobre la intel·ligència social amb l'objectiu de caracteritzar el banc de proves i així valorar la seua idoneïtat. Finalment, usem les propietats presentades per a caracteritzar alguns jocs socials i entorns multiagent, fem una comparació entre ells i discutim els seus punts forts i dèbils per a ser usats en l'avaluació de la intel·ligència social.

Acknowledgements

I still remember my first days in research, I was happy to have the opportunity to achieve one of my greatest aspirations; to achieve a PhD. At the very beginning of your doctoral thesis you feel excited about the new challenges that arise along the journey, until you realise where you really are. The development of a thesis is not at all easy. During its development there are thousands of situations that make you feel lost and overwhelmed due to the huge amount of new and baffling information that they pile on you. Once you have decided on the direction of your thesis, there comes the insecurity of not knowing whether you are focusing it properly. All PhD. students have the sensation that "there is no light at the end of the tunnel". I never imagined that reaching such a dream would be so complicated. Fortunately, there are always family, colleagues and friends whose support and patience help you to alleviate such a burden.

I would like to thank my parents, Juan and Amparo, for always believing in me, for the invaluable advice they have given me and for their sustained effort in making me the man I'm today; to my sister Natalia for putting up with me for all these years and always keeping us together; to my brother David for having accompanied me during all stages of my life, in both good times and bad times; and to the rest of my family, for their interest in my thesis and their understanding, especially when my thesis did not allow me to attend some of the family get togethers.

To my friends and other special people in my life for being there for me during all this time. Their support and words of encouragement gave me strength to go ahead in difficult moments. Without them I would have never been able to finish this doctoral thesis.

To my university colleagues and professors who have accompanied me on this journey, with whom I have, so far, spent the most stressful, endless and essentially gratifying years of my life, and especially to those who have managed to make it more bearable: Marco, Francisco, Fernando, Sonia, Julia, Salvador, Laura, Javier, César and Josep, among others. Also to David L. Dowe and Nader Chmait for inviting and welcoming me in Monash (their university) and succeeding in making my stay as professionally productive as personally enriching.

Finally, I want to thank my supervisor José for all his help and time spent with me, as well as for the patience to show me my mistakes and for being an example of hard work, constancy and wisdom.

To all, thanks for making this thesis possible.

Agradecimientos

Aún recuerdo mis primeros días en esto de la investigación, me sentía feliz por tener la oportunidad de realizar una de mis mayores aspiraciones; convertirme en doctor. Al empezar la tesis doctoral te sientes ilusionado, emocionado ante los nuevos retos que se abren a tu paso, hasta que te das cuenta de dónde te has metido. El desarrollo de una tesis no es nada fácil. Durante su desarrollo existen miles de situaciones que te hacen sentir perdido y abrumado por la ingente cantidad de nueva y desconcertante información que se te echa encima. Una vez decidido en qué dirección realizar la tesis, viene la inseguridad de no saber si se está enfocando adecuadamente. Además, a todo doctorando le llega el momento en el que "no ve la luz al final del túnel". Nunca imaginé que alcanzar este sueño fuera tan complicado. Por suerte, siempre hay familiares, compañeros y amigos que con su apoyo y paciencia consiguen aliviar tanta carga.

Me gustaría dar las gracias a mis padres, Juan y Amparo, por siempre confiar en mí, haberme sabido dar los mejores consejos y esforzarse tanto por hacerme ser el hombre que soy hoy; a mi hermana Natalia, por haberme aguantado durante todos estos años y habernos mantenido siempre unidos; a mi hermano David, por haberme acompañado durante todas las etapas de mi vida, tanto en los buenos momentos como en los malos; y al resto de mis familiares, por su interés en mi tesis y por su comprensión en algunas reuniones familiares a las que la tesis no me permitió asistir.

A mis amigos y personas especiales en mi vida, por haberme acompañado durante todo este tiempo. Sus apoyos y palabras de ánimo me han dado fuerzas para seguir adelante en los momentos difíciles. Sin ellos nunca hubiera podido terminar esta tesis doctoral.

A mis compañeros y profesores de universidad que me han acompañado en este trayecto, con los cuales he pasado, hasta el momento, los años más estresantes, interminables y sobre todo gratificantes de mi vida, y especialmente a aquellos que han conseguido hacerlo más llevadero: Marco, Francisco, Fernando, Sonia, Julia, Salvador, Laura, Javier, César y Josep, entre tantos otros. También a David L. Dowe y Nader Chmait por invitarme y acogerme en Monash (su universidad) y conseguir que mi estancia fuera tan productiva en lo profesional como enriquecedora en lo personal.

Por último, quiero dar las gracias a mi director José, por toda la ayuda y el tiempo que me ha dedicado, así como por la paciencia que ha tenido para mostrarme mis errores y ser un ejemplo de trabajo, constancia y sabiduría.

A todos, gracias por haber hecho posible esta tesis.

Contents

Chapter 1

Introduction

Evaluation tools are crucial in any discipline as a way to assess its progress and creations. The discipline of artificial intelligence (AI) has been concerned with the creation of algorithms that obtain good performance in some specific tasks. Recently, there is increasing interest and progress in AI systems that behave well in a more *general* range of situations. The evaluation procedures, benchmarks and computations are hence very different, depending on whether we focus on specialised benchmarks [72, 38, 24, 116] or on more general problems [96, 31, 94]. Despite this trend, AI is still lacking general, well-grounded and universally accepted intelligence measurement tools.

In fact, AI is a paradigmatic case of how useful these tools would be and how impeding this lack is. There are, of course, some tools, benchmarks and contests, aimed at the measurement of humanoid intelligence. However, the evolution and state of the art of AI is now more focussed towards social abilities, and here the measuring tools are still rather incipient.

In the past two decades, the notion of agent and the area of multi-agent systems have shifted AI to problems and solutions where 'social' intelligence is more relevant [26]. This shift towards a more social-oriented AI is related to the current view of human intelligence as highly social, actually one of the most distinctive features of human intelligence over other kinds of animal intelligence.

Dating back from the late nineties, we can find several works [19, 39, 69, 44] addressing the problem of measuring agent intelligence in a principled and general way. Using notions taken from (algorithmic) information theory, MML and two-part compression, Kolmogorov complexity and Solomonoff priors (see [71] for proper definitions of all these notions), some of these works present definitions and tests to evaluate agent intelligence.

A *universal* test, as introduced in [44], is a test which aims at evaluating any kind of subject including, e.g. humans and reinforcement learning agents. Some preliminary results of a universal test evaluating general intelligence [57, 62] show that the setting is able to compare and evaluate different kinds of agents, but it fails at placing them on the same scale, since humans usually get similar scores to those of other relatively simple agents. One possible explanation for these results is that it is virtually impossible to find other agents in the test, so social intelligence is not measured. The question, therefore, is what and how agents should be introduced in the test. This is related to the Turing test and the question of evaluating intelligence with games (also suggested in [44]), where the difficulty of a task is not only given by the complexity of the game, but from the opponent's intelligence.

Social intelligence has been defined in many ways in psychology and cognitive science, but it can be just worded, with the terminology of agents, as the ability to perform well in the context of other agents. The difference between social intelligence and general intelligence is that in the latter an agent could perform well if it were able to solve non-social tasks, such as escaping from a maze, solving a puzzle or predicting the next number in a series. On the contrary, social intelligence implies that tasks involve competing and/or collaborating with other agents. One problem of this definition is that we have to be more precise about what the 'other agents' are. If we evaluate humans, and the other agents are worms or sea sponges, then our intuitive notion of social intelligence does not work well, because working well in the context of other agents with low intelligence is not necessarily related to social intelligence as we know it. In psychometrics and human cognition, social intelligence clearly sets these other agents as other humans. But what about artificial agents? If we use a society of dull agents, the useful abilities might be very different to those which are required if we introduce an agent into, e.g. a society of humans.

The typical approach to observe and evaluate the social behaviour of an agent or group of agents is to test its performance interacting with several agents (homogeneous or heterogeneous) populating a selection of multi-agent environments [138]. However, this does not ensure that the agents we want to evaluate fully use their social abilities. For example, some agents could deliberately ignore the rest of agents in order to follow their own strategy. Also, the environments could be more focused on solving a certain problem rather than measuring the social abilities used by the agent we are evaluating. The use of specific environments (such as predator-prey, cooperation games, etc. [27, 83, 28]) ensures that some predefined social behaviours appear, but their generality is questionable and agent performance may depend on the specialisation to the game (good predators, good preys, etc.) instead of general social capabilities. In the end, this means that we must be careful about what elements (and more specifically, multi-agent environments and populating agents) are included in the test for the evaluation.

Nowadays, there is no clear procedure about how to define multi-agent settings where (1) social behaviour is encouraged and (2) social behaviour is not limited to the specific goals of the environment. Also, in the past, it has been difficult to easily regulate (3) how much important cooperation and competition are relatively and how they relate to rewards. Finally, (4) the relevance of the (social) intelligence of the other agents in the environments and their influence in the behaviour and performance of the agent to evaluate cannot be properly analysed with the current approaches.

1.1 Objectives

The goal of this thesis is to provide a firm scientific basis for the evaluation of social intelligence. Some significant questions that appear here are then whether it is possible to develop measurement tools that:

- Properly evaluate social intelligence universally (not only for artificial agents, but for any kind of interactive system), letting the evaluee interact with other (social) agents in a (social) environment and being able to evaluate both its competitive and cooperative social intelligence.

- Distinguish between general intelligence and social intelligence, fostering the evaluation in those scenarios where the interaction with other agents has an impact in performance.
- Permit us to assess the appropriateness of multi-agent environments and agents populating them, by providing some social properties they should meet in order to be used in a test to evaluate social intelligence.

1.2 Structure

The thesis is structured as follows:

Chapter 2) Background

This chapter explains some basic concepts which are necessary to understand some of the ideas presented in the following chapters. The presented concepts include:

- The definition of multi-agent environments, where several agents interact simultaneously in the same environment, also presenting some current games and scenarios that we use during the thesis as multi-agent environments.
- The area of reinforcement learning, where agents have to learn from experience how to behave in an environment using positive and/or negative signals called reinforcements, and presenting one of its well-known algorithms called Q-learning and some of its variants called SARSA and QV-learning.
- Kolmogorov complexity, which quantifies the size of the shortest program to describe strings.
- Probability distributions, such as the uniform and geometric distribution and the more complex universal distribution, which makes use of Kolmogorov complexity.
- A Monte Carlo method to approximate the value of the root of a given tree that needs high computational resources.

Chapter 3) State of the Art

For a long time, psychology has defined and evaluated intelligence and social intelligence for humans. More recently, the discipline of computer science has also been concerned with intelligence in order to make some progress on creating artificial intelligence. Although continuous progress has been achieved, nowadays we still do not have proper tools or tests to evaluate intelligence in AI and even fewer achievements have been done in evaluating social intelligence. This chapter makes a survey about the evaluation of social intelligence in psychometrics, the evaluation for animals, machines and a new paradigm focussed on the formal evaluation of intelligence using notions from algorithmic information theory.

Chapter 4) Extending a General Intelligence Test to Evaluate Social Intelligence

During a recent line of research to formally define intelligence, in 2007, a promising definition of intelligence called Universal Intelligence was presented, which basically defines the intelligence of an agent as its ability to perform well in the possibly infinite variety of environments. Following ideas from this definition and algorithmic information theory, a universal and formal, but at the same time practical, intelligence test was attempted. In this chapter, we extend this general intelligence test to consider several agents. This extension is performed in order to obtain some insights about the evaluation of social intelligence, paying special attention to how the inclusion of other agents influences the evaluee and its performance in an environment class created following the principles of a universal intelligence test.

This chapter is based on works from [69, 44]. Section 4.3 is based on the material in [56], which is an original work.

Chapter 5) Defining Social Intelligence Universally

Hitherto, the evaluation of social intelligence has typically been done in an informal and non-interacting way. This chapter presents a framework for a formal and parametric definition of social intelligence, which measures the performance of the evaluee interacting with other agents, which it has to compete and cooperate with. This parametrisation allows us to indicate any kind of social situation or testbed (in the form of agent and multi-agent environment sets and weights) of interest, allowing us, not only to define the social intelligence of an agent, but even to calculate it for a particular social situation. The definition also arranges the group of agents into teams, which fosters competitive and cooperative behaviours for the agents and facilitates the measurement for agents with low degrees of social intelligence. From this social intelligence definition, we define how to obtain a test to feasibly evaluate an agent.

This chapter is based on the material in [59], which is an original work.

Chapter 6) Experimental Analysis for Several Types of Environments and Agents

In order to validate the previously presented definition of social intelligence and find possible weaknesses on it, in this chapter we perform some experiments with Q-learning agents, modified to (somehow) behave more socially, interacting in several multi-agent environments. The experiments analyse the impact of the selection of environments and agents (and how this relativeness is understood), whether we can effectively gauge between competitive and cooperative behaviours for the agents by using teams, and finally, whether the definition actually focusses on social intelligence.

This chapter is based on the material in [59], which is an original work.

Chapter 7) Properties About Social Intelligence Testbeds

Since not all testbeds are suitable to evaluate social intelligence properly, we need to figure out a way to analyse any testbed, by means of its components, in order to determine its suitability to be used in a social intelligence test. In this chapter, we propose a formalisation of some property models in order to characterise any testbed. Several properties we present are parametrised and by using them we provide quantitative values for the testbeds. We define some properties that we consider are associated with social intelligence, such as social dependency and anticipation, and we analyse some properties that a good test of social intelligence should have, such as discrimination, grading, boundedness, symmetry, validity, reliability and efficiency. These properties help to: identify the components of social intelligence and its varieties; make clear that the mere appearance of other agents does not make a context social; and pave the way for the analysis of whether many multi-agent environments, games and tests found in the literature are useful for measuring social intelligence.

This chapter is based on the material in [61, 60], which are original works.

Chapter 8) Characterising Several Multi-Agent and Social Scenarios

Due to de current absence of environments specially designed to evaluate social intelligence, in this chapter, as a proof of concept, we use the previously defined properties to characterise some common multi-agent environments. With their characterisations we want to analyse their suitability as a test to evaluate social intelligence, compare them and analyse their differences. Besides, we want to analyse their strengths and weaknesses to figure out some of the facets that a good multi-agent environment should have to be used in a social intelligence test.

This chapter is partially based on the material in [60], which is an original work.

Chapter 9) Conclusions and Future Work

The final chapter discusses some conclusions and future work that arise from this thesis.

Appendices A, B and C

Appendices present how we calculate the property values for the matching pennies, prisoner's dilemma and predator-prey environments analysed in chapter 8.

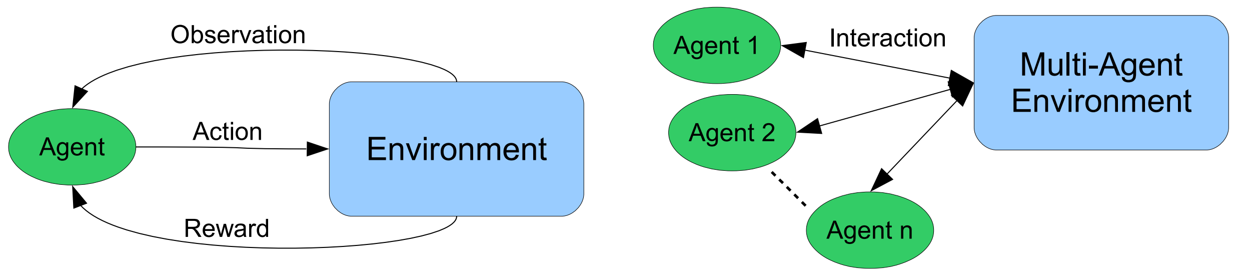
Chapter 2

Background

This chapter gives an introduction to the needed concepts and terminology and serves as a background for the following chapters.

2.1 Multi-Agent Environment

An environment can be seen as a place (or world) where an agent can interact through observations, actions and rewards, as seen in figure 2.1 (a). This general view of the interaction between an agent and an environment can be extended to various agents by letting them interact simultaneously with a multi-agent environment, as seen in figure 2.1 (b).



(a) Interaction between a single agent and an environment.

(b) Simultaneous interaction of multiple agents with a multi-agent environment.

Figure 2.1: Interaction between the agent(s) and the (multi-agent) environment.

A multi-agent environment is an interactive scenario with several agents. A multi-agent environment accepting n agents defines n parameters (one for each agent) that we denote as *agent slots*. We use $i = 1, \ldots, n$ to denote the agent slots. We call a *time step* to each simultaneous interaction of the n agents, where the order of events is always: observations, actions and rewards. $\mathcal{O}_i$ is the observation set that the agent in agent slot i can perceive from the environment, $\mathcal{A}_i$ is the action set provided by the environment that the agent in agent slot i can perform and $\mathcal{R}_i \subseteq \mathbb{Q}$ represents the possible rewards obtained by the agent in agent slot i from the environment. For each time step k, the agent in agent slot i must perceive an observation $o_{i,k} \in \mathcal{O}_i$, perform an action $a_{i,k} \in \mathcal{A}_i$ and obtain a reward $r_{i,k} \in \mathcal{R}_i$. We use o_k, a_k and r_k respectively to denote the joint observation, joint action and joint reward profiles of the n agents at time step k (i.e. $o_k = (o_{1,k}, \ldots, o_{n,k}) \in \mathcal{O}_1 \times \cdots \times \mathcal{O}_n$ represents the joint observation profile at time step k, and similarly for actions and rewards). For example, a sequence of two time steps in a multi-agent environment is then a string such as $o_1a_1r_1o_2a_2r_2$ and the string $o_{1,1}a_{1,1}r_{1,1}o_{1,2}a_{1,2}r_{1,2}$ denotes the sequence of observations, actions and rewards for the agent in agent slot 1.

At time step k, the term $\pi(a_{i,k}|o_{i,1}a_{i,1}r_{i,1}\ldots o_{i,k})$ is a probability measure, denoting the probability of the agent in agent slot i to perform action $a_{i,k}$ after the sequence of events $o_{i,1}a_{i,1}r_{i,1}\ldots o_{i,k}$. The observation provided by the multi-agent environment at time step k to the agent in agent slot i also has a probability measure $\omega(o_{i,k}|o_1a_1r_1\ldots o_{k-1}a_{k-1}r_{k-1})$. As with the observation, the reward provided at time step k by the environment to the agent in agent slot i depends on observations, actions and rewards at previous time steps $\rho(r_{i,k}|o_1a_1r_1\ldots o_ka_k)$. Note that the rewards obtained by each agent depend on the joint observations, actions and rewards of all the agents interacting in the environment, and not only on their own. A random agent (usually denoted as π_r) in agent slot i is an agent that chooses its actions from $\mathcal{A}_i$ using a uniform distribution.

In order to evaluate an ability (such as social intelligence), we basically need a system where agents can interact showing their behaviour and, at the same time, where the system can judge such behaviour (in the form of environment rewards) to give an assessment. Following this principle, a system that is not providing any kind of assessment to agents' behaviour cannot be used for evaluation purposes. This is the reason we use an interactive model, where agents are forced to interact with the environment and the environment provides them rewards based on their behaviour. For simplicity, we will assume a discrete time in this interaction.

Below, we present some of the multi-agent environments that we use in the following chapters.

2.1.1 Matching Pennies

Matching pennies [136] can be considered the simplest game in game theory featuring competition. It is a binary version of rock-paper-scissors. This game consists of two players (or agents) each flipping a coin. If both coins match player 1 wins, otherwise player 2 wins.

This game is played as a repeated game (i.e. it is the iterated matching pennies), which means that the game is played on a single time step and repeated for several time steps. Each player can see the actions performed by the other player, so they can use past time steps in order to predict the other player's strategy. For the agent in agent slot i, this environment only allows two actions $\mathcal{A}_i = \{\text{Head}, \text{Tail}\}$ and only provides two rewards $\mathcal{R}_i = \{-1, 1\}$, which correspond to lose and win respectively. For the agent in agent slot i, the environment provides an observation set $\mathcal{O}_i = \mathcal{A}_j \cup \{\lambda\}$ (where j represents the agent slot of the other agent) and the observation function ω returns to each agent the action performed by the other agent in the previous time step or λ if it is the first time step. Table 2.1 shows the reward function ρ as a reward matrix, which has the actions of both agents as input and their rewards as output.

	Head	Tail
Head	$(1, -1)$	$(-1, 1)$
Tail	$(-1, 1)$	$(1, -1)$

Table 2.1: Matching pennies' payoff matrix. Rows and columns represent the actions performed by the agents in agent slots 1 and 2 respectively. Cells content (X, Y) corresponds to the rewards obtained by the agents in agent slots 1 and 2 respectively when the actions leading to that cell are performed.

2.1.2 Prisoner's Dilemma

The prisoner's dilemma [93] is a simple and well-known game involving competition and cooperation. In this game, two prisoners (or agents) are suspects of a crime, and are asked whether the other prisoner is guilty of that crime. If both remain silent and do not betray the other, both spend a short time in prison. If one remains silent but not the other, the one who betrays reduces its time in prison to the minimum sentence, and the other prisoner obtains the maximum sentence. Finally, when both prisoners betray the other, both spend a long time in prison.

As happens with the matching pennies, this game is played as a repeated game (i.e. it is the iterated prisoner's dilemma), which means that the game is played on a single time step and repeated for several time steps. Each player can see the actions performed by the other player, so they can use past time steps in order to predict the other player's strategy. For the agent in agent slot i, this environment only allows two actions $\mathcal{A}_i = \{\text{Silent, Betray}\}$ and provides four rewards $\mathcal{R}_i = \{-4, -3, -2, -1\}$, which correspond to the time spent in prison (where $-i$ represents i units of time in prison). Instead, we can normalise $\mathcal{R}_i$ to have rewards between -1 and 1 as $\mathcal{R}_i = \{-1, -0.33, 0.33, 1\}$. For the agent in agent slot i, the environment provides an observation set $\mathcal{O}_i = \mathcal{A}_j \cup \{\lambda\}$ (where j represents the agent slot of the other agent) and the observation function ω returns to each agent the action performed by the other agent in the previous time step or λ if it is the first time step. Table 2.2 shows the reward function ρ as a reward matrix, which has the actions of both agents as input and their rewards as output.

	Silent	Betray
Silent	$(0.33, 0.33)$	$(-1, 1)$
Betray	$(1, -1)$	$(-0.33, -0.33)$

Table 2.2: Prisoner's dilemma's payoff matrix. Rows and columns represent the actions performed by the agents in agent slots 1 and 2 respectively. Cells content (X, Y) corresponds to the rewards obtained by the agents in agent slots 1 and 2 respectively when the actions leading to that cell are performed.

The prisoner's dilemma differs from matching pennies by including some cooperation, since both agents can cooperate by remaining silent to spend a relatively little time in prison.

2.1.3 Predator-Prey (Pursuit Game)

One typical multi-agent environment for cooperation that uses a 2D discrete space is a pursuit game called predator-prey [3], where the evaluee acts as a predator and has to cooperate/coordinate with other two predators in order to chase a prey. If they succeed chasing the prey, the goal is achieved. Figure 2.2 shows an example of a predator-prey scenario.

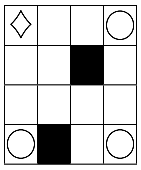


Figure 2.2: A predator-prey scenario with a 4x4 grid space. ◯ denotes a predator, $\diamondsuit$ denotes the prey and a black cell denotes a block. At each time step, agents can stay or move one cell horizontally or vertically, but blocks and boundaries cannot be crossed. The prey is chased once it shares a cell with a predator.

Many variants have been proposed about this scenario, which provides a high diversity of environments. Some examples include spaces with and without obstacles or boundaries, and many variants about the parameters have been considered: the distance of the grid that the agents can perceive, the number of predators or preys, the speed of the agents, etc. Even the definition of how the prey is chased has been modified, e.g. the prey is surrounded by the predators, or one predator chases the prey by occupying the same position. Some of these variants add a variety of social complexity, such as different levels of cooperation/competition by having to interact with different numbers of predators or preys, or having faster preys. These and other pursuit games have been widely studied and used in multi-agent systems (e.g. [27, 111, 17, 18, 82]).

Since we cannot use all the variants, we just select one of them. In what follows, we use the predator-prey environment shown in figure 2.2, which also shows its initial position. The pursuit game is typically performed in episodes. We make an episode to end after six time steps are performed. Even if the prey is chased, the interaction continue until the episode is finished.

For the agent in agent slot i, this environment allows four actions $\mathcal{A}_i = \{$Up, Right, Down, Left$\}$, which leads the agent to the cell facing this direction (when an agent performs an action leading to a block or boundary, the agent does not move). For the agent in agent slot i, the environment provides three rewards $\mathcal{R}_i = \{0, -6, 6\}$, which for the prey correspond to 'the episode is not finished', 'failed to survive' and 'achieved to survive' respectively, and for the predators correspond to 'the episode is not finished', 'failed the chase' and 'achieved the chase' respectively. At time step 1, the prey is located in the upper left corner and the three predators are located in the upper right, bottom left and bottom right corners. For the agent in agent slot i, the environment provides an observation set $\mathcal{O}_i$ which corresponds to the set of spaces with any possible location of the agents, and the observation function ω returns for every agent a description of the environment as, for example, figure 2.2. Table 2.3 shows the reward function ρ as a payoff matrix which has the current time step and the chasing situation as input and the agents' rewards as output. Note that after the six time steps, the average reward for each agent is -1 or 1 depending on whether the agent's team wins the chase.

2.1.4 Pac-Man

Pac-Man is a simple and well known game, but still complex enough to the state of the art in AI, which uses a 2D maze. The AI community has used this game as a testbed in order to

	Chased	Not chased
Time step 1 to 5	$(0, 0)$	$(0, 0)$
Time step 6	$(-6, 6)$	$(6, -6)$

Table 2.3: Predator-prey's payoff matrix. Rows represent the time step, while columns represent whether the prey has been chased or not. Cells content (X, Y) corresponds to the obtained rewards for the prey and each predator respectively.

evaluate their algorithms, e.g. [29, 118]. This game resembles a pursuit game, but this time the player represents the prey role (most of the time), so it must avoid being caught by the enemies (represented by ghosts). In order to win, Pac-Man must also collect all the pills that are spread through the space, which also provide some points. On the other hand, ghosts are appearing one by one over time, and they win if at least one of them is able to chase Pac-Man. If Pac-Man is able to reach certain locations in the space and eat specific pills, it becomes invulnerable for a short period of time, and receives additional points by chasing the ghosts. Figure 2.3 shows a Pac-Man game screenshot.

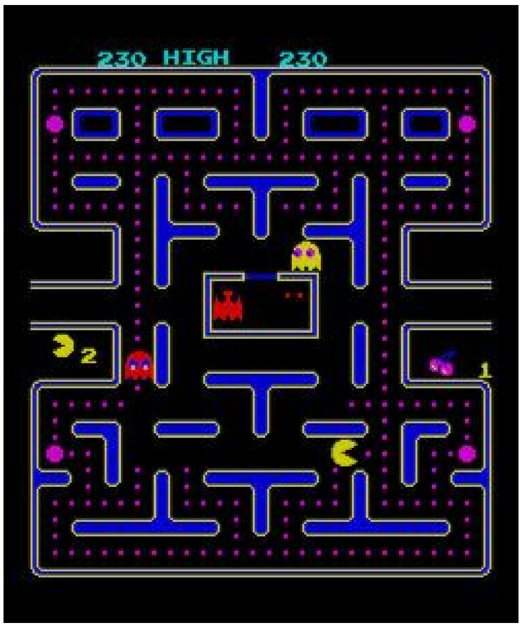


Figure 2.3: A Pac-Man game screenshot. Pac-Man is represented by a yellow circle with a mouth, which must avoid enemies represented by ghosts. Pac-Man must collect all the small pills and big pills change chasing agents' roles for a limited period of time. *Taken from* `http://www.freepik.com/` *with permission.*

For the agent in agent slot i, this environment allows four actions $\mathcal{A}_i = \{\text{Up, Right, Down, Left}\}$, which leads the agent to this direction (when an agent performs an action leading to a wall, the agent does not move). For the agent in agent slot i, the environment provides three rewards $\mathcal{R}_i = \{0, -1, 1\}$, which for Pac-Man correspond to 'do not eat any pill', 'be captured by a ghost' and 'eat a pill' or 'capture a ghost' respectively, and for the ghosts correspond to 'do not capture Pac-Man', 'be captured by Pac-Man' and 'capture Pac-Man' respectively. At time step 1, ghosts are located in the middle of the space and Pac-Man is located just below the ghosts. For the agent in agent slot i, the environment provides an observation set $\mathcal{O}_i$ which corresponds to the set of spaces with any possible location of the agents and the pills remaining, and the observation function ω returns for every agent a description of the environment as, for example, figure 2.3. The reward function ρ simply gives to Pac-Man rewards (or points) as long as it captures the pills that are spread in the space or, while it becomes invulnerable, additional points when chasing a ghost. Meanwhile, ghosts obtain rewards only when they chase Pac-Man and get negative rewards when get caught while Pac-Man is invulnerable.

2.1.5 RoboCup Soccer

As an example of a 3D space game we find the RoboCup Soccer competition [67]. Here, two artificial groups of agents (or teams) have to compete against each other in order to win a soccer match. The agents in each team must cooperate to make the ball reach the adversary's goal, while cooperate to avoid the adversary to score a goal. The game follows the rules of a typical soccer match. Figure 2.4 shows a picture of a RoboCup Soccer match, but actually we use a virtual version, which is just a simplification of the physical version.

Figure 2.4: A standard RoboCup Soccer platform with robots playing a match. *Used with permission from* `http://www.robocup2013.org/`*, photograph by Bart van Overbeeke.*

For the agent in agent slot i, this environment allows multiple actions $\mathcal{A}_i = \{$Up, Right, Down, Left, Kick, StandUp, $\dots\}$. For the agent in agent slot i, the environment provides three rewards $\mathcal{R}_i = \{0, -1, 1\}$, which correspond to 'the match is not finished' or 'tie the match', 'lose the match' and 'win the match' respectively. At time step 1, agents in one team are located on the left-hand side of the space and the other team is located on the right-hand side of the space. For the agent in agent slot i, the environment provides an observation set $\mathcal{O}_i$ which corresponds to the set of spaces with any possible location of the agents and the ball, and the observation function ω returns for every agent a description of the environment. The reward function ρ simply gives to each agent a reward of 0 while the match is being played, and when the match is finished, a reward of -1 when its team scored less goals than the other team, a reward of 0 when there is a tie and a reward of 1 when its team scored more goals than the other team. In order to ensure that, once the match is finished all agents obtain an average reward of $-1, 0$ or 1, $\mathcal{R}_i$ could be accordingly normalised.

2.2 Reinforcement Learning

The goal of reinforcement learning (RL) [110] is to make the agent learn how to solve a problem based on reinforcement signals (in the form of positive and negative rewards). Depending on the rewards the agent perceives, it reconfigures its behaviour modifying its future actions in order to obtain the maximum number of positive rewards. As a result, the agent 'learns' what to do to correctly solve these problems. In brief, reinforcement learning wants an agent to act in an environment in such a way that maximises the rewards it obtains.

The reinforcement signal may be immediate or delayed; an immediate signal represents a critic about each action just after the agent performs it. Therefore, the information provided by the reinforcement signal is local to each action. On the contrary, a delayed signal is not given after each action, but after completing a sequence of actions in order to solve a problem. In this case, this signal represents an overall assessment of the agent's behaviour. There also exist hybrid situations where positive rewards are not immediately provided after performing an action in the good direction, but there is neither a task nor problem to solve where rewards are not provided until the end.

There are many algorithms to address the reinforcement learning problem. Some of them make assumptions about the environments they interact with, as, for example, that they are 'Markov Decision Process' (MDP), while other algorithms try to address the problem by considering more general classes of environments.

2.2.1 Q-learning

One of the reference algorithms in reinforcement learning is the technique known as Q-learning [123, 124]. This reinforcement learning technique tries to learn an action-value function, indicating the expected utility of performing an action in a state. One of the strengths of Q-learning is its ability to compare the expected utility of its available actions without having a complex model of the environment, being as well one of its weaknesses, since usually it is not able to generalise enough to correctly "understand" the environment.

This technique consists of a set of states $\mathcal{S}$ (being each state a representation of the environment in a particular situation), a set of actions $\mathcal{A}_i$ (having the agent in agent slot i) and a quality value $q \in \mathbb{Q}$ for each pair of state-action. On each time step k, the agent models the environment as a state s_k, usually calculated as a simple function with the observation provided by the environment as a parameter (i.e. $s_k = f(o_{i,k})$). We use $Q(s, a)$ to denote the quality value of performing action a at state s and Q to denote the whole $Q(s, a)$ values.

The goal of Q-learning is to find (one of) the best Q that solves the problem, obtaining at the end the highest possible rewards. To do this, after each time step k the agent corrects its Q as follows:

$$Q(s_k, a_k) \leftarrow (1 - \alpha_k(s_k, a_k)) \times \underbrace{Q(s_k, a_k)}_{\text{old value}} + \alpha_k(s_k, a_k) \times \left(\overbrace{r_k + \gamma \underbrace{\max_a Q(s_{k+1}, a)}_{\text{estimate of optimal future value}}}^{\text{learned value}} \right) \tag{2.1}$$

where $\alpha_k(s_k, a_k)$ represents the 'learning rate' (it may be the same for all time steps and/or state-action pairs) and γ represents the 'discount factor'. r_k is the responsible to update Q, which is calculated as a function with the reward provided by the environment as a parameter (i.e. $r_k = g(r_{i,k})$). A simple option is to consider $r_k = r_{i,k}$.

Initially, Q must be filled with an 'initial value' Q_0. More formally:

$$\forall s, a : Q(s,a) \leftarrow Q_0 \tag{2.2}$$

Here we have many variants to select the Q_0 value. A Q_0 value equal to 0 makes the agent to quickly exploit positive rewards, while a very high Q_0 value makes the agent to quickly explore the environment. Since the algorithm goal is to find the best Q values, the closer (on average) Q_0 is to this (unknown) Q values, the better.

Finally, the agent just selects which action to perform as the action with the highest quality value in that state. More formally, at each time step k, the agent selects the next action to perform $a_{i,k}$ given the current state s_k with:

$$a_{i,k} = \arg\max_{a \in \mathcal{A}_i} Q(s_k, a) \tag{2.3}$$

But this policy may lead the agent only to a local maximum. In order to let the agent explore more states, we provide the algorithm with a 'random rate' β to sometimes select a random action. More formally:

$$a_{i,k} = \begin{cases} rand(\mathcal{A}_i) & \text{if } 0 \leq x < \beta \\ \arg\max_{a \in \mathcal{A}_i} Q(s_k, a) & \text{otherwise} \end{cases} \tag{2.4}$$

where x is a random number between 0 (inclusive) and 1 (exclusive) and $rand(\mathcal{A}_i)$ selects an action from $\mathcal{A}_i$ using a uniform distribution.

Learning rate α

The learning rate (with $0 < \alpha \leq 1$) determines to which extent the new information overrides the old information. A factor of 0 makes the agent not learn anything (this is the reason α is never equal to 0), while a factor of 1 makes the agent only consider the newest information.

Discount factor γ

The discount factor (with $0 \leq \gamma < 1$) determines the importance of future rewards. A factor of 0 makes the agent only consider current rewards. Meanwhile, a factor near to 1 makes the agent consider more long-term rewards. If the discount factor is set to be equal or greater than 1, the values of Q may diverge, which is usually avoided.

2.2.2 SARSA

A Q-learning variant known as SARSA [100] does not assume that the estimate of optimal future value (in equation 2.1) will actually be the one obtained in the next time step. Due to the exploration policy introduced by the random rate β, the next action may be different, and such randomly selected action may give the agent an unexpected and worse reward. Instead, SARSA waits until such next action is actually performed in order to give a value to the current action. In other words, this slightly different update policy makes SARSA to not only consider the action that produces the estimate of the optimal future value, which may not occur due to the exploration, but to also consider those actions that may occur during the exploration.

2.2.3 QV-learning

Another Q-learning variant known as QV-learning [140] considers a state-value function (V-function) in order to update its Q values. QV-learning is also enhanced by eligibility traces [109] to learn the state-values of the V-function using TD(λ) methods. The update done by QV-learning uses such a V-function instead of the estimate of optimal future value (in equation 2.1). The idea of using a V-function is that it does not consider actions and, therefore, it is more often updated and, supposedly, more informed about the state where the agent currently is interacting.

2.3 Kolmogorov Complexity

In algorithmic information theory, the Kolmogorov complexity of an object is the length of the shortest program that describes the object. Kolmogorov complexity is also known as 'descriptive complexity', 'Kolmogorov-Chaitin complexity', 'algorithmic entropy' or 'program-size complexity'.

To define the Kolmogorov complexity, we first must describe a Turing-complete description language for strings. Such a description language can be based on any computer programming language, such as C or Java. If p is a program (for example written in C) executed in a reference machine U that obtains as output the string x, then p is a description for x in the reference machine U. A typical assumption is to use a language that is prefix-free, which means that no string describing a program can be substring of any other program.

Once we find the program p with the minimum length that obtains x (with regard to the rest of programs that obtain x), then the length of p is the so-called Kolmogorov complexity.

Definition 1. The Kolmogorov complexity of a string x using a reference machine U is denoted as:

$$K_U(x) \triangleq \min_{p \text{ such that } U(p)=x} |p| \tag{2.5}$$

where $|p|$ denotes the length in bits of p and $U(p)$ denotes the result of executing p in U.

The importance of U mostly depends on the size of x. Since every Turing-complete machine can emulate any other, for every pair of Turing-complete machines U and V, there is a constant $c(U,V)$ (the length of the emulator of U in V) that only depends on U and V but not on x, so for all $x : |K_U(x) - K_V(x)| \leq c(U,V)$. The constant $c(U,V)$ will be relatively less significant as the length of x becomes larger.

For a better understanding of the Kolmogorov complexity, let us see an example:

azazazazazazazazazazazazazazazazazazazazazazazazazazazazazaz
lnkjdfglkfgvoijmnfgvil

As we can see, the first string is a succession of 'az' which can be described in natural language (in English) as 'az 30 times', using a total of 11 symbols (including spaces). However, for the second string, despite being much smaller than the first string, it is difficult to find a way to describe it with 11 symbols or less. Therefore, in English, it is more complex to describe the second string than the first one.

The main problem of the Kolmogorov complexity is that it is not computable due to the halting problem (or more generally, the *Entscheidungsproblem*). Let us imagine that we have a program p that provides x, whose length is $|p| = n$. This means that n is an upper bound to the Kolmogorov complexity of x, so $n \geq K(x)$. However, there may exist other programs that obtain x whose length is lower than n, but they need an almost infinite time to be executed. This is the reason this complexity is uncomputable in practice.

However, some other variants to Kolmogorov complexity are computable. One of them is the so-called Levin's Kt complexity.

Definition 2. The Levin's Kt complexity of a string x using a reference machine U is denoted as:

$$Kt_U(x) \triangleq \min_{p \text{ such that } U(p)=x} \{|p| + \log time(U, p)\} \tag{2.6}$$

where $|p|$ denotes the length in bits of p, $U(p)$ denotes the result of executing p in U and $time(U, p)$ denotes the (computational) time that U takes to execute p.

Once the time to obtain the string x is added to the formula, any program requiring an infinite time y to finish will be more complex than any other program obtaining x in a finite period of time, since $\lim_{y\to\infty} log(y) = \infty$.

For more information about Kolmogorov complexity, see [71].

2.4 Probability Distribution

A probability distribution (or simply 'distribution') is a function that gives a probability $Pr(x)$ to each element x from a particular set X.[1] Independently of X, in a distribution the sum of its elements probability must be 1. More formally:

$$\forall X : \sum_{x \in X} Pr(x) = 1 \tag{2.7}$$

Next we show the three distributions we use during this thesis.

2.4.1 Uniform Distribution

A uniform distribution just gives equal probability to each element of a particular set X. This is simply done by giving $1/n$ probability to each element $x \in X$, assuming that the number of elements in X is n. More formally:

$$\forall x \in X : Pr(x) = \frac{1}{|X|} \tag{2.8}$$

where $|X|$ denotes the number of elements in X.

[1] Actually, a distribution over X may be discrete or continuous depending on whether the elements of X are respectively discrete or continuous. In what follows, we only consider discrete distributions.

2.4.2 Geometric Distribution

A geometric distribution gives a probability to the number of binomial trials (i.e. an independent repetition of a random experiment with only two possible results: success or failure) needed to obtain the first success, where every trial has the same probability of success p.

The following equation calculates the probability of any natural number:

$$\forall x \in \{1, 2, 3, ...\} : Pr(x) = (1-p)^{x-1}p \tag{2.9}$$

where p denotes the probability of success.

2.4.3 Universal Distribution

Using Kolmogorov complexity (section 2.3), we define the universal distribution as follows:

Definition 3. The universal distribution, with respect to a prefix-free machine U, gives to each string x (from the set of all possible strings X) a probability as follows:

$$Pr_U(x) \triangleq 2^{-K_U(x)} \tag{2.10}$$

where $K_U(x)$ is the Kolmogorov complexity of the string x.

This distribution gives a higher probability to those objects whose description is small, and gives lower probability to those objects whose description is large. When U is universal (i.e. Turing-complete), this distribution is similar (depending on a constant) to the universal distribution for any other universal machine, since they can emulate each other. However, two universal distributions may give very different values to the most likely strings.

2.5 Monte Carlo Approximation

A Monte Carlo method is a way to approximate a result from a set of randomly sampled values. Such methods are normally used when it is difficult or impossible to obtain an exact value. This approximation is based on the idea that, by taking a representative selection of values, the result obtained from such values will be close to the real value. In order to obtain a representative selection, the sample procedure should not be biased. The use of a random selection of values should avoid any kind of bias.

In figure 2.5 we illustrate how to obtain a Monte Carlo approximation for the root value of a tree, where every leaf has a value and a non-leaf node value is calculated from its children values. As long as the number of nodes increases, it becomes more difficult to calculate the root value. To solve this problem, instead of calculating all nodes values, the algorithm approximates the value of each non-leaf node at level α (with $\alpha \geq 0$) and calculates the values for those nodes above the α level as usual. To approximate a node value, the algorithm randomly selects β (with $\beta \geq 1$) paths from all its node-to-leaf paths (possibly with repetitions) and only uses the value at their leaves.

There exist some trees where each non-leaf node calculates its value from the weighted sum of the values of its children. In such trees, when approximating the value of a node we use the following process in order to seek a representative selection. The weights are transformed into unit weights (i.e. for each non-leaf node, the sum of its children weights is 1), and the β paths are selected using these unit weights as probability distributions (section 2.4). Finally, each approximated node obtains its value as the mean of the leaves values of its β paths.

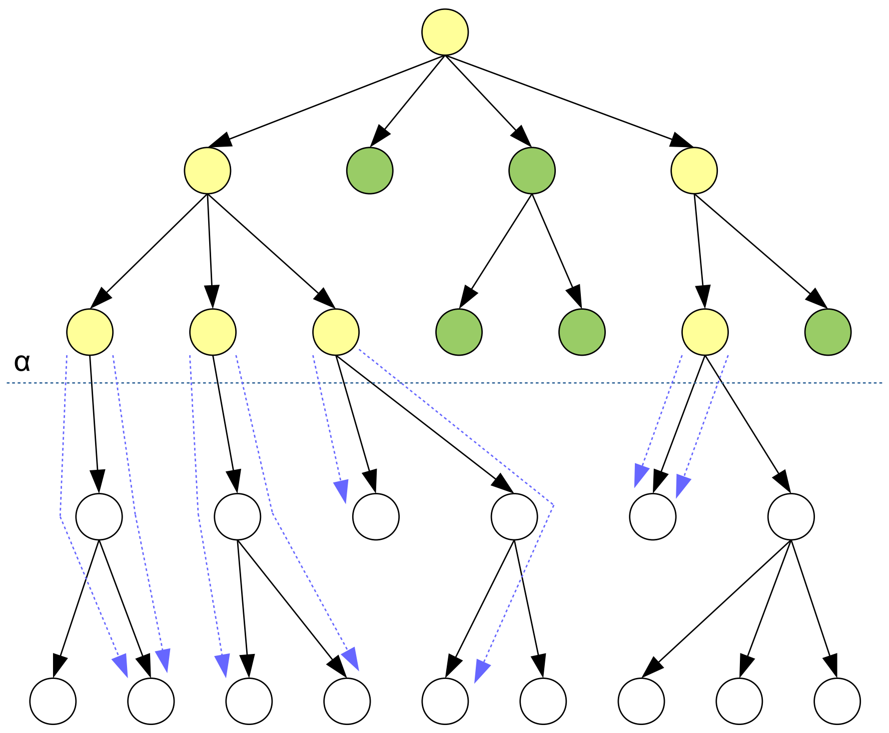


Figure 2.5: Monte Carlo method to calculate an approximation of the root value of the tree from its descendants. We represent an exact value with a green node and an approximated value with a yellow node. The method obtains a value for all nodes within the first α levels. For each yellow node at level α, blue arrows represent the β paths selected to approximate its value.

Chapter 3

State of the Art

3.1 Evaluation of Intelligence

The evaluation of intelligence and social intelligence has been studied in several disciplines during the last century. In this section we make a brief summary of the attempts to evaluate intelligence and social intelligence.

3.1.1 Psychometrics

The measurement of intelligence has its roots in Binet, which is considered the father of intelligence tests [8, 7, 6] and subsequent revisions [9, 5], later compiled and translated into English in [10]. His original study treated the problem of detecting those children who could not follow the regular rate of school. The study was focussed on elaborating a scale as a questionnaire with different questions related to reasoning and problem solving. They started with the assumption that the mental aptitude is a general and unique ability, and introduced the concept of “mental age” under the assumption that children’s mentality develops at a certain rate, so children from a certain biological age should perform well on a questionnaire made from items or exercises of that mental age. Then, a child that obtains a lower performance than those of his or her same biological age is considered to have a lower mental age. The scale was then revised and adapted to children of different ages, and was widely accepted due to its simplicity and ease of administration, making it very pragmatic. Nowadays, the concept of mental age has been abandoned and substituted with the average score of the population of the same age, thus a person’s score is compared with this average score.

In [106], Spearman proposed the theory of general intelligence. This general intelligence consists of a general factor (or g factor), which is mostly invariant during the person’s lifetime. He proposed that a person’s score in a mental test is composed of two parts, the general factor which is the same for every test, and a specific factor which may vary on each test for a mental aptitude. In this context, the g factor is nothing more than a score-factor which is invariant and cannot be trained. Thus, this g factor is omnipresent in all actions that require an intellectual aptitude. Nowadays, this theory of a general cognitive ability is still used and debated [91, 37, 88], where researchers defend that intelligence does not seem to constitute a unique ability as initially stated by Binet. Practically all current researchers agree on the existence of specific interrelated mental abilities or factors, although they differ in their number and nature.

In late 1930, Wechsler was trying to evaluate intelligence for adults, but having analysed tests from that period he concluded that they were not satisfactory. After revising all existing intelligence tests, he published the Wechsler-Bellevue Intelligence Scale (WBIS) [125] and its revision [126]. Such scale introduced some innovations to the evaluation of intelligence. It gets rid of the traditional quotient introduced by Binet, and replaces it with a non-scale with average at 100 for the average intelligence and with a standard deviation of 15. He also included his own classification of intelligence and introduced the concept of point scale, by assigning points to each item or exercise. This allowed the grouping of items by their content, instead of grouping them by the age of the evaluee as was traditionally done by Binet. Typical tests were composed only with a verbal scale, which was criticised for its emphasis on language and verbal skills. Wechsler added to his test a performance scale, where nonverbal items were included so the evaluee had to solve problems instead of only answering questions. The WBIS was later renamed as Wechsler Adult Intelligence Scale (WAIS) [128] and had several revisions [131, 133, 135]. Wechsler also created another version for children from 5 to 15 years old, the Wechsler Intelligence Scale for Children (WISC) [127], changed to children from 6 to 16 years old in posterior revisions [130, 132, 134].

Social intelligence (and its true distinction from general intelligence) has been a matter of study for many years. In psychometrics, social intelligence has its roots in [113], where Thorndike divided intelligence into three major abilities, which he identified as mechanical, social and abstract intelligence. While mechanical intelligence was referred to understand and manage physical objects and abstract intelligence to ideas and symbols, social intelligence was referred to understand and manage people. In its original definition, he defined social intelligence as “the ability to understand and manage men and women, boys and girls—to act wisely in human relations”. Since then, many other definitions have been proposed in psychometrics, such as the “ability to get along with others” [75], the “facility in dealing with human beings” [129], or more specific definitions including “[the] ability to get along with people in general, social technique or ease in society, knowledge of social matters, susceptibility to stimuli from other members of a group, as well as insight into the temporary moods or the underlying personality traits of friends and of strangers” [119]. Nonetheless, none of these definitions is sufficiently formal and operational to provide a clear measurement procedure. In fact, all these definitions require the definition of many other new concepts that appear in the definition.

The first test trying to measure social intelligence was the George Washington University Social Intelligence Test [77] and subsequent revisions [76, 78]. However, several controversies appeared about the test, such as whether social intelligence should be correlated with sociability or extraversion as done in the test (e.g. [114]). Furthermore, several studies revealed that test results were relatively high correlated with abstract intelligence [55] and partially with other multiple intelligence factors. Those difficulties for validating the results of the test led to lose interest on it.

From 1940, Spearman’s g factor obtained the main attention, where social intelligence was not considered an intelligence factor by itself, but only the ability to use general intelligence in the context of other persons. For instance, Wechsler considered that “social intelligence is just general intelligence applied to social situations” [129], so no subtest was specifically created to measure social intelligence. However, Wechsler considered that the picture arrangement and comprehension subtests of WAIS could be interpretable as measures of social competence, but there was not enough empirical experiments to support this claim [13].

In the late 1960s, Guilford revived the interest on social intelligence with his factorial definition of intelligence [34]. He proposed a taxonomy of intelligence with 120 possible factors. Each factor consisted on the combination of three different facets, which consisted on 5 kinds of operations, 4 kinds of contents and 6 kinds of products. Guilford stated that social intelligence was composed with factors from the content facet. Guilford's model was contrary to Spearman's *g* factor, gaining wide acceptance with those who did not consider the existence of a biological general factor of intelligence, while others criticised this model for being unnecessarily complex. Following this model, a test was created to measure social intelligence [86] but, again, the results correlated with those of general intelligence, losing validity.

From around 1985, psychologists put more attention on the understanding of social intelligence, leading to different theories about intelligence and letting its measurement aside. In [30], Gardner proposed his theory of multiple intelligences, where intelligence is not considered unique but as the combination of eight (but possibly more) different and semi-independent intelligences. Within his model, social intelligence can be identified as the interpersonal intelligence, where someone is able to identify, among other things, others' motivations and intentions. A different point of view is the Machiavellian Intelligence [12]. Here, Byrne and Whiten suggest that intelligence in humans has evolved from social interaction, i.e. interacting with other humans, viewing them as tools which can be manipulated.

Nowadays, from a theoretical perspective, it is still not clear whether social intelligence is simply the general intelligence applied to social situations, it is an actual factor of intelligence or it is an independent intelligence, although the latter has proliferated in the last years. Besides, it is still not clear how to empirically measure social intelligence, but it seems that a good solution could be to measure the cognitive strategies used to solve daily social problems.

3.1.2 Animals

Social intelligence is not only present in humans. Other animal species have also demonstrated this kind of intelligence. The evaluation of an animal is more difficult, as we cannot ask it to perform a test as we do with humans, so we make the animal confront other animals (usually of the same species) and see whether the subject deals with them or get along well. Also, tests include some food as rewards in order to motivate the animals to perform the tasks. This is the same configuration as in reinforcement learning (section 2.2). In social intelligence tests for animals, especially for those focussing on cooperation, they must obtain some food or reward that cannot be obtained by one individual alone, but two or more animals interacting are needed to get their reward. Some of the capabilities evaluated with those tests are their predisposition to deal with others, and their selfishness or altruism.

Although these tests measure some aspects of social intelligence, many have been devised to evaluate social intelligence for a particular species and for very specific tasks. In these tests, it is highly debatable whether the tasks are representative of a broader view of social intelligence. Also, it is usually very difficult to compare the results with those of other species. Fortunately, there have been some exceptions to this (species) specialisation, and they are proliferating in the past decade. For instance, some recent work has shown that social abilities can be compared in a systematic way between human children and non-human apes [51, 52].

Actually, comparative cognition [122, 102] is more and more concerned about performing tests that compare the abilities of many different species and also the abilities of individuals of different species. From this point of view, it should be more natural to provide a single test (with possibly many different customised interfaces and rewards) to assess every kind of species (or, in other words, a more general, or *universal* [44, 22, 46], test). To achieve this, such a test should be able to evaluate any level and spectrum of social intelligence, instead of focussing on the specific range and particular abilities of a single species.

3.1.3 Machines

When thinking about social tests and making them more species-independent, we can take the most general perspective, which leads us to the consideration of machines as well. However, evaluating social intelligence in machines has been quite different to the assessment of human and animal social intelligence, even from its beginning with the Turing Test [115]. Nonetheless, as occurs with animals, rewards or scores may be used as a measure of their performance and a way of giving feedback to make them show their abilities. Besides, environments must be presented in such a way that a machine can process the observations and perform a set of actions. This is done by providing them with sensors and actuators that interact with our physical world, or provide them with a logical or virtual environment.

Multi-agent learning (MAL) is a broad area of research where problems involving multiple agents have to be solved using machine learning (see [87] for a survey in cooperation). The environments used in tests for MAL tend to represent tasks that the agents must perform by interacting with other agents, so the performance is calculated as their capability to successfully cooperate with and/or compete against them to achieve some goals (see [3, 67] for two testbeds in multi-agent environments). In this way, the evaluation is a simulation in a social context, which is more directly linked to the definition of social intelligence.

Multi-agent systems (MAS) present rich and diverse possibilities for the interaction of distributed agents, both in competition [101, 93, 67, 4, 137] and cooperation [3, 111, 144, 107, 142]. Environments are usually selected to represent some particular problems for which MAL techniques are developed and evaluated. However, these evaluations lack some important features. They do not evaluate social intelligence in a general way, but they are typically designed to evaluate one kind of task. However, the most important problem is that they usually require very specific abilities, or when they require many, it is not clear how to disentangle them. For instance, if a MAS setting requires both competition and cooperation to solve a problem, it is not always easy to select or gauge the degree of relevance of each one in order to give more relevance to competition over cooperation, or vice versa. Nonetheless, the major issue is that many capabilities other than social intelligence also contaminate the results, which makes many MAS scenarios unsuitable if we want to measure social abilities only.

Efforts in MAS are usually put on creating or improving multi-agent algorithms. Few works are devoted to the evaluation perspective, and when it is the case, they are focussed on evaluating some characteristics of the agents such as their rationality, autonomy or reactivity (see [11] for an example). However, in order to evaluate social intelligence, it is convenient to put efforts not only on the agents, but also on the environments where they have to interact, treating the system as a whole.

Artificial general intelligence (AGI) research focusses on the original and ultimate goal of artificial intelligence, to create broad human-like intelligence by exploring, not only theoretical and experimental computer science, but all available paradigms or paths. AGI is a more complex area, because it is not based on the performance as in many artificial intelligence areas, but on more general-purpose systems with broader capabilities. AGI has been proliferating in the last decades. However, we still do not have a clear method to evaluate and compare their results [120, 121].

3.1.4 Formal Evaluation

As an alternative to the previous approaches in several disciplines[1], could we just start from a formal definition of an ability (such as social intelligence) and derive tests from it? This approach has been investigated for machine intelligence evaluation. Formal approaches started in the late 1990s using notions from Kolmogorov complexity, Solomonoff prediction and the MML principle [19, 47, 39, 40]. Legg and Hutter developed in [69] the so-called "Universal Intelligence", calculating it as the performance of the agent in a wide range of environments, weighted by a universal distribution. Adjusting to this definition, a framework to evaluate general intelligence [44] and an environment class [41] were proposed. In order to show their effectiveness, some experiments were performed [57, 62, 63, 70], but their results suggested that the framework still has some limitations. One of the possible reasons is that these environments lacked the richness of interaction. From the formal definition, it is virtually impossible to randomly generate an environment that contains some kind of social behaviour. Therefore, in order to overcome some of the limitations of these tests, some other agents need to be included in the environment to generate social situations.

In order to measure social intelligence in isolation, we need to provide an appropriate social environment class where only social intelligence is needed or, at least, where the degree of social intelligence needed can be fine tuned. For this purpose, in this thesis we provide a formal setting of social intelligence evaluation, where positions and roles of the agents used in the definition are taken into account.

In this context, one of the key issues is to determine what kind of agents we must include in the test to interact with and what their roles are. This boils down to choosing a distribution of agents. However, in order to provide an environment with some rich social situations, we need first to know the level of social intelligence of the agents provided by the distribution. This circular problem is turned into a recursive one in [48], where different levels of distributions are recursively provided by constructing a new distribution of agents from a prior distribution by selecting (or increasing the probability of) those agents with higher performance. However, it is not easy to derive a definition of social intelligence from here or a procedure to create environments that would be the base for social intelligence tests.

Overall, there are many different approaches for the study and evaluation of social intelligence, but we lack a comprehensive theory, well-grounded tools and wide comparisons to better understand the problem and find better measurement devices.

3.2 Works Focussed on the Interaction Between Agents

In the literature there are many works that investigate some aspects about the interaction between agents, such as their performance or how to improve and evaluate such agents to achieve better results in some environments. Although these works are focussing on the interaction between agents, they are not directly related or linked to the evaluation of social intelligence. In this section we discuss some of these other works.

[1]Not only as an alternative to MAS scenarios, but also to IQ tests, the Turing Test and other kind of tests (see [20] for a discussion).

In the context of social sciences (stretching from economics to AI), game theory [85] has studied the interaction of different agents in formalised structures (called games). For this purpose, game theory uses a formal approach to define a utility function, and the effort is made for finding the best strategy among all possible strategies, assuming that the rest of agents also try to obtain their best results following some kind of rational actions. Although game theory needs the interaction of several agents, the goal is not to evaluate social intelligence but rather to analyse how the agents (or just policies) behave in these games and whether they reach some kind of equilibrium. Several kinds of games try to represent or to analyse a variety of properties: cooperative or non-cooperative games, simultaneous or sequential games, normal-form or extensive-form games, zero-sum or general-sum games, and symmetric or asymmetric games [81, 85]. However, games that may have the most interesting properties or applications are not necessarily useful for testing. For instance, a game where equilibria are easy may be inappropriate if we want a discriminative test. Similarly, asymmetric games make it more difficult to assess agent performance, as they depend on the role each agent takes.

One important concept in game theory is the notion of zero-sum vs. non-zero-sum games. Zero-sum games are a particular set of games where a player's gains (or losses) are equally balanced by the other players' losses (or gains). These kinds of games are known as competitive games, since one's gains reduces the gains (or increases the losses) of the other player(s), making the players having opposed interests. When a zero-sum game only has two players it becomes a pure competitive game. But zero-sum games can also contain cooperation in games with three or more players. Two players can cooperate in order to compete against a third or more players. As the number of players increase, cooperation becomes more important. In contrast, general-sum games are those games where the payoffs sum more or less than zero, and games can be cooperative even for two players. Finally, another particular feature in game theory is that environments are generally simple (without objects or spaces) and it is just the continuous interaction between agents that matters.

The understanding of social behaviours has also been studied from the perspective of robotics, where robots (physical or simulated) are used in order to better understand the interaction between the agents [74]. These studies are performed by avoiding some convenient assumptions on their designs or models and adjusting their robots to a more realistic vision of the environment. However, such studies are designed to understand such social behaviours in order to have more information to improve their robots performance, while less effort is made on how to better understand the evaluation of such social behaviour.

In cooperative/competitive co-evolution (see [16] for a competitive example), the problem is to design algorithms or models to evolve the behaviour of a group of agents in cooperating/competing scenarios. This line of research is more concerned about creating specialised algorithms for specific environments and empirically see if new generations are more prepared to succeed in their scenarios. This approach can make the agents obtain good performance for some specific social contexts, but their environment-specialisation will probably make them useless for other situations.

Ad Hoc teamwork is a recent challenge where autonomous agents have to learn to coordinate and cooperate with previously unknown teammates in order to solve some task as a team [108]. Besides, this challenge is more concerned about creating autonomous agents. Stone et al. propose that these agents have to be evaluated by making them interact in a team with a set of agents in a variety of environments or domains. But this evaluation is not always easy to be performed. In fact, part of the difficulty comes from the set of agents to use. For example, a socially intelligent agent may obtain bad results while cooperating with a set of non-social agents. Moreover, when they exist, opponent agents are treated as part of the environment, which makes their evaluations more dependent to the specific agents they face. Additionally, [1] proposes how to evaluate an agent in those environments or domains by using what they call dimensions, which try to quantitatively represent some information about the environment (including opponent agents) when the autonomous agent is interacting on it. Our main criticism about this approach is that they assume that the agent to be evaluated has to know in advance some predefined behaviours that their teammates may be following, so they can measure how precise that knowledge is with respect to the real behaviour of its teammates.

Some researchers try to obtain some information about what happened during the evaluations of their agents in some scenarios. For this goal, they create some *measures* (see [65, 98] for some examples), which usually count the number of times a certain situation occurs. For example, in soccer, a measure could be the number of scored goals, the percentage of time that their team is in possession of the ball or the number of successful passes. Most of these measures are very domain- or task-specific. However, some measures are somehow independent to a domain or task, such as for example interference [33, 97], which is used to count the amount of time that agents need to manage a spacial collision, usually with another agent. Finally, researchers try to improve the results that their agents obtain in these measures to, hopefully, improve their performance on these scenarios. However, it is difficult to use such measures to represent a property of the environment or characterise the system where the agent is being evaluated.

The analysis of environments, benchmarks and testbeds have also received some attention. Several researchers have acknowledged that the environments used during the evaluation is an important element to consider [139, 138], since such environments establish the rules that agents must follow and represent the abilities that are taken into consideration. Moreover, it is not always clear what benchmarks and testbeds are actually evaluating and how to interpret the results that agents obtain on them [35]. In order to better understand what environments represent, some studies try to identify their components or provide for them a topology as well as some intuitive properties (e.g. [117, 79]). Although providing an appropriate scenario where evaluate the agents' abilities seems an essential issue, such works barely emphasise on analysing the appropriateness of the environments or testbeds for evaluating particular abilities such as social intelligence.

Chapter 4

Extending a General Intelligence Test to Evaluate Social Intelligence

The evaluation of social intelligence in a formal way does not appear to have been carefully analysed yet. It seems that a good starting point is to use a general intelligence test and analyse how it performs when evaluating social intelligence. From here, one possibility is to adjust an existing general intelligence test in such a way that some interactions with other agents are ensured to make it more ‘social’.

In this chapter, we perform some experiments on such ‘social’ test setting in order to examine the way in which some simple agents have a big impact on their performance while competing and cooperating with other (intelligent) agents. There are many possible ways of introducing cooperation and competition, which may lead to different experimental results, some of them similar to what has been previously studied in the AI literature. Here, we do not want to evaluate these choices, but to analyse how the degree of ‘intelligence’ of the agents in a multi-agent environment affects the role of cooperation and competition. This is crucial in order to get more information about how to perform a social intelligence test.

Below, we present the features and principles of such general intelligence test, how it has been ‘socially’ adjusted to perform the experiments and we discuss their results.

4.1 Universal Intelligence

In [69], Legg and Hutter provide a new definition of intelligence called Universal intelligence, which is based on Kolmogorov complexity (section 2.3) and Solomonoff’s inductive inference [104, 105]. This definition basically consists in measuring the performance of an agent interacting in a variety of environments, where this variety stands for the set of all possible environments. The term universal stands for its capacity to include in the definition all the possible environments by the use of a universal distribution (section 2.4.3). Another possibility would have been to select the environments with a uniform distribution, but for an infinite set of environments this distribution would not work, since each environment would have a probability equal to 0. We show Legg and Hutter's universal intelligence in the following definition:

Definition 4. The universal intelligence of an agent π, using the reference machine U, is:

$$\Upsilon_U(\pi) \triangleq \sum_{\mu \in M} Pr_U(\mu) \times V_\mu^\pi \tag{4.1}$$

where M denotes the infinite set of all environments, $Pr_U(\mu)$ denotes the universal probability to obtain μ with the reference machine U and V_μ^π denotes the expected cumulative reward of π interacting in μ.

This definition makes the agent interact in the set of all environments M giving them a weight based on their universal probability by a reference machine U. This assumes a coding of the environments as strings in U and that environments are recursively enumerable [68]. For each environment, the agent's performance is calculated as the expected cumulative reward as follows:

Definition 5. The expected cumulative reward of agent π interacting in environment μ is:

$$V_\mu^\pi \triangleq E\left(\sum_{k=1}^{\infty} r_k^{\mu,\pi}\right) \tag{4.2}$$

where $r_k^{\mu,\pi}$ denotes the reward obtained by π interacting in μ at time step k.

In order to prevent equation 4.1 from giving an infinite value, Legg and Hutter impose a restriction on environments rewards. This restriction consists in making the expected cumulative rewards to always be less than or equal to one. More formally:

$$\forall \mu, \pi : V_\mu^\pi = E\left(\sum_{k=1}^{\infty} r_k^{\mu,\pi}\right) \leq 1 \tag{4.3}$$

Finally, in equation 4.4 we just rewrite equation 4.1 by substituting the universal probability and the expected cumulative reward with their definitions:

$$\Upsilon_U(\pi) \triangleq \sum_{\mu \in M} Pr_U(\mu) \times V_\mu^\pi = \sum_{\mu \in M} 2^{-K_U(\mu)} \times E\left(\sum_{k=1}^{\infty} r_k^{\mu,\pi}\right) \tag{4.4}$$

4.2 Universal Anytime Intelligence Test

Following the ideas of the universal intelligence definition presented above, in [44] Hernandez-Orallo and Dowe propose a conceptual framework to construct a universal and anytime intelligence test. Unlike universal intelligence, here the term 'universal' refers that the test is able to evaluate intelligence for all kinds of systems or species. Moreover, the term anytime refers to the test being applicable to agents that interact with different time spans, any level of intelligence, as well as that it can be interrupted at any time, providing an approximation to the agent's intelligence (i.e. the longer the evaluation time, the better the result it obtains).

In order to provide all these features, the test considers several issues about the environments such as their difficulty and discrimination power. Also, they must react immediately and they must be balanced (a random agent should obtain an expected reward equal to 0). Also, the test must provide a mechanism to sample the environments such that their difficulty adjusts to the level of intelligence shown by the evaluee.

An option to select the environments would be to use a universal distribution as in equation 4.4. Although this option would work theoretically, it would give high probability to simple environments while providing a low probability to more complex environments. This would make the test spend most of the time and importance evaluating the agent in the same simplest environments, but the complex (and probably more interesting) environments would be almost ignored.

When selecting the environments, it is important to take into account that not all of them are useful to evaluate intelligence. Some of them could give the same rewards to all evaluated agents, which makes them not discriminate between non-similar agents, so they would be unsuitable to be used in a test. Besides, some environments could be similar to previous environments used during the test. It becomes clear the necessity to make a correct selection of the environments to be used during the evaluation.

To ensure that an environment is discriminative, evaluated agents must be able to influence on the rewards they obtain. An option to achieve this would be to make the environment reward-sensitive, which means that depending on the sequence of actions the agents perform, the environment provides a different sequence of rewards.

A desired property is that environments must be able to interact indefinitely and must be computable. Since their descriptions must be finite, there must exist some kind of pattern making them not stop. The environment should also distribute the rewards among the whole evaluation session, thus avoiding undesired situations where rewards are only provided in a certain moment of the evaluation and not providing rewards for the rest of the time. Otherwise, most of the evaluation time will be useless, since evaluated agents will not be able to improve their rewards.

To promote some balancedness regarding random agents' rewards, environments must be balanced. Basically this means that random agents must obtain an expected reward of 0 on these environments. More formally:

$$\forall \mu : E\left(\sum_{k=0}^{\infty} r_k^{\mu,\pi_r}\right) = 0 \tag{4.5}$$

where π_r denotes a random agent. To facilitate this, an approach could be to use symmetric rewards. That is:

$$\forall \mu, \pi, k : -1 \leq r_k^{\mu,\pi} \leq 1 \tag{4.6}$$

In order to ensure that the environments react immediately, a possibility is to simply avoid using those environments that take too much time to output observations and rewards. One option is to take this time into consideration in the probability to sample the environments, making them less probable as long as their time to output observations and rewards increase.

To permit a test to evaluate different kinds of systems, the test must be presented in such a way that there is no bias in favour of a particular system. For example, an environment with a 3D space would be easier for humans than for machines, since humans have been interacting and learning in such kind of spaces since their birth. Also, the interface could be partially-observable since not always the environment information is available for the evaluated agent, but fully-observable interfaces could also be used.

Finally, a practical test should be applied in a reasonable period of time. One way to achieve this is that the test must adapt itself accordingly to the evaluee's performance. To achieve this, since several environments to approximate the level of intelligence of the evaluee could be necessary, each of these environments could be used during a limited period of time in the evaluation. Then, the test would select the next environment depending on the performance shown by the evaluated agent on previous environments, providing more complex environments as long as the evaluee obtains good results, and sampling simplest ones when its results decrease. As long as the environment becomes more complex, this period of time could increase to let the evaluated agent enough time to understand the environment.

4.2.1 Lambda Environment Class

According to the above discussion, in order to evaluate general intelligence, in [41] a class of environments (called Λ) was presented. Next we show the main characteristics of this environment class.

The environment class is basically composed of a discrete space where the evaluated agent can interact, some objects that can move through the space and two special objects that are responsible to provide the rewards in the form of reward units.

The space is composed of cells with actions that connect those cells. Each cell has a reflexive action that leads to itself. A cell can contain the evaluated agent, any number of objects and a reward unit. Objects are visible to the evaluated agent, but reward units are not. Before starting the evaluation, the evaluated agent and each object are randomly (using a uniform distribution) placed in cells and they can move to an adjacent cell by performing the action that leads to that cell.

When generating the environment, the space and each object behaviour (or movements) are generated using a Turing-complete language (such as Markov algorithms). This allows the environment class to have a universal descriptive power to obtain any possible behaviour for the objects.

Two special objects called *Good* and *Evil* are the responsible to provide rewards. Good and Evil move through the space following both the same behaviour (or sequence or pattern of actions) and leaving a reward unit on each cell they visit: Good leaves a positive reward unit (1) while Evil leaves a negative reward unit (-1). On each time step, the evaluated agent obtains a reward equal to the reward unit present in the cell it is occupying and 0 if the cell does not contain a reward unit. The reason that both special objects let these reward units and follow the same pattern of actions is to ensure symmetry on rewards and a balanced environment (random agents score 0 on average).

Good and Evil cannot share the same cell. If that occurs one of them is randomly chosen to move, while the other remains on its cell. This avoids those situations where both objects remain indefinitely in the same cell, cancelling their reward units and, therefore, making the evaluated agent to always obtain the same reward of 0. On each new time step, rewards units are divided by a constant (as if reward units were fading in an exponential decay), but disappear when the evaluated agent obtains them. This makes Good and Evil leave a trail of rewards units as they move through the space following their pattern of actions. This makes the environments interact indefinitely and distribute rewards during the whole evaluation session.

The space is randomly generated in such a way that its topology follows a strongly connected graph, where vertices represent cells and edges represent actions. A strongly connected graph is a directed graph where every vertex is reachable from any other vertex in one or more moves.

This restriction in the topology of the space is to avoid that the evaluated agent falls into a heaven or hell subenvironment, where it can only obtain positive or negative rewards. This permits the evaluated agents to have influence on their future rewards and, therefore, allows to better discriminate them. Figure 4.1 shows an example of a randomly generated space.

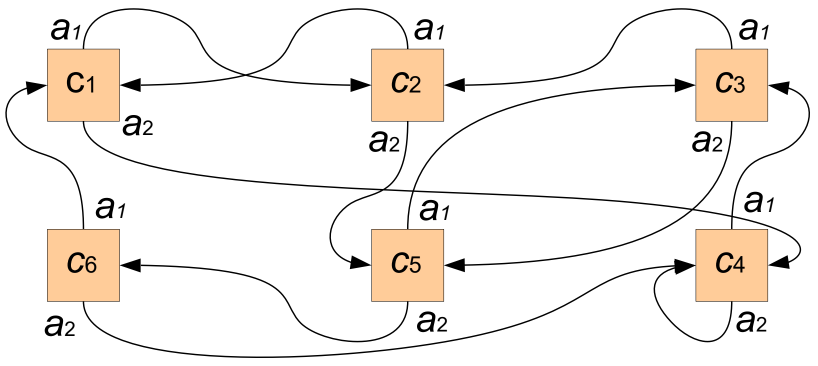


Figure 4.1: A space with 6 cells and 3 actions (a_0, a_1, a_2). Reflexive action a_0 is not shown.

Finally, the complexity of each environment can be determined by: the topology of its space (including the number of cells and actions) and the objects that are present (including their behaviour, movements or action pattern). In order to calculate the environment complexity, an option could be to generate a string describing the environment, and use this string to calculate (or approximate) the Kolmogorov complexity (section 2.3) of the environment.

4.2.2 A Prototype of a General Intelligence Test

Following the definition of the Λ environment class, some simplifications were performed to generate the environments [57]. For instance, speed is not considered, thus being a non-anytime version of the test. In addition, the prototype does not use a Turing-complete algorithm to generate the spaces nor the object behaviours. Spaces are generated by first determining the number of cells n_c, which is given by a number between 2 and 9, using a geometric distribution with $p = 0.5$ and normalising to sum up to 1. Similarly, the number of actions n_a is defined with a uniform distribution between 2 and n_c. Both cells and actions are indexed with natural numbers. There is a special action 0 which connects each cell with itself (the reflexive action). The arrows between cells are created by using a uniform distribution for each pair of cell and action, which assigns the destination cell for each pair.

The number of cells and actions is, of course, related to the complexity of the space, but not monotonically related to its Kolmogorov complexity or a computable variant such as Levin's Kt (section 2.3). Nonetheless, most of the actual grading of environments comes from the behaviour of Good and Evil. The sequence of actions for Good and Evil is defined by using a uniform distribution for each element in the sequence, and a geometric distribution to determine whether to stop generating the sequence, by using a probability of stopping $p = 0.5$. An example of sequence of actions for the space in figure 4.1 is 201210200, which means the execution of actions a_2, a_0, a_1, a_2, etc. Consider, e.g. that Good is placed at cell c_5. Since the sequence starts with '2', Good will move (via a_2) to cell c_6. For each time step the agents Good and Evil take the next action from the sequence and execute it. When the actions are exhausted, the sequence is started all over again. If an action is not allowed at a particular cell, the agent does not move.

The prototype lets Good, Evil and the evaluated agent interact for a certain number of time steps m, which is called an exercise (or episode). On each time step, reward units are divided by 2. For an exercise, the result or score of the evaluated agent in the environment is calculated as the average of its obtained reward units.

Figure 4.2 shows an example of the interface created for humans which is hopefully designed to be unbiased.

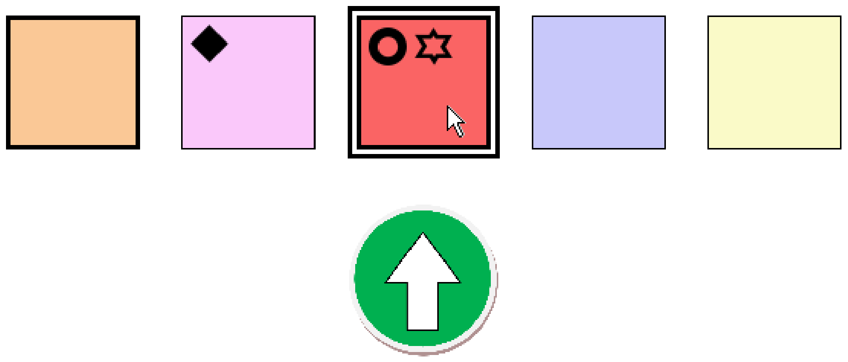


Figure 4.2: A snapshot of the interface for humans. The evaluated agent has just received a positive reward, shown with the green circle with an upwards arrow. The image also shows the evaluated agent located in cell 3, and Evil and Good are placed in cells 2 and 3 respectively. The evaluated agent can move to cell 1 and cell 3. Cell 3 is highlighted since the mouse cursor is over it.

Although the framework suggests to have a partially-observable interface, in the prototype it is fully-observable, so the evaluated agent can see all the cells and their contents (except reward units). The evaluated agent does not know in advance who Good is and who Evil is. It has to guess that.

The first evaluations using this prototype [57, 62] show that the setting is able to compare and evaluate different kinds of agents (humans and RL algorithms), but they do not properly reflect their supposed difference in intelligence, failing at placing them on the same scale, since humans usually get similar scores to those of other relatively simple agents.

4.3 Evaluation of Social Intelligence Using a General Intelligence Test

Many possible explanations are suggested in [62] about why the prototype of the previous section does not work in practice to evaluate general intelligence, with incremental knowledge acquisition and social intelligence being two of the abilities this test is not giving enough importance. With respect to the second issue, it is virtually impossible to find any social behaviour in the environments originating from the objects, so social intelligence is not measured in the test. In order to address this second issue, we must define environments which are more social. The question, therefore, is what and how agents should be introduced in the test, since the results of the evaluated agent will depend on the abilities of the other agents. This is related to the Turing Test and the question of evaluating intelligence with games (also suggested in [44]), where the difficulty is not only given by the complexity of the game, but from the opponent's intelligence. This leads to a circular problem: we need to know the opponent's intelligence first in order to know the difficulty of the problem. Figure 4.3 shows this situation. The question is what criteria we can use to introduce the other agents and how we can measure their (social) intelligence in advance.

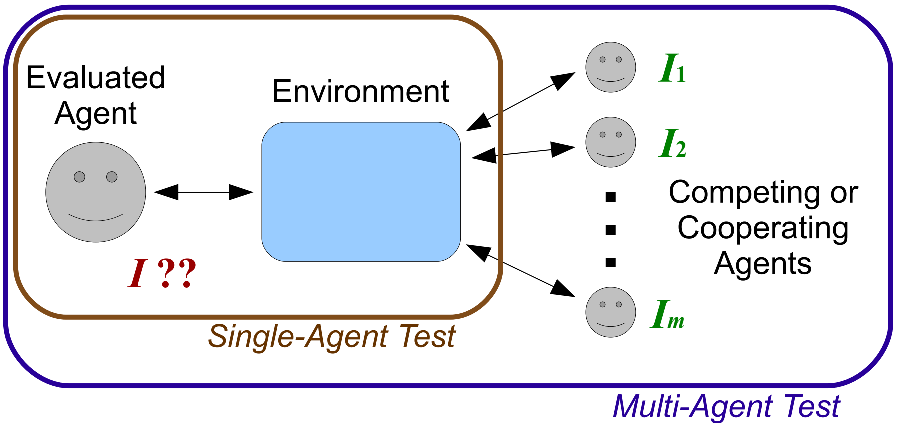


Figure 4.3: A multi-agent test compared to a single-agent test. In a multi-agent (social) intelligence test, other agents also interact (and become integral part) of the environment. In order to assess the intelligence of the evaluated agent, we need to know the intelligence of the other agents.

For the experiments, we use very simple reinforcement learning algorithms (section 2.2). The goal is neither showing how these algorithms behave nor comparing them. We just use them as off-the-shelf agents which can learn from an environment, in order to see how performance is affected by the introduction of more agents in an environment. Rather, the true goal is to analyse the behaviour of intelligence tests when environments are populated with agents, and how this affects the results of the evaluated agent. To have a broad picture, we examine several scenarios, some with competition and some with cooperation.

To evaluate social intelligence we need to develop intelligence tests in multi-agent scenarios. An easy way to achieve this is by adapting an existing intelligence test proposal into a multi-agent setting. This is what we do below.

4.3.1 Extending a General Intelligence Test to Consider Several Agents

The adaptation of the general intelligence test presented above to a multi-agent scenario gives us a fast and easy way to analyse what happens when we try to evaluate an agent interacting with other agents.

In particular we only need to consider how to include other agents in the environment. First, we let the environment include more agents. All agents can move freely to other cells independently of whether they are occupied or not by other agents. In other words, agents can share a cell. And second, Good and Evil are also considered agents but, since they are not even generally reactive, if no further agent is included, the environment cannot be considered a proper multi-agent system. Therefore, the behaviour of Good and Evil is slightly modified in such a way that they cannot step into a cell where any other agent is located. This re-introduces some degree of reactivity (with respect to the prototype), even in the case where no further agents other than Good and Evil are introduced.

The first question when several agents are introduced in the space is how the rewards are shared among them. A competitive, individualistic scenario is set by each agent trying to improve its own rewards. But we must address what happens when two or more agents share a reward unit. In such a case, one solution is to give as reward to each agent the reward unit divided by the number of agents in that cell. A second, relatively more difficult, question is how we can deal with cooperation. The easiest way of making this setting purely cooperative is by just putting all the obtained reward units in the same bag, so giving as reward to each agent an average of their reward units. With this, one should not be concerned about not getting some reward unit itself if some other agent is able to get it instead. What matters is the overall result. We can of course move between competition and cooperation by using the notion of team. All the members of a team put their reward units in the same bag, and each team should compete against the others as usual in games and economics.

Now we are ready to see what happens with a single-agent intelligence test when we convert it into multi-agent. But before that, we need to determine the agents that we use for the experiment. The agents are:

- Random: an agent which chooses randomly among the available actions using a uniform distribution.
- Q-learning (section 2.2.1): the most common reinforcement learning algorithm. We use the description of cells content as a state.
- SARSA (section 2.2.2): a well-known variant of Q-learning which also takes the future action into account.
- QV-learning (section 2.2.3) (without eligibility trace): a variant of Q-learning which partially resembles ActorCritic methods.

The three latter algorithms will be referred to as RL agents. In order to have a consistent view of the experiments, the parameters for all the RL agents algorithms (*learning rate* α, *discount factor* γ, *random rate* β and *initial value* Q_0) were fine-tuned on the single-agent scenarios, by using 1,000 exercises for each parameter variation, totalling a huge number of experiments to set the optimal parameters.

4.3.2 Evaluating Agents Isolatedly

We start our experiments with the scenario where agents are just taken and evaluated isolatedly. In addition, we just restrict the evaluation to environments which space has nine cells.

The result of figure 4.4 is clear (and consistent with the results in [57]). Random has an average reward of 0, as predicted by the theory. The three RL agents are very slow learners and only get closer to 0.5 after 10,000 time steps. Their behaviour is similar and the differences are small.

4.3.3 Evaluating Agents in a Competitive Scenario

More interesting things can be observed when we switch to the competitive scenario. Here all the agents are located in the space at the same time competing for reward units. As we can see in figure 4.5 Random gets a value which is even lower than 0, since most of the positive

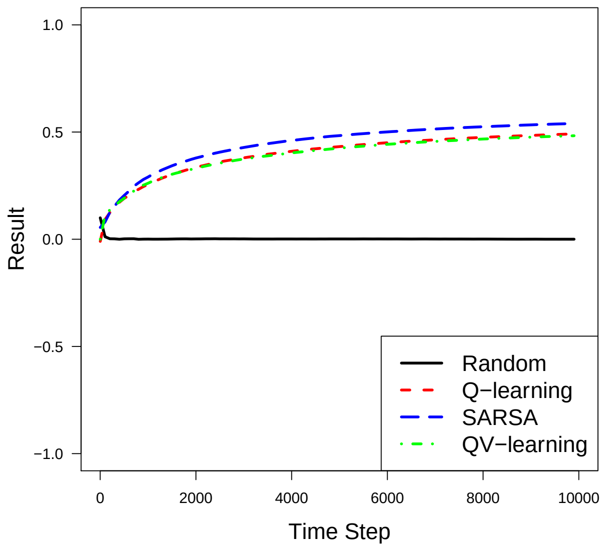


Figure 4.4: Isolated scenario. The four evaluated agents are evaluated independently over the same test. Agents' results (an average of rewards) after 10,000 time steps and 100 repetitions.

reward units are eaten by the other agents, leaving the negative rewards for Random. RL algorithms have a very poor result (not reaching 0.02 in 10,000 time steps). This is explained by the presence of Random, which makes the state-action tables of the RL algorithms grow considerably.

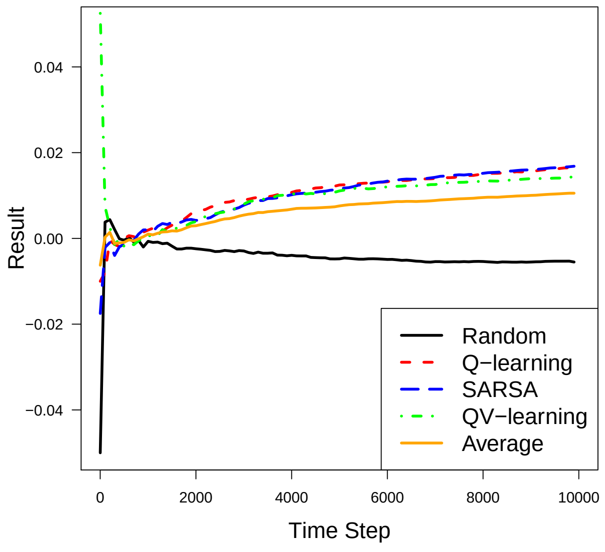


Figure 4.5: The three RL agents along with Random are evaluated at the same time competing for reward units. Agents' results (an average of rewards) after 10,000 time steps and 100 repetitions.

Finally, in order to further confirm that the problem is the state-action table space, we remove Random (which, given its random behaviour, can be considered noise), and we only leave the RL agents. We also increase the number of time steps to 100,000. This is shown in figure 4.6. Things improve slightly and, in the very long term, Q-learning and SARSA get close to 0.2, while QV-learning lags behind around 0.1. We see that just the presence of only two other agents makes their tables so big that they require more than 100,000 time steps to derive their Q values accurately.

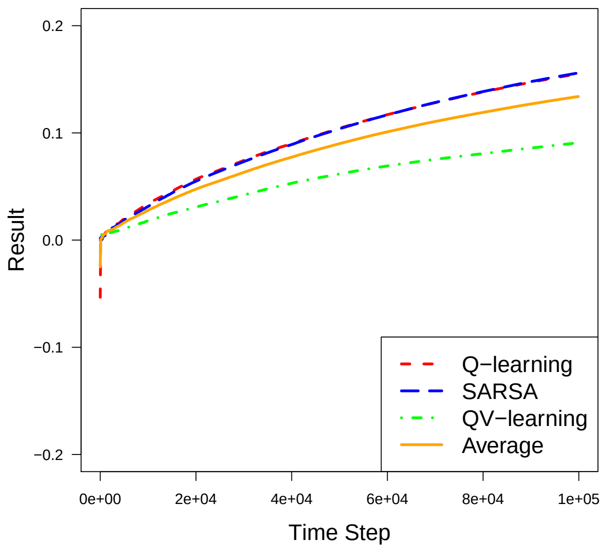


Figure 4.6: The three RL agents are evaluated at the same time competing for reward units. Agents' results (an average of rewards) after 100,000 time steps and 100 repetitions.

Apart from the comparative results, we see that performance depends on the other agents' policies, but especially on the ability of digesting the state-action table space, and how much noise (e.g. from Random) can be handled. Finally, it is interesting to mention that one of the properties of the Λ environment class that has been lost when other agents are introduced is the notion of balancedness, since a random agent typically scores worse than 0.

4.3.4 Evaluating Agents in a Cooperative Scenario

The next scenario we want to explore is when the four agents are prompted to cooperate. This is done by putting all the obtained reward units in the same bag, so the agents just see the reward as the average of their reward units.

Figure 4.7 changes from figure 4.5 very significantly. How can it be that moving from a competitive to a cooperative case we get worse results? The explanation is a little bit more convoluted. The problem of cooperation is the way we assign rewards. Since the reward they receive is the average of their reward units, it is much more difficult for them to determine the goodness of their actions, since their rewards are affected by other agents' movements. In other words, they lose 'individuality'.

This explanation is only part of the story if we compare figure 4.8 with figure 4.7 and figure 4.6. In this case, where Random has been removed, the results are slightly better than in the competitive case. However, this improvement is not uniform for the three RL agents. SARSA is clearly benefited in this situation, next comes Q-learning, while QV-learning is less able (or more altruistic) coping with the cooperation.

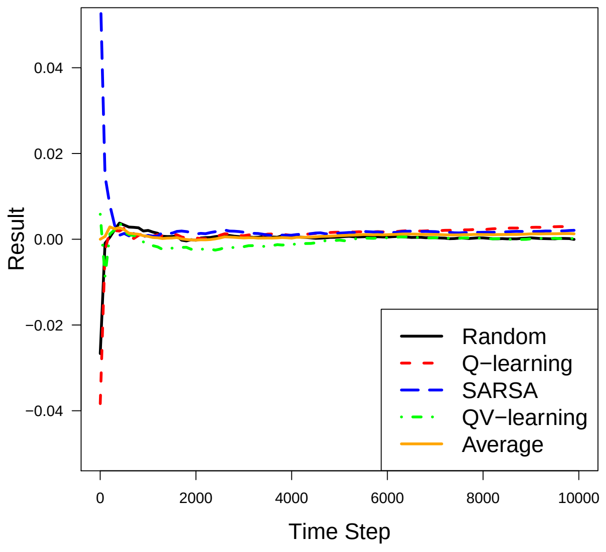


Figure 4.7: The three RL agents along with Random are evaluated at the same time cooperating for reward units (all the agents are in the same team). Agents' results (an average of rewards) after 10,000 time steps and 100 repetitions.

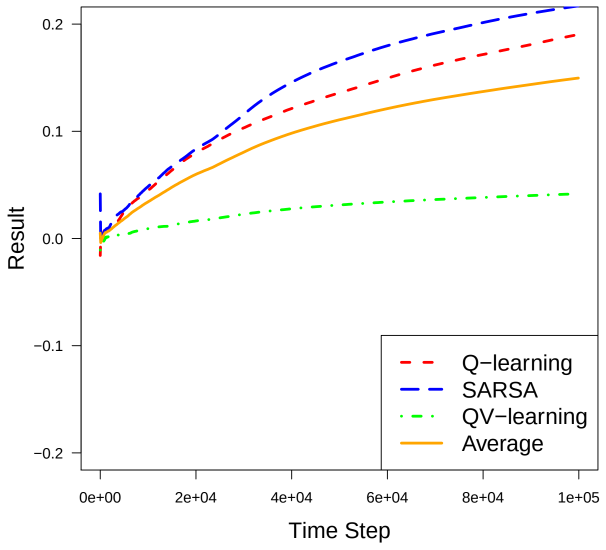


Figure 4.8: The three RL agents are evaluated at the same time cooperating for reward units (all the agents are in the same team). Agents' results (an average of rewards) after 100,000 time steps and 100 repetitions.

4.3.5 Scenario Measuring Both Competition and Cooperation

Finally, we examine another scenario where we now have competition and cooperation at the same time, using the notion of 'team'. We define two teams, one with two Q-learning agents and the other one with two SARSA agents. Inside each team, the obtained reward units go to the same bag, but different teams compete for the reward units. This is shown in figure 4.9. In general, the results are poorer than with three agents in the cooperative case (figure 4.8). This can be explained because here we have four agents instead of three, but also because having two teams is a more complex scenario than having just one.

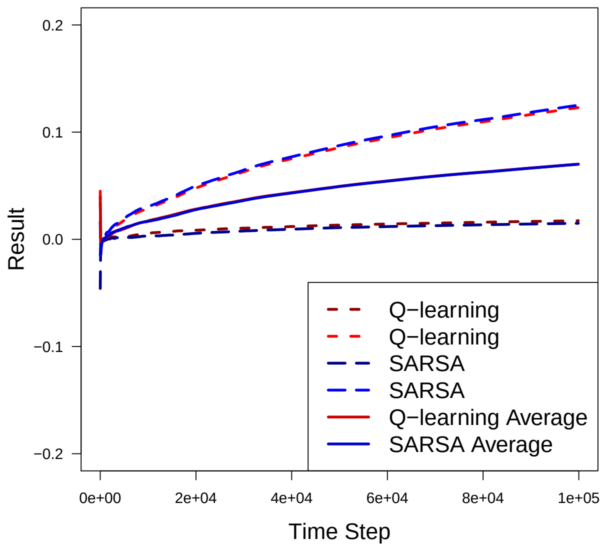


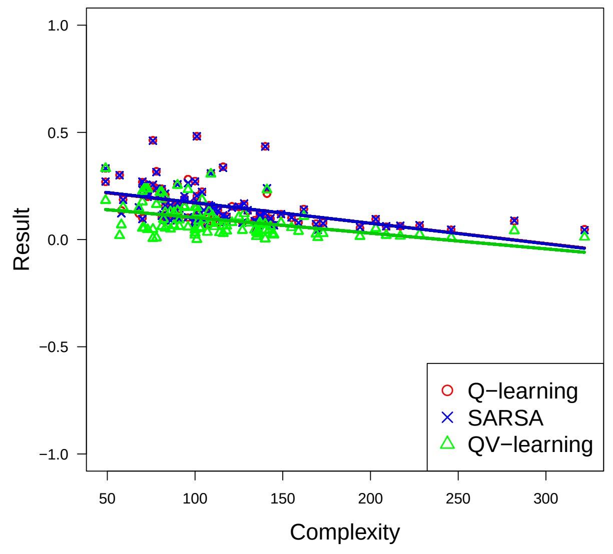
Figure 4.9: Two teams scenario. One team with two Q-learning agents against another team with two SARSA agents. Agents' results (an average of rewards) after 100,000 time steps and 100 repetitions.

The results show that there are no significant differences between both teams. However, there are important differences between the components of each team. This can be observed in figure 4.9, where we assign the best results in the team to the first entry and the worst results to the second entry. So, the plot just shows the difference in (average) performance between the best and the worst component in the team. We see that this difference is very significant. While usually an agent in the team performs around 0.1, the other agent stays at a very low result close to 0. It is not clear which role this second agent takes.

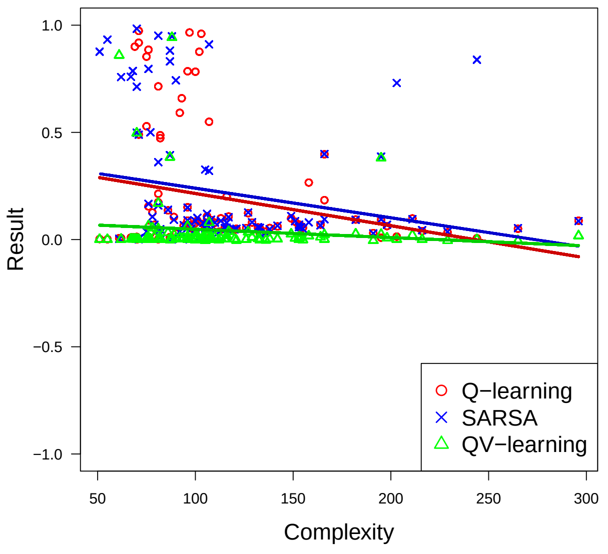
4.3.6 Discussion

We have analysed several multi-agent scenarios. A test which was originally designed to measure the intelligence of an agent in an environment without other agents is adapted to other scenarios where other agents are introduced in the environment, including cooperation and competition. As expected, working with many agents makes things much more complex. We see that performance can be seriously degraded by the inclusion of other agents with null intelligence, as a random agent. This is surprising if we look at this from the point of view of game theory (two-player games, in particular), but it is much more natural if we realise that it is more difficult to attain a goal if there is another agent bugging around (even randomly). This is extreme in the case of RL agents, because they cannot *learn* that random agents are just noise.

All this means that the difficulty of a task is no longer related to the complexity of the patterns in the environment in a tight way, as it was for the single-agent situation. We can see this by comparing the complexity of the environment (excluding the evaluated agents) and the results for the scenarios where only the three RL agents are used, i.e. figure 4.6 and figure 4.8. In order to approximate the environment complexity, we use the size of a compressed coding of the concatenation of the description of the space S and the description of the pattern for Good and Evil, denoted by P. More formally, we calculate an approximation to its (Kolmogorov) complexity, denoted by K^{approx} as $K^{approx} = LZ(S, P)$ where LZ is just the 'gzip' method given by the *memCompress* function in R, a GNU project implementation of Lempel-Ziv coding [145]. This comparison is shown in figure 4.10. We see that there is still a relation between the complexity of the environment and the result, while this relation is stronger for Q-learning and SARSA in the cooperative case. In fact, the results for Q-learning and SARSA are very good when the complexity is very low. This means that in very simple cases RL agents are able to perform well, even in social scenarios. This seems to suggest that the difficulty of a social environment is a cumulative issue, which adds the complexity of the environment and the complexity/performance/noise of other agents.



(a) Competitive scenario.



(b) Cooperative scenario.

Figure 4.10: Relation between environment complexity and agents' result for the scenarios where only the three RL agents are used. Linear regression is also shown for each agent.

In fact, it would be extremely informative (with regard to the creation of universal environments) to repeat the experiment performed with humans and RL agents in [62] by using one of these simple multi-agent environments. We guess that while humans would still be able to manage, the collapse that we observe in the RL agents would show that the mere introduction of some simple social behaviour may show the real differences between these two types of agents. We guess that humans will still be able to manage, mostly because they handle noise much better and the real differences with RL agents would show up.

However, before constructing a social intelligence test based on these ideas, we need to better understand some phenomena that take place when we include other agents and let them compete and cooperate. Naturally, many other experiments must be done before embarking on the challenge of a true (and feasible) social intelligence test. In this regard, the experiments shown in this section could be extended in many ways. For instance, for the RL agents, we only consider model-free techniques whose search space grows geometrically as more agents are there. It would be interesting to see the results for model-based techniques or RL algorithms using function approximations, as well as other RL algorithms which work better when the Markov property does not hold (which is the general case in multi-agent systems). Also, some other RL algorithms which are specialised for multi-agent settings, such as Frequently Adjusted Multi-agent Q-learning [64] might give different results. Other issues which could be reconsidered is the way we modify the reward system to make the test competitive or cooperative. One possibility would be to remove the agents Good and Evil and let all the agents (and perhaps other objects) be able to generate rewards for the other agents. This would make things more difficult, but it would allow for more elaborate scenarios for competition, cooperation and communication.

Summing up, in order to evaluate social intelligence, we have pushed forward the idea of 'multi-agent intelligence test', which is a situated intelligence test where there are other agents in the environments. This is a new notion, since the kind of intelligence tests we are used to are typically those where the evaluated agent has to solve some tasks or where it has to be interrogated by other agents (interviews, Turing test, etc.), but the other agents are not *inside* the test. The closest notion is an old companion of artificial intelligence, *games*, especially multiplayer games, but it has only been recently proposed as a testbed for measuring intelligence [44]. However, the role of the opponent and its intelligence has not been clarified to date, especially if we want a test to give an absolute result, not only comparing a pair of agents.

Chapter 5

Defining Social Intelligence Universally

Once analysed the implications about the evaluation of an agent interacting with other agents, we can make our first steps towards the creation of a social intelligence test. In particular, the social intelligence test we are looking for should have the following requirements:

- It evaluates social intelligence for all kinds of interactive systems or species.
- Its measurement can be directly obtained from the operational definition.
- The evaluee has to interact in the environments with other agents.
- The evaluee has to use its social abilities and their consequences have to be perceived.
- The agent's performance is affected when the agent has to compete and cooperate with other socially intelligent agents.
- The evaluated agent must use its social intelligence to understand and/or have influence over other agents' policies in such a way that this is useful to accomplish its goals.
- The role every agent takes in the environment is relative to the agent to evaluate.

And, obviously, other requirements that every test should have, among others:

- It must be a finite procedure that can be feasibly applicable to an agent.
- It should be applied in a reasonable period of time to be practical.
- The exercises included in the test have to discriminate, with respect to the overall result, between non-similar agents.

In order to formally measure an ability, it seems clear that the first thing we need is a precise and formal definition about what we want to measure. in such a case, a formal social intelligence definition.

One way of reaching a universal definition of social intelligence is to consider more specific definitions and generalising them for any kind of subject. Thorndike's definition of social intelligence refers to "men, women, boys and girls" [113]. The approach we propose would generalise Thorndike's view with the variety of species in animal cognition, but also including machines, robots and other artificial systems. This is in the spirit of universal psychometrics [46], where we must consider any kind of agent (natural or artificial). Any of these systems can, in principle, be evaluated and can also be subject of interaction with the evaluee.

This approach can take us to definitions such as "performance of an agent in a wide range of environments while interacting with other agents" [56], as carried out along the previous chapter. As a result, we see clearly that social intelligence is a *relative* property, where we need to specify these other agents (and the range of environments)[1].

With this approach, we distinguish those traits that have positive consequences on the performance (rewards) from those that are associated to social intelligence but do not necessarily lead to better performance (such as being generous, open, extroverted, etc.). In other words, we understand that an agent is socially intelligent if it has the ability to perform better in a social environment, but not if it is very *sociable* but showing very poor performance. In the end, we want an operational definition such that its measurement can be directly linked to it, and not derived by some other traits that are usually associated to social intelligence in humans and animals.

So we must focus our attention on the specification of the set of environments used for measuring and, most especially, on the characteristics of other agents. Nonetheless, it is important to determine the *role* these agents take in the environment relative to the agent to evaluate. For instance, the environment can be populated by very intelligent agents, but the possibilities of an evaluated agent to achieve its goals depends on whether these agents are allies or enemies, or more generally if they are cooperative or competitive. The key issue is to establish whether the other agents' goals and interests are compatible with one's goals. The concept is complex, as alliances can be created and broken even if no clear teams are established from the beginning (which is an interesting property of social intelligence). Nonetheless we have to consider the notion of *role* from the beginning and make it visible at the top, jointly with the kind of environment and the kind of agents.

These roles or alliances determine two major social behaviours: cooperation and competition. These are in fact linked to the issue that some agents share some goals while some other agents compete or are against other agents' goals. If we think of rewards (with some kind of utility function) as a general way of expressing goals, interests and even resources they share or compete for, we can distinguish two major kinds of social intelligence:

Definition 6. Competitive social intelligence is the capability to obtain the best performance in a multi-agent environment where other agents compete for the same rewards.

Definition 7. Cooperative social intelligence is the capability to obtain the best performance in a multi-agent environment where other agents share the same rewards.

Note that both definitions are not exclusive, as there are environments where both competitive and cooperative behaviours are possible. This is similar to the several degrees of general-sum games in game theory. Nonetheless, it would be very useful to have some way to analyse competition and cooperation separately (as two main facets of social intelligence). How

[1]Note that the evaluee should not know the environments and agents with which it will be evaluated in order to avoid overspecialisation in the evaluation.

clear-cut such separation can be done is an open question, as both abilities are occasionally correlated. For instance, the creation of alliances in a purely competitive scenario leads to temporary or permanent cooperation, where the other agents are seen in an instrumental way. In figure 5.1 we can see how these competitive and cooperative social intelligences are related to social (and general) intelligence.

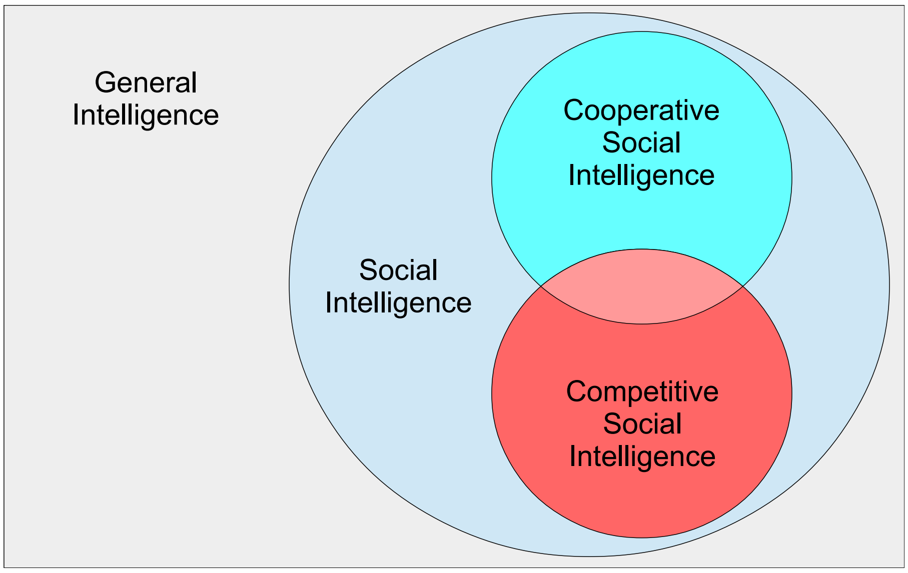

General Intelligence
Cooperative Social Intelligence
Social Intelligence
Competitive Social Intelligence

Figure 5.1: Venn diagram showing the relations between social intelligence types.

In what follows, we see how these informal definitions can be formalised and integrated.

5.1 Teams

We need to address a characterisation of agent slots, such that we can specify how agents participate in the environment. This actually means that we need to decide how the environment distributes rewards among the agents. An easy possibility will be to make each agent get its rewards without further constraints over other agents' rewards. With this configuration (e.g. general-sum games), both competition and cooperation may be completely useless for most environments, as the rewards are not limited or linked to the other agents. In contrast, if we set that the total set of rewards is limited in some way, we will foster competition, as happens in zero-sum games. But in any of these two cases it is not ensured that cooperation will occur. Alliances and coalitions [14, 36, 89, 15, 95] could arise sporadically between at least two agents in order to improve their rewards, probably by bothering or defending against a third agent. However, with low levels of social intelligence, alliances and coalitions seem unlikely to happen, since they would need some predisposed social abilities to maintain them in groups. For this reason, we need to find a way to make agents cooperate, or at least to make it more likely before any (sophisticated) alliance or coalition can emerge on its own. One possible answer to make the agents compete and/or cooperate is the use of teams, defined as follows:

Definition 8. Agent slots i and j are in the same team iff for every time step k any agents in i and j obtain the exact same reward regardless of which agents are present in the multi-agent environment.

which means that all agents in a team receive exactly the same rewards. This differs from alliances and coalitions, where the agents could receive different rewards. In fact, teams are fixed and cannot be changed by the agents. Also, we do not use the terms alliance or coalition as we do not use any sophisticated mechanism to award rewards, related to the contribution of each agent in the team, as it is done with the Shapley Value [99]. We just distribute rewards equitably. However, teams also allow the formation of alliances or coalitions between several (possibly individual) teams, or also between agents of different teams.

The inclusion of teams is a useful tool to evaluate the agents' abilities for cooperation and/or competition. In fact, in AI and game theory, we tend to characterise environments depending on their payoff structures as, for example, cooperative, (pure) competitive or conflicting interest games. This characterisation relies on the distribution of the rewards obtained by the agents, which typically depends on whether agents have similar, different or independent goals. The inclusion of teams allows us to explicitly arrange agents with similar goals in the same team, while arranging agents with different or independent goals in other teams.

5.2 Multi-Agent Environment Using Teams

At this moment, we are ready to define a multi-agent environment with parametrised agents by only specifying its agent slots and team arrangement.

Definition 9. A multi-agent environment μ accepting $N(\mu)$ agents (i.e. the number of agent slots in μ) is a tuple $\langle \mathcal{O}, \mathcal{A}, \mathcal{R}, \omega, \rho, \tau \rangle$, where $\mathcal{O}$, $\mathcal{A}$, $\mathcal{R}$ represent the observation sets, action sets and reward sets respectively (i.e. $\mathcal{O} = (\mathcal{O}_1, \dots, \mathcal{O}_{N(\mu)})$, $\mathcal{A} = (\mathcal{A}_1, \dots, \mathcal{A}_{N(\mu)})$ and $\mathcal{R} = (\mathcal{R}_1, \dots, \mathcal{R}_{N(\mu)})$) and ω and ρ are the observation function and reward function respectively. Finally, τ is a partition on the set of agent slots $\{1, \dots, N(\mu)\}$, where each set in τ represents a team.

Note that with this definition the agents are not included in the environment. For instance, noughts and crosses (tic-tac-toe) could be defined as an environment μ_{nc} with two agents, where $\tau = \{\{1\}, \{2\}\}$ defines the partition of agent slots into teams, which represents that this game allows two teams, and one agent in each. Another example is RoboCup Soccer [67], whose τ would be $\{\{1, 2, 3, 4, 5\}, \{6, 7, 8, 9, 10\}\}$, which represents that there are two teams, with agent slots $\{1, 2, 3, 4, 5\}$ in one team and agent slots $\{6, 7, 8, 9, 10\}$ in the other team.

Some environments typically do not take into account whether the agent slots are part of a team to provide them with rewards, having that $\mathcal{R}_i$ and $\mathcal{R}_j$ can be different even when i and j are in the same team. Following this idea, for any agent slot i and time step k, we denote $r_{i,k} \in \mathcal{R}_i$ as an *individual reward*. Here, in order to ensure the equality relation of rewards between agents in the same team (definition 8), we just provide for each agent slot i in that team and any time step k with what we denote as *team reward* $\bar{r}_{i,k}$ (this is the actual reward obtained by the agents), calculated as the mean of their members' individual rewards:

$$\forall k, t \in \tau : \bar{r}_{i,k} = \frac{1}{|t|} \sum_{i \in t} r_{i,k} \tag{5.1}$$

where k is the time step, τ is the partition of agent slots into teams and $|t|$ is the number of agent slots in t.

5.3 Agents' Setup

Once environments are defined, without including the agents, now we can define an *instantiation* of an environment with a particular *agent line-up*. Formally, a line-up l is a list of agents. For instance, if we have an agent set $\Pi = \{\pi_1, \pi_2, \pi_3, \pi_4\}$, a line-up from this set could be $l_1 = (\pi_2, \pi_3)$. The use of the same agent twice is allowed, so $l_2 = (\pi_1, \pi_1)$ is also a line-up. We denote by $\mu[l]$ the instantiation of an environment μ with a line-up l in such a way that the ith agent of l occupies the agent slot i of μ, provided that the length of l is greater than or equal to the number of agents allowed by μ (if l has more agents, the excess is ignored). For instance, for the noughts and crosses, an instantiation would be $\mu_{nc}[l_1]$. Note that different instantiations over the same environment would normally lead to different results.

When evaluating an agent in a multi-agent environment, it is usually desired not to evaluate it in all agent slots, but only in some of them (as, for example, when evaluating the goalkeeper in soccer), and use the rest of agent slots to populate the multi-agent environment with the agents we want it to interact with. Since line-ups could take into account undesired situations, where the agent to evaluate is located in an undesired agent slot, we make use of agent line-up patterns. An *agent line-up pattern* $\dot{l}$ is a list of agents where one or more elements are not instantiated. We can instantiate (or insert) an agent into a non instantiated position to create more specific line-up patterns. The instantiation of an agent π at position i on line-up pattern $\dot{l}$ of length n is denoted by $\dot{l} \overset{i}{\leftarrow} \pi$, which is exactly $\dot{l}_{1:(i-1)} \cdot \pi \cdot \dot{l}_{(i+1):n}$, where $l \cdot m$ denotes the concatenation of lists l and m and $l_{j:k}$ denotes the elements in l from position j to k. This notation can be extended to instantiate several agents simultaneously using $\dot{l} \overset{i,\ldots,j}{\longleftarrow} \pi_1, \ldots, \pi_n$ to represent $\dot{l} \overset{i}{\leftarrow} \pi_1 \cdots \overset{j}{\leftarrow} \pi_n$. Once a line-up pattern $\dot{l}$ has all its elements instantiated becomes a line-up l. For instance, a line-up pattern for the agent set Π could be $\dot{l}_3 = (\pi_3, *)$, where $*$ represents an element that is not instantiated, and $\dot{l}_3 \overset{2}{\leftarrow} \pi_4$ instantiates position 2 with agent π_4, converting the line-up pattern into the line-up $l_3 = (\pi_3, \pi_4)$. Note that environments can only be instantiated with line-ups, so first we need to instantiate all the elements from a line-up pattern to convert it to a line-up, and then use it to instantiate the multi-agent environment.

We use $\dot{L}^n_{-i,\ldots,j}(\Pi)$ to denote the set of all the line-up patterns of length n with agents of Π where positions $i, \ldots, j$ are not instantiated. For instance, $\dot{L}^n_{-i}(\Pi)$ defines the set of all possible line-up patterns $\{\dot{l}_{1:i-1} \cdot * \cdot \dot{l}_{i+1:n}\}$ using agents from Π.

We typically use a line-up pattern with positions $i, \ldots, j$ not being instantiated to evaluate agents in agent slots $i, \ldots, j$ of an environment, while the rest of positions in the line-up pattern contains the agents they have to interact with.

We use discrete weight functions of non-negative rational numbers[2] for the environments, their agent slots and the line-up patterns. These weights represent each element relevance within a set (or group) of elements. $w_M(\mu)$ denotes a weight for environment μ from a certain set M, $w_S(i, \mu)$ denotes a weight for agent slot i of a certain environment μ and $w_{\dot{L}}(\dot{l})$ denotes a weight for a line-up pattern $\dot{l}$ formed with agents from a certain set of agents Π, giving weights to the agents in the line-up pattern and their positions. Note that both w_M and $w_{\dot{L}}$ could be integrated into a single weight $w_{\dot{L},M}(\dot{l}, \mu)$ for *instantiated environments*. However, we want to decouple agents and environments and use both of them as independent parameters. To make the agents and the environment independent we work with the next assumption.

Assumption 1. If $w_{\dot{L},M}(\dot{l}, \mu)$, where $\exists i : \dot{l} \in \dot{L}^{N(\mu)}_{-i}(\Pi)$, is independent of μ then:

[2]We make use of rational numbers instead of real numbers to let the evaluation be computable.

$$\forall \dot{l}, \mu : w_{\dot{L},M}(\dot{l}, \mu) = w_{\dot{L}}(\dot{l})$$

which means that an agent line-up has the same weight (or importance) independently of the multi-agent environment.

Finally, we use $R_i^K(\mu[l])$ to denote the expected result that the ith agent in l obtains in μ (also in agent slot i) during K time steps. If K is omitted, we assume $K = \infty$. It might be difficult to obtain an agent's expected result when either μ and/or some of the agents in l are stochastic. In order to obtain the expected result of an agent in such situations, some methods (e.g. Monte Carlo at section 2.5) may be used to approximate this value.

5.4 A Formal Definition of Social Intelligence

Having these ideas in mind we can now attempt a first definition of social intelligence. We first fix the line-up and vary on the possible environments.

Definition 10. We define the social intelligence of an agent π interacting in agent line-up l which contains π at position i, over a set of multi-agent environments M accepting at least i agents and at most $|l|$ agents (being $|\cdot|$ the length of a list), weighting the multi-agent environments by w_M as:

$$\Upsilon_i(l, M, w_M) \triangleq \sum_{\mu \in M} w_M(\mu) R_i(\mu[l]) \tag{5.2}$$

where $|M| \geq 1$ and $\forall \mu \in M : N(\mu) \geq 2$.

When M contains two or more environments whose rewards have different domains, we could just normalise their rewards. Another option when evaluating various agents is to rank them for each environment according to their results, and use their ranks instead of their expected results, allowing us to compare the ranking performance of several agents.

Alternatively, we can think about a definition of the social intelligence of an agent for a varying set of line-up patterns while fixing the environment.

Definition 11. We define the social intelligence of an agent π in agent slot i, interacting in a multi-agent environment μ accepting at least i agents, with a set of agent line-up patterns defined over agent set Π and $w_{\dot{L}}$ as a weight for agent line-up patterns:

$$\Upsilon_i(\pi, \Pi, w_{\dot{L}}, \mu) \triangleq \sum_{\dot{l} \in \dot{L}_{-i}^{N(\mu)}(\Pi)} w_{\dot{L}}(\dot{l}) R_i(\mu[\dot{l} \overset{i}{\leftarrow} \pi]) \tag{5.3}$$

where $N(\mu) \geq 2$ and $|\Pi| \geq 1$.

Note that now $\dot{l}$ has always $N(\mu)$ elements when instantiated, so now no upper restriction exists over the number of agent slots of μ.

We can integrate both equations 5.2 and 5.3, also summing the performance over all possible agent slots of the environments of M, weighting the agent slots of each environment by w_S as follows:

Definition 12. The social intelligence of π interacting with the class of agents Π with a weight for agent line-up patterns $w_{\dot{L}}$, in a set of multi-agent environments M with a weight for multi-agent environments w_M and a weight for agent slots w_S is defined as:

$$\Upsilon(\pi, \Pi, w_{\dot{L}}, M, w_M, w_S) \triangleq \sum_{\mu \in M} w_M(\mu) \sum_{i=1}^{N(\mu)} w_S(i, \mu) \sum_{\dot{l} \in \dot{L}_{-i}^{N(\mu)}(\Pi)} w_{\dot{L}}(\dot{l}) R_i(\mu[\dot{l} \overset{i}{\leftarrow} \pi]) \tag{5.4}$$

where $|M| \geq 1, \forall \mu \in M : N(\mu) \geq 2$ and $|\Pi| \geq 1$.

This equation now relaxes the lower restriction of the number of agent slots on the environments to be at least 2.

The interpretation of the above definition is the expected performance of agent π interacting with all possible instantiated environments generated using the set of agents Π and set of environments M, with π interacting in all possible agent slots in each environment. When we are interested in evaluating the social intelligence in the broadest way, we need to instantiate the set of environments M and agents Π with classes of environments and agents respectively that cover this ability appropriately, or use two infinite sets with all possible social environments and agents[3]. To measure this ability for a particular situation, M and Π can be instantiated with the variables of interest, such as suggested for M in [112] for the evaluation of intelligence.

When every environment in M gives several rewards, each of which indicating the agent's performance to achieve a particular goal, definition 12 can be easily extended by giving a weight to each goal.

But it is not necessary to build specific line-up patterns for each environment. Instead, environments and line-up patterns can be independently provided beforehand, without need of being interrelated. For this purpose, the positions of the agents in the line-up patterns can be assumed independent:

Assumption 2. If $w_\Pi(\pi, i)$ defines the weight for the agent π appearing at position i in a line-up pattern, we assume that the agent has the same weight independently of its position. Formally:

$$\forall \pi, i : w_\Pi(\pi, i) = w_\Pi(\pi)$$

Under assumption 2, $w_{\dot{L}}$ can be derived as a function of terms from w_Π. Finally, we assume that the line-up pattern weight only depends on its agents weights (independently of their position).

Assumption 3. We calculate $w_{\dot{L}}$ as a product of agent weights w_Π as:

$$\forall i, n, \Pi, \dot{l} \in \dot{L}_{-i}^{n}(\Pi) : w_{\dot{L}}(\dot{l}) = \prod_{1 \leq k < i} w_\Pi(\dot{l}_{k:k}) \prod_{i < k \leq n} w_\Pi(\dot{l}_{k:k}) \tag{5.5}$$

[3]When using an infinite set with all possible social environments, the definition can be simplified. For every environment μ_1 and agent slot i in it, there is always an environment μ_2 with exactly the same behaviour where agent slot i becomes 1. That means that, after properly adjusting environment weights w_M, we could easily get rid of the summation over agent slots and work just with agent slot 1 for agent π. In other words, this would be like considering that the evaluated agent always interacts in agent slot 1. For practical reasons, when using a finite set of environments we included agents slots in order to avoid the tedious work of defining several environments for the same domain.

Now we can obtain the expected performance of agent π interacting with all possible line-up patterns generated using the set of agents Π, and in a set of environments M with π interacting in all possible agent slots in each environment.

Proposition 1. Under assumption 3, social intelligence as per equation 5.4 is also defined as:

$$\Upsilon(\pi,\Pi,w_{\dot{L}},M,w_M,w_S) =$$
$$=\sum_{j=2}^{\infty}\sum_{i=1}^{j}\sum_{\dot{l}\in\dot{L}_{-i}^{j}(\Pi)}\left(\prod_{1\le k<i} w_\Pi(\dot{l}_{k:k})\prod_{i<k\le j} w_\Pi(\dot{l}_{k:k})\right)\sum_{\mu\in M^j} w_M(\mu)w_S(i,\mu)R_i(\mu[\dot{l}\overset{i}{\leftarrow}\pi])$$

where M^j denotes all the environments in M with j agent slots, $|M|\ge 1$ and $|\Pi|\ge 1$.

Proof. Definition 12 ranges over environments, their agent slots and then over line-up patterns, but we could express an equivalent equation by ranging over line-up patterns first and environments and their agent slots next:

$$\Upsilon(\pi,\Pi,w_{\dot{L}},M,w_M,w_S) = \sum_{\mu\in M} w_M(\mu)\sum_{i=1}^{N(\mu)} w_S(i,\mu)\sum_{\dot{l}\in\dot{L}_{-i}^{N(\mu)}(\Pi)} w_{\dot{L}}(\dot{l})R_i(\mu[\dot{l}\overset{i}{\leftarrow}\pi]) =$$

$$\overset{ass.3}{=}\sum_{\mu\in M} w_M(\mu)\sum_{i=1}^{N(\mu)} w_S(i,\mu)\sum_{\dot{l}\in\dot{L}_{-i}^{N(\mu)}(\Pi)}\left(\prod_{1\le k<i} w_\Pi(\dot{l}_{k:k})\prod_{i<k\le N(\mu)} w_\Pi(\dot{l}_{k:k})\right)R_i(\mu[\dot{l}\overset{i}{\leftarrow}\pi]) =$$

$$=\sum_{j=2}^{\infty}\sum_{\mu\in M^j} w_M(\mu)\sum_{i=1}^{j} w_S(i,\mu)\sum_{\dot{l}\in\dot{L}_{-i}^{j}(\Pi)}\left(\prod_{1\le k<i} w_\Pi(\dot{l}_{k:k})\prod_{i<k\le j} w_\Pi(\dot{l}_{k:k})\right)R_i(\mu[\dot{l}\overset{i}{\leftarrow}\pi]) =$$

$$=\sum_{j=2}^{\infty}\sum_{i=1}^{j}\sum_{\dot{l}\in\dot{L}_{-i}^{j}(\Pi)}\left(\prod_{1\le k<i} w_\Pi(\dot{l}_{k:k})\prod_{i<k\le j} w_\Pi(\dot{l}_{k:k})\right)\sum_{\mu\in M^j} w_M(\mu)w_S(i,\mu)R_i(\mu[\dot{l}\overset{i}{\leftarrow}\pi])$$

□

This shows how we can parametrise the definition in terms of the weight of the other participants (w_Π) independently of their order in line-up patterns. For instance, the weight for each agent could depend on its (social) intelligence, provided we are able to estimate this value. The use of a product of weights makes sense if w_Π is a unit measure.

Proposition 1 is not only useful for parametrising the definition in terms of the agents in isolation, but also because it decouples agents from environments. This makes sense in the context of social intelligence evaluation, as we want to consider other agents that are able to work in different environments, and not very specific agents that only work in one environment.

Definition 12 and its reformulation by proposition 1 integrates all kinds of social behaviour, as it does not distinguish between agents appearing in the same team or opponent teams. For instance, if we consider a set Π with very intelligent agents, some environments (and line-up patterns) will be easier if many of these agents appear in the same team, but will be harder if they appear in opponent teams. Also, the aggregation may consider many other environments where no social behaviour takes place, or even non-social agents. This means that the above equations are a skeleton for the definition, but we still need to better analyse the pair (Π, $w_{\dot{L}}$) or (Π, w_Π) and the trio (M, w_M, w_S).

5.5 Social Intelligence Test

A definition is not a test, most especially because many definitions range over infinite sets or an infinite number of time steps. A test must be a finite procedure that can be feasibly applicable to an agent. For the moment, we focus on non-adaptive tests, which are based on performing just a finite number of finite experiments or exercises (episodes), which are independent of the previous ones.

Consequently, a test to evaluate social intelligence is defined using the previous definition of $\Upsilon(\pi, \Pi, w_{\dot{L}}, M, w_M, w_S)$, where Π is sampled with some distribution, M is sampled with some distribution and the number of time steps for each experiment is limited in some way. Sampling is understood to be without replacement when there is determinism (it does not make sense to repeat the same exercise if the result is already known) but is understood to be with replacement for non-deterministic agents or environments. We denote by $S \sim^n [A]_p$ a sample S of n elements from set A using probability distribution p for the powerset of A, i.e. for 2^A, only giving a probability > 0 to subsets with n elements. The use of a distribution over samples instead of a distribution over exercises gives more flexibility about the conditions that we could establish on the sampling procedure. For instance, we could define a sample probability such that high diversity is ensured (apart from high accumulated relevance of the exercises that are chosen) or such that a range of difficulties is covered. Keep in mind that with this definition, the issues of with replacement or without replacement are re-understood as whether these samples allow repeated values or not. With this notation, we can give the following definition of test:

Definition 13. A *test* over Υ (definition 12 in section 5.4), denoted by $\hat{\Upsilon}[p_\Pi, p_M, p_S, p_K, n_E]$, is a sample of n_E exercises from all those summed in the definition, using agent distribution p_Π, a multi-agent environment distribution p_M, an agent slot distribution p_S, and a distribution on the number of time steps p_K.

$$\hat{\Upsilon}[p_\Pi, p_M, p_S, p_K, n_E](\pi, \Pi, w_{\dot{L}}, M, w_M, w_S) \triangleq \eta_{\mathcal{E}} \sum_{\langle \mu, i, \dot{l} \rangle \in \mathcal{E}} w_M(\mu) w_S(i, \mu) w_{\dot{L}}(\dot{l}) R_i^K(\mu[\dot{l} \xleftarrow{i} \pi])$$

where $\eta_{\mathcal{E}}$ normalises the formula with $\eta_{\mathcal{E}} = \frac{1}{\sum_{\langle \mu, i, \dot{l} \rangle \in \mathcal{E}} w_M(\mu) w_S(i,\mu) w_{\dot{L}}(\dot{l})}$, K is chosen using probability distribution p_K and the exercises $\mathcal{E}$ are sampled as:

$$\mathcal{E} \sim^{n_E} \left[\bigcup_{\mu \in M} \bigcup_{i=1}^{N(\mu)} \left\{ \left\langle \mu, i, \dot{l} \right\rangle : \dot{l} \in \dot{L}_{-i}^{N(\mu)}(\Pi) \right\} \right]_{p_{\mathcal{E}}}$$

with $p_{\mathcal{E}}$ being a distribution on the set of triplets $\left\langle \mu, i, \dot{l} \right\rangle$ based on p_M, p_S and p_Π.

Note that we use p_Π for the line-up patterns, which could be the line-up pattern probability derived as the product of the probabilities of each agent in the line-up pattern following assumption 2, as in equation 5.5.

It is important not to confuse the probabilities of sampling the line-up patterns, environments, agent slots and number of time steps (p_Π, p_M, p_S, p_K) with any weight defined on them, in particular, the weights $w_{\dot{L}}$, w_M and w_S defined on line-up patterns, environments and agent slots respectively. While weights represent the relevance of an environment, its agent slots and line-up pattern for the definition (so it determines the abilities, roles and agents that have higher weight in the formula), the distributions are just a way of sampling the usually large or infinite set of environments, agent slots and agents. Weights and distributions might be related (or may be equal in order to ensure fast convergence to the actual value), but some other considerations may suggest that a less relevant case (low weight) can be sampled with high probability, as it may be highly representative or more robust, for instance. Actually, we want that a diversity of cases is sampled, rather than similar cases that will provide redundant information. This is why we use a distribution on 2^A and not on A because otherwise we would not be able to measure how good (e.g. informative) a set of exercises is.

Chapter 6

Experimental Analysis for Several Types of Environments and Agents

Social abilities may be more or less important depending on the situation the agent is facing. It seems that the performance of cooperative agents should improve as the environment focuses more on cooperation, as well as competitive agents in more competitive scenarios. In this chapter we do some experiments to see whether, by the use of different partitions of agent slots used to group agents together, the environment becomes a more cooperative (respectively competitive) scenario and, therefore, cooperative (respectively competitive) agents improve their results. We also analyse not only the impact of a particular environment, but also whether the agents they have to face have relevance on their performance. In order to do so, we analyse the results obtained by several agents with different social aptitudes (more competitive or cooperative behaviours), while interacting in some environments with several partition of agent slots and with different line-ups.

6.1 Experiment Configuration

We wished to use some agents specially designed for *general* cooperation and competition, but since we could not find such agents (at least in a way they could be applied to several environments without further assumptions or constraints), we had to use a general agent and adapt it to be more socially interactive. We also wanted simple agents such that the analysis of results were easier. For this experiments, we make use of some Q-learning agents (section 2.2.1). In order to make them more or less cooperative/competitive oriented, we feed their state-action Q values (i.e. r_k in equation 2.1) with the result of a function based on their individual rewards or team rewards (section 5.2). We call this their "motivation".[1] Here we show the agents used for the experiments:[2]

[1]Note that the Q-learning agents we obtain are completely equal except for their motivations.

[2]Although technically agents do not have access to individual rewards, we assume they do since we are not interested in providing realistic agents but only in evaluating agents.

- Random: An agent whose actions are selected using a uniform distribution (section 2.4.1).
- QLSelfish: A Q-learning agent whose motivation is to increase its individual reward. The original Q-learning behaviour without using teams.
- QLCommunal: A Q-learning agent whose motivation is to increase its team reward.
- QLMerciful: A Q-learning agent whose motivation is to increase the minimum individual reward in its team.
- QLHarmful: A Q-learning agent whose motivation is to increase the difference between its individual reward and the sum of the individual rewards of agents in other teams.

For this selection of agents we hypothesise that agents encouraging its team rewards are co-operative or team-oriented, while agents only concerned about their own individual rewards or about bothering other agents are competitive or self-oriented.

Now we show the pseudocode for the algorithm we use for the experiments (see algorithm 1). Input variables are accessible in all functions. The QLearning agent has the typical Q-learning behaviour. lr, df, rr, iv refer to the *learning rate* α, *discount factor* γ, *random rate* β and *initial value* Q_0 parameters for a Q-learning agent. We use Π_e and Π_o to respectively represent the set of agents to evaluate and the set of agents to populate the environment. The algorithm makes use of six functions: *flatten*, *size* and *initialise* have their standard interpretation, while *expectedResult*, *createLineups* and *train* are defined below:

- *expectedResult*(μ, i, l, K, β) refers to the R function defined in section 5.3 (page 44) which calculates the expected result of agent in the ith position of line-up l interacting in environment μ (also in agent slot i) during K time steps. We use an average of rewards as the utility function to calculate an agent's result. The expected result of an agent is approximated with a Monte Carlo method (section 2.5), approximating this value by sampling β root-to-leaf paths (i.e. $\alpha = 0$).
- *createLineups*(Π, S, π, i) creates the set of all line-ups with *size(S)* number of agents (being S a set of agent slots), formed with agents from Π but ensuring that π is located in agent slot i.
- *train*(μ, l, n_E) creates a line-up with the agents of l in the same order after training them interacting in environment μ during n_E episodes (or exercises).

Following the definitions in chapter 5, we calculate the social intelligence of an agent in an environment as its expected result[3] interacting in all possible line-ups and agent slots using uniform weights for $w_{\hat{L}}$ and w_S. Before evaluating an agent in a line-up, all the agents in this line-up are trained simultaneously (similarly as it is done in co-evolution [16]).

[3]The evaluation is actually calculated as the average of 200 repetitions to approximate the expected average rewards of the agents.

Algorithm 1 Experiment.

```
1: Input:
2: $S_e$                    ▷ Agent slots where the agents are evaluated
3: $T$                      ▷ Partitions of agent slots where the agents are evaluated
4: $\mu$                    ▷ Environment where the agents are evaluated
5: $n_E$                    ▷ Number of episodes that the agents are trained
6: $K$                      ▷ Number of time steps that the agents interact with the environment
7: $\beta$                  ▷ Number of samples to approximate an agent's expected result
8: Output:
9: results

1: Begin
2:     $\Pi_{QL}$ = {QLSelfish, QLCommunal, QLMerciful, QLHarmful}
3:     initialiseQL($\Pi_{QL}$)
4:     $\Pi = \Pi_{QL} \cup$ {Random}
5:     return $\{(\tau$, evaluateAgents$(\Pi, \Pi, \tau)) \mid \tau \in T\}$
6: End

1: procedure INITIALISEQL($\Pi_{QL}$)
2:     Params = {(lr, df, rr, iv) | lr $\in$ [0.05, 0.15, ..., 0.95] $\wedge$ df $\in$ [0, 0.1, ..., 0.9] $\wedge$ rr = 0.05 $\wedge$ iv = 10}
3:     bestParam = $\arg\max_{param \in Params}$ evaluateQLParam(param)
4:     for all $\pi_{QL} \in \Pi_{QL}$ do
5:         initialise($\pi_{QL}$, bestParam)        ▷ Sets the parameters to a Q-learning agent
6:     end for
7: end procedure

1: function EVALUATEQLPARAM(parameters)
2:     initialise(QLearning, parameters)        ▷ Sets the parameters to QLearning
3:     return $\sum$ { result | (QLearning, result) $\in$ evaluateAgents({QLearning}, {Random}, $\tau$) $\wedge$ $\tau \in T$ }
4: end function

1: function EVALUATEAGENTS($\Pi_e$, $\Pi_o$, $\tau$)
2:     agentsResult = $\emptyset$
3:     $S$ = flatten($\tau$)        ▷ Flattens the structure into a set
4:     initialise($\mu, \tau$)        ▷ Restructures the partition of agent slots of the environment
5:     for all $\pi_e \in \Pi_e$ do
6:         agentResult = 0
7:         for all $i \in S_e$ do
8:             agentSlotResult = 0
9:             $L$ = createLineups($\Pi_o, S, \pi_e, i$)
10:            for all $l \in L$ do
11:                $\hat{l}$ = train($\mu, l, n_E$)
12:                agentSlotResult += expectedResult($\mu, i, \hat{l}, K, \beta$)
13:            end for
14:            agentResult += agentSlotResult / size($L$)
15:        end for
16:        agentsResults = agentsResults $\cup$ ($\pi_e$, agentResult / size($S_e$))
17:    end for
18:    return agentsResults;
19: end function
```

6.2 Prisoner's Dilemma (3-Players Version)

For our first experiment, we use the prisoner's dilemma environment (section 2.1.2) which is a simple and well-known game involving competition and cooperation. In order to better analyse the importance of the partition of agent slots, we use a 3-players version[4], where more complex teams may exist. Next we explain the difference with the original version.

In this game, one or more prisoners (a team) are locked in the same jail cell, while the rest of the prisoners are locked in other jail cell(s) in the same way. The prisoners that share a jail cell have the same team reward (i.e. time avoided in prison), which is calculated as the mean of all individual rewards of prisoners in that team. Each player can see the actions performed by the other players. For the agent in agent slot i, the environment provides an observation set $\mathcal{O}_i = \mathcal{A}_j \times \mathcal{A}_k \cup \{\lambda\}$ such that $i \neq j \neq k \wedge j < k$, and for each agent the observation function ω returns the actions performed by the other agents in the previous time step or λ if it is the first time step. In this game, three partitions of agent slots may exist: $\tau_1 = \{\{1,2,3\}\}$, $\tau_2 = \{\{1,2\},\{3\}\}$ and $\tau_3 = \{\{1\},\{2\},\{3\}\}$. For this experiment we evaluate and analyse the agents interacting in all of them and in all agent slots. Table 6.1 shows the individual reward matrix[5] for the agent in agent slot i (i.e. $\mathcal{R}_i$), which has the actions of the agents in the three agent slots as input and its individual reward as output.

	Silent	Betray
Silent	7	3
Betray	3	0

(a) Agent remains silent.

	Silent	Betray
Silent	9	5
Betray	5	1

(b) Agent betrays.

Table 6.1: Prisoner's dilemma's (a 3-players version) individual payoff matrix. Rows and columns represent the actions of the agents in the other agent slots.

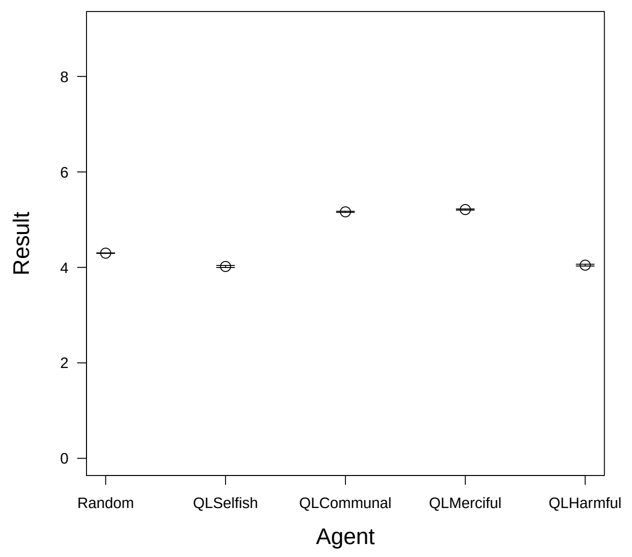
For this environment, according to algorithm 1, the parameters obtained for the Q-learning agents are *learning rate* = 0.05 and *discount factor* = 0.6. Next we show the results obtained by the agents for each partition of agent slots after first training them and then evaluating them during 500 time steps with 100 Monte Carlo samples to approximate their expected results (an average of rewards), and averaging over 200 repetitions.

In figure 6.1 (a) we see the results obtained for the 1-team partition of agent slots {{1, 2, 3}}. As we notice from the figure, having a cooperative behaviour, as QLCommunal and QLMerciful do, provides far better results than non-cooperative behaviours. Also, competitive oriented agents (QLSelfish and QLHarmful) obtain bad results, even worse than Random. It could be argued that these results only represent the learning speed as if the agent were given more time to learn they could reach similar values. In figure 6.1 (b) we see the results that the agents obtain during the training phase, which show that this is not the case.

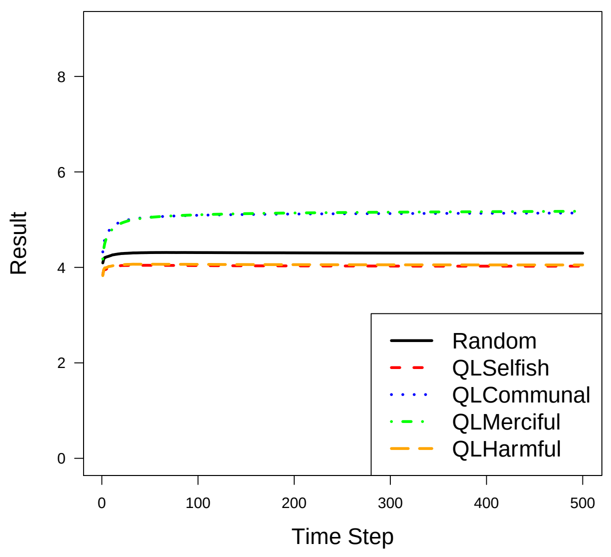
But regardless of the partition of agent slots, this environment is typically encouraging cooperation over competition, so it could also be argued that the results are independent of the

[4]This 3-players version of the prisoner's dilemma has been freely adapted from web page `http://www.classes.cs.uchicago.edu/archive/1998/fall/CS105/Project/node6.html`

[5]The numbers in the matrix have been chosen to meet the following rules: "1) Betray should be the dominant choice for each agent. 2) An agent should always be better off if more of the other agents choose to remain silent. And, 3) if one agent's choice is fixed, the other two agents should be left in a two-player prisoner's dilemma."



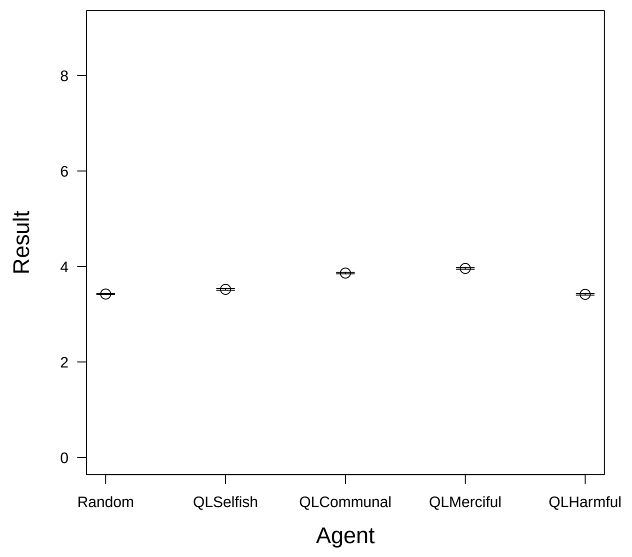
(a) Mean with standard error of the mean.



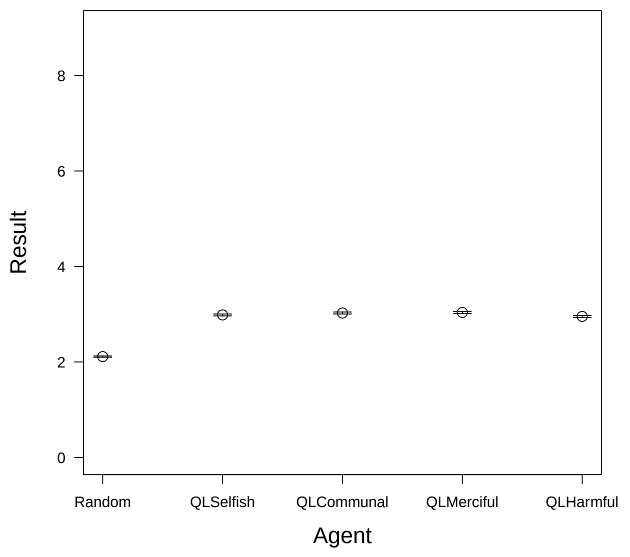
(b) Training phase during 500 time steps.

Figure 6.1: Agents' results for the partition of agent slots into teams {{1, 2, 3}} with 100 Monte Carlo samples to approximate their expected results (an average of rewards) and after averaging over 200 repetitions.

partition of agent slots. In figure 6.2 (a) we see the results for another partition of agent slots {{1, 2}, {3}}.



(a) Partition of agent slots into teams {{1, 2}, {3}}.



(b) Partition of agent slots into teams {{1}, {2}, {3}}.

Figure 6.2: Mean with standard error of the mean of agents' results with 100 Monte Carlo samples to approximate their expected results (an average of rewards) and after averaging over 200 repetitions.

After modifying the partition of agent slots, the picture changes slightly. Cooperative agents still maintain better results than competitive agents, but the difference is smaller.

Finally, in figure 6.2 (b) we provide a 3-team partition of agent slots $\{\{1\}, \{2\}, \{3\}\}$. Here we see how this modification has a clear impact on agents' results. With this partition of agent slots, after making the agents to have totally opposed goals, cooperative and competitive agents' results become quite similar. In fact, QLSelfish, QLCommunal and QLMerciful must reach the same result since their behaviour in one team with only one agent is the same. Meanwhile, although QLHarmful is a competitive agent, it still has some difficulties to reduce other agents' rewards while increasing its owns. At least, with this 3-team partition of agent slots the difference in performance between cooperative and competitive behaviours is drastically reduced.

With this environment, we manage to show that it is possible to balance an environment to a more cooperative/competitive scenario by placing agents in the same or different teams. However, only one environment seems not enough to reach a conclusion. Next we show some other environments.

6.3 Lambda Environment

The next environment (actually a prototype) we use for this experiment is the lambda environment (section 4.2.1). Since this environment was initially designed for one agent, here we extend it to include any number of agents, similarly as done in section 4.3.1 but with small differences.

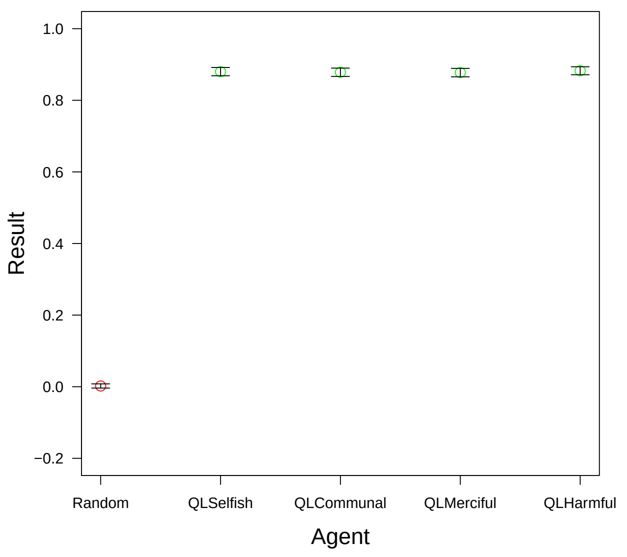
For this experiment, we instantiate environments with 6 cells and 3 actions, reward units are divided by 2 after each time step, and the pattern of actions of Good and Evil is randomly generated to have (on average) 6 actions. The environment is modified in such a way that teams can exist. This modification is done straightforwardly by adding more agents to the environment and placing them in teams. We let the environment have either 1, 2 or 3 agents, which leads us to six different partitions of agent slots: $\tau_1 = \{\{1\}\}$, $\tau_2 = \{\{1,2\}\}$, $\tau_3 = \{\{1\},\{2\}\}$, $\tau_4 = \{\{1,2,3\}\}$, $\tau_5 = \{\{1,2\},\{3\}\}$ and $\tau_6 = \{\{1\},\{2\},\{3\}\}$. Again, we evaluate and analyse the agents interacting in all of them and in all agent slots.

In this environment, if we let the agents interact an infinite number of time steps, a Q-learning agent usually obtains its maximum possible result. For this reason, we analyse in which cases the Q-learning agents will reach their maximum possible result in the limit. Since in most cases their final performance will be the same, here instead we are more concerned about analysing their learning speed while dealing with other agents.

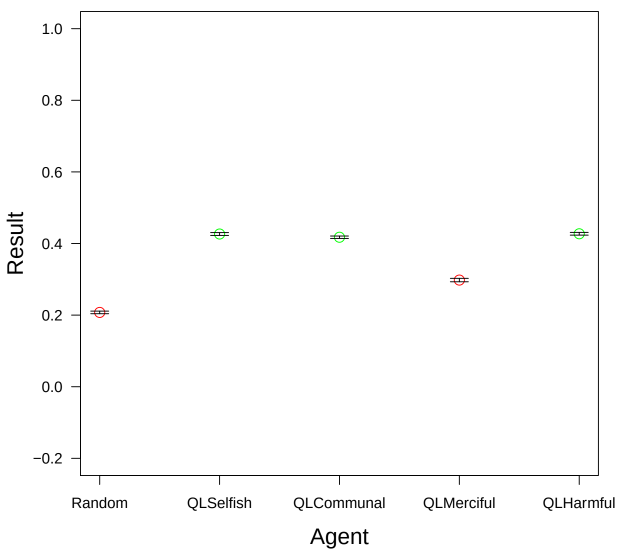
For this environment, according to algorithm 1, the parameters obtained for the Q-learning agents are *learning rate* $= 0.95$ and *discount factor* $= 0.3$. Next we show the results of each agent obtained for each partition of agent slots after first training them and then evaluating them during 10,000 time steps with 5 Monte Carlo samples to approximate their expected results (an average of rewards), and averaging over 200 repetitions.

In figure 6.3 (a) we see the results obtained for the partition of agent slots $\{\{1\}\}$. This configuration is the same as the original lambda environment. As there is only one agent interacting, all Q-learning agents' behaviours are similar, so they have similar results. As expected, Random obtains a result near to 0.

Next we include some cooperation into the environment by putting two agents in the same team. In figure 6.3 (b) we see the results obtained for the partition of agent slots $\{\{1, 2\}\}$. Here, QLMerciful has a lower result than the other Q-learning agents. This is because when it



(a) Partition of agent slots into teams {{1}}.



(b) Partition of agent slots into teams {{1, 2}}.

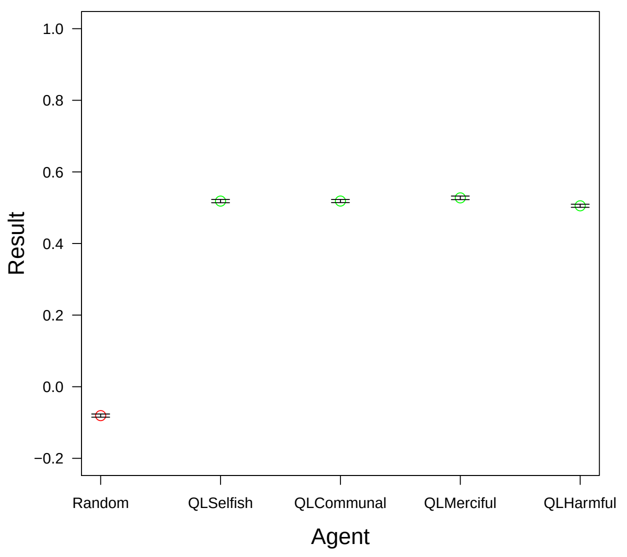
Figure 6.3: Mean with standard error of the mean of agents' results with 5 Monte Carlo samples to approximate their expected results (an average of rewards) and after averaging over 200 repetitions. Green and red dots respectively mean the agents reach or not their maximum possible result after an infinite number of time steps.

has to interact with Random, it is trying to help it to obtain good rewards, but clearly Random is not going to obtain good rewards by itself. In fact, Random obtains good rewards on this configuration, but this is because the other agents share their reward units with it, so despite it is no contributing to obtain good reward units, it is profiting from other agents' reward units.

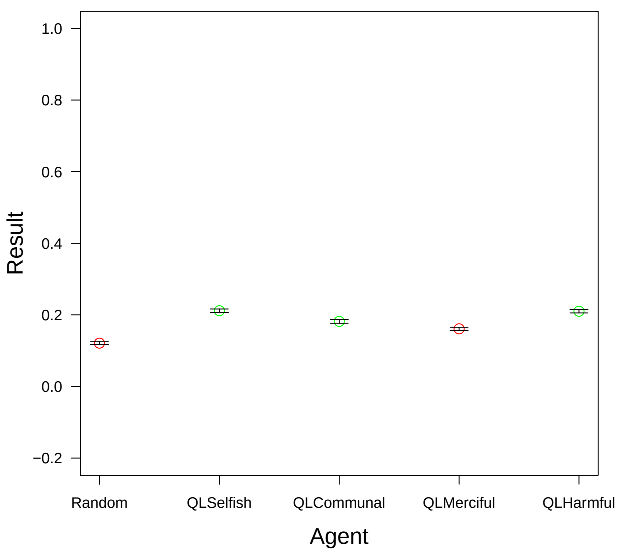
In figure 6.4 (a) we see the results obtained for the partition of agent slots {{1}, {2}}, where agents are placed in different teams, having opposed interests. Here, agents have to compete for reward units and competition is encouraged. As a result, Q-learning agents have good results, since they only have to take care of their own. Also, QLHarmful obtains a slightly worse result than the other Q-learning agents, since it also wants to decrease others rewards, this makes it to sometimes lose good rewards.

Next we increase the number of agents to see more diversity in the teams. In figure 6.4 (b) we see the results obtained for a partition of agent slots with three agents {{1, 2, 3}}, where all of them are arranged in the same team. Once we move to a configuration with three agents, Q-learning agents need much more time to reach their best results. This configuration has a lot of agents interacting together, so Q-learning agents need much more time to distinguish which part of the rewards comes from their own actions and which from other agents' actions. We also see that QLSelfish and QLHarmful learn faster than the other agents. This is because they behave to obtain good rewards by themselves and they do not take into account other agents' rewards.

In figure 6.5 (a) we see the results obtained for the next partition of agent slots {{1, 2}, {3}}, where we have two teams with one agent in one team and two agents in the other, obtaining both cooperation and competition, which makes it a more interesting configuration to analyse. First, Random is evaluated in more cooperative scenarios than competitive ones, so it is taking more advantage of the good reward units shared by the other agents, than reward units lost



(a) Partition of agent slots into teams {{1}, {2}}.



(b) Partition of agent slots into teams {{1, 2, 3}}.

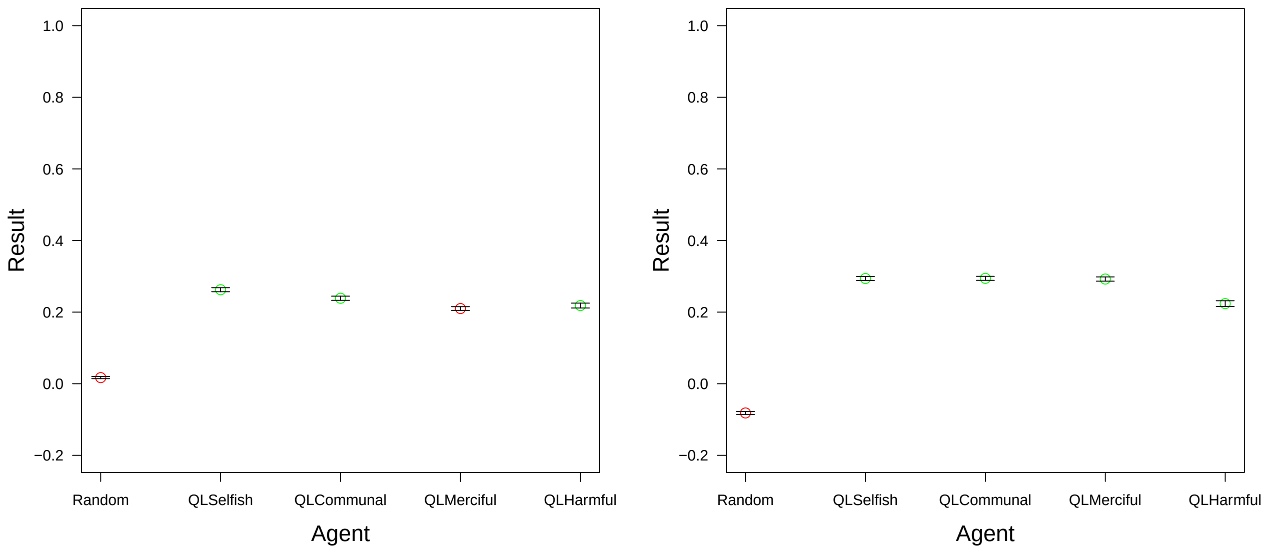
Figure 6.4: Mean with standard error of the mean of agents' results with 5 Monte Carlo samples to approximate their expected results (an average of rewards) and after averaging over 200 repetitions. Green and red dots respectively mean the agents reach or not their maximum possible result after an infinite number of time steps.

when interacting alone. As occurred in previous cooperative configurations, QLMerciful loses some good rewards since it is sometimes in the same team than Random, making its behaviour of helping its teammate useless. Again, QLSelfish is the fastest Q-learning agent to obtain its best result, and again it is because it does not have to consider the results of the other agents. QLCommunal also leads to good results, but it is not as fast as QLSelfish, since it also has to adapt to its teammate's actions. QLHarmful is usually satisfied when the opponent team obtains bad rewards, forgetting to improve its own rewards, making it to slowly learn how to correctly achieve good results.

Finally, in figure 6.5 (b) we see the results obtained for the partition of agent slots {{1}, {2}, {3}}, where the three agents have opposed interests. Here we see that QLSelfish, QLCommunal and QLMerciful reach similar results as expected, since none of them have to take into account other agents in their team. But QLHarmful takes longer to reach the same result, since it also has to concern about making decrease other agents' rewards. In fact, these results are similar (not regarding agents' results numbers, but their grading) to partition of agent slots {{1}, {2}} shown in figure 6.4 (a), since agents are similarly arranged into teams by only having one agent on each team.

With this environment, we could see that QLSelfish is learning faster to obtain better rewards, and the other Q-learning agents take more time to reach the same results (or they never reach them). This is because QLSelfish does not take into account the actions of the other agents, which can be considered as noise, so they do not distract it to perform well.

Although this same environment has been used in section 4.3, the goal of this experiment is clearly different. Here we are interested in analysing the effect that team partitions have on agents that have some social behaviours. Here we show an example where partitions of agent



(a) Partition of agent slots into teams {{1, 2}, {3}}.

(b) Partition of agent slots into teams {{1}, {2}, {3}}.

Figure 6.5: Mean with standard error of the mean of agents' results with 5 Monte Carlo samples to approximate their expected results (an average of rewards) and after averaging over 200 repetitions. Green and red dots respectively mean the agents reach or not their maximum possible result after an infinite number of time steps.

slots do not provide the desired result of encouraging cooperation/competition, at least for this agent population of Q-learning and random agents. Although the agents will finally reach similar results, it is clear that team-oriented agents are not favoured just by including them in teams, and self-oriented agents do not always obtain better results when they have to compete for reward units. This result can be explained because the environment does not foster the interaction between the agents, so individual actions are better rewarded and, therefore, the partition of agent slots cannot benefit from neither cooperative nor competitive actions.

6.4 Predator-Prey (Pursuit Game)

Our last environment is a pursuit game called predator-prey (section 2.1.3).

For this environment, $\tau = \{\{1\}, \{2, 3, 4\}\}$ represents the partition of agent slots. The first team $\{1\}$ contains the prey and the second team $\{2, 3, 4\}$ contains three predators. Agent slot 1 starts in the upper left corner and agent slots $2, 3$ and 4 start in the upper right, bottom left and bottom right corners respectively. Since some of the agents we use differ in how they treat individual and team rewards, we slightly change the rewards provided by the environment (i.e. $\mathcal{R}_i$) in order to let the agents use individual rewards. However, their team rewards are calculated as usual. Table 6.2 shows the individual reward matrix which has the current time step and the chasing situation as input and the agents' individual rewards as output. The predator who chased the prey obtains an individual reward of 18, while the rest of predators obtain an individual reward of 0. If two or more predators chase the prey at the same time step, the 18 individual rewards are distributed equally. Note that the obtained team rewards are the same, so agents' results are consistent with the original $\mathcal{R}_i$.

	Chased	Not chased
Time step 1-5	0	0
Time step 6	-6	6

(a) Prey individual rewards.

	Chased	Not chased
Time step 1-5	$(0, 0)$	0
Time step 6	$(18, 0)$	-6

(b) Predator individual rewards.

Table 6.2: Predator-prey's individual payoff matrix. Rows represent the time step, while columns represents whether the prey has been chased or not. In table (b), cells content (X, Y) in the prey chased situation corresponds to the individual reward (X) shared by the predators who chased the prey, and the individual reward (Y) for the rest of predators.

For this environment, according to algorithm 1, the parameters obtained for the Q-learning agents are *learning rate* $= 0.95$ and *discount factor* $= 0.9$. In figure 6.6 we show the results for each agent interacting as a predator (agent slots 2, 3 and 4) with 100 Monte Carlo samples to approximate their expected results (an average of rewards), for the partition of agent slots {{1}, {2,3,4}} after training them during 1,500 episodes and averaging over 200 repetitions.

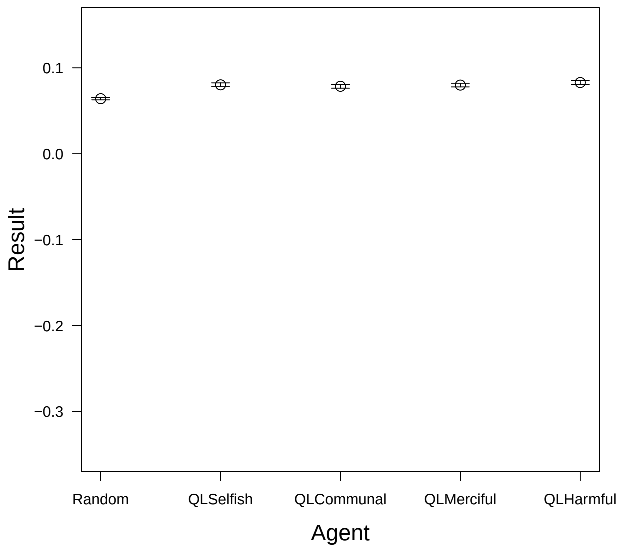
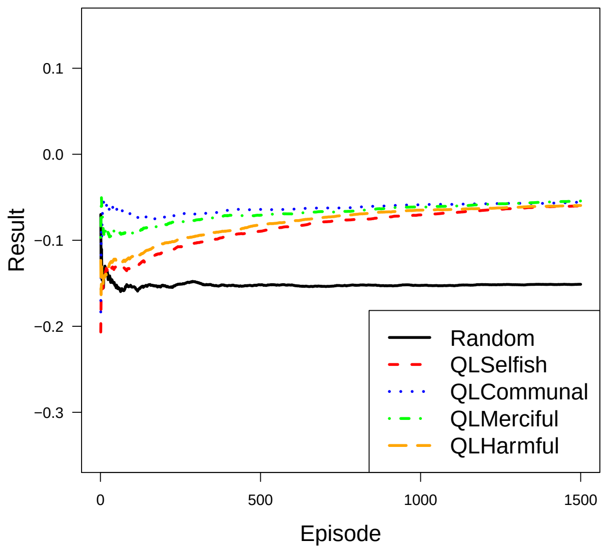


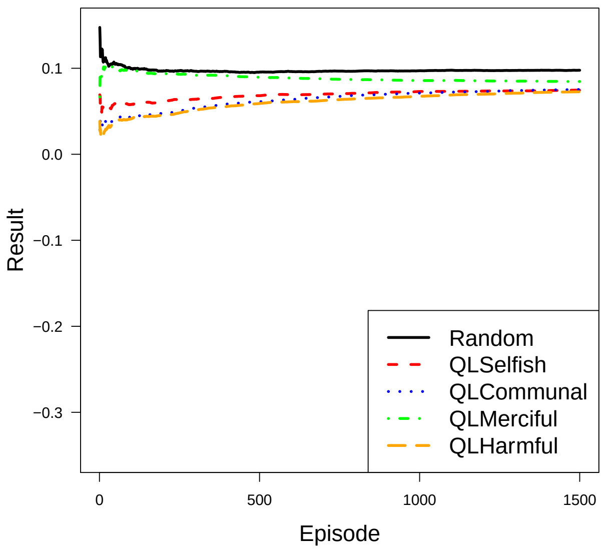
Figure 6.6: Mean with standard error of the mean of agents' results interacting as a predator with 100 Monte Carlo samples to approximate their expected results (an average of rewards), for the partition of agent slots into teams {{1}, {2, 3, 4}} after training them during 1,500 episodes and averaging over 200 repetitions.

As expected, Random obtains the worst result, since it is not possible for the other two predators to coordinate with it to chase the prey. For the rest of agents, QLSelfish and QL-Harmful lead the results, although they are very close. In fact, we cannot conclude that any of the Q-learning agents is better than the others.

As there are not many meaningful combinations of partition of agent slots in this environment to analyse, and the previous one has not shown a clear result, now we analyse whether it is possible to obtain different gradings for the evaluated agents depending on which agents we use to populate the environment. In other words, we analyse the influence of the composition of the teams. In figure 6.7 we can see the training phase of different situations when the evaluated agents interact along with two Random agents as predators. In figure 6.7 (a), the prey is also a Random agent, while figure 6.7 (b) shows the results when Q-learning agents act as the prey.



(a) Random interacts as the prey.



(b) Q-learning agents interact as the prey.

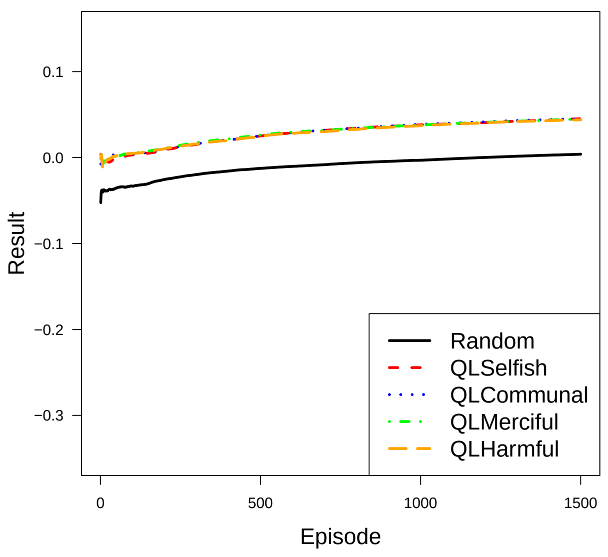
Figure 6.7: Agents' results for the training phase during 1,500 episodes with 100 Monte Carlo samples to approximate their expected results (an average of rewards) having two Random agents as teammates.

In figure 6.7 (a), Q-learning agents obtain quite similar results around -0.06, while the result of Random is stabilised around -0.15. As expected, three Random agents cooperating as predators have more difficulties to chase a Random agent prey, while two Random agents along with a Q-learning agent cooperating as predators chase it more often.

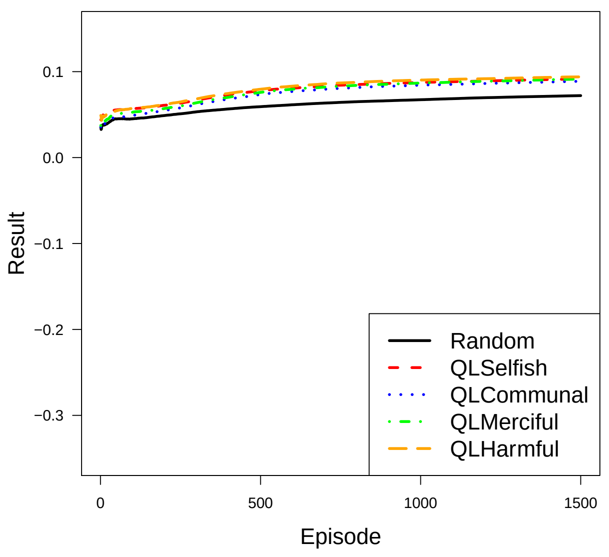
In figure 6.7 (b), the agents obtain better results, since it is easier to predict the movements of a prey when it follows some kind of simple strategy. The result of Random stabilises around 0.1, while Q-learning agents obtain slightly worse results. QLMerciful seems to quickly stabilise around 0.08, while QLSelfish, QLCommunal and QLHarmful have to learn several episodes to reach a slightly lower result (although it seems that with more episodes they will reach the result of QLMerciful). The explanation is simple. In this scenario, the prey has to escape from two Random agents in a small space, which becomes quite difficult, and when the other predator also acts randomly, it is still more difficult to anticipate the three predators' actions. But when the third predator is actually trying to chase the prey, it is easier to anticipate its movements and, therefore, easier to escape.

Now let us see in figure 6.8 what happens when the two predator teammates are also social agents. We can see in figure 6.8 (a) that social agents' results have clearly increased when they have to chase a Random agent prey (with respect to figure 6.7 (a)), as well as Random also increases its results since it can benefit from social agents coordination. In figure 6.8 (b), the change is more visible. Social agents have slightly increased from figure 6.7 (b) as they now have two social agents as predator teammates to coordinate with. Random has worsen, even to the point of having worse results than the social agents. This is consistent with the idea that social behaviour only pays off when there are other social agents in the environment.

With this environment we see that we can obtain different rankings for the several agents depending on the other agents present in the line-up. This shows us that social intelligence is



(a) Random interacts as the prey.



(b) Q-learning agents interact as the prey.

Figure 6.8: Agents' results for the training phase during 1,500 episodes with 100 Monte Carlo samples to approximate their expected results (an average of rewards) having two social agents as teammates.

not only dependent on the environment used for the evaluation or the partition of agent slots, but also on the agents used to populate the environment.

6.5 Aggregation of Results: Towards Social Intelligence Evaluation

Now let us calculate the social intelligence (Υ) of the set of agents used for the experiments. We use uniform weights for environments w_M, agent slots w_S and line-ups $w_{\dot{L}}$ to give similar importance to all environments, agent slots and line-ups, but with $w_S(1, \mu) = 0$ for the predator-prey since, in this environment, we are only interested in agents' performance interacting as a predator (agent slots 2, 3 and 4). In previous subsections, we already calculated the performance of the agents for all the agent slots of interest and line-ups, using uniform weights for agent slots and line-up patterns. Actually, when we change the partition of agent slots in an environment, we are creating a (slightly) different environment. Using equation 5.4 with a uniform weight for environments w_M, let us first calculate the social intelligence for the three prisoner's dilemma environments (table 6.3), the six lambda environments (table 6.4) and the predator-prey environment (table 6.5).

As we can see, each environment has different social intelligences for the agents. Random is clearly the worst of the agents. In the predator-prey all Q-learning agents have similar results, but there exists a grading among the agents in the other two environments and they are different depending on which environment we look at. In the prisoner's dilemma, team-oriented agents clearly have the best results, whilst in the lambda environment results are more equal.

Finally, we can also calculate the social intelligence in the three environments for all social scenarios disaggregating by partition of agent slots. Since the rewards of the environments do

	{{1, 2, 3}}	{{1, 2}, {3}}	{{1}, {2}, {3}}	Υ
Random	4.29808	3.42243	2.11226	3.27759
QLSelfish	4.01913	3.52103	2.98515	3.50843
QLCommunal	5.16402	3.86189	3.02557	4.01716
QLMerciful	5.21163	3.95900	3.03797	4.06953
QLHarmful	4.04678	3.41647	2.95404	3.47243

Table 6.3: Agents' social intelligence for the prisoner's dilemma.

	{{1}}	{{1, 2}}	{{1}, {2}}
Random	0.00210	0.20756	-0.08068
QLSelfish	0.88013	0.42655	0.51851
QLCommunal	0.87829	0.41755	0.51864
QLMerciful	0.87735	0.29792	0.52756
QLHarmful	0.88257	0.42723	0.50538

	{{1, 2, 3}}	{{1, 2}, {3}}	{{1}, {2}, {3}}	Υ
Random	0.12106	0.01710	-0.08158	0.03092
QLSelfish	0.21164	0.26259	0.29388	0.43221
QLCommunal	0.18182	0.23876	0.29435	0.42156
QLMerciful	0.16105	0.21003	0.29240	0.39438
QLHarmful	0.21015	0.21836	0.22382	0.41125

Table 6.4: Agents' social intelligence for the lambda environment.

	{{1}, {2, 3, 4}} : Υ
Random	0.06413
QLSelfish	0.08025
QLCommunal	0.07856
QLMerciful	0.07996
QLHarmful	0.08293

Table 6.5: Agents' social intelligence for the predator-prey.

not have the same domain, we cannot use their expected result to perform an aggregation of R as for definition 12. Instead, here we use the ranking of the agents. We consider that each agent belongs to a tie group of agents (TGA), where all agents within the group obtain the same rank value. We use the mean (M) and standard error (SE) to determine whether an agent belongs to a tie group. For agent i we calculate a tie acceptance range as $TAR_i = [M_i - SE_i, M_i + SE_i]$. An agent i belongs to a TGA iff $\forall_{j \in TGA} : TAR_i \cap TAR_j \neq \emptyset$. If an agent can belong to different tie groups of agents, it will belong to the group with most agents. Now, we calculate agents' ranking performance for the three environments on cooperative, competitive and mixed scenarios by using partition of agent slots where all agent slots are in the same team (table 6.6), different teams (table 6.7) and mixed (table 6.8) respectively[6]:

Now we can see that Random obtains the worst result in all kinds of scenarios. In the

[6]PD, LE and PP represent the prisoner's dilemma, lambda environment and predator-prey environments respectively.

	PD	LE	Υ
Random	3	5	4.00000
QLSelfish	4.5	1.5	3.00000
QLCommunal	2	3	2.50000
QLMerciful	1	4	2.50000
QLHarmful	4.5	1.5	3.00000

Table 6.6: Agents' ranking performance for cooperative partitions of agent slots into teams ({{1, 2, 3}}).

	PD	LE	Υ
Random	5	5	5.00000
QLSelfish	2	2	2.00000
QLCommunal	2	2	2.00000
QLMerciful	2	2	2.00000
QLHarmful	4	4	4.00000

Table 6.7: Agents' ranking performance for competitive partitions of agent slots into teams ({{1}, {2}, {3}}).

	PD	LE	PP	Υ
Random	4.5	5	5	4.83333
QLSelfish	3	1	2.5	2.16666
QLCommunal	2	2	2.5	2.16666
QLMerciful	1	3.5	2.5	2.33333
QLHarmful	4.5	3.5	2.5	3.50000

Table 6.8: Agents' ranking performance for mixed partitions of agent slots into teams ({{1, 2}, {3}} for PD and LE, and {{1}, {2, 3, 4}} for PP).

cooperative scenario, team-oriented agents obtain the best positions, while in the competitive scenario, QLSelfish, QLCommunal and QLMerciful obtain the same results, while QLHarmful is the worst of the Q-learning agents. This grading is explained because when all agent slots are in different teams the first three agents lead to the same behaviour, obtaining similar results, and QLHarmful is right behind them. In the mixed scenario, the grading is more diverse. In this case, QLSelfish and QLCommunal obtain the best results.

6.6 Discussion

We have seen some interesting and diverse results from these experiments. We have seen that the arrangement of agents into different teams can have an impact on how cooperative/competitive the environment is, making different social agents obtain different results depending on the partition of agent slots. This is the case of the prisoner's dilemma, an initially cooperative environment, where the results of team-oriented social agents worsen as long as the partition of agent slots becomes more competitive. But this does not work for every environment (as we saw in the lambda environment), where making the environment more cooperative/competitive does not provide the expected result since the environment is not evaluating social behaviour. Also, other environments (such as the predator-prey) are less malleable to use partitions of agent slots. So, if we want to evaluate social intelligence using this setting, we should carefully choose which environments we want to use. Besides, as shown in the predator-prey environment, we saw that social intelligence is not only dependent on the rules of the environments, but also on which agents are used to populate them. For instance, we saw that social agents reach better results than a random agent when they have to interact with social agents as teammates. This forces us to be careful about which set of agents we select to populate the environments. Finally, with this setting we showed agents' results for each environment, providing us their social intelligence for that particular (or specialised) environment. But, more importantly, we could also include a variety of environments, allowing us to evaluate their social intelligence (or ranking performance) in a more general sense.

From a general perspective, we have to realise that here we used a random agent as well as a set of *modified* Q-learning as social agents to perform the experiments. But Q-learning was not initially meant to be a social agent. Maybe with more *real* social agents this picture would change and partitions of agent slots could better encourage cooperation/competition. But still with such low social intelligence agents, this experiments show that the inclusion of partitions of agent slots by itself is not enough (although it helps) to ensure a cooperative or competitive scenario, so social intelligence cannot be evaluated in all scenarios. The environments and the agents used have a big impact on the social aspect of the testbed.

In the end, we have seen several scenarios where we obtained different results with the inclusion of partitions of agent slots. This shows that we have a powerful tool to study both cooperative and competitive scenarios. However, not every environment is equally useful for this. In particular, we can barely take advantage of this tool in environments where agents cannot benefit from interacting with other agents. This suggests that we should also analyse the environments themselves in order to see how suitable they are to evaluate social intelligence. Nonetheless, the key idea is the aggregation of results for several environments, as given by definition 12, as we have done in the final part of the experiments. Once the environments and the agents are properly selected, this definition can be used to measure an agent's social intelligence.

Chapter 7

Properties About Social Intelligence Testbeds

In order to evaluate social intelligence and distinguish it from general intelligence, we need tests where social abilities have to be used and, also, where we can perceive their consequences. This means that not every multi-agent environment is useful for measuring social intelligence and not every set of populating agents is also useful. We want tests such that the evaluated agent must use its social intelligence to understand and/or have influence over other agents' policies in such a way that this is useful to accomplish its goals. We also need situations where common general intelligence is not enough. In a way, we want to *subtract* (from the summation of all multi-agent environments and line-up patterns) those problems (as defined by sets of multi-agent environments and populating agents) where general intelligence is enough (and social intelligence is useless) and those where intelligence (of any kind, social or non-social) is useless.

We investigate some property models that are hence desirable (or necessary) for a social intelligence testbed, and more specifically for its set of multi-agent environments M and set of agents to populate them Π_o, to measure social intelligence. For many of the properties we present, we have been inspired by many and different research areas such as game theory, ecology, statistics or psychology. Actually, we define these properties for one environment but they are easily extensible to a family or distribution of environments using a weight function. Similarly, some of the properties below are presented for two agent slots but they could be extended to three or more agent slots as well. In particular, we provide a property about interactivity, designed to measure how the populating agents reflect on the actions performed by the evaluated agents. Similarly, non-neutralism properties have been designed to measure how the populating agents reflect on the rewards obtained by the evaluated agents. Basically, interactivity and non-neutralism have been designed in order to measure how the populating agents affect on how the evaluated agents interact. Furthermore, we included two anticipation properties (competitive and cooperative) in order to measure how the knowledge of the populating agents behaviours can be exploited by the evaluated agents to perform better. We also provide some other properties that are more of a practical nature: such as the degree of discrimination and grading of the environments, measuring whether different agents obtain different results and they can be ordered based on their results respectively; the boundedness and team symmetry of agent slots, facilitating the evaluation procedure; or the reliability and efficiency of a test, included to determine whether such a test can be easily applied. Finally, the validity property has been included with the goal to identify how a test to evaluate social intelligence should be.

Figure 7.1 gives a summary of the properties we consider for a social intelligence testbed. They have different purposes and reach different levels of formalisation. Many of the properties (the quantitative ones) follow the structure $Prop(\Pi_e, w_{\Pi_e}, \Pi_o, w_{\dot{L}}, \mu, w_S)$, i.e. given the two agent sets Π_e and Π_o (for evaluated agents and populating agents respectively), the weights for them (w_{Π_e} for evaluated agents and $w_{\dot{L}}$ for line-up patterns respectively), a multi-agent environment μ and its agent slot weight w_S, we obtain a value for the property $Prop$. Note that we use a set for evaluated agents Π_e and a weight over them w_{Π_e}. This is a mechanism we use to calculate the properties not only for one evaluated agent, but for several of them in a general way. We have designed this quantitative properties in such a way that can be applied with the parameters used (appropriately adjusted) in the definition of social intelligence (i.e. the parameters in definition 12).

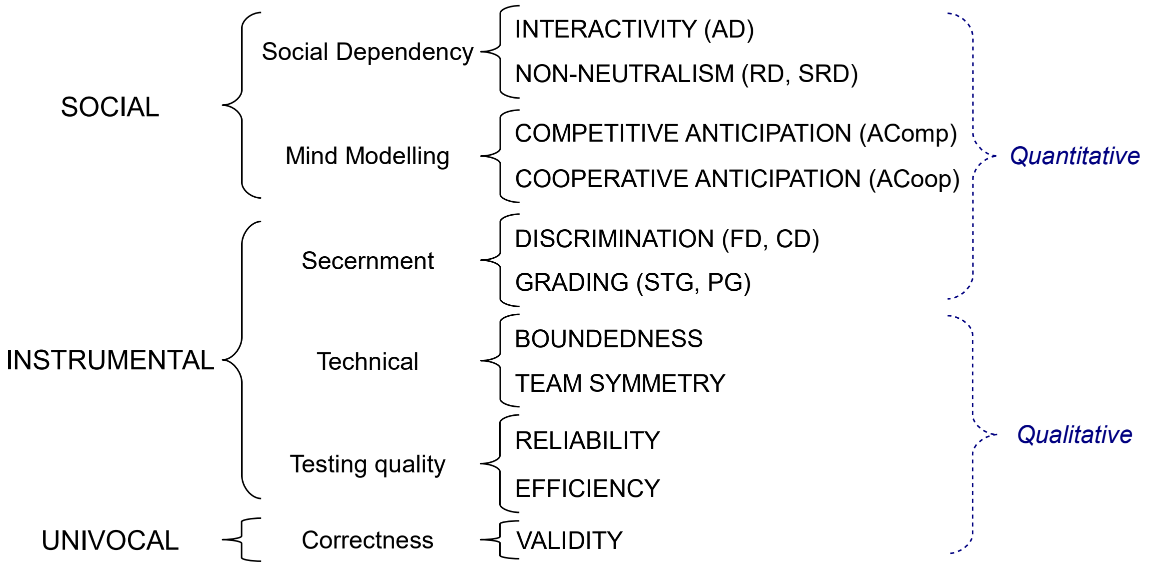


Figure 7.1: Taxonomy of property models about social intelligence testbeds, grouped in three main categories (social, instrumental and univocal) and six subcategories (social dependency, mind modelling, secernment, technical, testing quality and correctness).

In some of the properties we present, we use $\breve{A}_i^K(\mu[l])$ to denote the distribution (a probability measure) of action sequences that the ith agent in line-up l performs in the multi-agent environment μ (also in agent slot i) during K time steps. If K is omitted, we assume $K = \infty$, i.e. infinite sequences of actions for an endless episode. If μ and the agents on l are deterministic then this boils down to a probability measure giving probability 1 to one single sequence, the sequence of actions performed by the ith agent of l on μ. Similarly, we use $\breve{R}_i^K(\mu[l])$ to denote the distribution of reward sequences that the ith agent in l obtains in μ (also in agent slot i) during K time steps.

With these properties we try to represent how appropriate a multi-agent environment μ and the set of populating agents Π_o are in order to evaluate the social intelligence of a given set of agents Π_e. Next, we analyse and formalise them.

7.1 Boundedness

One desirable property is that rewards are bounded, otherwise the value of Υ (e.g. equation 5.4) could diverge. Any arbitrary choice of upper and lower bounds can be scaled to any other choice so, without loss of generality, we can assume that all of them are bounded between -1 and 1. Formally:

$$\forall i, k : -1 \leq r_{i,k} \leq 1 \tag{7.1}$$

Note that they are bounded for every time step. So, if we use a bounded utility function to calculate the agent's result (e.g. an average of rewards), then $R_i^K(\cdot)$ is also bounded.

However, bounded expected results do not ensure that the measurement from Υ is bounded. In order to ensure bounded measurements of social intelligence and the properties we present, we also need to consider that weights are bounded, i.e. there are constants c_M, c_S, c_{Π_e} and $c_{\dot{L}}$ such that:

$$\forall M : \sum_{\mu \in M} w_M(\mu) = c_M \tag{7.2}$$

$$\forall \mu : \sum_{i=1}^{N(\mu)} w_S(i, \mu) = c_S \tag{7.3}$$

$$\forall \Pi_e : \sum_{\pi \in \Pi_e} w_{\Pi_e}(\pi) = c_{\Pi_e} \tag{7.4}$$

$$\forall i, n, \Pi_o : \sum_{\dot{l} \in \dot{L}^n_{-i}(\Pi_o)} w_{\dot{L}}(\dot{l}) = c_{\dot{L}} \tag{7.5}$$

Equation 7.5 can also be extended for two or more non-instantiated positions.

A convenient choice would be to have $c_M = c_S = c_{\Pi_e} = c_{\dot{L}} = 1$, and these weights would become unit measures (which should not be confused with the probabilities used to sample elements in a test). With these conditions on rewards and weights, Υ and $\hat{\Upsilon}$ are bounded[1], as well as some of the properties we present below.

An optional property that might be interesting occasionally is to consider environments whose reward sum is constant. Without loss of generality, we can take this constant to be zero, which leads to the well-known notion of zero-sum games in game theory.

Definition 14. A multi-agent environment μ is zero-sum if and only if:

$$\forall k : \sum_{i=1}^{N(\mu)} r_{i,k} = 0 \tag{7.6}$$

The above definition may be too strict when we have environments with an episode goal at the end, but we want some positive or negative rewards to be given while agents approach the goal. A more convenient version follows:

[1]Note that we are talking about the measure. For instance, Υ can be a measure that represents the, e.g. sigmoid function of an unbounded magnitude, easily recovered with a logit or probit function.

Definition 15. A multi-agent environment μ is zero-sum in the limit iff:

$$\lim_{K\to\infty} \sum_{k=1}^{K} \sum_{i=1}^{N(\mu)} r_{i,k} = 0 \tag{7.7}$$

With teams, the previous definition could be modified in such a way that:

$$\lim_{K\to\infty} \sum_{k=1}^{K} \sum_{t\in\tau} \sum_{i\in t} r_{i,k} = 0 \tag{7.8}$$

So the sum of the agents' rewards in a team does not need to be zero but the sum of all agents' rewards does. For instance, if we have a team with agents $\{1, 2\}$ and another team with agents $\{3, 4, 5\}$, then a result (in the limit) of $1/4$ for agents 1 and 2 implies $-1/6$ for each of the agents in the other team.

The zero-sum properties are appropriate for competition between teams. In fact, if we have two agents and each in a different team then we have *pure competitive* environments. We can have both competition and cooperation by using teams in a zero-sum environment, where agents in a team cooperate and agents in different teams compete. If we want to evaluate *pure cooperation* (with one or more teams) then zero-sum environments are not appropriate.

7.2 Interactivity

By interactivity we mean that agents' actions have implications on the actions (and ultimately on rewards) of the other agents. This is a key property as the existence of several agents in an environment does not ensure, per se, any social behaviour. In fact, it is important to realise that the use of several agents and their arrangement into teams does not ensure that some social behaviour can ever take place. Imagine a non-social environment, such as finding the way out of a maze (without other agents). Rewards depend on the agent finding the way out or not. While this is clearly non-social, we can use this environment as a building block and create a multi-agent environment that takes two agents but makes them play separately on two equal mazes. We can generate rewards in at least four different ways: (1) we can give rewards separately without any modification on the outputs of the building blocks, (2) we can normalise them to a constant or a zero-sum, (3) we can average both rewards and give them to both agents or (4) any other combination of the rewards, including a stochastic (non-deterministic) combination. Note that none of these four options would contain or foster any kind of social behaviour. In fact, no agent is aware of the other agent's presence (apart from the effect on rewards, which could be attributed to some randomness of the environment). However, the rewards can get highly correlated (as in ways 2 or 3 above) and do so in a non-additive or functional way.

The explanation of why there is no social behaviour in this case is that one agent can not have influence on the actions that the other agent has to perform. As a result, the big issue about choosing social contexts is how to determine that an agent has influence on other agents' actions. In fact, this is at the roots of definitions of interaction [23, 141, 21, 58] and the distinction between several kinds of interaction [66]. In fact, a similar detection of interactivity has already been attempted under the name of reactivity [1]. Some other approaches have looked for some common information content between the peers. However, as pointed out by [21], 'this may originate from a common source', so common or mutual information is not sufficient for interaction to have taken place.

So we need a measure of interaction that is not based on correlation or common information content. However, the degree of influence that other agents may have on the actions of the agent we are evaluating is difficult to grasp; as environments and agents can be non-deterministic, changes can appear just randomly.

7.2.1 Action Dependency

We need to take a different approach. The key idea defines interaction in terms of sensitivity to other agents or, in other words, whether the inclusion of different agents in the multi-agent environment has an effect on what the evaluated agent does. One formalisation of this idea goes as follows:

Definition 16. The action dependency degree for the evaluated agent π interacting in agent slot i in multi-agent environment μ with a set of populating agents Π_o with a weight of agent line-up patterns $w_{\dot{L}}$, is given by:

$$AD_i(\pi, \Pi_o, w_{\dot{L}}, \mu) \triangleq \eta_{\dot{L}^2} \sum_{\dot{u},\dot{v} \in \dot{L}_{-i}^{N(\mu)}(\Pi_o) | \dot{u} \neq \dot{v}} w_{\dot{L}}(\dot{u}) w_{\dot{L}}(\dot{v}) \Delta_{\mathcal{A}^+}(\breve{A}_i(\mu[\dot{u} \overset{i}{\leftarrow} \pi]), \breve{A}_i(\mu[\dot{v} \overset{i}{\leftarrow} \pi])) \quad (7.9)$$

where $\eta_{\dot{L}^2}$ normalises the formula with $\eta_{\dot{L}^2} = \frac{1}{\sum_{\dot{u},\dot{v} \in \dot{L}_{-i}^{N(\mu)}(\Pi_o)|\dot{u}\neq\dot{v}} w_{\dot{L}}(\dot{u}) w_{\dot{L}}(\dot{v})}$, $\Delta_{\mathcal{A}^+}$ denotes a divergence function between distributions of action sequences, $|\Pi_o| \geq 2$, $N(\mu) \geq 2$ and $\exists \dot{u}, \dot{v} \in \dot{L}_{-i}^{N(\mu)}(\Pi_o) | \dot{u} \neq \dot{v}, w_{\dot{L}}(\dot{u}) > 0$ and $w_{\dot{L}}(\dot{v}) > 0$.

This equation basically calculates how the actions performed by π depend on what other agents are present on the environment.

Note that $\breve{A}$ returns a distribution of action sequences if the environment or any of the agents is non-deterministic. If AD_i is high, then the proportion of cases where two line-up patterns lead to different action sequences for π is high. This means that π is highly sensitive in this environment about who else is interacting in it. Conversely, if for many pairs of line-up patterns the action sequences of π are similar, this means that π's actions are not usually affected by other agents. Note that having action dependency does not imply that π knows which actions are performed by the other agents, but that their actions have implications in π's actions.

The previous definition is relative to a distribution of line-up patterns on a population of agents, but it is given for a particular evaluated agent π interacting in a particular agent slot i. We may have that one evaluated agent can be very insensitive to line-up pattern changes, but other evaluated agents can be more sensitive in the same situation. Similarly, being in agent slot i may make the evaluated agent ignore the actions of the other agents, but other agent slots could be more willing to make it take into account the other agents' actions. If we want to generalise this for a set of evaluated agents Π_e and aggregate for all agent slots, then we have:

Definition 17. The action dependency degree for a set of evaluated agents Π_e with associated weight w_{Π_e} in multi-agent environment μ with weight of agent slots w_S, a set of populating agents Π_o and a weight of agent line-up patterns $w_{\dot{L}}$, is given by:

$$AD(\Pi_e, w_{\Pi_e}, \Pi_o, w_{\dot{L}}, \mu, w_S) \triangleq \sum_{\pi \in \Pi_e} w_{\Pi_e}(\pi) \sum_{i=1}^{N(\mu)} w_S(i, \mu) AD_i(\pi, \Pi_o, w_{\dot{L}}, \mu) \quad (7.10)$$

where $N(\mu) \geq 1$ and $|\Pi_e| \geq 1$.

It certainly remains to clarify what $\Delta_{\mathcal{A}^+}$ can be. For deterministic cases, a possibility would be to represent the action sequences as strings and use an edit distance. However, for non-deterministic cases we need to find alignments between the distributions or aggregate action sequences into some prototypes and compare them. One simple approach for both (the deterministic and non-deterministic cases) could be based on comparing action frequencies (independently of their order) or n-grams. Note that different $\Delta_{\mathcal{A}^+}$ functions may lead to different interpretations of action influence. For instance, there can be environments where a first few actions are interactive, but then no interaction takes places any more. In this case, the action sequences may be very different, but the degree, or more precisely, the timespan of interaction is small (like a butterfly effect).

7.3 Non-Neutralism

The existence of interaction between agents does not ensure that these interactions are meaningful in terms of rewards. For instance, two agents can influence each other's actions, but they may not affect each other's rewards. This, in ecological terms, is known as 'neutralism'. In fact, in ecology, given two species, there are seven possible combinations of positive, negative or no effect between them, leading to six forms of symbiosis [73]: neutralism (0,0), amensalism (0,-), commensalism (+,0), competition (-,-), mutualism (+,+), and predation/parasitism (+,-). In our case, as we want to characterise environments that may contain individuals (possibly more than two), we can simplify this to neutralism, cooperation (including commensalism and mutualism) and competition (including the rest). In other words, we want to analyse whether interaction has no effect on rewards (and ultimately on agents' expected results), has a positive relation or a negative one.

7.3.1 Reward Dependency

So, the first thing that we need to determine is whether the evaluated agent's rewards are affected by the presence of other agents, i.e. there is a dependency in rewards. This is very similar to the action dependency seen above:

Definition 18. The reward dependency degree for the evaluated agent π interacting in agent slot i in multi-agent environment μ with a set of populating agents Π_o and a weight of agent line-up patterns $w_{\dot{L}}$, is given by:

$$RD_i(\pi, \Pi_o, w_{\dot{L}}, \mu) \triangleq \eta_{\dot{L}^2} \sum_{\dot{u},\dot{v} \in \dot{L}_{-i}^{N(\mu)}(\Pi_o) | \dot{u} \neq \dot{v}} w_{\dot{L}}(\dot{u}) w_{\dot{L}}(\dot{v}) \Delta_{\mathbb{Q}^+}(\breve{R}_i(\mu[\dot{u} \overset{i}{\leftarrow} \pi]), \breve{R}_i(\mu[\dot{v} \overset{i}{\leftarrow} \pi])) \quad (7.11)$$

where $\eta_{\dot{L}^2}$ normalises the formula with $\eta_{\dot{L}^2} = \frac{1}{\sum_{\dot{u},\dot{v} \in \dot{L}_{-i}^{N(\mu)}(\Pi_o) | \dot{u} \neq \dot{v}} w_{\dot{L}}(\dot{u}) w_{\dot{L}}(\dot{v})}$, $\Delta_{\mathbb{Q}^+}$ denotes a divergence function for distributions of rational number sequences, $|\Pi_o| \geq 2$, $N(\mu) \geq 2$ and $\exists \dot{u}, \dot{v} \in \dot{L}_{-i}^{N(\mu)}(\Pi_o) | \dot{u} \neq \dot{v}, w_{\dot{L}}(\dot{u}) > 0$ and $w_{\dot{L}}(\dot{v}) > 0$.

For $\Delta_{\mathbb{Q}^+}$ we could use a function similar to the one used in $\Delta_{\mathcal{A}^+}$ (section 7.2.1), calculating the divergence between distributions of rational number sequences. Another possibility would be to first calculate the expected result for both distributions of reward sequences, and then use a divergence function between rational numbers $\Delta_{\mathbb{Q}}$. Following this possibility, we can use $\Delta_{\mathbb{Q}}(a,b) = 1 - \delta(a,b)$, where δ is the Kronecker delta function ($\delta(a,b) = 1$ if $a = b$ and 0 otherwise). With this choice, equation 7.11 would boil down to the probability that, when π is instantiated into two line-up patterns in position i, its expected results interacting in agent slot i of μ are different (using weight $w_{\dot{L}}$). Another choice could be relative absolute difference, i.e. $\Delta_{\mathbb{Q}}(a,b) = \frac{|a-b|}{|a|+|b|}$.

We can generalise this for a set of evaluated agents Π_e and aggregate for all agent slots:

Definition 19. The reward dependency degree for a set of evaluated agents Π_e with associated weight w_{Π_e} in multi-agent environment μ with weight of agent slots w_S, a set of populating agents Π_o and a weight of agent line-up patterns $w_{\dot{L}}$ is given by:

$$RD(\Pi_e, w_{\Pi_e}, \Pi_o, w_{\dot{L}}, \mu, w_S) \triangleq \sum_{\pi \in \Pi_e} w_{\Pi_e}(\pi) \sum_{i=1}^{N(\mu)} w_S(i,\mu) RD_i(\pi, \Pi_o, w_{\dot{L}}, \mu) \tag{7.12}$$

where $N(\mu) \geq 1$ and $|\Pi_e| \geq 1$.

So now we measure how dependent the rewards are in general (for any evaluated agent in Π_e and agent slot).

The previous definition may slightly resemble the Shapley Value [99] in cooperative game theory, but here we are not concerned with how relevant each agent is in a team (whether its contribution is higher than the contribution of its teammates), but to see whether there is effect on the rewards.

7.3.2 Slot Result Dependency

Both definitions 17 and 19 are necessary as we can have reward dependency without action dependency and action dependency without reward dependency. That is we could have an agent that always performs the same actions and depending on the other agents it could obtain different rewards. Similarly, an agent could perform different actions depending on the actions performed by the other agents, but obtaining always the same rewards. An ideal situation would be an environment where the populating agents are able to influence the rewards obtained by the evaluated agent, as well as the actions that it can perform. Ultimately, such an environment would force the evaluated agent to take into consideration the other agents' behaviour in order to improve its own results. This will make the evaluating agent use its social intelligence to understand these other agents and act accordingly to perform better.

Now, we are interested in telling the *sign* of this dependency, i.e. whether the evaluated agent has to consider the agents interacting in other agent slots as cooperative or competitive.

Definition 20. The slot result dependency for the evaluated agent π interacting in agent slot i with agent slot j (with i $\neq$ j) in multi-agent environment μ with a set of populating agents Π_o and a weight of agent line-up patterns $w_{\dot{L}}$ is given by:

$$SRD_{i,j}(\pi, \Pi_o, w_{\dot{L}}, \mu) \triangleq corr_{\dot{l} \in \dot{L}_{-i}^{N(\mu)}(\Pi_o)}[w_{\dot{L}}(\dot{l})](R_i(\mu[\dot{l} \xleftarrow{i} \pi]), R_j(\mu[\dot{l} \xleftarrow{i} \pi])) \tag{7.13}$$

where $corr_{x \in X}[w](a,b)$ is a weighted (w) correlation function between a and b for the elements of X, $|\Pi_o| \geq 1$ and $N(\mu) \geq 2$.

Any correlation function can be used here (Pearson, Spearman, etc.). Clearly, if two agent slots are in the same team, from definition 8, we have that its SRD is 1. In the case of a zero-sum environment with only two teams, any two agent slots of different teams have a SRD of -1 (provided there is at least one 'match' which is not a tie). Note that, as usual with correlation measures, if we have that two agent slots are reward independent, then its SRD is 0. However, having $SRD = 0$ does not necessarily imply independency. We would need to calculate the reward dependency degree and then ask whether the slot result dependency is positive or negative for pairs of agent slots.

Now, we can generalise this for a set of evaluated agents Π_e and aggregate for all combinations of pairs of agent slots:

Definition 21. The slot result dependency for a set of evaluated agents Π_e with associated weight w_{Π_e} in multi-agent environment μ with weight of agent slots w_S, with a set of populating agents Π_o and a weight of agent line-up patterns $w_{\dot{L}}$ is given by:

$$\begin{aligned} SRD(\Pi_e, w_{\Pi_e}, \Pi_o, w_{\dot{L}}, \mu, w_S) \triangleq \sum_{\pi \in \Pi_e} w_{\Pi_e}(\pi) \eta_{S_1^2} \sum_{i=1}^{N(\mu)} w_S(i,\mu) \times \\ \times \left(\sum_{j=1}^{i-1} w_S(j,\mu) SRD_{i,j}(\pi, \Pi_o, w_{\dot{L}}, \mu) + \sum_{j=i+1}^{N(\mu)} w_S(j,\mu) SRD_{i,j}(\pi, \Pi_o, w_{\dot{L}}, \mu) \right) \end{aligned} \tag{7.14}$$

where $\eta_{S_1^2}$ normalises the formula with $\eta_{S_1^2} = \frac{1}{\sum_{i=1}^{N(\mu)} w_S(i,\mu) \left(\sum_{j=1}^{i-1} w_S(j,\mu) + \sum_{j=i+1}^{N(\mu)} w_S(j,\mu) \right)}$, $N(\mu) \geq 2$, $|\Pi_e| \geq 1$, $|\Pi_e| \geq 1$ and $\exists i,j | 1 \leq i \leq N(\mu), 1 \leq j \leq N(\mu), i \neq j, w_S(i,\mu) > 0$ and $w_S(j,\mu) > 0$.

In practice, in order to evaluate social abilities, we require environments with high RD. Then, depending on the use of teams and normalisations, we can gauge whether we want to evaluate competition or cooperation, and have some positive SRD with some agent slots and some negative SRD with some other agent slots. This is easily obtained by using teams.

7.4 Anticipation

One crucial property that is related to social intelligence is anticipation, which means that the evaluated agents can benefit from anticipating other agents' moves or, in more general terms, by having a theory of others' minds. Having an environment where anticipating abilities are useful will permit the evaluated agent to take advantage of its understanding of other agents in order to improve its results. While a formalisation of this concept is very elusive, we can at least introduce an approximation.

In anticipation we usually expect that the evaluated agents can perform better if the agents they interact with can be well anticipated. This is difficult to define in general, but we can introduce a simplified approach based on the idea that one evaluated agent *anticipates* if its expected result interacting with a (generally) non-random agent is higher than its expected result interacting with a random agent. We generalise this as follows:

Definition 22. The anticipation benefit for the evaluated agent π interacting in agent slot i with respect to agent slot j in multi-agent environment μ with a set of populating agents Π_o and a weight of agent line-up patterns $w_{\dot{L}}$ is given by:

$$Ant_{i,j}(\pi, \Pi_o, w_{\dot{L}}, \mu) \triangleq \sum_{\dot{l} \in \dot{L}_{-i,j}^{N(\mu)}(\Pi_o)} \sum_{\pi_o \in \Pi_o} w_{\dot{L}}(\dot{l} \overset{j}{\leftarrow} \pi_o) \frac{1}{2} \left(R_i(\mu[\dot{l} \overset{i,j}{\leftarrow} \pi, \pi_o]) - R_i(\mu[\dot{l} \overset{i,j}{\leftarrow} \pi, \pi_r]) \right) \tag{7.15}$$

where π_r is a random agent, $\frac{1}{2}$ normalises the formula, $N(\mu) \geq 2$ and $|\Pi_o| \geq 1$.

$Ant_{i,j}$ uses as reference the expected result of the evaluated agent when it interacts with a random agent (which movements are impossible to anticipate, making the evaluated agent's anticipation abilities useless), and calculates the benefit that the evaluated agent can obtain when it interacts with populating agents instead (which movements could be anticipated)[2].

Definition 22 provides us information about the evaluated agent's exploitation of the other agents' behaviour in order to improve its own results. Note that such definition does not specify the relation between the agent slot that occupies the evaluated agent and the agent slot that occupies each of the populating agents. Next we see how we consider this relation in the context of competition and cooperation when using teams.

7.4.1 Competitive Anticipation

We usually expect that the evaluated agents can perform better if their opponents/competitors can be well anticipated. When using teams, the position of the opponents is clearly stated. So, in order to calculate this anticipation about competitive agents (or competitive anticipation), we must consider those agent slots that do not pertain to the team where the evaluated agent is.

Here, we generalise this concept for a set of evaluated agents Π_e and for all combinations of pairs of agent slots in different teams:

Definition 23. The competitive anticipation benefit for a set of evaluated agents Π_e with associated weight w_{Π_e} interacting in multi-agent environment μ with weight of agent slots w_S, with a set of populating agents Π_o and a weight of agent line-up patterns $w_{\dot{L}}$ is given by:

$$\begin{aligned} AComp(\Pi_e, w_{\Pi_e}, \Pi_o, w_{\dot{L}}, \mu, w_S) \triangleq \sum_{\pi \in \Pi_e} w_{\Pi_e}(\pi) \eta_{S_2^2} \sum_{t_1,t_2 \in \tau | t_1 \neq t_2} \sum_{i \in t_1} w_S(i,\mu) \sum_{j \in t_2} w_S(j,\mu) \times \\ \times Ant_{i,j}(\pi, \Pi_o, w_{\dot{L}}, \mu) \end{aligned} \tag{7.16}$$

where $\eta_{S_2^2}$ normalises the formula with $\eta_{S_2^2} = \frac{1}{\sum_{t_1,t_2 \in \tau | t_1 \neq t_2} \sum_{i \in t_1} w_S(i,\mu) \sum_{j \in t_2} w_S(j,\mu)}$, τ is the partition of agent slots into teams of multi-agent environment μ, $N(\mu) \geq 2$, $|\Pi_e| \geq 1$ and $\exists t_1, t_2 \in \tau | t_1 \neq t_2, \exists i \in t_1, j \in t_2 | w_S(i,\mu) > 0$ and $w_S(j,\mu) > 0$.

If *AComp* is positive this means that the evaluated agents behave better against (generally) non-random agents than against random agents. One good example of the above definition is when t_1 is a predator team and t_2 is a prey team (and vice-versa). If the evaluated agents

[2]Note that, in the reference we use, the expected result of the evaluated agent could be different of zero. In fact, this reference provides us with some information about how important the evaluated agent's non-anticipating abilities are and, therefore, corrects the importance that the anticipating abilities have in the expected result. Possibly, an indicator about the reference might be useful to better interpret the result of this property.

in a set perform better with non-random preys than with random preys then the multi-agent environment shows a benefit for this set (and for these agent slots). Of course, this depends on Π_o, but if we include non-random opponents with some movement patterns and/or some degree of intelligence, the definition becomes more meaningful.

7.4.2 Cooperative Anticipation

On the other hand, we usually expect that the evaluated agents can perform better if their cooperators can be well anticipated. Similarly as done with competitive anticipation, we generalise this concept for a set of evaluated agents Π_e and for all combinations of pairs of agent slots in the same team:

Definition 24. The cooperative anticipation benefit for a set of evaluated agents Π_e with associated weight w_{Π_e} interacting in multi-agent environment μ with weight of agent slots w_S, with a set of populating agents Π_o and a weight of agent line-up patterns $w_{\dot{L}}$ is given by:

$$\begin{aligned} ACoop(\Pi_e, w_{\Pi_e}, \Pi_o, w_{\dot{L}}, \mu, w_S) \triangleq \sum_{\pi\in\Pi_e} w_{\Pi_e}(\pi)\eta_{S_3^2}\sum_{t\in\tau}\sum_{i,j\in t|i\neq j} w_S(i,\mu)w_S(j,\mu)\times \\ \times\, Ant_{i,j}(\pi, \Pi_o, w_{\dot{L}}, \mu) \end{aligned} \tag{7.17}$$

where $\eta_{S_3^2}$ normalises the formula with $\eta_{S_3^2} = \frac{1}{\sum_{t\in\tau}\sum_{i,j\in t|i\neq j} w_S(i,\mu)w_S(j,\mu)}$, τ is the partition of agent slots into teams of multi-agent environment μ, $N(\mu) \geq 2$, $|\Pi_e| \geq 1$ and $\exists i, j \in t \in \tau | i \neq j, w_S(i,\mu) > 0$ and $w_S(j,\mu) > 0$.

For cooperative anticipation, the use of a random agent for definition 22 may not work in some cases if there are more than two agents in a team, as cooperation may only take place when all of them cooperate and not only a pair (if a random agent is included, this can be very disruptive). In this case, this definition could be extended to reach the number of agents in the team instead.

When using teams, the position of teammates cooperators is clearly stated, but agents in different teams could also form an alliance to cooperate. In order to calculate this anticipation about alliances, we could extend equation 7.17 by also considering those situations where agent slots in other teams may form an alliance with the evaluated agent. In such a case, definition 24 should include a weight to provide the appropriate importance when cooperation occurs within a team and when it occurs between agent slots of different teams. Another option (and probably easier) would be to calculate this alliance anticipation as a different property, letting the evaluated agent's ability to anticipate teammates actions as we defined in definition 24, and proposing a new definition to calculate the evaluated agent's ability to ally with agents in different teams and anticipate them to improve its results.

Here, we have defined the anticipation with only two agent slots, but competition and cooperation (and possibly alliance) can also appear with three or more agents.

7.5 Secernment

It is an important characteristic for a test to be able to give different results for different evaluated agents. Otherwise, if the results are the same (or very similar) for most evaluated agents, we get little information. In other words, we want tests (and with special attention, the environment and the set of populating agents) to secern, to be discriminative.

7.5.1 Fine and Coarse Discrimination

Although there are many approaches to the idea of discriminating power (e.g. [43]), one simple notion that accounts for this concept quite well is the variance of results. In order to formalise this notion of variance (or number of different values) of the expected result of the set of evaluated agents, we can just compare pairs of values as follows:

Definition 25. The fine discriminating power for evaluated agents π_1 and π_2 interacting in agent slot i in multi-agent environment μ with a set of populating agents Π_o and a weight of agent line-up patterns $w_{\dot{L}}$ is given by:

$$FD_i(\pi_1, \pi_2, \Pi_o, w_{\dot{L}}, \mu) \triangleq \sum_{\dot{l} \in \dot{L}_{-i}^{N(\mu)}(\Pi_o)} w_{\dot{L}}(\dot{l}) \Delta_{\mathbb{Q}}(R_i(\mu[\dot{l} \overset{i}{\leftarrow} \pi_1]), R_i(\mu[\dot{l} \overset{i}{\leftarrow} \pi_2])) \tag{7.18}$$

where $\Delta_{\mathbb{Q}}$ denotes a divergence function for rational numbers, $N(\mu) \geq 1$ and if $N(\mu) > 1$ then $|\Pi_o| \geq 1$ otherwise $|\Pi_o| \geq 0$.

This measures the expected result divergence of two evaluated agents placed both in agent slot i of μ in the same line-up patterns, or in other words, whether the evaluation of two different agents obtains different results for them. If $\Delta_{\mathbb{Q}}$ is some kind of numeric difference (e.g. the absolute difference or the squared difference), then this measure would be similar to some kind of dispersion of expected results (like a variance). If $\Delta_{\mathbb{Q}}(a, b) = 1 - \delta(a, b)$ (with δ being the Kronecker delta function) we have that this measures the number of times two different evaluated agents score differently.

We can generalise this for a set of evaluated agents Π_e and aggregate for all agent slots:

Definition 26. The fine discriminating power for a set of evaluated agents Π_e with associated weight w_{Π_e} in multi-agent environment μ with weight of agent slots w_S, with a set of populating agents Π_o and a weight of agent line-up patterns $w_{\dot{L}}$ is given by:

$$\begin{gathered} FD(\Pi_e, w_{\Pi_e}, \Pi_o, w_{\dot{L}}, \mu, w_S) \triangleq \eta_{\Pi^2} \sum_{\pi_1, \pi_2 \in \Pi_e | \pi_1 \neq \pi_2} w_{\Pi_e}(\pi_1) w_{\Pi_e}(\pi_2) \sum_{i=1}^{N(\mu)} w_S(i, \mu) \times \\ \times FD_i(\pi_1, \pi_2, \Pi_o, w_{\dot{L}}, \mu) \end{gathered} \tag{7.19}$$

where $N(\mu) \geq 1$, η_{Π^2} normalises the formula with $\eta_{\Pi^2} = \frac{1}{\sum_{\pi_1, \pi_2 \in \Pi_e | \pi_1 \neq \pi_2} w_{\Pi_e}(\pi_1) w_{\Pi_e}(\pi_2)}$, $|\Pi_e| \geq 2$ and $\exists \pi_1, \pi_2 \in \Pi_e | \pi_1 \neq \pi_2$, $w_{\Pi_e}(\pi_1) > 0$ and $w_{\Pi_e}(\pi_2) > 0$.

Being able to discriminate in terms of pair of evaluated agents for each line-up pattern in an environment can be generalised with the overall result of a social intelligence measure (Υ), namely:

Definition 27. The coarse discriminating power for the evaluated agents π_1 and π_2 interacting in agent slot i in multi-agent environment μ with a set of populating agents Π_o and a weight of agent line-up patterns $w_{\dot{L}}$ is given by:

$$CD_i(\pi_1, \pi_2, \Pi_o, w_{\dot{L}}, \mu) \triangleq \Delta_{\mathbb{Q}}(\Upsilon_i(\pi_1, \Pi_o, w_{\dot{L}}, \mu), \Upsilon_i(\pi_2, \Pi_o, w_{\dot{L}}, \mu)) \tag{7.20}$$

where $\Delta_{\mathbb{Q}}$ denotes a divergence function for rational numbers and $\Upsilon_i(\pi, \Pi_o, w_{\dot{L}}, \mu)$ denotes a measure of the social intelligence of π interacting in agent slot i of multi-agent environment μ with the set of populating agents Π_o with associated line-up pattern weight $w_{\dot{L}}$. As an example of Υ we could use the one presented in equation 5.3 (with Π_o as Π).

We generalise this for a set of evaluated agents Π_e and aggregate for all agent slots:

Definition 28. The coarse discriminating power for a set of evaluated agents Π_e with associated weight w_{Π_e} in multi-agent environment μ with weight of agent slots w_S, with a set of populating agents Π_o and a weight of agent line-up patterns $w_{\dot{L}}$ is given by:

$$\begin{aligned} CD(\Pi_e, w_{\Pi_e}, \Pi_o, w_{\dot{L}}, \mu, w_S) \triangleq \eta_{\Pi^2} \sum_{\pi_1,\pi_2 \in \Pi_e | \pi_1 \neq \pi_2} w_{\Pi_e}(\pi_1) w_{\Pi_e}(\pi_2) \sum_{i=1}^{N(\mu)} w_S(i,\mu) \times \\ \times CD_i(\pi_1, \pi_2, \Pi_o, w_{\dot{L}}, \mu) \end{aligned} \tag{7.21}$$

where $N(\mu) \geq 1$, η_{Π^2} normalises the formula with $\eta_{\Pi^2} = \frac{1}{\sum_{\pi_1,\pi_2 \in \Pi_e | \pi_1 \neq \pi_2} w_{\Pi_e}(\pi_1) w_{\Pi_e}(\pi_2)}$, $|\Pi_e| \geq 2$ and $\exists \pi_1, \pi_2 \in \Pi_e | \pi_1 \neq \pi_2$, $w_{\Pi_e}(\pi_1) > 0$ and $w_{\Pi_e}(\pi_2) > 0$.

In both *fine* and *coarse* discrimination the goal is to check whether two evaluated agents obtain similar results. The difference resides at the level we check this similarity. While the *fine* discrimination checks the similarity for each line-up pattern, the *coarse* discrimination is more oriented to check the overall similarity.

7.5.2 Strict Total and Partial Grading

A multi-agent environment and a set of populating agents being discriminative when comparing a set of evaluated agents does not mean that there is a gradation or order between the results for this set of evaluated agents. For instance, if we have three agents π_1, π_2 and π_3 that we want to evaluate in a competitive environment with two agent slots divided into two teams, and we get that π_1 scores better when interacts with π_2, π_2 scores better when interacts with π_3 and π_3 scores better when interacts with π_1, then there is no way to establish a gradation for these three agents. Idealistically, we would like to have a strict total order, but this is unrealistic for many environments and sets of populating agents.

So the idea we will pursue is to evaluate how close an environment and a set of populating agents are to this ideal situation, from the perspective of the expected results of the evaluated agents (without using an aggregated rating system[3]):

Definition 29. The strict total grading quality for the evaluated agents π_1, π_2 and π_3 interacting in agent slots i and j (with i $\neq$ j) in multi-agent environment μ with a set of populating agents Π_o and a weight of agent line-up patterns $w_{\dot{L}}$ is given by:

$$STG_{i,j}(\pi_1, \pi_2, \pi_3, \Pi_o, w_{\dot{L}}, \mu) \triangleq \sum_{\dot{l} \in \dot{L}_{-i,j}^{N(\mu)}(\Pi_o)} w_{\dot{L}}(\dot{l}) STO_{i,j}(\pi_1, \pi_2, \pi_3, \dot{l}, \mu) \tag{7.22}$$

where $N(\mu) \geq 2$, if $N(\mu) > 2$ then $|\Pi_o| \geq 1$ otherwise $|\Pi_o| \geq 0$ and $STO_{i,j}(\pi_1, \pi_2, \pi_3, \dot{l}, \mu)$ (where $\dot{l}$ has all its elements instantiated except positions i and j) is 1 if there is a permutation

[3]A common approach is to create a rating when we have many experiments, as done with sport ratings, such as the ELO rating [25] in chess.

of the three evaluated agents such that there is a strict total order in their expected results when placed by pairs in $\dot{l}$ interacting in μ in agent slots i and j, and 0 otherwise. Formally, it is 1 iff there is a permutation (π'_1, π'_2, π'_3) such that: $R_i(\mu[\dot{l} \overset{i,j}{\leftarrow} \pi'_1, \pi'_2]) < R_j(\mu[\dot{l} \overset{i,j}{\leftarrow} \pi'_1, \pi'_2])$, $R_i(\mu[\dot{l} \overset{i,j}{\leftarrow} \pi'_2, \pi'_3]) < R_j(\mu[\dot{l} \overset{i,j}{\leftarrow} \pi'_2, \pi'_3])$ and $R_i(\mu[\dot{l} \overset{i,j}{\leftarrow} \pi'_1, \pi'_3]) < R_j(\mu[\dot{l} \overset{i,j}{\leftarrow} \pi'_1, \pi'_3])$. For instance, if we have three evaluated agents a, b and c in a multi-agent environment μ and agent line-up pattern $\dot{l}$ with positions i and j not instantiated, and their expected results when placed by pairs in $\dot{l}$ shows us that $b < a$, $a < c$ and $b < c$, then we have $STO_{i,j}(a, b, c, \dot{l}, \mu) = 1$ with the permutation (b, a, c).

If equation 7.22 is equal to 1, intuitively this would mean that there exists an order between the three agents, and when two of them interact together, the one with the better position would always obtain a better result. Now, we generalise this for a set of evaluated agents Π_e and aggregate for all combinations of pairs of agent slots:

Definition 30. The strict total grading quality for a set of evaluated agents Π_e with associated weight w_{Π_e} in multi-agent environment μ with weight of agent slots w_S, with a set of populating agents Π_o and a weight of agent line-up patterns $w_{\dot{L}}$ is given by:

$$STG(\Pi_e, w_{\Pi_e}, \Pi_o, w_{\dot{L}}, \mu, w_S) \triangleq \eta_{\Pi^3} \sum_{\pi_1,\pi_2,\pi_3 \in \Pi_e | \pi_1 \neq \pi_2 \neq \pi_3} w_{\Pi_e}(\pi_1) w_{\Pi_e}(\pi_2) w_{\Pi_e}(\pi_3) \eta_{S_1^2} \sum_{i=1}^{N(\mu)} w_S(i,\mu) \times$$
$$\times \left(\sum_{j=1}^{i-1} w_S(j,\mu) STG_{i,j}(\pi_1, \pi_2, \pi_3, \Pi_o, w_{\dot{L}}, \mu) + \sum_{j=i+1}^{N(\mu)} w_S(j,\mu) STG_{i,j}(\pi_1, \pi_2, \pi_3, \Pi_o, w_{\dot{L}}, \mu) \right) \tag{7.23}$$

where η_{Π^3} normalises the formula with $\eta_{\Pi^3} = \frac{1}{\sum_{\pi_1,\pi_2,\pi_3 \in \Pi_e | \pi_1 \neq \pi_2 \neq \pi_3} w_{\Pi_e}(\pi_1) w_{\Pi_e}(\pi_2) w_{\Pi_e}(\pi_3)}$, $\eta_{S_1^2}$ normalises the formula with $\eta_{S_1^2} = \frac{1}{\sum_{i=1}^{N(\mu)} w_S(i,\mu) \left(\sum_{j=1}^{i-1} w_S(j,\mu) + \sum_{j=i+1}^{N(\mu)} w_S(j,\mu) \right)}$, $|\Pi_e| \geq 3$, $\exists \pi_1, \pi_2, \pi_3 \in \Pi_e | \pi_1 \neq \pi_2 \neq \pi_3, w_{\Pi_e}(\pi_1) > 0, w_{\Pi_e}(\pi_2) > 0$ and $w_{\Pi_e}(\pi_3) > 0$, $N(\mu) \geq 2$ and $\exists i, j | 1 \leq i \leq N(\mu), 1 \leq j \leq N(\mu), i \neq j, w_S(i,\mu) > 0$ and $w_S(j,\mu) > 0$.

The previous definition considers all pairs of agent slots, even if both agent slots of the pair pertain to the same team, which by definition 8 will make STG to become 0 for such pair. If such situation wants to be avoided, we could just consider those pairs that its agent slots do not pertain to the same team.

Definition 30 only considers strict total orders, and is useful to determine whether we can find a strict total order for the evaluated agents. However, this does not say much about the existence of grading 'inconsistencies', such as non-discriminative cases such as $\pi_1 = \pi_2$, $\pi_2 = \pi_3$ and $\pi_1 = \pi_3$ which, for the above definition, are considered in the same way as not ordering cases such as $\pi_1 > \pi_2$, $\pi_2 > \pi_3$ and $\pi_1 < \pi_3$. In order to distinguish these cases, we can give a new definition as follows:

Definition 31. The partial grading quality for a set of evaluated agents Π_e with associated weight w_{Π_e} in multi-agent environment μ with weight of agent slots w_S, with a set of populating agents Π_o and a weight of agent line-up patterns $w_{\dot{L}}$ is defined as in definition 30 with the use of a partial order with $\leq$ instead of a strict total order with $<$. It is denoted by PG.

If STG and PG are high, this means that a derivation of a rating is highly consistent to what we see when using evaluated agents from Π_e on agent slots i and j. A very similar property is known as monotonicity in [50, sec. 5], [49] showing an agent set for the matching pennies game that is non-monotonic. Nonetheless, a partial order can still be constructed for the agent set of all finite state machines for this game [53].

The existence of a meaningful rating allows for subselections of Π_o according to this rating, which can be used to furbish the tests with high-rank agents that can lead to more sophisticated social environments (making it more or less difficult depending on whether it is used for the same team or for opponents).

7.6 Validity

Validity is the most important property of a cognitive test in psychometrics. In our context, the validity of a definition means that it accounts for the notion we expect it to grasp. For instance, if we say that a given definition of Υ measures social intelligence but it actually measures arithmetic abilities then the definition is not valid. Ultimately, this depends on the choice of Π_o and M as the core of Υ (e.g. equation 5.4).

Poor validity may have two sources (or may appear in two different variants): a definition may be too specific (it does not account for all the abilities the notion is thought to consider) or it is too general (it includes some abilities that are not part of the notion to be measured). In other words, the measure should account for *all, but not more,* of the concept it tries to represent. We refer to these two issues of validity as the generality and the specificity of the measure. While validity is not usually seen as an instrumental property, we have to say that the choices of Π_o and M may both have generality and specificity, which eventually can compensate, but could lead to a test that is not very effective. That means that we should try to find proper choices such that they fit the concept we want to measure precisely.

Regarding generality, we should be careful about the use of very restrictive choices for Π_o and M. It could be possible to find a single multi-agent environment that meets all the properties seen in the previous sections. However, using just one environment is prone to specialisation, as usual in many AI benchmarks. For instance, if we use a particular maze as an environment with a set of particular populating agents, then we can have good scores by evaluating a very specialised agent for this situation, which may be unable to succeed in other mazes or problems. For instance, chess with current chess players is an example where a specialised system (e.g. Deep Blue) is able to score well, while it is clearly useless for other problems. A similar over-specialisation may happen if the populating agent set is too small. This is usual in biology, where some species specialise for predating (or establishing a symbiosis) with other species.

Consequently, the multi-agent environment set and the populating agent set must be general enough to avoid that some predefined or hardwired policies could be optimal for these sets. This is the key issue of a (social) *intelligence* test; it must be as general as possible. We need to choose a diverse multi-agent environment set. One possibility is to consider all multi-agent environments (as done in [69, 44] with single-agent environments), and another is to find a multi-agent environment class that is sufficiently representative (as attempted in [41] with single-agent environments).

Similarly, we need to consider a set of populating agents that leads to a diversity in line-up patterns. This set should incorporate many different types of agents: random agents, agents with some predetermined policies, agents that are able to learn, human agents, agents with low social intelligence, agents with high social intelligence, etc. The set of all possible agents (either artificial or biological) is known as *machine kingdom* in [45, 46] and raises many questions about the feasibility of any test considering this astronomically large set. Also, there are doubts about what the weight for this universal set should be when forming line-up patterns (i.e. $w_{\dot{L}}$). Instead, some representative kinds of agents could be chosen to populate the environments. In this way, we could aim at social intelligence relative to a smaller (and well-defined) set of populating agents, possibly specialising the definition by limiting the resources, the program size [42] or the intelligence of these agents [48].

Regarding specificity, it is equally important for the measurement to only include those environments and populating agents that really reflect what we want to measure. For instance, it is desirable that the evaluation of an ability is done in an environment where no other abilities are required, or in other words, we want that the environment evaluates the ability in isolation. Otherwise, it will not be clear which part of the result comes from the ability to be evaluated, and which part comes from other abilities. Although it is very difficult to avoid any contamination, the idea is to ensure that the role of these other abilities are minor, or are taken for granted for all the agents we want to evaluate.

The properties of dependency (interactivity, non-neutralism) and anticipation (both competitive and cooperative) seen in previous sections have been included for the sake of specificity. We are certainly not interested in non-social environments as this would contaminate the measure with other abilities. In fact, one of the recurrent issues in defining and measuring social intelligence is to be specific enough to distinguish it from general intelligence.

Unlike all other properties in this chapter, validity precisely accounts for how well the definition reflects the natural or intuitive notion that is to be measured.

The assurement about validity must come then from the use of a formal definition (e.g. definition 12), with a meaningful instantiation for the sets of multi-agent environments and populating agents, and also from the experimental results that can be obtained through the tests derived from the definition. Note that in psychometrics there is usually a lack of a proper definition of the cognitive abilities of interest (e.g. psychometrics has not presented a representational definition of intelligence), so validity is applied to a test and not to the definition of a cognitive ability. In fact, the concept is frequently derived from the test, as has happened with the modern view of intelligence, as ‘the ability measured by IQ tests’.

7.7 Reliability

Another key issue in psychometric tests is the notion of reliability, which means that the measurement is close to the actual value. Note that this is different to validity, which refers about the true identification or definition of the actual value. In other words, if we assume validity, i.e. that the definition is correct[4], reliability refers to the quality of the measurement with respect to the actual value. More technically, if the actual value of π for an ability ϕ is v then we want a test to give a value which is close to v. The cause of the divergence may be systematic (bias), non-systematic (variance) or both.

First, we need to realise that reliability applies to tests (e.g. definition 13). Reliability is then defined by considering that a test can be repeated many times, so becoming a random variable that we can compare to the true value. Formally:

Definition 32. Given a definition of a cognitive ability Υ and a test over it $\hat{\Upsilon}$, the test error is given by:

[4]The lack of a proper definition for many abilities makes reliability refer to the quality of the result of a single application of the test in comparison to the idealised average result if the test could be repeated indefinitely.

$$TE(\hat{\Upsilon}) \triangleq Mean((\hat{\Upsilon} - \Upsilon)^2) \tag{7.24}$$

where the mean is calculated over the repeated application of the test (to one subject or more subjects).

The reason for defining test error as the mean *squared* error (and not an absolute error) is a customary choice in many measures of error, as we can decompose it into the squared bias $(Mean(\hat{\Upsilon}) - \Upsilon)^2$ and the variance of the error $Var(\hat{\Upsilon} - \Upsilon)$. These values can be calculated for just one evaluated agent or for all evaluated agents.

Following the definition of test at section 5.5, if the bias is not zero this means that the procedure to sample the exercises and/or the number of time steps is inappropriate, and the choices for p_{Π_o} (p_Π), p_M, p_S and p_K in definition 13 must be revised. If there is a high variance, this suggests that the number of exercises n_E is too small and we need more (i.e. exercises in a tests) to get a less volatile result, or that the exercises run for a very short time.

In this sense, note that some of the properties studied in previous sections can hold for Υ but may be significantly different for unreliable tests (i.e. approximations) $\hat{\Upsilon}$.

The reliability $Rel(\hat{\Upsilon})$ can be defined as a decreasing function over $TE(\hat{\Upsilon})$, such as $Rel(\hat{\Upsilon}) = e^{-TE(\hat{\Upsilon})}$. The estimation of $TE(\hat{\Upsilon})$ or $Rel(\hat{\Upsilon})$ depends on knowing the true value of Υ. This is not possible in practice for most environments, so Υ will need to be estimated for large samples and compared with an actual test (working with a small sample). Because of the difficulties of estimating this, in what follows we will just give a qualitative assessment. In order to determine the reliability of a test, we will obtain various samples of exercises and calculate whether the results obtained in such exercises correlate with their expected results. For doing this, we determine a test is reliable if such correlation (e.g. a Pearson Correlation) is equal to or greater than 0.7.[5]

7.8 Efficiency

This property refers to how efficient a test is in terms of the (computational) time required to get a reliable score. It is easy to see that efficiency and reliability are opposed. If we were able to perform an infinite number of infinite exercises, where exercises are selected without bias, then we would have $\hat{\Upsilon} = \Upsilon$, with perfect reliability, as we would exhaust Π_o and M. However, as we usually try to make tests not only finite but more efficient, we lose reliability because of the sampling procedure. If done properly, it is usually the variance component of the reliability decomposition that is affected if we keep the bias close to 0 even with very low values for the number of exercises (n_E in definition 13).

Efficiency can be defined as a ratio between the reliability and the time taken by the test (depending mostly on n_E and p_K in definition 13, but also on p_{Π_o} (p_Π), p_M and p_S).

Definition 33. Given a definition of a cognitive ability Υ and a test over it $\hat{\Upsilon}$, the efficiency is given by:

$$Eff(\hat{\Upsilon}) \triangleq Rel(\hat{\Upsilon})/Time(\hat{\Upsilon}) \tag{7.25}$$

where *Time* is the average time taken by test $\hat{\Upsilon}$. *Time* can be measured as physical (real) time or as computational time (steps).

[5] [84] suggests 0.7 as an acceptable reliability coefficient.

While this is the way it should be measured, the big issue is how to choose multi-agent environments and populating agents such that a high efficiency is attained. Clearly, if the selected environments are insensitive to agents' actions or require too many actions to affect rewards, then this will negatively affect efficiency. As we are interested in social abilities, interactivity and non-neutralism must be high, as otherwise most time steps will be useless to get information about the evaluated agent. This of course includes cases where the evaluated agent is stuck or bored because its opponents (or teammates) are too good or too bad, or the environment leads it to heaven or hell situations where its actions are almost irrelevant.

Naturally, a way of making tests more efficient is by the use of adaptive tests, as in computerised adaptive testing. Here we do not explore this possibility as our definition of test in section 5.5 is not adaptive (for adaptive versions of universal tests, the reader is referred to [44, 46]).

Similarly to reliability, we will just give a qualitative assessment of efficiency. For doing this, we determine the efficiency of the test based on the number of repetitions that is needed to obtain a reliable test. If such number of repetitions is lower than 20, we say the test is efficient.

7.9 Team Symmetry

In game theory, a symmetric game is a game where the payoffs for playing a particular strategy depend only on the other strategies employed by the rest of agents, not on who is playing each strategy. This property is very useful for evaluating purposes, as the results would be independent of the position of the evaluated agent in the environment.

When using teams, this definition of symmetry must be reconsidered. The previous definition means that for each pair of line-ups with the same agents but in different order, the agents maintain their previous results. But with the inclusion of teams this definition is not appropriate. For example, using a multi-agent environment with the partition of agent slots $\tau = \{\{1,2\},\{3,4\}\}$ and line-up $l = (\pi_1, \pi_2, \pi_3, \pi_4)$, we have that agents π_1 and π_2 must both obtain the same result, as π_3 and π_4 as well. Following the definition and switching the positions of π_2 and π_3 we obtain line-up $l' = (\pi_1, \pi_3, \pi_2, \pi_4)$, which now means that agents π_1 and π_3 must have the same results (since they are now in the same team) while maintaining their previous results, as π_2 and π_4 as well. This situation can only occur when all agent slots (and therefore teams) obtain equal results.

Instead, we extend this definition of symmetry to include teams. First, we denote by $\sigma(l)$ the set of all possible line-ups that we can obtain by permuting the positions of the agents of line-up l. This set corresponds with the one used in game theory to define symmetry. To adapt this set to include teams, we must select a subset of line-ups from $\sigma(l)$ respecting the teams defined in τ. We denote this subset with $\sigma(l,\tau)$, where we only select line-ups from $\sigma(l)$ if original teams are maintained. Following the example, line-up l' is not included in $\sigma(l,\tau)$ since π_1 and π_3 from l' were not in the same team in l (as π_2 and π_4 as well). However, $l'' = (\pi_3, \pi_4, \pi_2, \pi_1)$ is included in $\sigma(l,\tau)$, since both pair of agents (π_1, π_2) and (π_3, π_4) are still in the same team. From here, we define symmetry for a multi-agent environment that uses teams as follows:

Definition 34. We say a multi-agent environment μ is *team symmetric* if and only if every team in τ has the same number of elements and:

$$\forall i, K, \Pi, l \in L^{N(\mu)}(\Pi), l' \in \sigma(l,\tau) : R_i^K(\mu[l]) = R_{i'}^K(\mu[l']) \tag{7.26}$$

where i' represents the position of agent $l_{i:i}$ in l' and whatever the utility function used to calculate agents' results.

Note that we impose that every set in τ must have the same number of elements. This is because we only consider multi-agent environments to be team symmetric if we can evaluate an agent in every agent slot and obtain the same result. Having teams with different number of elements will not allow us to do this.

This definition now fits our goal of team symmetry. But too few multi-agent environments will fit this definition because it is too restrictive. However, we could divide this definition of team symmetry into two parts depending on the relation between the agent slots.

For the first part we look at the relation between the agent slots within each team:

Definition 35. We say a multi-agent environment is *Intra-Team Symmetric* when the agents within every team can be swapped without affecting their results.

This kind of team symmetry will allow us to evaluate the agents in an environment without taking into account their positions within their teams.

For the second part we look at the relation between the agent slots in different teams:

Definition 36. We say a multi-agent environment is *Total Inter-Team Symmetric* if every pair of teams has the same number of elements and they can be swapped without affecting their results.

This kind of team symmetry will allow us to evaluate a team of agents in a multi-agent environment without taking into account in which set of τ it is situated.

Definition 34 corresponds with both Intra-Team and Total Inter-Team Symmetry, where every team of agents can be located in every set of τ and in different order, maintaining their performance expectation.

However, although a multi-agent environment whose teams do not have the same number of elements is not ideal for evaluating purposes, some partial symmetry among its teams may still exist:

Definition 37. We say a multi-agent environment is *Partial Inter-Team Symmetric* if not all teams have the same number of elements and every pair of teams having the same number of elements can be swapped without affecting their results.

Team symmetry is not a necessary condition for social behaviours, but it is a very practical one for measurement as the result does not depend on the agent slot we use to evaluate and all agent slots are useful for evaluating the same ability.

7.10 Summary of Properties

In tables 7.1, 7.2 and 7.3 we can see a summary of all previous properties. Tables 7.1 and 7.2 show the *quantitative* properties, while table 7.3 shows the *qualitative*[6] properties. This completes our picture jointly with figure 7.1.

The quantitative properties are divided into two kinds. The properties whose values range from 0 to 1 (i.e. table 7.1) determine the percentage of compliance that the multi-agent environment μ and the set of populating agents Π_o have about this property when evaluating

[6]Some of them can in principle be quantified, but here we only give a qualitative assessment.

Property model	Meaning	Lowest value	Highest value
$AD(\Pi_e, w_{\Pi_e}, \Pi_o, w_{\dot{L}}, \mu, w_S)$ Social:Social dependency:Interactivity	The **action dependency** of the evaluated agents on the agent line-up pattern they encounter.	The evaluated agents do not take into account other agents' actions in their behaviour.	The evaluated agents behave completely depending on the other agents' actions.
$RD(\Pi_e, w_{\Pi_e}, \Pi_o, w_{\dot{L}}, \mu, w_S)$ Social:Social dependency:Non-neutralism	The **reward dependency** of the evaluated agents on the agent line-up pattern they encounter.	Each evaluated agent obtains the same reward sequence independently of the agent line-up pattern.	The agents in the agent line-up pattern directly exercise influence on the reward sequences of each evaluated agent.
$FD(\Pi_e, w_{\Pi_e}, \Pi_o, w_{\dot{L}}, \mu, w_S)$ Instrumental:Secernment:Discrimination	The **fine discrimination** between pairs of evaluated agents when interacting with the same agent line-up patterns.	Every evaluated agent obtains the same expected result for each agent line-up pattern.	Every evaluated agent obtains different expected results for each agent line-up pattern.
$CD(\Pi_e, w_{\Pi_e}, \Pi_o, w_{\dot{L}}, \mu, w_S)$ Instrumental:Secernment:Discrimination	The **coarse discrimination** between pairs of evaluated agents.	Every evaluated agent obtains the same (social) intelligence value.	Every evaluated agent obtains different (social) intelligence values.
$STG(\Pi_e, w_{\Pi_e}, \Pi_o, w_{\dot{L}}, \mu, w_S)$ Instrumental:Secernment:Grading	The **strict total grading** measures the level of strict grading ($<$) between the evaluated agents.	There is no strict total order between any trio of evaluated agents.	There is a strict total order between all the evaluated agents.
$PG(\Pi_e, w_{\Pi_e}, \Pi_o, w_{\dot{L}}, \mu, w_S)$ Instrumental:Secernment:Grading	The **partial grading** measures the level of partial grading ($\leq$) between the evaluated agents.	There is no partial order between any trio of evaluated agents.	There is a partial order between all the evaluated agents.

Table 7.1: Summary of the quantitative property models, whose values range from 0 to 1, about a multi-agent environment μ with agent slot probability w_S, evaluated agent set Π_e with weights w_{Π_e} and populating agent set Π_o with agent line-up pattern weights $w_{\dot{L}}$. For each property model the table shows its arguments, a brief description and a description of the situations when their lowest and highest values occur.

a set of agents Π_e. Therefore, the lower the value the worse the testbed is in regard to this property, and the higher the value the better. On the contrary, the properties whose values range from -1 to 1 (i.e. table 7.2) must not be interpreted in the same way. Instead, these

Property model	Meaning	Lowest value	Highest value
$SRD(\Pi_e, w_{\Pi_e}, \Pi_o, w_{\dot{L}}, \mu, w_S)$ Social:Social dependency:Non-neutralism	The **slot result dependency** measures how competitive or cooperative the multi-agent environment is.	The multi-agent environment is completely competitive.	The multi-agent environment is completely cooperative.
$AComp(\Pi_e, w_{\Pi_e}, \Pi_o, w_{\dot{L}}, \mu, w_S)$ Social:Mind modelling:Competitive	The benefit of **anticipating competitive** agents.	Every evaluated agent completely fails at anticipating competitive agents' behaviour.	Every evaluated agent perfectly anticipates competitive agents' behaviour.
$ACoop(\Pi_e, w_{\Pi_e}, \Pi_o, w_{\dot{L}}, \mu, w_S)$ Social:Mind modelling:Cooperative	The benefit of **anticipating cooperative** agents.	Every evaluated agent completely fails at anticipating cooperative agents' behaviour.	Every evaluated agent perfectly anticipates cooperative agents' behaviour.

Table 7.2: Summary of the quantitative property models, whose values range from -1 to 1, about a multi-agent environment μ with agent slot probability w_S, evaluated agent set Π_e with weights w_{Π_e} and populating agent set Π_o with agent line-up pattern weights $w_{\dot{L}}$. For each property model the table shows its arguments, a brief description and a description of the situations when their lowest and highest values occur.

Property	Meaning
Boundedness Instrumental:Technical	Rewards and weights are bounded, so the social intelligence definition and test and all quantitative property models are also bounded.
Validity Univocal:Correctness	The test evaluates what it is supposed to evaluate and nothing more.
Reliability Instrumental:Testing quality	The result of an agent's ability in the test is close to its actual value.
Efficiency Instrumental:Testing quality	Decreases with the amount of (computational) time used in the test to obtain a reliable score.
Team Symmetry Instrumental:Technical	Desirable condition to simplify the measurement in such a way that only one agent slot has to be used to evaluate an agent.

Table 7.3: Summary of the qualitative properties (or for which we give a qualitative assessment) about a social intelligence testbed, providing a brief description of what each property represents.

properties measure to which kind of type the testbed is more focussed on, and not a level of accomplishment or quality.

The set of properties we propose here provides key information about the testbed we are analysing. First, we can measure the influence that a set of populating agents Π_o produces on the set of agents we want to evaluate Π_e. Second, we can analyse to what extent the anticipation abilities are useful for the set of agents we want to evaluate Π_e interacting with a set of populating agents Π_o. Third, we can determine whether cooperation or competition is given more importance in the testbed. Fourth, we estimate the discriminative power that the testbed has for the evaluation of different agents Π_e. Fifth, the grading power of the testbed indicates how effective it is to rank the agents we want to evaluate Π_e. And sixth, we have some instrumental properties that are convenient to convert the definition into a practical test.

We presented the properties in such a way that the multi-agent environment and both sets of agents (Π_e and Π_o) are included. These properties can also analyse the testbed without including the set of evaluated agents that we want to evaluate by letting Π_e unfixed, i.e. by not specifying a particular set of evaluated agents. This could be done by letting Π_e be the set of all possible agents that will be evaluated with this testbed. Another more practical option would be to use instead a representative sample of the agents that will be evaluated. A third (and easier) option would be to use the set of populating agents also as the set of evaluated agents. Although this last set of evaluated agents will not represent accurately the agents that are pretended to be evaluated, it could be used in order to obtain an idea about the properties of the testbed. Similarly, the properties of a multi-agent environment can also been calculated by letting both sets of agents (Π_e and Π_o) unfixed.

Note that the properties we present need to specify several functions, such as the correlation function, the utility function or divergence functions between distributions, so depending on the functions we use, the same testbed could obtain different values for the properties. When trying to assess a realistic testbed, we would need to decide what functions are most appropriate before obtaining any conclusion about the testbed.

Finally, although possible, the calculation of the properties for complex scenarios may be prohibitively expensive. Instead, we could just approximate them. One possibility would be to not consider all the pairs of line-up patterns and agent slots, but make a representative sample of them. A similar solution could be used to approximate the expected result of an agent in complex stochastic scenarios, such as, e.g. a Monte Carlo approximation (section 2.5).

The property models we present in this chapter can be used to characterise social intelligence testbeds, letting us to make a comparison between them and allowing us to identify their strengths and weaknesses.

Chapter 8

Characterising Several Multi-Agent and Social Scenarios

Many games and multi-agent environments have been proposed as testbeds to evaluate the performance of an agent (or group of agents) interacting with other agents [90, 138, 103, 143]. Typically, these games and multi-agent environments are created or selected to represent a specific problem or family of problems to analyse or solve, although they are not specifically designed to evaluate social intelligence itself. In this chapter, we use our properties to characterise some of them and analyse their suitability to be used in a test to evaluate social intelligence. Since we are interested in developing social intelligence tests, it is first mandatory to evaluate whether these other previous testbeds could be valid as they are (or with minor modifications). Otherwise, they can still be a good source of inspiration to figure out new multi-agent environment classes by reusing some of their ideas or hybridising some of their features.

We would have liked to explore many games and multi-agent environments, but we can just practically do a selection of some of the most common and representative in the area of multi-agent systems, game theory and (social) computer games. We will focus on some environments whose specification is complete, so we can analyse them with respect to the properties seen in the previous chapter. In particular, the environments that we analyse in this chapter are presented in section 2.1, which are: matching pennies, prisoner's dilemma, predator-prey (a pursuit game), Pac-Man and RoboCup Soccer.

8.1 Graphical Analysis for the Properties

Before starting with the environments, we introduce some indicators and a graphical representation that we illustrate on a figurative multi-agent environment.

In order to assess interactivity, non-neutralism, anticipation and other properties for a multi-agent environment μ, we need to specify the evaluated agent set Π_e with associated weight w_{Π_e}, the agent set Π_o that populates the environment, line-up pattern weights $w_{\dot{L}}$ and agent slot weights w_S. One choice for Π_e and Π_o would be to consider any possible agent that is expressible using a given policy language. This, however, would make the calculation of most properties difficult (if not impossible). A better approach would be to use a (representative) sample of all agents or a sample of a meaningful class. Instead of that, and in order to give a more general picture of the environment itself, we will show the range of values that each property can have (for every possible agent set), and how much this range can be restricted (for better or worse) depending on which Π_o we select. In fact, when evaluating a set of agents Π_e in a certain setup, we must provide which set of agents Π_o populates the environment but, when assessing the properties of a testbed, we could let Π_o unfixed in order to better evaluate how the environment behaves for such Π_e. Finally, the use of different weights can lead to different ranges for the properties but, in order to simplify our calculations, in what follows we assume uniform unit weights for evaluated agents w_{Π_e}, line-up patterns $w_{\dot{L}}$ and agent slots w_S. The same happens with which utility function we use to calculate an agent's result (section 5.3 on page 44), the correlation function we use to calculate the correlation on rewards of agents in different agent slots (section 7.3.2 in definition 20), and divergence functions we use to calculate the divergence between actions and rewards distributions (sections 7.2, 7.3 and 7.5.1). In what follows we assume an average of rewards as the utility function, any correlation function where two equal and two inverse values obtain 1 and -1 respectively, and we use $\Delta_{\mathbb{Q}}(a', b')$ instead of $\Delta_{\mathbb{Q}^+}(a, b)$ where a' and b' are the expected results of a and b respectively, and $\Delta_{\mathcal{A}^+}(a, b)$ and $\Delta_{\mathbb{Q}}(a, b)$ return 0 if a and b are equal and 1 otherwise.

We divided the properties into three types. In the first type we have the properties which have a quantitative value that can range between 0 and 1. In the second type we have the properties which have a quantitative value that can range between -1 and 1. And in the third type are the properties for which we provide a qualitative value.

For the first two types of properties we calculate the range that each property can have in an environment. For this, we need to calculate the lowest and highest values that this range can have for each quantitative property *Prop*. To achieve this, we select Π_e and Π_o (from the set of all possible Π_e and Π_o such that *Prop* is defined) that obtain the lowest and highest values respectively. We define *General* as follows:

Definition 38. We denote *General* to be the range of values from $General_{\min}(Prop, \mu)$ to $General_{\max}(Prop, \mu)$, where:

$$General_{\min}(Prop, \mu) \triangleq \min_{\Pi_e, \Pi_o} Prop(\Pi_e, w_{\Pi_e}, \Pi_o, w_{\dot{L}}, \mu, w_S) \tag{8.1}$$

$$General_{\max}(Prop, \mu) \triangleq \max_{\Pi_e, \Pi_o} Prop(\Pi_e, w_{\Pi_e}, \Pi_o, w_{\dot{L}}, \mu, w_S) \tag{8.2}$$

where the weights for evaluated agents w_{Π_e}, agent line-up patterns $w_{\dot{L}}$ and agent slots w_S are uniform unit weights and the pair $\langle \Pi_e, \Pi_o \rangle$ is selected (from the set of all possible $\langle \Pi_e, \Pi_o \rangle$ such that *Prop* is defined) to minimise/maximise the values of a quantitative property *Prop* for multi-agent environment μ.

For the first type of properties *Prop*, we can select some set of populating agents Π_o to obtain a situation where *General* is restricted in such a way that $General_{\max}(Prop, \mu)$ decreases. In particular, we are interested in the setup with the "lowest maximum", where Π_o minimises this maximum. We define *Left* as follows:

Definition 39. We denote *Left* to be the most restricted range of values from $Left_{\min}(Prop, \mu)$ to $Left_{\max}(Prop, \mu)$ that we can obtain when Π_o is selected (from the set of all possible Π_o

such that *Prop* is defined) to decrease the values of a quantitative property *Prop* which range is between 0 and 1 for multi-agent environment μ, where:

$$Left_{\min}(Prop, \mu) \triangleq General_{\min}(Prop, \mu) \tag{8.3}$$

$$Left_{\max}(Prop, \mu) \triangleq \min_{\Pi_o} \max_{\Pi_e} Prop(\Pi_e, w_{\Pi_e}, \Pi_o, w_{\dot{L}}, \mu, w_S) \tag{8.4}$$

In the same way, for the first type of properties *Prop* we can select some set of populating agents Π_o to obtain a situation where *General* is restricted in such a way that $General_{\min}(Prop, \mu)$ increases. In particular, we are interested in the setup with the "highest minimum", where Π_o maximises this minimum. We define *Right* as follows:

Definition 40. We denote *Right* to be the most restricted range of values from $Right_{\min}(Prop, \mu)$ to $Right_{\max}(Prop, \mu)$ that we can obtain when Π_o is selected (from the set of all possible Π_o such that *Prop* is defined) to increase the values of a quantitative property *Prop* which range is between 0 and 1 for multi-agent environment μ, where:

$$Right_{\min}(Prop, \mu) \triangleq \max_{\Pi_o} \min_{\Pi_e} Prop(\Pi_e, w_{\Pi_e}, \Pi_o, w_{\dot{L}}, \mu, w_S) \tag{8.5}$$

$$Right_{\max}(Prop, \mu) \triangleq General_{\max}(Prop, \mu) \tag{8.6}$$

For the first type of properties, the *General*, *Left* and *Right* ranges become better as long as their minimum and maximum values become higher. If the *Left* range values are lower, this would mean that a bad selection of Π_o is disastrous for the quality of the testbed. If *Right* range values are higher would mean that there is a good selection of Π_o which improves the quality of the testbed. The comparison between *Left* and *Right* with *General* shows us the importance that a good selection for the set of populating agents Π_o has for a property *Prop* in a multi-agent environment μ. As these three ranges become more different, the selection of populating agents Π_o becomes more important in order to provide a better quality for the testbed.

In figure 8.1 we present the properties of a figurative multi-agent environment divided in three sections. The top section represents five quantitative properties of the first type[1] (whose range can be between 0 and 1). The middle section represents the three quantitative properties of the second type (whose range can be between -1 and 1). Finally, the bottom section represents the five qualitative properties.

We can see that each property of the first type has the early mentioned *General*, *Left* and *Right* ranges represented with three bands. The first property (Action Dependency) has a *General* range from 0 to 1, represented with the first band. This is the broadest range that this kind of property can have. This means that this environment can have any value for this property depending on the sets of populating agents Π_o and evaluated agents Π_e. The second band represents its *Left* range, which is equal to $[0, 0]$. In this case, there exists a set of populating agents Π_o that restricts this range to the minimum possible range. The third band represents its *Right* range, which remains from 0 to 1. Now, no set of populating agents Π_o can be selected to restrict this range. In the next four properties we see some other examples for the three ranges that this type of property can have. As we can see in the last property of

[1]Since FD and CD are similar properties, we decided to just calculate the FD property in order to simplify the analysis.

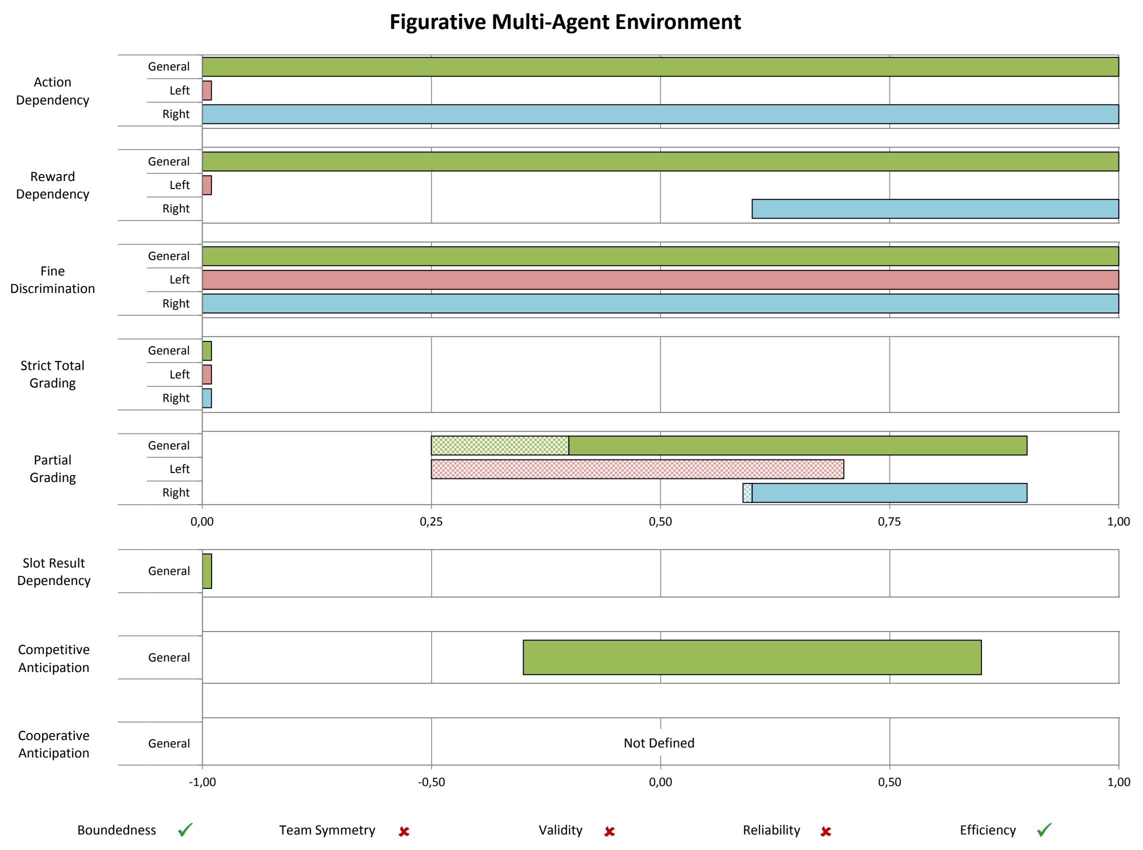


Figure 8.1: Properties for a figurative multi-agent environment. Lighter bands mean the values are not formally calculated, but an estimation is given.

this type (Partial Grading), we use a lighter colour to represent that (part of) a range is not formally calculated, but instead we provide an estimation.

Next, we arrive to the second type of properties. Here the values for the *General* range can be between -1 and 1. Unlike the previous case, we do not provide *Left* and *Right* ranges for this type of properties. This is because these properties represent for which kind of social intelligence the environment is more oriented, so there are not really good or bad ranges.

In the first property of this second type (Slot Result Dependency), we see that the *General* range is equal to $[-1, -1]$, indicating that this environment is purely competitive. The next property shows another example for this type of properties. Meanwhile, the last property has a label with the text "Not Defined". This is because for this environment the property is not defined, so we cannot represent it.

Finally, we arrive to the last type of properties. Here, we denote whether the environment meets the properties or not by using a tick (✓) or cross ($\times$) mark respectively.

To determine the reliability of an environment, we perform 100 exercises where only such environment is used. All exercises have the same configuration except for the line-ups. To obtain a variety of line-ups, each of them is formed with Q-learning agents (section 2.2.1) with all their parameters randomly selected to be uniformly distributed between 0 and 1.

Now that we have explained how we represent the properties, let us start analysing some true multi-agent environments.

8.2 Matching Pennies

The first multi-agent environment we analyse is the matching pennies (section 2.1.1), which can be considered the simplest game in game theory featuring competition. Clearly, in this game, $\tau = \{\{1\}, \{2\}\}$ represents the partition of agent slots into teams, i.e. it has two teams and only one agent slot in each.

Next we discuss this game with respect to the properties seen in chapter 7. We can see a summary of the properties for the matching pennies in figure 8.2. The proofs of the figure values are found in appendix A.

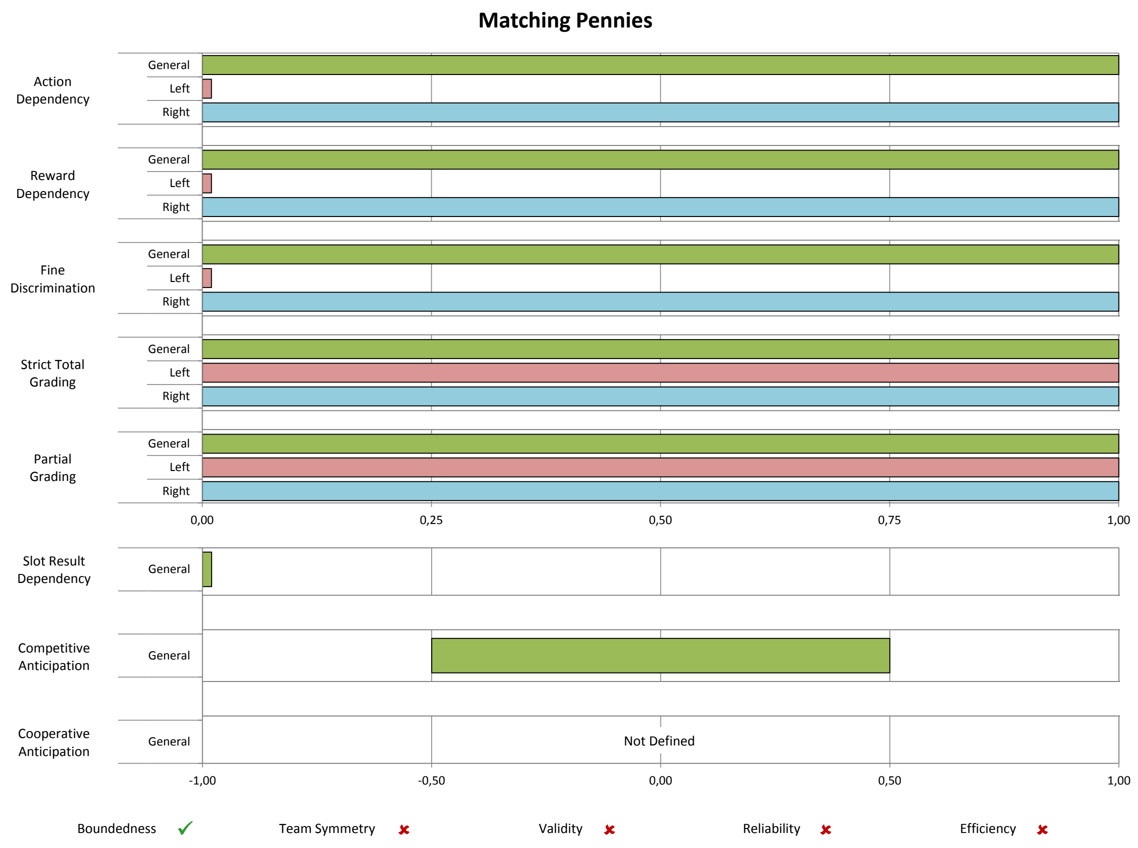


Figure 8.2: Properties for the matching pennies game using uniform unit weights for w_{Π_e}, $w_{\dot{L}}$ and w_S, using $\Delta_{\mathbb{Q}}(a', b')$ instead of $\Delta_{\mathbb{Q}^+}(a, b)$ where a' and b' are the expected results of a and b respectively, and $\Delta_{\mathcal{A}^+}(a, b)$ and $\Delta_{\mathbb{Q}}(a, b)$ return 0 if a and b are equal and 1 otherwise. We use an average of rewards as the utility function to calculate an agent's result and any correlation function where two equal and two inverse values obtain 1 and -1 respectively.

In order to calculate the reliability of this environment, we let it run for 10 time steps and we accurately calculate the expected results of the 100 exercises.

If we start with the properties, since rewards are always between -1 and 1, we see that this game is *bounded*. This means that the value of Υ and many other properties will be bounded. Also, matching pennies is a well known zero-sum game, which means that the payoffs of the agents always sum zero (as we can see in table 2.1) and, therefore, agents have totally opposed interests.

We next move to the team symmetry property. This game has only two teams with one agent slot on each team. So, in order to prove that this multi-agent environment is not team symmetric, we only need to find a pair of line-ups $l_1 = (\pi_1, \pi_2)$ and $l_2 = (\pi_2, \pi_1)$ where the sequence of rewards for π_1 and/or π_2 differs in both line-ups. This becomes trivial by using the same agent π_t (which always performs Tail) as both π_1 and π_2 (i.e. $l_1 = l_2 = (\pi_t, \pi_t)$) and check whether the agent obtains the same expected result in both agent slots. Since π_t gets an expected result of 1 in agent slot 1 and an expected result of -1 in agent slot 2, we can conclude that this multi-agent environment is *not team symmetric*. This forces us to calculate some other properties for all the agent slots.

If we look to the *General* range for the action dependency (AD) property, we see that it goes from 0 to 1. That means that the evaluated agents can either interact without noticing the other agent or can perform actions depending on which agent they encounter. But some particular selection of Π_o could make this environment to have a too restrictive $Left$ range with respect to this property, making it equal to $[0, 0]$, so no evaluated agent could perform different actions depending on which agent it interacts with. In addition, we see that no particular selection of Π_o can restrict the *Right* range, which remains from 0 to 1.

When we look at the *General* range for the reward dependency (RD) property, we see that it goes from 0 to 1. This means that the evaluated agents can either obtain the same expected result or can obtain different expected results depending on which agent they encounter. But some particular selection of Π_o could make this environment to have a too restrictive $Left$ range with respect to this property, making it equal to $[0, 0]$. In addition, we see that no particular selection of Π_o can restrict the *Right* range, which remains between 0 and 1.

The *General* range for the fine discrimination (FD) property goes from 0 to 1. This means that two different evaluated agents can either obtain the same expected result or can obtain different expected results. The $Left$ range can be restricted to be equal to $[0, 0]$, and the *Right* range goes from 0 to 1. This means that, with a bad selection of Π_o, no pair of evaluated agents can be differentiated in terms of performance, and it does not exist a Π_o to always differentiate any pair of evaluated agents.

The *General* range for the strict total grading (STG) and partial grading (PG) properties has, as in all the previous properties, a minimum value of 0 and a maximum value of 1. This means that we cannot provide an ordering for some sets of evaluated agents, but we can provide it for some other sets of evaluated agents. In both STG and PG, $Left$ cannot be restricted by any Π_o, remaining from 0 to 1, and the same occurs with *Right*, which cannot be restricted by any Π_o, remaining from 0 to 1.

This environment always has a *General* range for the slot result dependency (SRD) property equal to $[-1, -1]$. This means that both agent slots have opposed interests, i.e. it is entirely competitive.

The *General* range for the competitive anticipation (AComp) property goes from -0.5 to 0.5. These values tell us that an evaluated agent can improve its expected results by correctly anticipating the other agent's actions, but an incorrect anticipation worsens its expected results.

On the contrary, we cannot evaluate the cooperative anticipation (ACoop) property in this environment. This is because each of the two teams has only one agent slot (in addition, this is also a zero-sum game), so the formula cannot be applied.

We now discuss the validity property. Matching pennies is a very simple game. As a result, it seems clear that it is not general enough to be used (alone) as the basis of a social intelligent test. Nonetheless, there are different opinions about this, as it has been suggested that matching pennies *could* be an intelligence test on its own, under the name 'Adversarial Sequence Prediction' [53, 54]. In fact, a tournament was organised in 2011 where computer algorithms competed[2] and very interesting emergence phenomena were observed. Only strategies that were able to see patterns in the other players scored well (better than random). There are of course some counter-intuitive things about this game. Actually, random agents obtain exactly the same expected result interacting with any opponent, even with very intelligent ones. This raises concerns about the validity of this environment for a (social) test since, in a (social) intelligence test, the average result of a random policy should not obtain a good score, since random agents are clearly not (social) intelligent. Also, matching pennies expected results can be non-monotonic for a set of agents. In [50] there is an example of an agent set for matching pennies that is non-monotonic (so $PG < 1$). Nonetheless, partial orders can still be constructed for the agent set of all finite state machines [54]. Another important problem about matching pennies is that it only evaluates pure competition (it is a zero-sum game), and no form of co-operation can be found (although some versions, or the ternary extension, rock-paper-scissors, could allow for cooperation). Finally, another strong argument against the validity of this game as a good environment (alone) for a social intelligence test is that some current AI systems may score better than humans, even though these AI systems are not (socially) intelligent at all and they are designed to play matching pennies only.

Finally, as we can see in figure 8.3, matching pennies is *not reliable* since the results obtained in a single test only correlate 0.18 with their expected results. Besides, the environment is *not efficient* since a lot of repetitions of the test (1866) are needed to reach a reliable result (0.7 correlation).

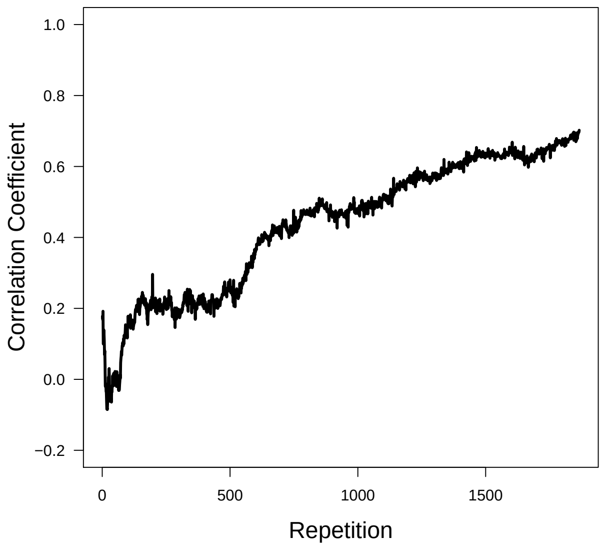


Figure 8.3: Pearson correlation coefficients between the average results of several repetitions of 100 exercises and their expected results of a test using the matching pennies environment.

Overall, matching pennies is an interesting game, but it lacks the suitability and generality that a social intelligence test should have. None of the quantitative properties of the first type (those whose range can be between 0 and 1) can be restricted by using a proper set of populating agents, so we cannot ensure that some social interactions will happen when evaluating an agent's social intelligence. Moreover, this game only focusses on evaluating competition as seen in the slot result dependency, not measuring at all any cooperative social intelligence from the agents.

[2]See `http://matchingpennies.com/tournament/`.

Nonetheless, it is a very simple game that illustrates how the range values of several properties can be calculated and provide very useful information about how a multi-agent environment or game behaves. In the end, it has been a useful exercise before analysing more sophisticated scenarios below.

8.3 Prisoner's Dilemma

The second multi-agent environment we analyse is the prisoner's dilemma (section 2.1.2) which is a simple and well-known game involving competition and cooperation. In this game, $\tau = \{\{1\},\{2\}\}$ represents the partition of agent slots, which has two teams and only one agent slot in each team.

We can see a summary of the properties for the prisoner's dilemma in figure 8.4. The proofs of the figure values are found in appendix B.

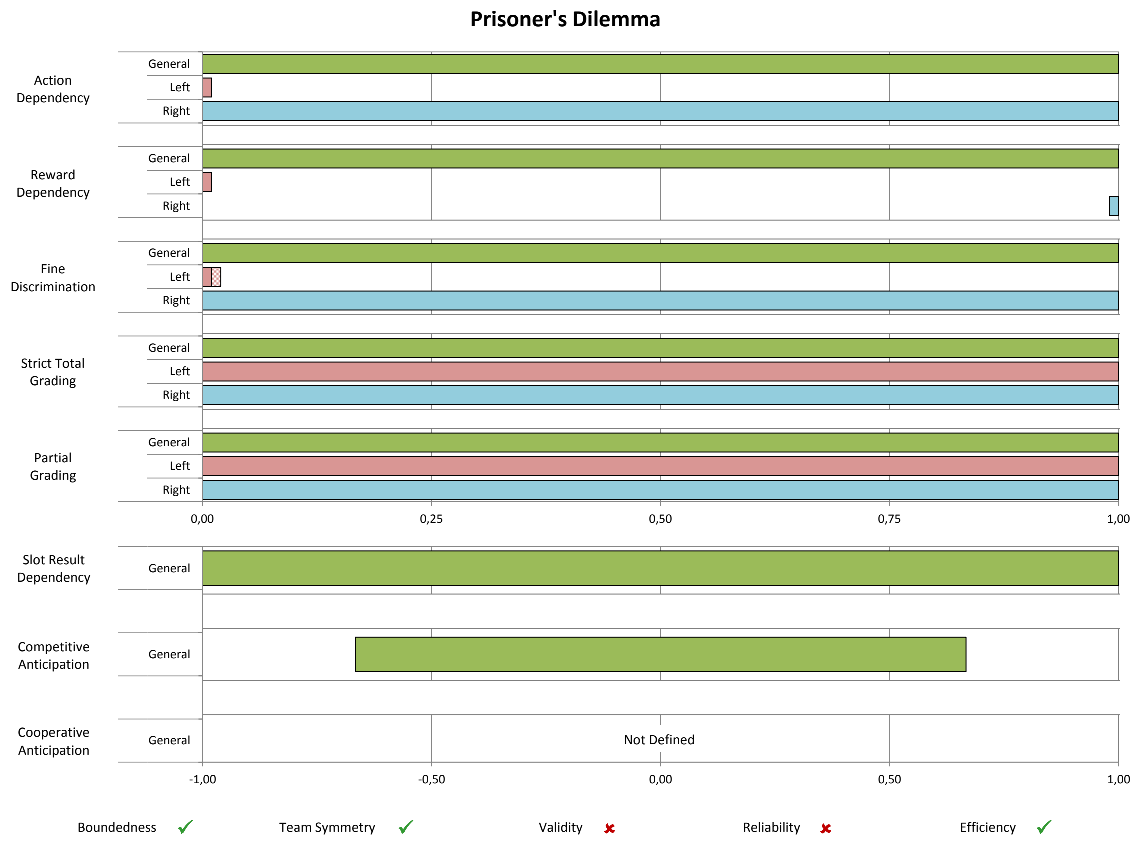


Figure 8.4: Properties for the prisoner's dilemma game using uniform unit weights for w_{Π_e}, $w_{\dot{L}}$ and w_S, using $\Delta_{\mathbb{Q}}(a', b')$ instead of $\Delta_{\mathbb{Q}^+}(a, b)$ where a' and b' are the expected results of a and b respectively, and $\Delta_{\mathcal{A}^+}(a, b)$ and $\Delta_{\mathbb{Q}}(a, b)$ return 0 if a and b are equal and 1 otherwise. We use an average of rewards as the utility function to calculate an agent's result and any correlation function where two equal and two inverse values obtain 1 and -1 respectively. Lighter bands mean the values are not formally calculated, but an estimation is given.

In order to calculate the reliability of this environment, we let it run for 10 time steps and we accurately calculate the expected results of the 100 exercises.

Similarly as happens with matching pennies, since rewards are always between -1 and 1, we see that this game is *bounded*. In this game some cooperation appears when both agents decide to remain silent, so they obtain the maximum joint reward (0.33). But betraying can provide the best reward to one agent if the other agent still remains silent, so remaining silent now provides the worst reward (-1). Finally, if both decide to betray, both obtain the worst joint reward (-0.33). Again, the value of Υ and many properties will be bounded.

Analysing the team symmetry property, we can see that the payoff matrix (table 2.2) is clearly symmetric for both agent slots. This makes that the payoffs of any strategies made by the agents do not depend on which agent slots they are, since they will obtain the same rewards. From this observation we can conclude that this multi-agent environment is *team symmetric*, which allows us to calculate some other properties only for one agent slot and assume that it is maintained for the other.

If we look at the ranges for the action dependency (AD) property, we encounter exactly the same scenario than in the matching pennies. That means that evaluated agents can either interact without noticing the other agent, obtaining a value of 0, or can always perform actions depending on the agent they encounter, obtaining a value of 1. But some particular selections of Π_o can provide a too restricted *Left* range, forcing this value to be equal to $[0,0]$, so no evaluated agent from Π_e can behave differently depending on the populating agents of Π_o. In addition, no particular selection of Π_o can restrict the *Right* range, remaining from 0 to 1.

We start to find some differences with the matching pennies when we analyse the reward dependency (RD) property. As in matching pennies, the *General* range for this property goes from 0 to 1, so the expected results of the evaluated agents can either depend or not on which agent they encounter. Also, some particular selection of Π_o could make this environment to have a too restrictive *Left* range with respect to this property, making it equal to $[0,0]$, so no evaluated agent obtains different expected results depending on which agent it interacts with. But, a good selection of Π_o can restrict the *Right* range making it equal to $[1,1]$. This means that the evaluated agents obtain different expected results depending on which agent they interact with.

We now move to the fine discrimination (FD) property. The *General* range for this property goes from 0 to 1. This means that two different evaluated agents can either obtain the same expected result or different expected results depending on the set of populating agents Π_o we select. It exists a particular Π_o which restricts the *Left* range to be equal to $[0,0]$, meaning that the environment does not discriminate the evaluated agents, since every evaluated agent obtain the same expected result. It is not possible to restrict the *Right* range in such a way that we can always discriminate every pair of evaluated agents, remaining from 0 to 1, so it is possible to find two different evaluated agents from some particular Π_e obtaining the same expected result.

The *General* range for the strict total grading (STG) and partial grading (PG) properties can, as in all the previous properties, reach a minimum value of 0 and a maximum value of 1 for this environment. This means that we cannot provide an ordering for some sets of evaluated agents, but we can provide it for some other sets of evaluated agents. In both STG and PG, the *Left* range cannot be restricted by any Π_o, remaining from 0 to 1, and the same occurs with the *Right* range, which cannot be restricted by any Π_o, remaining from 0 to 1.

The *General* range for the slot result dependency (SRD) property goes from -1 to 1. This provides very different expected results depending on the agents' strategies.

The *General* range for the competitive anticipation (AComp) property goes from -0.66 to 0.66. Anticipating the strategy of the other agent can be really useful to obtain a good expected result, but it can also provide a really bad expected result if the strategy is not correctly anticipated.

We cannot evaluate the cooperative anticipation (ACoop) property in this environment. This is because, as in the matching pennies game, each of the two teams has only one agent slot, so the formula cannot be applied. This seems counter-intuitive since agents' actions can lead to cooperation among the agents, but this cooperation is not meant to improve the agent's own rewards, but to improve the other agent's rewards (which is in a different team). Indeed, if we reframe the game by using only one team and calculating the team reward as the mean of the agents' rewards, then both agents can cooperate to obtain the best team reward. This would lead us to a situation where a bad cooperative anticipation between the agents' actions can negatively affect the team reward, but also, a good cooperative anticipation will have good benefits for the team.

We now discuss the validity property. The prisoner's dilemma is similar in simplicity to the matching pennies game. But in this game, competition is not so strong, providing some cooperation between the two teams and making this game more general than the matching pennies. But in this game we cannot evaluate cooperation within a team, so it is not general enough to evaluate social intelligence. Actually, some simple strategies can clearly make the adversary's results get stuck, forcing it to obtain bad rewards independently of its strategy. This raises concerns about the validity of this environment for a (social) test, since a (social) intelligence test should not give bad results to (social) intelligent agents.

Finally, as we can see in figure 8.5, prisoner's dilemma is *not reliable* since the results obtained in a single test only correlate 0.21 with their expected results. However, the environment is *efficient* since few repetitions of the test (13) are needed to reach a reliable result (0.7 correlation).

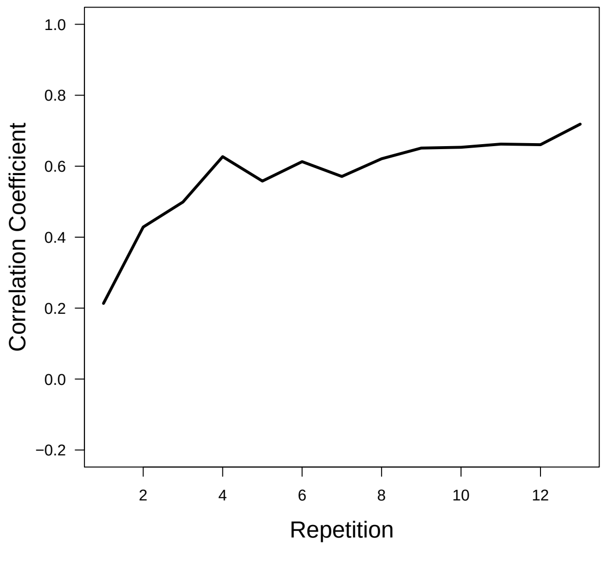


Figure 8.5: Pearson correlation coefficients between the average results of several repetitions of 100 exercises and their expected results of a test using the prisoner's dilemma environment.

The prisoner's dilemma resembles the matching pennies in many aspects, but it is slightly a more complex game. As many similarities exist between both environments, many properties remain equal.

Let us see a more complex multi-agent environment.

8.4 Predator-Prey (Pursuit Game)

The next multi-agent environment we analyse is a pursuit game called predator-prey (section 2.1.3). $\tau = \{\{1\}, \{2, 3, 4\}\}$ represents the partition of agent slots. The first team $\{1\}$ contains the prey and the second team $\{2, 3, 4\}$ contains three predators. Agent slot 1 starts in the upper left corner and agent slots $2, 3$ and 4 start in the upper right, bottom left and bottom right corners respectively. For the analysis of this multi-agent environment we follow the same procedure as for the previous ones. This is the reason why we allow the evaluated agent to interact in every agent slot, even as the prey. But, as mentioned above, if we only let the agent interact as a predator, the values of the properties could be different.

We can see a summary of the properties for the predator-prey in figure 8.6. The proofs of the figure values are found in appendix C. We assume that each agent knows in which agent slot it starts.

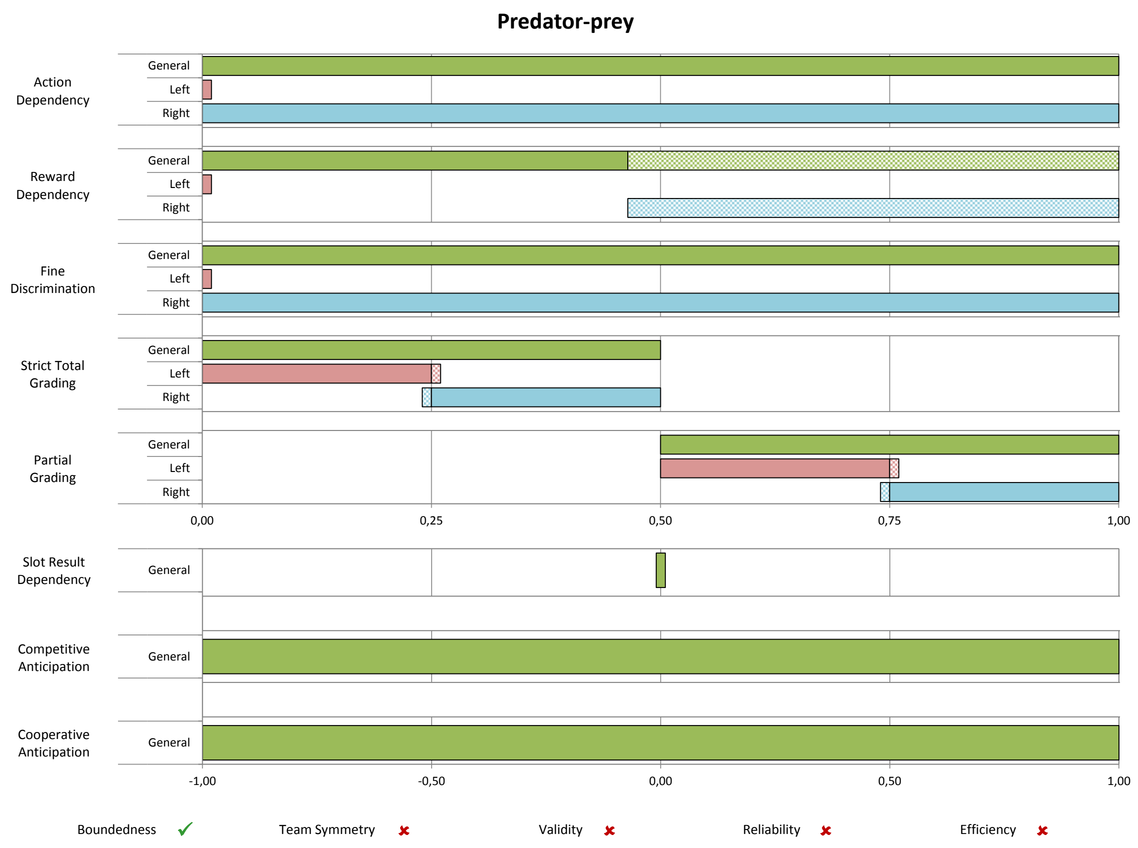


Figure 8.6: Properties for the predator-prey environment using uniform unit weights for w_{Π_e}, $w_{\dot{L}}$ and w_S, using $\Delta_{\mathbb{Q}}(a', b')$ instead of $\Delta_{\mathbb{Q}^+}(a, b)$ where a' and b' are the expected results of a and b respectively, and $\Delta_{\mathcal{A}^+}(a, b)$ and $\Delta_{\mathbb{Q}}(a, b)$ return 0 if a and b are equal and 1 otherwise. We use an average of rewards as the utility function to calculate an agent's result and any correlation function where two equal and two inverse values obtain 1 and -1 respectively. Lighter bands mean the values are not formally calculated, but an estimation is given.

In order to calculate the reliability of the environment, we estimate the expected results of the 100 exercises with a Monte Carlo approximation (section 2.5).

When we start with the properties we see that this multi-agent environment is *bounded* since its rewards are between -6 and 6, so average rewards when the episode is finished (after six time steps) are between -1 and 1. As with the previous environments, the value of Υ and many properties will be bounded.

Analysing the team symmetry property, we clearly see that this multi-agent environment is *not team symmetric*. First, prey and predator teams do not have the same number of agent slots. And second, only by changing the positions of the agents within the predator team does not provide the same expected results for the agents, owing to they start in different positions in the space. This forces us to calculate all the other properties for all the agent slots.

If we look at the action dependency (AD) property, we cannot see any difference with the previous environments, even having a so different scenario. For the *General* range, evaluated agents can either interact without noticing the other agents, providing a minimum value of 0, or can perform different actions depending on the agents they encounter, providing a maximum value of 1. But some particular bad selections of Π_o could restrict the $Left$ range to be equal to $[0, 0]$. On the contrary, no particular selections of Π_o can restrict the *Right* range for this property, remaining from 0 to 1.

When we look at the ranges for the reward dependency (RD) property, we can see that the *General* range goes from 0 to 1 so the expected results of the evaluated agents can either differ or not depending on the agents they encounter. A really bad selection of Π_o can make the $Left$ range too restrictive, staying on $[0, 0]$ no matter which Π_e we are evaluating. But we can restrict the *Right* range by selecting a properly Π_o, obtaining a range from 0.4643 to 1. This would force that almost half of the expected results of the evaluated agents would be different depending on the line-up pattern they interact with.

We now move to the ranges for the fine discrimination (FD) property. The *General* range goes from 0 to 1, i.e. the expected result of one evaluated agent can be different or not from the expected result of another evaluated agent depending on which set of populating agents Π_o is selected. But a bad selection of Π_o can restrict too much the $Left$ range, making it equal to $[0, 0]$, so every evaluated agent would obtain the same expected result. However, the *Right* range cannot be restricted, remaining from 0 to 1, since there are always two different evaluated agents that can obtain the same expected result, independently of the set of populating agents Π_o.

The strict total grading (STG) and partial grading (PG) properties are clearly different from the previous environments. The *General* range for the strict total grading property can only go from 0 to 0.5, so we cannot even have a strict total ordering for all the evaluated agents. In addition, its $Left$ range can be restricted to be from 0 to 0.25, so it would be possible to obtain some strict total ordering. At least, its *Right* range can be restricted to be from 0.25 to 0.5, which somehow alleviates this situation. But instead, its partial grading has really good ranges. Its *General* range goes from 0.5 to 1, its $Left$ range can be restricted to be from 0.5 to 0.75 and its *Right* range can be restricted to be from 0.75 to 1. This makes this environment good for grading the evaluated agents, so even with a bad selection of Π_o we would still be able to obtain some partial orders between some of the evaluated agents, and provides a promising partial grading for the evaluated agents if Π_o is well selected.

The slot result dependency (SRD) has a *General* range equal to $[0, 0]$. This particular value comes from the opposite results of the prey and predators. As early mentioned, if we had only used the agent slots of the predators for evaluating the agents, we would have had another range.

The *General* range for the competitive anticipation (AComp) property goes from -1 to 1. Anticipating the strategy of the other team can really be useful to improve the expected result, but a bad anticipation really penalises the expected result of an evaluated agent.

In this environment we can find cooperation between the agents within the predator team. The *General* range for the cooperative anticipation (ACoop) goes from −1 to 1. This makes a good coordination within the predator team to always chase the prey, but a bad coordination can let the prey to escape.

We now discuss the validity property. In the predator-prey environment we can encounter both competition between the prey and predators, and cooperation among the predators, which makes it a complex game to evaluate social intelligence. Both competition and cooperation seem important in this environment, giving us a general situation to evaluate social intelligence in a broad way. Also, agents that are evaluated in this environment can have good orderings as properties STG and PG reflect, making it a good environment to classify the agents. However, the abilities that the agents need to accomplish their goals are not balanced. It is easier for the predator team to win the chase if they cooperate adequately, where the prey will not have a chance to survive. Also, the $Left$ ranges of the properties are usually very restrictive when Π_o is not selected carefully. So it is really necessary to select a correct set of populating agents, but still, once a certain level of (social) intelligence is reached we cannot evaluate higher levels of (social) intelligence, since their result will remain equal. From here, we can say that the social intelligence that this environment is evaluating is clearly limited to a certain level, and once this level is reached the results will not vary as happens in the $Left$ range for the RD property. Summarising, this environment allows us to evaluate both competition and cooperation which makes it good to evaluate social intelligence, but this can only be evaluated until a certain level of (social) intelligence. Over this level, the environment lacks mechanisms to let the evaluated agent show its actual level of (social) intelligence, so with this environment we are not able to measure such agent's social intelligence level. As a result, the environment is *not valid* for interesting levels of social intelligence.

Finally, as we can see in figure 8.7, the predator-prey is *not reliable* since the results obtained in a single test have no correlation with their expected results. Besides, the environment is *not efficient* since a lot of repetitions of the test (51) are needed to reach a reliable result.

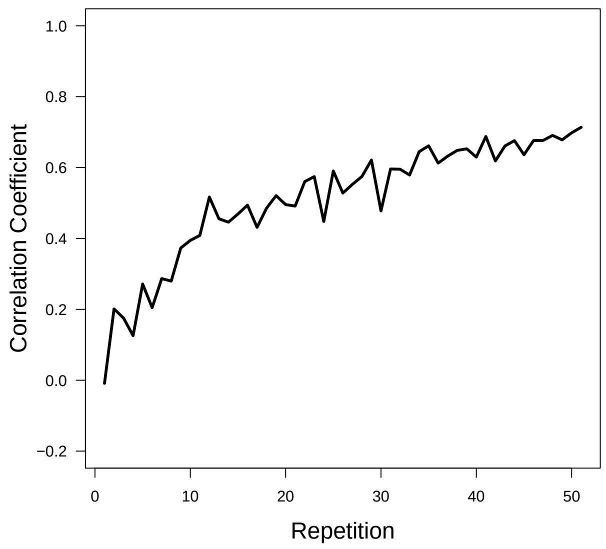


Figure 8.7: Pearson correlation coefficients between the average results of several repetitions of 100 exercises and their expected results of a test using the predator-prey environment.

The predator-prey gives us a more complex multi-agent environment than the previous ones. However, it can be a good multi-agent environment for a testbed to evaluate social intelligence if Π_o is wisely chosen. But we must be careful, since we could obtain a really poor testbed if Π_o is not properly selected. The results we have obtained came from our choice to include the possibility to evaluate the agents also interacting as the prey. A more classical approach (where agents are only evaluated interacting as predators) could have given us a different picture for this environment.

8.5 Pac-Man

Computer games are also currently used as mainstream environments to evaluate AI systems. One example of the use of games for evaluating AI is the ALE (Arcade Learning Environment) [2], a framework where a set of arcade computer games are used to evaluate the performance of current AI algorithms. The following multi-agent environment we analyse here is a computer game called Pac-Man (section 2.1.4). $\tau = \{\{1\}, \{2, 3, 4, 5\}\}$ represents the partition of agent slots. The first team $\{1\}$ contains Pac-Man and the second team $\{2, 3, 4, 5\}$ contains four ghosts. For the analysis of this multi-agent environment we follow the same procedure as for the previous ones. This is the reason why we allow the evaluated agent to interact in every agent slot, even as the ghosts.

From the huge diversity of situations that can occur in this environment, it is difficult to formally analyse some of the properties as we did with the previous environments. As long as the games and multi-agent environments are more complex, it becomes more difficult to determine their actual levels of cooperation and competition, and more effort is needed to formalise them and find some Π_e and Π_o to find the properties ranges. Instead, we analyse this environment in an informal way. However, we could still use the properties to assess such a testbed when both sets of agents Π_e and Π_o are fixed.

In figure 8.8 we show a summary with an estimation of the properties for Pac-Man. Regarding reliability and efficiency, due to the intractability to accurately approximate their expected results we do not perform the calculations, but we give our estimation about what we think it would happen.

When we start with the properties we see that this game is *not bounded*, since Pac-Man can obtain more and more points (or rewards) as long as it continues surpassing levels. This makes that Υ and many properties will not be bounded for this environment. Reframing the game by calculating an average of points by time as rewards in order to make it bounded will change the goal of the game significantly. Since we give an estimation for the properties, we give them bounded values.

Also, this multi-agent environment is *not team symmetric*. On one hand, both teams do not have the same number of agent slots, which makes the multi-agent environment *not Total Inter-Team Symmetric*. On the other hand, we could say that the multi-agent environment is Intra-Team Symmetric, since every ghost has the same probability to chase Pac-Man, but this is not exact, since each ghost appears in different moments of the game, so swapping their behaviour could not provide exactly the same expected results, making the multi-agent environment *not Intra-Team Symmetric*.

The action dependency (AD) property seems to be as in previous environments. All agents have the possibility to ignore the actions of the other agents or act according to what these other agents did in previous time steps.

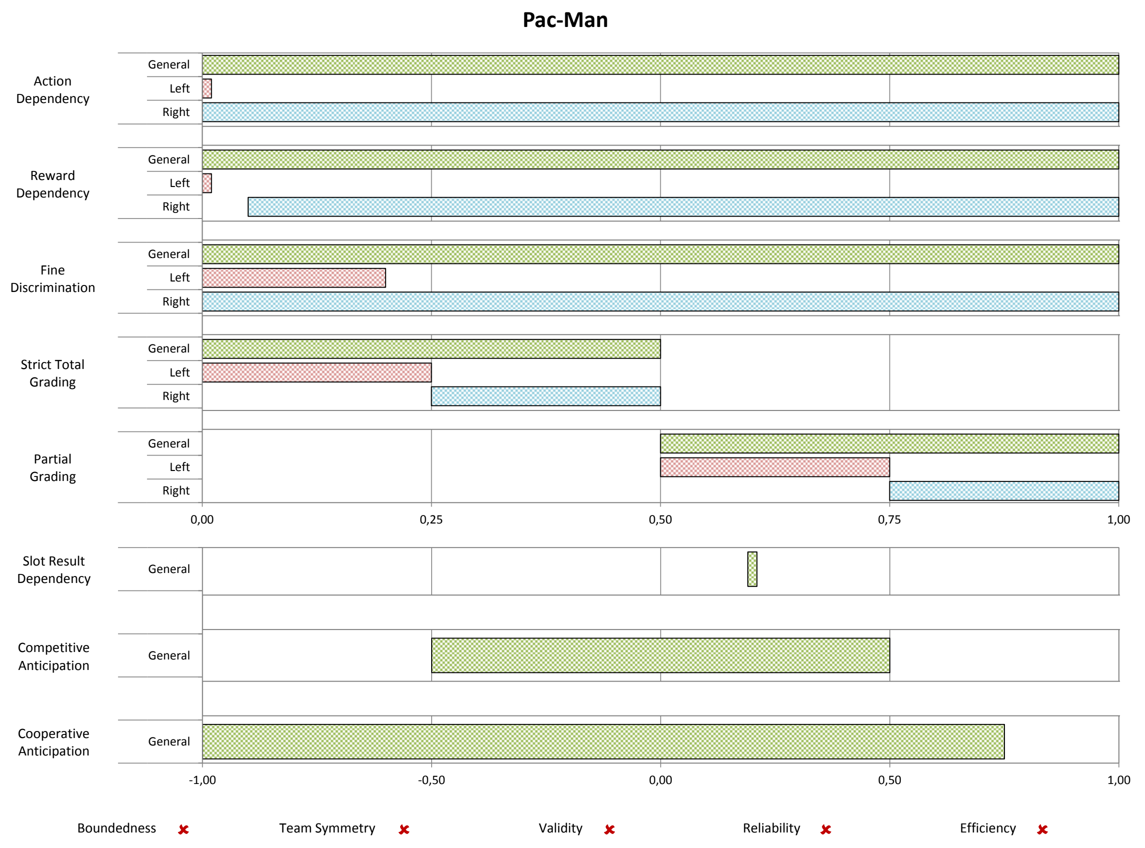


Figure 8.8: Estimation of the properties for the Pac-Man game using uniform unit weights for w_{Π_e}, $w_{\dot{L}}$ and w_S, using $\Delta_{\mathbb{Q}}(a', b')$ instead of $\Delta_{\mathbb{Q}^+}(a, b)$ where a' and b' are the expected results of a and b respectively, and $\Delta_{\mathcal{A}^+}(a, b)$ and $\Delta_{\mathbb{Q}}(a, b)$ return 0 if a and b are equal and 1 otherwise. We use an average of rewards as the utility function to calculate an agent's result.

When we look at the reward dependency (RD) property, it could obtain some different values. Indeed, it is too easy to chase Pac-Man if the four ghosts cooperate coherently, but a bad behaviour for the ghosts can facilitate the game for Pac-Man. In addition, small differences in the behaviour of the agents can provide very different results as, for example, a ghost passes near Pac-Man and decides to chase or to avoid it. This small difference in behaviour will provide high differences in their results. However, when a ghost is far from Pac-Man, small differences in its behaviour will probably lead to similar results.

The fine discrimination (FD) property also has a huge range of values. As mentioned above, the behaviour of two different evaluated agents can both obtain the same or very different expected results, highly depending on the behaviour of the other agents. However, the points that Pac-Man obtains can somehow depend on Pac-Man's behaviour itself (at least during the first moments of the game) but, since eventually good coordinated ghosts will always chase it, Pac-Man results are still limited.

It seems difficult to know whether we can establish a grading between the evaluated agents in this environment. But we venture that the grading properties could be similar to the ones provided in the predator-prey environment, since both environments have many similarities.

It is also difficult to provide a slot result dependency, since rewards obtained by one team typically do not reflect on the other team. For example, every point obtained by Pac-Man does not directly have influence on the ghost team's rewards, and chasing Pac-Man only prevents it from obtaining more points. But, if we just assume that both teams rewards are always different, we can do as we did in the SRD for the predator-prey environment (which has a similar configuration) to obtain an approximated value, meaning that the slot result dependency is more focused on cooperation than competition.

If we look at the anticipation properties, it is possible that anticipating competitors does not have a huge reflect on rewards, but still can provide some good rewards. This is specially seen while interacting as Pac-Man. It can avoid being chased by the ghosts and achieve more points, but this anticipation will eventually become useless, since ghosts have an enormous advantage to chase it. In cooperative anticipation, it is possible that one ghost can do worse than a random agent (as, for example, perfectly following another ghost's movements, making its presence useless), leading to really bad values for this property, but a good cooperating anticipation can make chasing Pac-Man easier.

This environment is *not very reliable* as it depends on many small details, such as which direction takes each agent on each deviation or whether Pac-Man captures the ghosts when it becomes invulnerable. Also, it is *not efficient*. We would need to run the game at least hundreds of times to get some stability in the agents' results, since many of the first actions obtain the same rewards but the crucial part of the game comes when the pills become scattered.

Finally, with this environment we can both evaluate competition and cooperation. We can find competition, since each team can only gain rewards by making the other team lose rewards. Additionally, in the ghost team, cooperation is also needed to properly chase Pac-Man. However, for this environment, the selection of populating agents is crucial, a set of predators with high (and more interesting) level of (social) intelligence can make Pac-Man efforts useless, which will always obtain bad results, making this game *not valid* to evaluate social intelligence for more relevant situations, similarly as happened with the predator-prey environment.

8.6 RoboCup Soccer

The last multi-agent environment we analyse is a 3D space game called RoboCup Soccer (section 2.1.5). $\tau = \{\{1,2,3,4,5\},\{6,7,8,9,10\}\}$ represents the partition of agent slots, where agent slots 1 and 6 represent the goalkeepers.

As happens with the previous environment, this game has a huge diversity of situations that can occur (including physical and virtual versions), which makes difficult to formally analyse its $General$, $Left$ and $Right$ ranges. Again, in such complex scenario, it becomes more difficult to determine the actual levels of cooperation and competition. But also, due to the high level of complexity of this game, teams tend to some specialisation, with each player focussing on some specific aspects of the game instead of focusing on the problem in a general way. Again, we analyse this environment in an informal way.

Similarly as we did in the previous environment, in figure 8.9 we show a summary with an estimation of the properties for RoboCup Soccer.
Similarly with the previous environment, regarding reliability and efficiency, we give our estimation about what we think it would happen.

When we start with the properties we see that this game is *bounded*, since rewards (1 for win, 0 for a tie and -1 for lose) are bounded. This makes that Υ and many properties will be bounded.

When we look at the team symmetry property, both teams have the same number of agent slots and, if we ignore which team starts with the possession of the ball on each half, it makes the

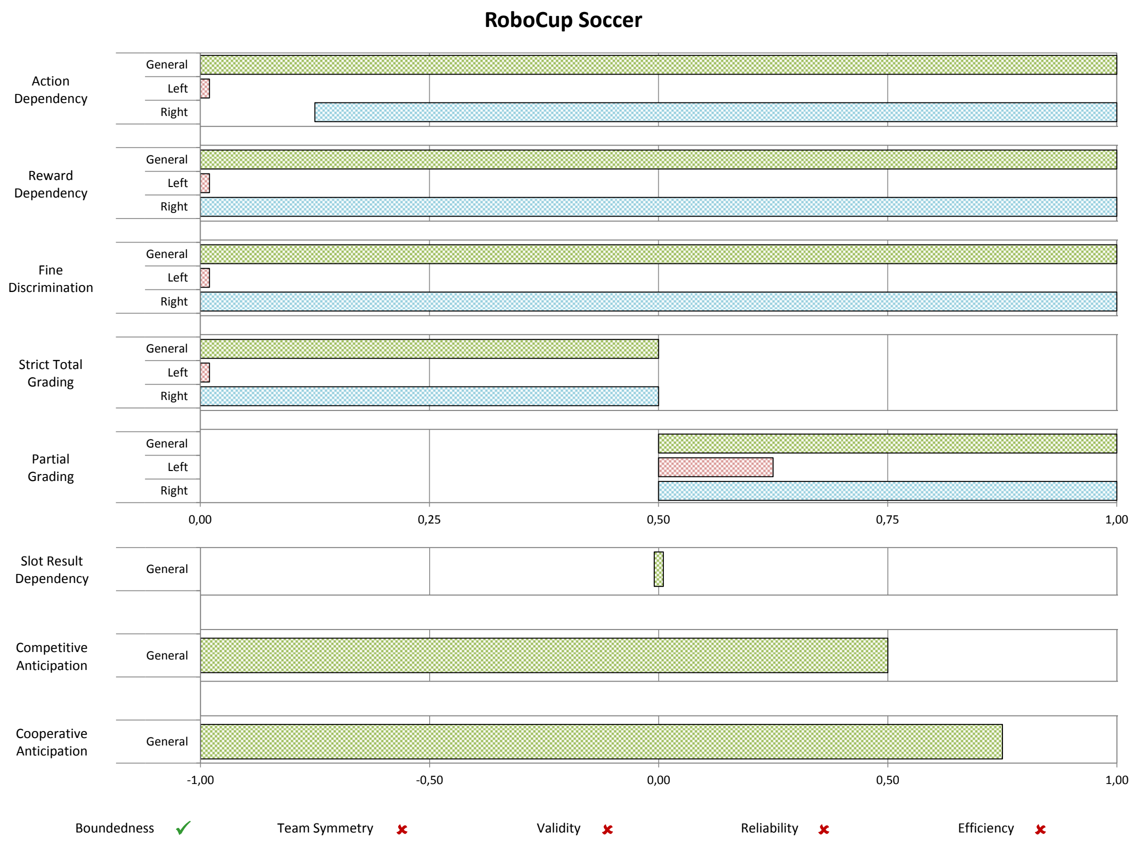


Figure 8.9: Estimation of the properties for the RoboCup Soccer game using uniform unit weights for w_{Π_e}, $w_{\dot{L}}$ and w_S, using $\Delta_{\mathbb{Q}}(a', b')$ instead of $\Delta_{\mathbb{Q}^+}(a, b)$ where a' and b' are the expected results of a and b respectively, and $\Delta_{\mathcal{A}^+}(a, b)$ and $\Delta_{\mathbb{Q}}(a, b)$ return 0 if a and b are equal and 1 otherwise. We use an average of rewards as the utility function to calculate an agent's result.

multi-agent environment *Total Inter-Team Symmetric*, so both teams can swap their positions and the expected result will remain the same. But, if we want to swap the positions of two agents within the same team, they would not obtain the same expected results (as, for example, the goalkeeper has different rules), so the multi-agent environment is *not Intra-Team Symmetric*. Since for the team symmetry condition we need the multi-agent environment to be both *Total Inter-Team Symmetric* and *Intra-Team Symmetric*, we can conclude that the RoboCup Soccer is *not team symmetric*.

The action dependency (AD) property is similar to the previous environments. All the evaluated agents have the possibility to act differently depending on the agents they encounter, but, at least, in this game an agent can affect the actions of the evaluated agents. For example, one agent can knock the evaluated agent down to the ground, so now it is only able to stand up.

We now see the reward dependency (RD) property, which can have a huge range. A change in the line-up of course can change the result of the match. Conversely, only changing one agent in the line-up can completely change the match result to make a team lose. And we can reason in the same way for the fine discrimination (FD) property, since the behaviour of only one agent (the evaluated agent) can also make its team to either win/lose the game, or obtain the same expected result. However, it is not always the case that the presence of the evaluated agent in a team has a difference on its results, specially when its teammates are bad players and the opponent team is really good.

It is not easy to determine if there exists some grading between the evaluated agents in this environment. Instead, we can take a look at some professional (human) soccer leagues. It is not unusual to see situations where two teams repeatedly tie, and a third team beats one of them while loses against the other one repeatedly. This situation shows us that there is no strict grading between teams (and neither is between their players) for this game. However, some teams usually win against other teams, specially against teams of inferior leagues.

The slot result dependency is straightforward for this environment. We have two teams with five agents on each team. Every agent within a team obtain the same expected result, while the other agents within the other team obtain the opposite expected result. Using a correlation function over these expected results and over all agent slots, we obtain a slot result dependency equal to $[0, 0]$.

If we look at the anticipation properties, correctly anticipating both competitive and cooperative can provide a high advantage, so the team of the evaluated agent can score more goals and win the game more easily. A bad competitive anticipation can lead the opponent to win the game, while a good one can provide good expected results. In cooperative anticipation, the evaluated agent could play worse if it is not correctly anticipating its teammates, but a good anticipation can provide them really good expected results. In both cases, good anticipation never ensures to score more goals and, therefore, obtain better results. But a very good understanding of the behaviour of the teammates can provide better chances to win the game, since defending every possible attack is not always possible, but only understanding the adversary does not ensure by itself to score more goals.

Let us now consider the validity of this environment. First, with this environment we can both evaluate competition and cooperation. We can find competition, since both teams must compete to win the game. Additionally, the agents within each team can cooperate to mislead the other team and score more goals. Second, increasing the social intelligence of the agents typically increases the difficulty of the match, since more skilled agents score goals more easily, and also defend better, preventing the other team to score. This makes the game useful to match a high variety of skill levels. But conversely, this game also evaluates some other skills than social intelligence, such as for example, their ability to predict the movement that the ball will do when it is kicked. In principle, there are reasons to consider this environment a valid scenario to evaluate social intelligence. However, since the agents need more than their social intelligence in order to play the game, we think that it is not specific enough, and an evaluation using this kind of environment does not only evaluate social intelligence, but also other abilities such as motion understanding. For this reason we consider this environment *not valid* to evaluate social intelligence.

This environment is *not reliable*. In this environment, it is widely known that two teams that are facing each other do not necessarily obtain the same result in every match, and several matches are needed to figure out which of them is better. This depends highly on the decisions made by the agents, since a small change can lead to a very different result on the match. Also, the result of the game depends on luck (at least for the physical version), since the players cannot always predict the correct movement of the ball. This situation is aggravated when several matches must correlate with their expected results, where hundreds or thousands of repetitions are needed to obtain a reliable test, making the environment *not efficient*.

8.7 Discussion

So far, we have analysed some ranges of values that the properties presented in chapter 7 can have for the five multi-agent environments we have selected. Next, let us group the environment properties by the different ranges in order to make a comparison between the environments through their properties.

In figure 8.10 we see the *General* range of the quantitative properties for the five multi-agent environments.

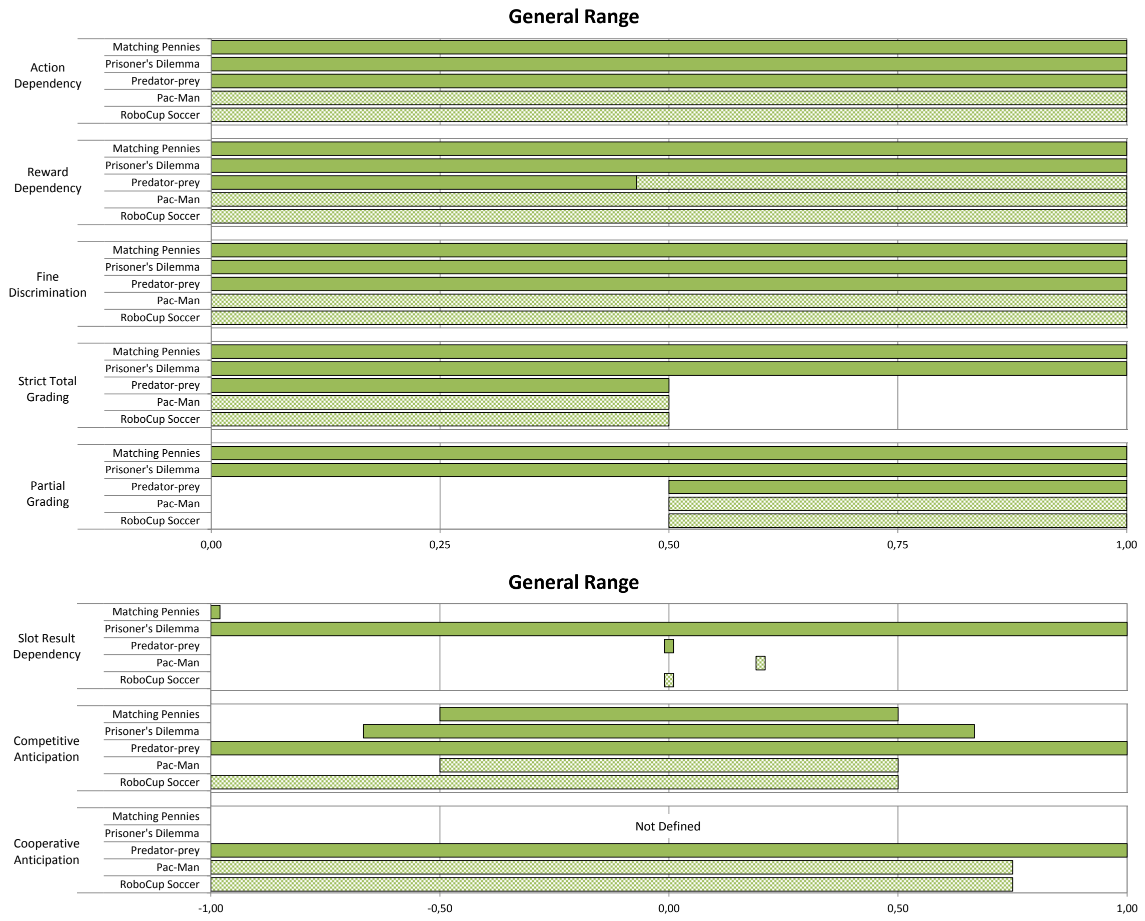


Figure 8.10: *General* range for the quantitative properties of the five multi-agent environments analysed, using uniform unit weights for w_{Π_e}, $w_{\dot{L}}$ and w_S, using $\Delta_{\mathbb{Q}}(a', b')$ instead of $\Delta_{\mathbb{Q}^+}(a, b)$ where a' and b' are the expected results of a and b respectively, and $\Delta_{\mathcal{A}^+}(a, b)$ and $\Delta_{\mathbb{Q}}(a, b)$ return 1 if a and b are equal and 0 otherwise. We use an average of rewards as the utility function to calculate an agent's result and any correlation function where two equal and two inverse values obtain 1 and -1 respectively. Lighter bands mean the values are not formally calculated, but an estimation is given.

As we can see in the top part of figure 8.10, there are few differences with respect to this range for the environments. In the first three properties all the environments have the broadest possible range, so considering all possible agents leads to virtually any possibility in any of them. We only see some difference for the grading properties in the last three environments. While the first two environments have the broadest possible range, the last three environments seems that as long as the strict total grading gets worse, the partial grading gets a better range. This is simply explained because agents in the same team obtain the same rewards, so it is not possible to obtain a strict total ordering for them while it is still possible for agents in different teams. Obviously, the agents in the same team always have a partial ordering, making this range higher.

In the bottom part of figure 8.10 we can see more differences. In the slot result dependency property we can see the first big difference between the environments. The majority of these environments have a unique (and usually different) value for this property. While the matching pennies is completely competitive, Pac-Man is slightly more oriented to cooperation, and predator-prey and RoboCup Soccer are neutral (i.e. provide both competition and cooperation to the same extent). But, the prisoner's dilemma is the only one that has the broadest possible range instead of having a predetermined configuration, so this environment allows the agents to dynamically cooperate and compete with other agents depending on which actions they perform.

Finally, we have the two anticipation properties. In both of them, the agents obtain better expected results when they correctly anticipate, but their expected results get worse when they incorrectly anticipate.

In the competitive anticipation we see some small differences between the environments. While a correct competitive anticipation provides good expected results, some environment could provide much worse expected results when incorrectly anticipating competitive agents, as we can see for the RoboCup Soccer. For the cooperative anticipation we see a different picture. In this case two of the environments do not have this property, since they do not provide teams where the evaluated agent can cooperate with a teammate. However, the last three environments provide teams where cooperative anticipation can be useful between cooperative agents, and also they (almost) provide the broadest range for this property.

It is much more interesting to see what happens with these multi-agent environments if we make bad selections for Π_o. In figure 8.11 we see the $Left$ range of the quantitative properties which range is between 0 and 1 for the five multi-agent environments.

We can see that both the action and reward dependency properties have the worst possible range for all the environments. This means that a bad selection of Π_o would be disastrous with respect to these properties. In fact, this is not surprising, since Π_o could be populated only with agents having the same exact behaviour, so the evaluated agents would not be able to behave or obtain different rewards depending on which populating agents from Π_o they are interacting with.

For the fine discrimination property, we can see that (almost) all the environments have a very low range, so the evaluated agents can hardly be discriminated.

The strict total grading property clearly gives us an order for the environments. With regard to this property, RoboCup Soccer is clearly the worst environment, while predator-prey and Pac-Man follow it. Meanwhile, matching pennies and prisoner's dilemma cannot be restricted with any particular Π_o, obtaining a good range for this property.

Finally, the partial grading has different ranges for the environments. Predator-prey and Pac-Man have the same range, and they also have a clearly better range than the RoboCup Soccer. Both matching pennies and the prisoner's dilemma have the broadest possible range. At this point, it is not clear which pair matching pennies and the prisoner's dilemma, or predator-prey and Pac-Man has better ranges. From one side, the ranges for matching pennies and prisoner's dilemma go from 0 to 1, so a selection of Π_o does not necessarily worsen their ranges for this property, but still it is possible to have a bad value. From the other side, the worst

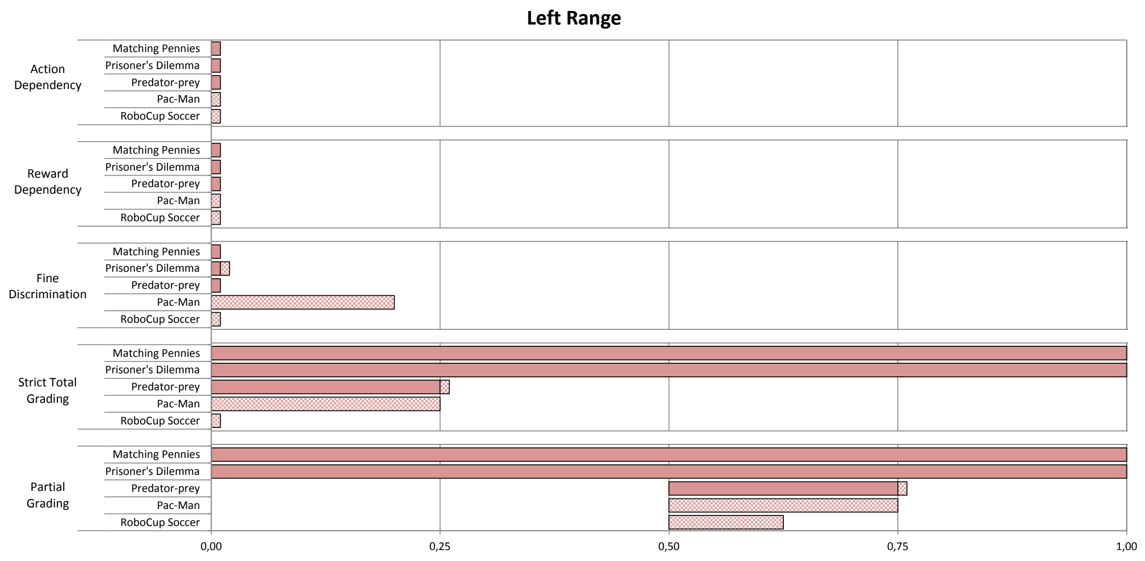


Figure 8.11: *Left* range for the quantitative properties which range is between 0 and 1 of the five multi-agent environments analysed, using uniform unit weights for w_{Π_e}, $w_{\dot{L}}$ and w_S, using $\Delta_{\mathbb{Q}}(a', b')$ instead of $\Delta_{\mathbb{Q}^+}(a, b)$ where a' and b' are the expected results of a and b respectively, and $\Delta_{\mathcal{A}^+}(a, b)$ and $\Delta_{\mathbb{Q}}(a, b)$ return 1 if a and b are equal and 0 otherwise. We use an average of rewards as the utility function to calculate an agent's result. Lighter bands mean the values are not formally calculated, but an estimation is given.

value for the predator-prey and Pac-Man cannot be as bad as it can be in matching pennies and the prisoner's dilemma. However, a bad selection of Π_o does worsen their best values for this property.

Now we see the effect that good selections for Π_o can have on some quantitative properties for the multi-agent environments. This is possibly the most interesting picture, because it gives us the best we could do with a right choice of Π_o. In figure 8.12 we see the *Right* range of the quantitative properties which range is between 0 and 1 for the five multi-agent environments.

We can see that the action dependency and fine discrimination properties cannot improve much by selecting an appropriate Π_o. At least RoboCup Soccer can slightly restrict its action dependency range, which means that even with the best possible choice of Π_o we cannot ensure that there will be a high action dependency.

For the reward dependency property the ranges vary. Matching pennies and RoboCup Soccer cannot restrict their ranges. Pac-Man can slightly restrict this range and predator-prey does not have a bad range, so depending on which line-up pattern the evaluated agents encounter, they certainly obtain some differences in their expected results. But prisoner's dilemma can ace this property, making the expected results different for every evaluated agent in Π_e depending on the line-up pattern they interact with.

For the strict total grading we find more differences. It is not clear which environment has a better range, but at least we can say that RoboCup Soccer has the worst range among the five environments. However, it is not as clear which of the other four environments has the best range. From one side, the ranges for matching pennies and the prisoner's dilemma go from 0 to 1, so a selection of Π_o does not necessarily improve their ranges for this property, but still it is possible to have a good value. From the other side, the best value for the predator-prey and Pac-Man cannot be as good as it can be in matching pennies and prisoner's dilemma. However,

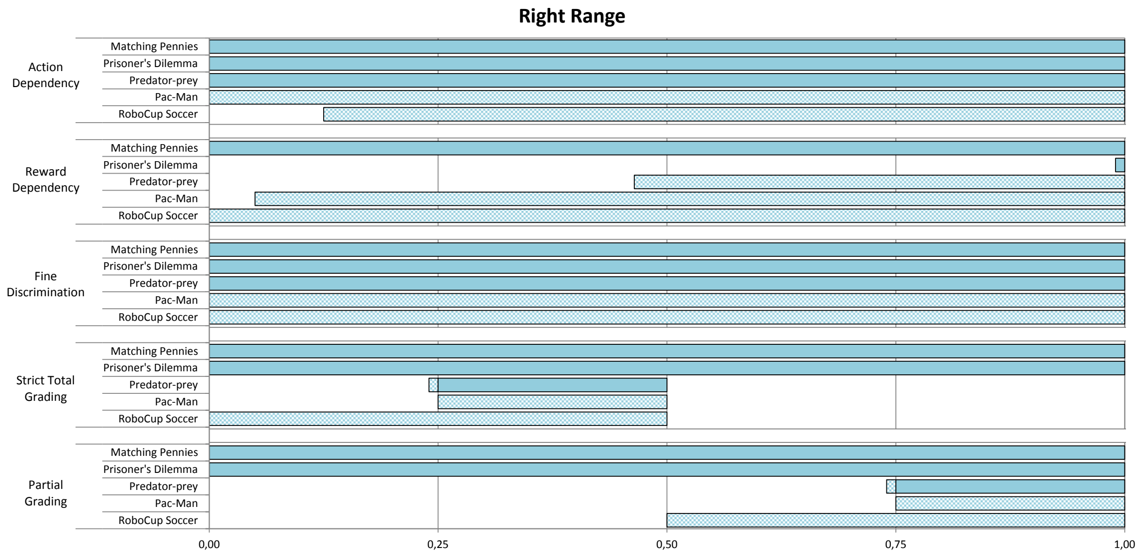


Figure 8.12: *Right* range for the quantitative properties which range is between 0 and 1 of the five multi-agent environments analysed, using uniform unit weights for w_{Π_e}, $w_{\dot{L}}$ and w_S, using $\Delta_{\mathbb{Q}}(a', b')$ instead of $\Delta_{\mathbb{Q}^+}(a, b)$ where a' and b' are the expected results of a and b respectively and $\Delta_{\mathcal{A}^+}(a, b)$ and $\Delta_{\mathbb{Q}}(a, b)$ return 1 if a and b are equal and 0 otherwise. We use an average of rewards as the utility function to calculate an agent's result. Lighter bands mean the values are not formally calculated, but an estimation is given.

a good selection of Π_o does improve their worst values for this property.

Finally, the partial grading gives us more information about the orderings. RoboCup Soccer has improved its range, but it is still worse than the predator-prey and Pac-Man, which now are clearly the best to obtain an ordering. However, matching pennies and the prisoner's dilemma cannot restrict their ranges, making them the worst of the five environments.

Overall, the *Right* range is more informative. Note that the Π_o that we use to calculate the values for each environment property is not necessarily the same. We just obtain the values locally for each property. That means that some values could not be achievable at the same time.

Lastly, in table 8.1 we see a summary of the qualitative properties to obtain a practical test for the five multi-agent environments.

	Boundedness	Team Symmetry	Validity	Reliability	Efficiency
Matching Pennies	✓	✗	✗	✗	✗
Prisoner's Dilemma	✓	✓	✗	✗	✓
Predator-prey	✓	✗	✗	✗	✗
Pac-Man	✗	✗	✗	✗	✗
RoboCup Soccer	✓	✗	✗	✗	✗

Table 8.1: Qualitative properties of the five multi-agent environments analysed.

As we have seen, almost all the environments have bounded rewards. This allows us to provide a bounded value for Υ and many properties. But only the prisoner's dilemma is team symmetric, so in order to evaluate an agent in the other environments, we need to evaluate them in all the agent slots. With respect to the validity property, no multi-agent environment is correctly evaluating social intelligence. Some of them are not sufficiently general, as happens with the matching pennies or the prisoners' dilemma. We have the opposite situation with RoboCup Soccer, where more abilities are evaluated and it seems difficult to isolate social intelligence from these other abilities. Finally, other environments can only evaluate the social intelligence to a certain degree, as happens in the predator-prey and Pac-Man. With respect to the reliability property, none of these environments is reliable, since the correlations between the results obtained by the evaluated agents and their expected results are too weak. Lastly, only the prisoner's dilemma is an efficient environment in terms of the number of repetitions needed to reach a reliable result (only 13 repetitions are needed). We need to average the results of more than 20 repetitions of the test for the other environments to obtain values close to their expected results.

From the analysis and comparison between the properties of the multi-agent environments we made in this chapter, we can provide some insights. First, we give some findings about the five multi-agent environments.

- As we have seen in our analysis, these multi-agent environments are typically covering anticipation well. Competitive anticipation is well covered in all the environments, while cooperative anticipation is not defined for the first two environments, but the last three are covering it very well. It also seems that the partial grading is generally well covered, so we can find some partial orderings between the evaluated agents.

- We can find some other properties that the multi-agent environments are not covering well. One example is the action dependency, where (almost) every environment analysed obtained the same poor ranges. This property is something which is not usually thought of when designing a multi-agent environment, but the possibility of having influence on the actions that other players can do is an interesting thing to consider when designing a multi-agent environment and, in particular, if we want to evaluate social intelligence. Another property which is not usually well covered is the slot result dependency, where these environments are typically only giving one value. We do not mean that the environments do not have good values but, instead, having a broader range of values would provide us with a more interesting scenario, where the relations of competition and cooperation between agents can change dynamically.

- As we can see from their qualitative properties, four of the five multi-agent environments we selected have some difficulties to be used in a practical test. None of them provides team symmetry to simplify the evaluation. The range of abilities required to succeed in these environments are not appropriate to be used in a test to evaluate social intelligence accurately, as well as the reliability of the environments is compromised even for slightly sophisticated agents (such as Q-learning agents) and only one of the five environments is efficient. At least, the environments usually provide bounded rewards, so we could calculate a bounded value of the (social) intelligence of the agents we want to evaluate.

- With these properties, we obtained different ranges of $General$, $Left$ and $Right$ for each of the five multi-agent environments. In addition, we could see that some little changes over the definition of a multi-agent environment (as occurs with the matching pennies and prisoner's dilemma) are clearly reflected with these properties. In fact, every kind of multi-agent environment will have particular ranges of values for these properties, with which we will be able to select the (social) environment(s) that best fits our goals (e.g. select an environment focused on anticipating other agents).

- As we have seen, a good selection of the set of populating agents Π_o is crucial in order to obtain appropriate social intelligence testbeds. But, how could we provide a Π_o which is appropriate to evaluate the social ability of an agent in a large number of multi-agent environments? This is a difficult task. When starting the evaluation, since the social intelligence of the evaluated agent is not known, it would be appropriate to use one Π_o which agents are not too socially smart during the first exercises. As long as the evaluation goes ahead and the social intelligence of the evaluee is better known, it would be better to use another Π_o whose agents are conforming to this level of social intelligence. In order to solve this problem, we could provide a unique Π_o and use some kind of distribution which is continuously evolving, giving more probability to the agents which are obtaining better results on these exercises, as in the spirit of the Darwin-Wallace distribution [48].

From the previous analysis, we can now distinguish the features of the multi-agent environments that could be reused for the design of better multi-agent environments to measure social intelligence more effectively. The first environment we saw is matching pennies, but it does not seem to have any particular useful feature from the properties we analysed. Next we saw the prisoner's dilemma environment, which is similar to the matching pennies with some little modifications. The prisoner's dilemma offers some nice features to include in a social intelligence test. First, we notice its capability to dynamically change the relation between the agent slots, providing a competitive and cooperative environment at the same time depending on the agents' actions. Second, the evaluated agent can obtain drastically different results when it interacts with very different populating agents from Π_o. And third, its team symmetry and efficiency makes this environment a good candidate to provide a simple test. The third environment was the predator-prey. This is the first environment that we analysed providing several agents in (at least) one team. From this team of agents, it is possible to anticipate cooperative agents in such a way that really good expected results can be achieved when it is done correctly, and an incorrect anticipation can provide really bad expected results. The same occurs while anticipating competitive agents, but in this case, both teams can anticipate the agents in the other team. Besides, we can obtain good partial gradings for the evaluated agents. However, Pac-Man and RoboCup Soccer do not provide significant features beyond those provided by predator-prey. At least, in RoboCup Soccer it is possible to exert a slight influence on other agents' actions, but only to some extent.

Conversely, we also distinguish those features that we do not want to appear in multi-agent environments for social intelligence testbeds. The first feature we distinguish is that none of these environments is valid to evaluate social intelligence since they are evaluating: 1) more abilities than necessary, as in RoboCup Soccer where the agents need their motion understanding to play the game; 2) not enough abilities, as in matching pennies and prisoner's dilemma where the agents cannot cooperate with agents in the same team; or 3) is only valid for lower levels of social intelligence, as in predator-prey and Pac-Man where the predators and ghosts can easily chase the prey and Pac-Man respectively once they reach a certain level of (social) intelligence. Also, none of the environments provide reliable results, probably due to that some little changes in the behaviour of the agents can create a butterfly effect, making the agents to obtain very different results (as occurs in the last three environments). Also, the multi-agent environments are typically not team symmetric, which will force to evaluate the agents in all agent slots, and we usually need many repetitions evaluating the agents in these environments to obtain a reliable result. In these five multi-agent environments, it is weird (if not impossible) to find a situation where an agent can directly influence on which actions are available for one (or more) of the other agents. The capability to directly influence on the available actions of the rest of agents could provide us a richer social environment, helping us to force the evaluated agent to consider the other agents if it wants to improve its performance. Also, some environments (matching pennies and prisoner's dilemma) are not suitable to let the agents anticipate cooperation within a team, since they do not provide the agents a team of agents to cooperate with them. Finally, when we see in more detail some environments, we notice that predator-prey and Pac-Man sometimes provide a really difficult/hostile scenario, where the predators and ghosts respectively have an enormous advantage to win the game.

Even if it is not the goal of this thesis (but a future work in our path to create a social intelligence test), we consider that a good multi-agent environment to measure social intelligence should have (at least) these characteristics: 1) It should provide two or more teams to interact with, and two or more agent slots on each team. By having this, the agents would be able to compete against the other team(s), cooperate with the agent(s) within their team and, if the environment provides more than two teams, cooperate with other teams to improve one's own results. This would provide anticipation to the environment, so the agents would be able to competitive and cooperative anticipate other agents; 2) The multi-agent environment should allow the agents to influence in some way the rewards obtained by the other agents, providing some reward dependency; 3) There should not be easy equilibria in the multi-agent environment. If such circumstance occurs, most of the agents (the intelligent ones) would always perform the same actions, which would limit the results obtained by the agents. Avoiding easy equilibria would provide the multi-agent environment with more discriminative power, reward dependency and grading for the agents; 4) The multi-agent environment should allow the agents to influence in some extent the actions that the other agents can perform, creating richer social situations and providing some action dependency; 5) There should be limited rewards that the agents can obtain and the payoff of the agents should only depend on the actions they perform. This would provide us a bounded and team symmetric multi-agent environment, which would be ideal to create a practical social test; And 6) the multi-agent environment should provide different kind of spaces where the agents can move. This would avoid the agents to specialise to a particular space, which would make the multi-agent environment more valid to evaluate social intelligence in a more broader way.

During this chapter, we have characterised all the multi-agent environments individually. However, we remind that we could have created a more general testbed by grouping them in a set for environments M and weighted the values obtained for each property with a weight for environments w_M. The values obtained for the properties would have been more representative of a more general testbed.

Finally, what can we say about the properties? Are they sufficient to characterise any testbed or multi-agent environment? How should they be used? Are the $Left$ and $Right$ ranges more meaningful than the $General$ ranges? Some insights below.

- With these properties we are able to obtain different values for each multi-agent environment, which gives us some idea about their strengths and weaknesses.

- We only used one evaluated agent from a set of evaluated agents Π_e interacting with a set of populating agents Π_o. More specialised properties (and even a definition of social collective intelligence) can be easily extended by, for example, dividing the set of populating agents Π_o into two sets (i.e. one for opponent players and one for team players) or, instead of making the evaluated agents interact in the environments in isolation, include together in the environment a group (or collective) of evaluated agents.

- There are also other issues which may not be covered on these properties, as for example communication between teammates. They do not provide information about misleading opponents, or the possibility of the agents to influence the actions of other agents on its benefit. Also, the properties do not show us the contribution of the agents to their teams' rewards, as well as the impact that their inclusion in the line-up has in the rewards of the other teams.

- These properties provide us some interesting information about the testbeds such as the fine and coarse discrimination, which give us a measure of their discriminative power. Other interesting properties are the action, reward and slot result dependency, giving us an idea about the existing dependency between the actions/rewards of the agents, and which relation between the agent slots is given more importance in the testbed or multi-agent environment (i.e. it is a more competitive- or cooperative-oriented testbed or multi-agent environment), and if this relation is static or can change during the evaluation.

- We used the $Left$ and *Right* ranges in order to compare for each property how a particular good or bad selection of Π_o can affect that property in a multi-agent environment. Conversely, an actual test should provide a unique Π_o to evaluate the agents, obtaining a unique range for each property. This selection of Π_o will (most probably) make the properties to barely look like the $Left$ or *Right* ranges we calculated for these five multi-agent environments, providing instead more varied ranges. Therefore, a comparison between some testbeds or multi-agent environments with their Π_o fixed would give us a more clear idea about their differences.

In this chapter, we have provided some examples of how the properties presented in chapter 7 can be used to obtain some useful information about the suitability of a testbed or multi-agent environment in order to evaluate social intelligence. The identification of suitable testbeds is crucial in order to determine whether the improvements on performance of socially intelligent agents are really general improvements towards the creation and development of such agents or, on the contrary, these improvements do not really correspond to a better understanding of social behaviours, but to a better understanding of other unrelated characteristics.

Chapter 9

Conclusions and Future Work

Social intelligence has been an important area of study in psychology, comparative cognition and economics for more than a century, and more recently, in artificial intelligence, notably in the area of multi-agent systems. However, despite the fact that other tests have been created to evaluate other cognitive abilities, nowadays it is still difficult to find a proper test to evaluate social intelligence. Also, current tests tend to be focussed on evaluating the ability of a single species and it is even more complicated to find a test to evaluate social intelligence that is applicable to machines. In fact, tests designed to succeed on a task that requires social intelligence usually also require other abilities to succeed in the task, making them not appropriate for the evaluation of social intelligence. This lack of general socially-oriented tests may be due to the absence of a precise (and formal) definition of social intelligence and theoretical tools to assess the suitability of testbeds for the evaluation of social intelligence.

In this thesis we have made a survey about the evaluation of social intelligence, we have analysed some repercussions about evaluating social intelligence using a general intelligence test, we have formalised a proper definition of social intelligence (as an aggregation of results on multi-agent environments interacting with other agents) and some useful social properties to characterise testbeds. In particular, the contributions are:

- We have analysed what social behaviour and its evaluation implies, reviewing what it means from several disciplines.
- We have considered various options for a definition, and we finally proposed a formal and parametrised definition of social intelligence. This definition formalises the notion of the performance of an agent interacting with any set of (social) agents in a variety of multi-agent environments. The definition allows us to provide different sets of environments and agents to perform the evaluation and parametrise them, allowing us to arrange the agents into teams to help us evaluate cooperation and/or competition for some environments. For this definition we propose two alternatives: one starting from the environments and including the agents afterwards, and the other way around, selecting the agents first and placing them later into the set of environments. We also have indicated how a test can be constructed using this definition.

- We have analysed the effect that the arrangement of agents into teams and the selection of agents to populate the environments have in a social test. We showed that the arrangement of agents into teams can foster cooperation and/or competition in some multi-agent environments, but is not enough to obtain an appropriate testbed or multi-agent environment. We also have seen the dependency of the agents used to populate the environment, so social agents are more or less effective not only depending on the environment, but also on the agents that populate that environment.

- We have proposed some properties along with their formal definitions in order to better analyse the appropriateness of a testbed or simply a multi-agent environment to evaluate social intelligence. Some of them (action dependency, reward dependency, fine and coarse discrimination and strict total and partial grading) are conceived as the degree of compliance that the testbed has about these properties, while others (slot result dependency and cooperative and cooperative anticipation) just indicate which type (competitive or cooperative) the testbed is more focussed on.

- With the properties proposed, we have analysed and compared several environments and games from artificial intelligence and game theory (where social intelligence has an important role) to see which properties they meet and which can be improved in order to evaluate social intelligence.

This definition of social intelligence along with the social properties proposed here are a first attempt in order to determine whether a testbed is useful for the assessment of social intelligence. As far as we know, this is the first approximation to provide a formal definition of social intelligence along with some useful properties to judge a certain testbed. A formal definition of social intelligence should facilitate us the development, comparison and improvement of socially intelligent agents. Besides, being able to determine the (social) characteristics that are present on a (social) testbed will help in the development of more general agents [32]. Also, they can help us to prevent some of the risks of creating artificial social intelligence, such as the creation of excessively competitive but uncooperative agents [80] or unethical or not empathic agents [92]. Indeed, further research will provide us with more information about which properties can be improved and information about other properties to complement the ones presented here.

There is still work to be done to achieve our goal of providing a practical test to measure social intelligence. The definition of social intelligence proposed here along with the social properties we propose to characterise a testbed or multi-agent environment and determine whether it is useful to evaluate social intelligence have some open features to be solved.

1. Our definition uses various sets and weights in the formula as parameters. How do we obtain an appropriate set of agents? We noted the importance of providing a good environment class where only social abilities are required to succeed on it but, how to provide this environment class? How to assess the appropriateness of environments? Which distributions and weights of agents and environments are appropriate to use in a social intelligence test?

2. It is not clear what utility function the agents must have to calculate their results. Should rewards be regulated using a discount factor as usual in reinforcement learning? Should we give more importance to later rewards, when the agents are supposed to understand how to behave? Or is it better to use an average, giving the same importance to all the rewards? Similar issues happen with the correlation function and divergence functions.

3. Every test should provide which level of difficulty it is evaluating. The difficulty of the environment could be calculated as, for example, the performance of a distribution of agents' policies with different levels of complexity, as presented in [43]. We postulated that the level of difficulty should be determined by the agents included in the environment (and their intelligence), the partition of agent slots that determines how teams are formed and the environment where the agent is evaluated. We determined how the first two parameters should influence the formula, but without indicating the formula itself. Should the level of intelligence of the agents weight more than the partition of agent slots, or should it be otherwise? How can we consider the environment in this formula?

4. We only evaluated one agent π interacting with a set of agents Π_o. But a more specialised definition of social (multi-agent or collective) intelligence and social properties about it can be easily extended by including more parameters as, for example, dividing the set of agents Π_o into two sets (i.e. one for team players and one for opponent players) or, instead of evaluating the agent in isolation, evaluating together a group (or collective) of agents.

5. We noticed that when a space is used in the definition of an environment, the agents necessarily require some spatial intelligence. In theory, we should calculate which part of the result comes from spatial intelligence and subtract it, but this seems very difficult. Alternatively, we could figure out a multi-agent environment class where no other abilities needed to interact will be really useful. Or at least with a wide variety such that, on average, these other abilities do not bias the result.

6. We could also use this definition of social intelligence by letting the agents cooperate placing them into individual teams. Life has taught us that alliances and coalitions can arise from several agents, even when they do not share the same objectives, to improve the chances of success. It would be interesting to analyse whether a test evaluating only competition can indirectly evaluate this spontaneous cooperation.

7. Social intelligence is linked to communication and language. We have not included any property or feature in the definitions to account for the presence of communication and language, or to facilitate that. Clearly, communication is possible through actions. Even language can be transmitted by the agent actions with a proper coding. However, this could be rendered more easily to agents. Nonetheless, any particular communication protocol can make the test non-universal. Instead we think that some extra actions that could be observed immediately by other agents (or by a subset of them) could be basic enough as a signal.

In this thesis we provided a formal definition that, when the multi-agent environments and populating agents used are properly selected, allows us to evaluate the social intelligence of any agent interacting in teams. Besides, the social properties we present go beyond the simple properties of game theory in many ways, opening a number of possibilities for the evaluation of testbeds and multi-agent environments. Of course, further research is needed to clarify many open questions, as happens with the arguably easier problem of non-social intelligence evaluation, which has not been fully achieved and it is still being investigated. The evaluation of social intelligence is still more convoluted. We have set some formal principles and made the difficulties arise. This thesis provides the basis of how we can evaluate social intelligence in a formal way following these principles.

Bibliography

[1] Samuel Barrett and Peter Stone. "An Analysis Framework for Ad Hoc Teamwork Tasks". In: *Proceedings of the Eleventh International Conference on Autonomous Agents and Multiagent Systems*. Vol. 2. Richland, South Carolina, United States of America: International Foundation for Autonomous Agents and Multiagent Systems, 2012, pp. 357–364.

[2] Marc G. Bellemare, Yavar Naddaf, Joel Veness, and Michael Bowling. "The Arcade Learning Environment: An Evaluation Platform for General Agents". In: *Journal of Artificial Intelligence Research* 47 (May 2013), 253–279.

[3] Miroslav Benda, Vasudevan Jagannathan, and Rajendra T. Dodhiawala. *On Optimal Cooperation of Knowledge Sources - An Empirical Investigation*. Tech. rep. Seattle, Washington, United States of America: Boeing Advanced Technology Center, Boeing Computing Services, July 1986.

[4] Darse Billings. "Thoughts on RoShamBo". In: *International Computer Games Association* 23.1 (Mar. 2000), pp. 3–8.

[5] Alfred Binet. "Nouvelles recherches sur la mesure du niveau intellectuel chez les enfants d'école". In: *L'Année Psychologique* 17.1 (1910), pp. 145–201.

[6] Alfred Binet and Thomas Simon. "Application des méthodes nouvelles au diagnostic du niveau intellectuel chez des enfants normaux et anormaux d'hospice et d'école primaire". In: *L'Année Psychologique* 11.1 (1904), pp. 245–336.

[7] Alfred Binet and Thomas Simon. "Méthodes nouvelles pour le diagnostic du niveau intellectuel des anormaux". In: *L'Année Psychologique* 11.1 (1904), pp. 191–244.

[8] Alfred Binet and Thomas Simon. "Sur la nécessité d'établir un diagnostic scientifique des états inférieurs de l'intelligence". In: *L'Année Psychologique* 11.1 (1904), pp. 163–190.

[9] Alfred Binet and Thomas Simon. "Le développement de l'intelligence chez les enfants". In: *L'Année Psychologique* 14.1 (1907), pp. 1–94.

[10] Alfred Binet and Thomas Simon. *The Development of Intelligence in Children (the Binet-Simon Scale)*. Translated by Elizabeth S. Kite from articles in L'Année Psychologique 11.1, pp. 163-336; 14.1, pp. 1-90; and 17.1, pp. 145-201. The Williams & Wilkins Company, May 1916, p. 355.

[11] Pierpaolo Di Bitonto, Maria Laterza, Teresa Roselli, and Veronica Rossano. "Evaluation of Multi-Agent Systems: Proposal and Validation of a Metric Plan". In: *Transactions on Computational Collective Intelligence VII*. Lecture Notes in Computer Science 7270 (Apr. 2012), pp. 198–221.

[12] Richard W. Byrne and Andrew Whiten, eds. *Machiavellian Intelligence: Social Expertise and the Evolution of Intellect in Monkeys, Apes, and Humans.* Oxford University Press, July 1988, p. 430.

[13] Jonathan M. Campbell and David M. McCord. “Measuring Social Competence with the Wechsler Picturei Arrangement and Comprehension Subtests”. In: *Assessment* 6.3 (Sept. 1999), pp. 215–223.

[14] Theodore Caplow. *Two Against One: Coalitions in Triads.* Englewood Cliffs, New Jersey, United States of America: Prentice-Hall, 1968, p. 183.

[15] Georgios Chalkiadakis and Craig Boutilier. “Sequentially optimal repeated coalition formation under uncertainty”. In: *Autonomous Agents and Multi-Agent Systems* 24.3 (May 2012), pp. 441–484.

[16] Dave Cliff and Geoffrey F. Miller. “Co-evolution of Pursuit and Evasion II: Simulation Methods and Results”. In: *From Animals to Animats 4: Proceedings of the Fourth International Conference on Simulation of Adaptive Behavior.* Complex Adaptive Systems. The MIT Press, Sept. 1996, pp. 506–515.

[17] Jörg Denzinger and Matthias Fuchs. “Experiments in Learning Prototypical Situations for Variants of the Pursuit Game”. In: *Proceedings of the Second International Conference on Multi-Agent Systems.* Menlo Park, California, United States of America: AAAI Press, 1996, pp. 48–55.

[18] Jörg Denzinger and Michael Kordt. “Evolutionary On-line Learning of Cooperative Behavior with Situation-Action-Pairs”. In: *Proceedings of the Fourth International Conference on Multi-Agent Systems.* IEEE Press, 2000, pp. 103–110.

[19] David L. Dowe and Alan R. Hájek. “A non-behavioural, computational extension to the Turing Test”. In: *Proceedings of the International Conference on Computational Intelligence and Multimedia Applications.* River Edge, New Jersey, United States of America: World Scientific, 1998, pp. 101–106.

[20] David L. Dowe and José Hernández-Orallo. “IQ tests are not for machines, yet”. In: *Intelligence* 40.2 (Mar. 2012), pp. 77–81.

[21] David L. Dowe and José Hernández-Orallo. “On interaction complexity, (space-time) resolution and intelligence”. In: *ReteCog II Workshop: Interaction.* Zaragoza, Spain, Jan. 2013, p. 5.

[22] David L. Dowe and José Hernández-Orallo. “How universal can an intelligence test be?” In: *Adaptive Behavior* 22.1 (Feb. 2014), pp. 51–69.

[23] David L. Dowe, José Hernández-Orallo, and Paramjit K. Das. “Compression and Intelligence: Social Environments and Communication”. In: *Proceedings of the Fourth Conference on Artificial General Intelligence.* Vol. 6830. Lecture Notes in Artificial Intelligence. Springer, 2011, pp. 204–211.

[24] Chris Drummond and Nathalie Japkowicz. “Warning: statistical benchmarking is addictive. Kicking the habit in machine learning”. In: *Journal of Experimental & Theoretical Artificial Intelligence* 22.1 (2010), pp. 67–80.

[25] Arpad E. Elo. *The rating of chessplayers, past and present.* Arco Pub., 1978, p. 206.

[26] Joshua M. Epstein. *Generative Social Science: Studies in Agent-Based Computational Modeling.* Princeton University Press, 2006, p. 356.

[27] Peter Frankl. “On a pursuit game on Cayley graphs”. In: *Combinatorica* 7.1 (Mar. 1987), pp. 67–70.

[28] Feng Fu, Martin A. Nowak, and Christoph Hauert. “Invasion and expansion of cooperators in lattice populations: Prisoner’s dilemma vs. snowdrift games”. In: *Journal of Theoretical Biology* 266.3 (Oct. 2010), pp. 358–366.

[29] Marcus Gallagher and Amanda Ryan. “Learning to Play Pac-Man: An Evolutionary, Rule-based Approach”. In: *Evolutionary Computation, 2003. CEC ’03.* Vol. 4. IEEE Press, 2003, pp. 2462–2469.

[30] Howard Gardner. *Frames of Mind: The Theory of Multiple Intelligences.* New York, New York, United States of America: Basic Books, 1983, p. 440.

[31] Michael Genesereth and Yngvi Björnsson. “The International General Game Playing Competition”. In: *AI Magazine* 34.2 (2013), pp. 107–111.

[32] Ted Goertzel. “The path to more general artificial intelligence”. In: *Journal of Experimental & Theoretical Artificial Intelligence* 26.3 (Apr. 2014), pp. 343–354.

[33] Dani Goldberg and Maja J. Matarić. “Interference as a Tool for Designing and Evaluating Multi-Robot Controllers”. In: *Proceedings of the Fourteenth National Conference on Artificial Intelligence.* Menlo Park, California, United States of America: AAAI Press, 1997, pp. 637–642.

[34] J. P. Guilford. *The Nature of Human Intelligence.* New York, New York, United States of America: McGraw-Hill, 1967, p. 538.

[35] Steve Hanks, Martha E. Pollack, and Paul R. Cohen. “Benchmarks, Test Beds Controlled Experimentation, and the Design of Agent Architectures”. In: *AI Magazine* 14.4 (1993), pp. 17–42.

[36] Alexander Harcourt and Frans B. M. de Waal, eds. *Coalitions and alliances in humans and other animals.* New York, New York, United States of America: Oxford University Press, 1992, p. 531.

[37] C. M. A. Haworth, M. J. Wright, M. Luciano, N. G. Martin, E. J. C. de Geus, C. E. M. van Beijsterveldt, M. Bartels, D. Posthuma, D. I. Boomsma, O. S. P. Davis, Y. Kovas, R. P. Corley, J. C. DeFries, J. K. Hewitt, R. K. Olson, S-A Rhea, S. J. Wadsworth, W. G. Iacono, M. McGue, L. A. Thompson, S. A. Hart, S. A. Petrill, D. Lubinski, and R. Plomin. “The heritability of general cognitive ability increases linearly from childhood to young adulthood”. In: *Molecular Psychiatry* 15.11 (Nov. 2010), pp. 1112–1120.

[38] Malte Helmert. “Complexity results for standard benchmark domains in planning”. In: *Artificial Intelligence* 143.2 (Feb. 2003), pp. 219–262.

[39] José Hernández-Orallo. “Beyond the Turing Test”. In: *Journal of Logic, Language and Information* 9.4 (Oct. 2000), pp. 447–465.

[40] José Hernández-Orallo. “On the Computational Measurement of Intelligence Factors”. In: *Measuring the Performance and Intelligence of Systems: Proceedings of the 2000 PerMIS Workshop.* National Institute of Standards and Technology, 2001, pp. 72–79.

[41] José Hernández-Orallo. “A (hopefully) Unbiased Universal Environment Class for Measuring Intelligence of Biological and Artificial Systems”. In: *Proceedings of the Third Conference on Artificial General Intelligence.* Vol. 11. Advances in Intelligent Systems Research. Atlantis Press, June 2010, pp. 182–183.

[42] José Hernández-Orallo. *Complexity distribution of agent policies.* Tech. rep. Valencia, Spain: Universitat Politècnica de València, Feb. 2013, p. 49.

[43] José Hernández-Orallo. “On environment difficulty and discriminating power”. In: *Autonomous Agents and Multi-Agent Systems* 29.3 (May 2015), pp. 402–454.

[44] José Hernández-Orallo and David L. Dowe. “Measuring universal intelligence: Towards an anytime intelligence test”. In: *Artificial Intelligence* 174.18 (Dec. 2010), pp. 1508–1539.

[45] José Hernández-Orallo and David L. Dowe. “On *Potential* Cognitive Abilities in the Machine Kingdom”. In: *Minds and Machines* 23.2 (May 2013), pp. 179–210.

[46] José Hernández-Orallo, David L. Dowe, and María Victoria Hernández-Lloreda. “Universal psychometrics: Measuring cognitive abilities in the machine kingdom”. In: *Cognitive Systems Research* 27 (Mar. 2014), pp. 50–74.

[47] José Hernández-Orallo and Neus Minaya-Collado. “A Formal Definition of Intelligence Based on an Intensional Variant of Kolmogorov Complexity”. In: *Proceedings of the International ICSC Symposium on Engineering of Intelligent Systems.* ICSC Press, 1998, pp. 146–163.

[48] José Hernández-Orallo, David L. Dowe, Sergio España, María Victoria Hernández-Lloreda, and Javier Insa-Cabrera. “On More Realistic Environment Distributions for Defining, Evaluating and Developing Intelligence”. In: *Proceedings of the Fourth Conference on Artificial General Intelligence.* Vol. 6830. Lecture Notes in Artificial Intelligence. Springer, 2011, pp. 82–91.

[49] José Hernández-Orallo, Javier Insa-Cabrera, David L. Dowe, and Bill Hibbard. “Turing Machines and Recursive Turing Tests”. In: *Revisiting Turing and his Test: Comprehensiveness, Qualia, and the Real World.* The Society for the Study of Artificial Intelligence and Simulation of Behaviour, 2012, pp. 28–33.

[50] José Hernández-Orallo, Javier Insa-Cabrera, David L. Dowe, and Bill Hibbard. “Turing Tests with Turing Machines”. In: *Alan Turing Centenary.* Vol. 10. EPiC. EasyChair, 2012, pp. 140–156.

[51] Esther Herrmann, Josep Call, María Victoria Hernández-Lloreda, Brian Hare, and Michael Tomasello. “Humans Have Evolved Specialized Skills of Social Cognition: The Cultural Intelligence Hypothesis”. In: *Science* 317.5843 (Sept. 2007), pp. 1360–1366.

[52] Esther Herrmann, María Victoria Hernández-Lloreda, Josep Call, Brian Hare, and Michael Tomasello. “The Structure of Individual Differences in the Cognitive Abilities of Children and Chimpanzees”. In: *Psychological Science* 21.1 (Jan. 2010), pp. 102–110.

[53] Bill Hibbard. “Adversarial Sequence Prediction”. In: *Proceedings of the First Conference on Artificial General Intelligence.* Vol. 171. Amsterdam, Netherlands: IOS Press, Feb. 2008, pp. 399–403.

[54] Bill Hibbard. “Measuring Agent Intelligence via Hierarchies of Environments”. In: *Proceedings of the Fourth Conference on Artificial General Intelligence.* Vol. 6830. Lecture Notes in Artificial Intelligence. Springer, 2011, pp. 303–308.

[55] Thelma Hunt. “The measurement of social intelligence”. In: *Journal of Applied Psychology* 12.3 (June 1928), pp. 317–334.

[56] Javier Insa-Cabrera, José Luis Benacloch-Ayuso, and José Hernández-Orallo. “On Measuring Social Intelligence: Experiments on Competition and Cooperation”. In: *Proceedings of the Fifth Conference on Artificial General Intelligence.* Vol. 7716. Lecture Notes in Computer Science. Springer, 2012, pp. 126–135.

[57] Javier Insa-Cabrera, David L. Dowe, and José Hernández-Orallo. “Evaluating a Reinforcement Learning Algorithm with a General Intelligence Test”. In: *Advances in Artificial Intelligence: Proceedings of the Fourteenth Conference of the Spanish Association for Artificial Intelligence, CAEPIA 2011.* Vol. 7023. Lecture Notes in Artificial Intelligence. Springer, 2011, pp. 1–11.

[58] Javier Insa-Cabrera and José Hernández-Orallo. “Interaction settings for measuring (social) intelligence in multi-agent systems”. In: *ReteCog II Workshop: Interaction.* Zaragoza, Spain, Jan. 2013, p. 4.

[59] Javier Insa-Cabrera and José Hernández-Orallo. “A formal, parametrised setting to evaluate social intelligence”. In: *Computational Intelligence* (2015). Submitted.

[60] Javier Insa-Cabrera and José Hernández-Orallo. “Characterisation of Social Intelligence Testbeds using Quantitative Properties”. In: *Autonomous Agents and Multi-Agent Systems* (2015). Submitted.

[61] Javier Insa-Cabrera and José Hernández-Orallo. “Instrumental Properties of Social Testbeds”. In: *Proceedings of the Eighth Conference on Artificial General Intelligence.* Vol. 9205. Lecture Notes in Artificial Intelligence. Springer, 2015, pp. 101–110.

[62] Javier Insa-Cabrera, David L. Dowe, Sergio España, María Victoria Hernández-Lloreda, and José Hernández-Orallo. “Comparing Humans and AI Agents”. In: *Proceedings of the Fourth Conference on Artificial General Intelligence.* Vol. 6830. Lecture Notes in Artificial Intelligence. Springer, 2011, pp. 122–132.

[63] Javier Insa-Cabrera, José Hernández-Orallo, David L. Dowe, Sergio España, and María Victoria Hernández-Lloreda. “The ANYNT Project Intelligence Test: Λ_{one}”. In: *Revisiting Turing and his Test: Comprehensiveness, Qualia, and the Real World.* The Society for the Study of Artificial Intelligence and Simulation of Behaviour, 2012, pp. 20–27.

[64] Michael Kaisers and Karl Tuyls. “Frequency Adjusted Multi-agent Q-learning”. In: *Proceedings of the Ninth International Conference on Autonomous Agents and Multiagent Systems.* Vol. 1. Richland, South Carolina, United States of America: International Foundation for Autonomous Agents and Multiagent Systems, 2010, pp. 309–316.

[65] Gal A. Kaminka, Ian Frank, Katsuto Arai, and Kumiko Tanaka-Ishii. “Performance Competitions as Research Infrastructure: Large Scale Comparative Studies of Multi-Agent Teams”. In: *Autonomous Agents and Multi-Agent Systems* 7.1 (July 2003), pp. 121–144.

[66] David Keil and Dina Goldin. “Indirect Interaction in Environments for Multi-agent Systems”. In: *Environments for Multi-Agent Systems II: Second International Workshop, E4MAS 2005.* Vol. 3830. Lecture Notes in Computer Science. Springer, 2006, pp. 68–87.

[67] Hiroaki Kitano, Minoru Asada, Yasuo Kuniyoshi, Itsuki Noda, and Eiichi Osawa. “RoboCup: The Robot World Cup Initiative”. In: *Proceedings of the First International Conference on Autonomous Agents.* New York, New York, United States of America: ACM Press, 1997, pp. 340–347.

[68] Shane Legg. “Machine Super Intelligence”. PhD thesis. Lugano, Switzerland: University of Lugano, June 2008, p. 184.

[69] Shane Legg and Marcus Hutter. “Universal Intelligence: A Definition of Machine Intelligence”. In: *Minds and Machines* 17.4 (Dec. 2007), pp. 391–444.

[70] Shane Legg and Joel Veness. “An Approximation of the Universal Intelligence Measure”. In: *Algorithmic Probability and Friends: Bayesian Prediction and Artificial Intelligence.* Vol. 7070. Lecture Notes in Artificial Intelligence. Springer, 2013, pp. 236–249.

[71] Ming Li and Paul M. B. Vitányi. *An Introduction to Kolmogorov Complexity and Its Applications.* 3rd ed. Springer-Verlag, 2008, p. 790.

[72] Vladimir Lifschitz. “Benchmark problems for formal nonmonotonic reasoning”. In: *Non-Monotonic Reasoning: Proceedings of the Second International Workshop.* Vol. 346. Lecture Notes in Computer Science. Springer, May 1989, pp. 202–219.

[73] Michael L. McKinney and Robert M. Schoch. *Environmental Science: Systems and Solutions.* Jones & Bartlett, 2003, p. 560.

[74] Sara Mitri, Steffen Wischmann, Dario Floreano, and Laurent Keller. “Using robots to understand social behaviour”. In: *Biological Reviews* 88.1 (Feb. 2013), pp. 31–39.

[75] Fred A. Moss and Thelma Hunt. “Are You Socially Intelligent?” In: *Scientific American* 137.2 (Aug. 1927), pp. 108–110.

[76] Fred A. Moss, Thelma Hunt, and K. T. Omwake. *Manual for the Social Intelligence Test, Revised Form.* Washington D. C., United States of America: The Center for Psychological Service, 1949.

[77] Fred A. Moss, Thelma Hunt, K. T. Omwake, and M. M. Ronning. *Social Intelligence Test.* Washington D. C., United States of America: The Center for Psychological Service, 1927.

[78] Fred A. Moss, Thelma Hunt, K. T. Omwake, and L. G. Woodward. *Manual for the George Washington University Series Social Intelligence Test.* Washington D. C., United States of America: The Center for Psychological Service, 1955.

[79] Lisa J. Moya and Andreas Tolk. “Towards a Taxonomy of Agents and Multi-Agent Systems”. In: *Proceedings of the 2007 Spring Simulation Multiconference.* Vol. 2. San Diego, California, United States of America: Society for Computer Simulation International, Mar. 2007, pp. 11–18.

[80] Vincent C. Müller. “Risks of general artificial intelligence”. In: *Journal of Experimental & Theoretical Artificial Intelligence* 26.3 (Apr. 2014), pp. 297–301.

[81] Roger B. Myerson. *Game Theory: Analysis of Conflict.* Harvard University Press, Sept. 1997, p. 600.

[82] Majid Nili Ahmadabadi and Masoud Asadpour. “Expertness Based Cooperative Q-Learning”. In: *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics* 32.1 (Feb. 2002), pp. 66–76.

[83] Geoff Nitschke. “Co-evolution of cooperation in a Pursuit Evasion Game”. In: *Proceedings IROS 2003, International Conference on Intelligent Robots and Systems.* Vol. 2. IEEE Press, 2003, pp. 2037–2042.

[84] Jum C. Nunnally. *Psychometric Theory*. 2nd ed. New York, New York, United States of America: McGraw-Hill, 1978, p. 701.

[85] Martin J. Osborne. *An Introduction to Game Theory*. Oxford University Press, Aug. 2003, p. 560.

[86] Maureen O'Sullivan, J. P. Guilford, and R. de Mille. *The Measurement of Social Intelligence*. Los Angeles, California, United States of America: University of Southern California, 1965, p. 39.

[87] Liviu Panait and Sean Luke. "Cooperative Multi-Agent Learning: The State of the Art". In: *Autonomous Agents and Multi-Agent Systems* 11.3 (Nov. 2005), pp. 387–434.

[88] Matthew S. Panizzon, Eero Vuoksimaa, Kelly M. Spoon, Kristen C. Jacobson, Michael J. Lyons, Carol E. Franz, Hong Xian, Terrie Vasilopoulos, and William S. Kremen. "Genetic and environmental influences on general cognitive ability: Is g a valid latent construct?" In: *Intelligence* 43 (Mar. 2014), pp. 65–76.

[89] Arvind Parkhe. "Strategic Alliance Structuring: A Game Theoretic and Transaction Cost Examination of Interfirm Cooperation". In: *The Academy of Management Journal* 36.4 (Aug. 1993), pp. 794–829.

[90] Barney Pell. "A Strategic Metagame Player for General Chess-Like Games". In: *Computational Intelligence* 12.1 (Feb. 1996), pp. 177–198.

[91] Robert Plomin and Frank M. Spinath. "Genetics and general cognitive ability (g)". In: *Trends in Cognitive Sciences* 6.4 (Apr. 2002), pp. 169–176.

[92] Alexey Potapov and Sergey Rodionov. "Universal Empathy and Ethical Bias for Artificial General Intelligence". In: *Journal of Experimental & Theoretical Artificial Intelligence* 26.3 (Apr. 2014), pp. 405–416.

[93] William Poundstone. *Prisoner's Dilemma*. Anchor Books, Feb. 1993, p. 294.

[94] Diego Pérez, Spyridon Samothrakis, Julian Togelius, Tom Schaul, Simon M. Lucas, Adrien Couëtoux, Jerry Lee, Chong-U Lim, and Tommy Thompson. "The 2014 General Video Game Playing Competition". In: *IEEE Transactions on Computational Intelligence and AI in Games* (2015). To Appear.

[95] Talal Rahwan, Tomasz Michalak, Michael Wooldridge, and Nicholas R. Jennings. "Anytime coalition structure generation in multi-agent systems with positive or negative externalities". In: *Artificial Intelligence* 186 (July 2012), pp. 95–122.

[96] Brandon Rohrer. "Accelerating progress in Artificial General Intelligence: Choosing a benchmark for natural world interaction". In: *Journal of Artificial General Intelligence* 2.1 (Dec. 2010), pp. 1–28.

[97] Avi Rosenfeld, Gal A. Kaminka, and Sarit Kraus. "A Study of Scalability Properties in Robotic Teams". In: *Coordination of Large-Scale Multiagent Systems*. Ed. by Paul Scerri, Régis Vincent, and Roger Mailler. Springer, 2006. Chap. 2, pp. 27–51.

[98] Avi Rosenfeld, Gal A. Kaminka, Sarit Kraus, and Onn Shehory. "A study of mechanisms for improving robotic group performance". In: *Artificial Intelligence* 172.6-7 (Apr. 2008), pp. 633–655.

[99] Alvin E. Roth, ed. *The Shapley value: Essays in honor of Lloyd S. Shapley*. Cambridge University Press, 1988, p. 338.

[100] G. A. Rummery and M. Niranjan. *On-Line Q-Learning Using Connectionist Systems.* Tech. rep. Cambridge, England: Cambridge University Engineering Department, Sept. 1994, p. 20.

[101] John Rust, John H. Miller, and Richard G. Palmer. “Behavior of Trading Automata in a Computerized Double Auction Market”. In: *The Double Auction Market: Institutions, Theories, and Evidence.* Ed. by Daniel Friedman and John Rust. Vol. 15. Studies in the Sciences of Complexity. Addison-Wesley, 1992. Chap. 6, pp. 155–198.

[102] Sara J. Shettleworth. *Fundamentals of Comparative Cognition.* Oxford University Press, Mar. 2012, p. 192.

[103] Yoav Shoham and Kevin Leyton-Brown. *Multiagent Systems: Algorithmic, Game-Theoretic, and Logical Foundations.* Cambridge University Press, Dec. 2008.

[104] Raymond J. Solomonoff. “A Formal Theory of Inductive Inference. Part I”. In: *Information and Control* 7.1 (Mar. 1964), pp. 1–22.

[105] Raymond J. Solomonoff. “A Formal Theory of Inductive Inference. Part II”. In: *Information and Control* 7.2 (June 1964), pp. 224–254.

[106] Charles E. Spearman. ““General Intelligence”, Objectively Determined and Measured”. In: *The American Journal of Psychology* 15.2 (Apr. 1904), pp. 201–292.

[107] Esben H. Østergaard, Gaurav S. Sukhatme, and Maja J. Matarić. “Emergent Bucket Brigading: A simple mechanism for improving performance in multi-robot constrained-space foraging tasks”. In: *Proceedings of the Fifth international conference on Autonomous agents.* New York, New York, United States of America: ACM Press, 2001, pp. 29–30.

[108] Peter Stone, Gal A. Kaminka, Sarit Kraus, and Jeffrey S. Rosenschein. “Ad Hoc Autonomous Agent Teams: Collaboration without Pre-Coordination”. In: *Proceedings of the Twenty-Fourth Conference on Artificial Intelligence.* AAAI Press, 2010, pp. 1504–1509.

[109] Richard S. Sutton. “Learning to Predict by the Methods of Temporal Differences”. In: *Machine Learning* 3.1 (Aug. 1988), pp. 9–44.

[110] Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction.* Cambridge, Massachusetts, United States of America: The MIT Press, May 1998, p. 338.

[111] Ming Tan. “Multi-Agent Reinforcement Learning: Independent vs. Cooperative Agents”. In: *Proceedings of the Tenth International Conference on Machine Learning.* Morgan Kaufmann, June 1993, pp. 330–337.

[112] Kristinn R. Thórisson, Jordi Bieger, Stephan Schiffel, and Deon Garrett. “Towards Flexible Task Environments for Comprehensive Evaluation of Artificial Intelligent Systems and Automatic Learners”. In: *Proceedings of the Eighth Conference on Artificial General Intelligence.* Vol. 9205. Lecture Notes in Artificial Intelligence. Springer, 2015, pp. 187–196.

[113] Edward L. Thorndike. “Intelligence and its uses”. In: *Harper’s Magazine* 140.2 (Jan. 1920), pp. 227–235.

[114] Robert L. Thorndike and Saul Stein. “An evaluation of the attempts to measure social intelligence”. In: *Psychological Bulletin* 34.5 (May 1937), pp. 275–285.

[115] Alan M. Turing. “Computing Machinery and Intelligence”. In: *Mind* 59.236 (Oct. 1950), pp. 433–460.

[116] Andrew J. Turner and Julian F. Miller. “Cartesian Genetic Programming encoded Artificial Neural Networks: A Comparison using Three Benchmarks”. In: *Proceedings of the Fifteenth Annual Conference on Genetic and Evolutionary Computation.* New York, New York, United States of America: ACM Press, 2013, pp. 1005–1012.

[117] Paul Valckenaers, John Sauter, Carles Sierra, and Juan Antonio Rodriguez Aguilar. “Applications and environments for multi-agent systems”. In: *Autonomous Agents and Multi-Agent Systems* 14.1 (Feb. 2007), pp. 61–85.

[118] Joel Veness, Kee Siong Ng, Marcus Hutter, William Uther, and David Silver. “A Monte-Carlo AIXI Approximation”. In: *Journal of Artificial Intelligence Research* 40 (Jan. 2011), pp. 95–142.

[119] Philip E. Vernon. “Some Characteristics of the Good Judge of Personality”. In: *The Journal of Social Psychology* 4.1 (1933), pp. 42–57.

[120] Pei Wang. “What Do You Mean by “A””. In: *Proceedings of the First Conference on Artificial General Intelligence.* Vol. 171. Amsterdam, Netherlands: IOS Press, Feb. 2008, pp. 362–373.

[121] Pei Wang. “The evaluation of AGI systems”. In: *Proceedings of the Third Conference on Artificial General Intelligence.* Vol. 11. Advances in Intelligent Systems Research. Atlantis Press, June 2010, pp. 164–169.

[122] Edward A. Wasserman and Thomas R. Zentall, eds. *Comparative Cognition: Experimental Explorations of Animal Intelligence.* Oxford University Press, Apr. 2009, p. 720.

[123] Christopher J. C. H. Watkins. “Learning from Delayed Rewards”. PhD thesis. Cambridge, England: King’s College, May 1989, p. 234.

[124] Christopher J. C. H. Watkins and Peter Dayan. “Q-Learning”. In: *Machine Learning* 8.3-4 (May 1992), pp. 279–292.

[125] David Wechsler. *The Measurement of Adult Intelligence.* The Williams & Wilkins Company, 1939, p. 229.

[126] David Wechsler. *The Wechsler-Bellevue Intelligence Scale, Form II.* The Psychological Corporation, 1946, p. 96.

[127] David Wechsler. *Wechsler Intelligence Scale for Children.* New York, New York, United States of America: The Psychological Corporation, 1949.

[128] David Wechsler. *Manual for the Wechsler Adult Intelligence Scale.* Oxford, England: The Psychological Corporation, 1955, p. 110.

[129] David Wechsler. *The Measurement and Appraisal of Adult Intelligence.* 4th ed. Baltimore, Maryland, United States of America: The Williams & Wilkins Company, 1958, p. 324.

[130] David Wechsler. *Wechsler Intelligence Scale for Children - Revised.* San Antonion, Texas, United States of America: The Psychological Corporation, 1974, p. 191.

[131] David Wechsler. *Wechsler Adult Intelligence Scale - Revised.* The Psychological Corporation, 1981.

[132] David Wechsler. *Wechsler Intelligence Scale for Children - Third Edition.* San Antonion, Texas, United States of America: The Psychological Corporation, 1991.

[133] David Wechsler. *Wechsler Adult Intelligence Scale - Third Edition.* San Antonion, Texas, United States of America: The Psychological Corporation, 1997.

[134] David Wechsler. *Wechsler Intelligence Scale for Children - Fourth Edition.* San Antonion, Texas, United States of America: The Psychological Corporation, 2003.

[135] David Wechsler. *Wechsler Adult Intelligence Scale - Fourth Edition.* NCS Pearson, 2008.

[136] Jörgen W. Weibull. *Evolutionary Game Theory.* Cambridge, Massachusetts, United States of America: The MIT Press, 1995.

[137] Michael P. Wellman, Peter R. Wurman, Kevin O'Malley, Roshan Bangera, Shou–de Lin, Daniel Reeves, and William E. Walsh. "Designing the Market Game for a Trading Agent Competition". In: *Internet Computing, IEEE* 5.2 (Mar. 2001), pp. 43–51.

[138] Danny Weyns, H. Van Dyke Parunak, Fabien Michel, Tom Holvoet, and Jacques Ferber. "Environments for Multiagent Systems State-of-the-Art and Research Challenges". In: *Environments for Multi-Agent Systems: First International Workshop, E4MAS 2004.* Vol. 3374. Lecture Notes in Computer Science. Springer, 2005, pp. 1–47.

[139] Danny Weyns, Michael Schumacher, Alessandro Ricci, Mirko Viroli, and Tom Holvoet. "Environments in multiagent systems". In: *The Knowledge Engineering Review* 20.2 (June 2005), pp. 127–141.

[140] Marco A. Wiering. "QV(λ)-learning: A New On-policy Reinforcement Learning Algorithm". In: *Proceedings of the Seventh European Workshop on Reinforcement Learning.* 2005, pp. 17–18.

[141] Paul L. Williams and Randall D. Beer. *Generalized Measures of Information Transfer.* Tech. rep. Bloomington, Indiana, United States of America: Indiana University, Feb. 2011, p. 6.

[142] Dan Xiao and Ah-Hwee Tan. "Cooperative reinforcement learning in topology-based multi-agent systems". In: *Autonomous Agents and Multi-Agent Systems* 26.1 (Jan. 2013), pp. 86–119.

[143] Zhanna V. Zatuchna and Anthony Bagnall. "Learning Mazes with Aliasing States: An LCS Algorithm with Associative Perception". In: *Adaptive Behavior* 17.1 (Feb. 2009), pp. 28–57.

[144] Byoung-Tak Zhang and Dong-Yeon Cho. "Co-evolutionary Fitness Switching: Learning Complex Collective Behaviors Using Genetic Programming". In: *Advances in Genetic Programming.* Ed. by Lee Spector, William B. Langdon, Una-May O'Reilly, and Peter J. Angeline. Vol. 3. The MIT Press, 1999. Chap. 18, pp. 425–445.

[145] Jacob Ziv and Abraham Lempel. "A Universal Algorithm for Sequential Data Compression". In: *IEEE Transactions on Information Theory* 23.3 (May 1977), pp. 337–343.

Appendices

The appendices give the proofs showing how we obtained the values of the properties for some multi-agent environments analysed in chapter 8.

Before starting with each of the multi-agent environments, we prove a lemma that is helpful for the $Left$ and $Right$ ranges.

We could calculate $Left$ and $Right$ using Π_e and Π_o with a high number of agents. However, the more agents we include the more difficult the calculation becomes. Instead of this, and in order to simplify calculations, we can just use the minimum necessary number of agents in Π_e and Π_o for that property to obtain the maximum/minimum value following the idea on lemma 1:

Lemma 1. *In order to calculate $Left/Right$ maximum/minimum value for a property $Prop$, the length of the set of evaluated agents $|\Pi_e|$ and the length of the set of populating agents $|\Pi_o|$ can be respectively equal to the minimum number of evaluated agents n and populating agents m needed to calculate $Prop$.*

Proof. Let $\Pi_e = \{\pi_1, \ldots, \pi_n, \ldots, \pi_p\}$ be the set of evaluated agents with weight w_{Π_e} in a multi-agent environment μ with weight of agent slots w_S using a set of populating agents Π_o and $w_{\dot{L}}$ as a weight for agent line-up patterns.

Let us suppose that we want to calculate the value for a property $Prop$ which needs n evaluated agents to be defined, its definition calculates first the value for each evaluated agent π and then these values are weighted using $w_{\Pi_e}(\pi)$ to provide the property value. Following this definition we obtain a list of values $(v_1, \ldots, v_n, \ldots, v_p)$ (one for each evaluated agent) that are weighted with w_{Π_e} to obtain the property value v. If we get rid of the evaluated agent which obtains the maximum value for $Prop$ and we normalise w_{Π_e} to sum 1 after removing the weight for this evaluated agent, then the new property value v' will always be lower than v. We can repeat this process until n agents remain in Π_e (i.e. $|\Pi_e| = n$) to obtain v_{min} for this set of evaluated agents. An analogous process applies to obtain v_{max} by getting rid of the evaluated agent which obtains the minimum value for $Prop$.

The same reasoning applies to the properties that calculate each value using a pair of evaluated agents, but in this case we get rid of the agent whose sum of values (the values of the pairs where this agent is used) is highest/lowest. Also, the same reasoning applies for Π_o. □

Appendix A

Matching Pennies Properties

In this section we prove how we obtained the values for the properties for the matching pennies environment (section 2.1.1). We use uniform unit weights for w_{Π_e}, $w_{\dot{L}}$ and w_S. To calculate some of the values for the properties, we make use of lemma 2.

Lemma 2. In the matching pennies environment and for every agent slot, introducing a random agent π_r in an agent line-up always provides an expected result equal to 0 for both agents.

Proof. A random agent π_r has a probability of $p^k_{r,h} = p^k_{r,t} = \frac{1}{2}$ to perform both Head and Tail at time step k. Let us denote with π_s the agent that π_r is interacting with, and denote with $p^1_{s,h}$ the probability of performing Head and $p^1_{s,t} = 1 - p^1_{s,h}$ the probability of performing Tail at the first time step for π_s.

To calculate the expected reward of an agent, we sum the possible rewards that this agent can obtain multiplied by the probability that these rewards occur. When we calculate the expected reward for π_r for the first time step in the matching pennies environment μ in any agent slot i, we obtain:

$$\forall i : R^1_i(\mu[\dot{l} \overset{i}{\leftarrow} \pi_r]) = p^1_{r,h} \left(p^1_{s,h} \times r^1_{h,h,i} + p^1_{s,t} \times r^1_{h,t,i}\right) + p^1_{r,t} \left(p^1_{s,h} \times r^1_{t,h,i} + p^1_{s,t} \times r^1_{t,t,i}\right)$$

where $\dot{l}$ contains π_s in agent slot j (having $i \neq j$), $r^k_{a_1,a_2,i}$ is the reward that the agent in agent slot i obtains at time step k when one agent performs a_1 and the other agent performs a_2.

From the matching pennies' payoff matrix (table 2.1), we can see that for every agent slot i, $r_{h,h,i} = r_{t,t,i}$ and $r_{h,t,i} = r_{t,h,i}$, so we name them $r_{e,i}$ and $r_{d,i}$ respectively. We can also see that the reward values are the inverse of each other, having $r_{d,i} = -r_{e,i}$. Renaming the rewards in the formula and rearranging it we obtain:

$$\begin{aligned}
\forall i : R_i^1(\mu[\dot{l} \overset{i}{\leftarrow} \pi_r]) &= p_{r,h}^1 \left(p_{s,h}^1 \times r_{e,i} + p_{s,t}^1 \times r_{d,i}\right) + p_{r,t}^1 \left(p_{s,h}^1 \times r_{d,i} + p_{s,t}^1 \times r_{e,i}\right) = \\
&= p_{r,h}^1 \left(p_{s,h}^1 \times r_{e,i} + p_{s,t}^1 \times (-r_{e,i})\right) + p_{r,t}^1 \left(p_{s,h}^1 \times (-r_{e,i}) + p_{s,t}^1 \times r_{e,i}\right) = \\
&= p_{r,h}^1 \left(r_{e,i} \times \left(p_{s,h}^1 - p_{s,t}^1\right)\right) + p_{r,t}^1 \left((-r_{e,i}) \times \left(p_{s,h}^1 - p_{s,t}^1\right)\right) = \\
&= p_{r,h}^1 \left(r_{e,i} \times \left(p_{s,h}^1 - p_{s,t}^1\right)\right) - p_{r,t}^1 \left(r_{e,i} \times \left(p_{s,h}^1 - p_{s,t}^1\right)\right) = \\
&= \left(p_{r,h}^1 - p_{r,t}^1\right) \times \left(r_{e,i} \times \left(p_{s,h}^1 - p_{s,t}^1\right)\right)
\end{aligned}$$

And since π_r gives the same probability to both Head and Tail (i.e. $p_{r,h}^1 = p_{r,t}^1 = \frac{1}{2}$) we obtain the following expected reward:

$$\forall i : R_i^1(\mu[\dot{l} \overset{i}{\leftarrow} \pi_r]) = \left(\frac{1}{2} - \frac{1}{2}\right) \times \left(r_{e,i} \times \left(p_{s,h}^1 - p_{s,t}^1\right)\right) = 0$$

We calculated the expected reward for the first time step. At this point π_s could change its behaviour depending on what happened on the previous time step, using different probabilities $p_{s,h}^2$ and $p_{s,t}^2$ for time step 2. But note that it does not matter which probabilities $p_{s,h}^n$ and $p_{s,t}^n$ we use, the expected reward will still be 0. Since all the expected rewards are 0, any utility function using these expected rewards obtains an expected result equal to 0. Obviously, since this is a zero-sum game, when π_r obtains an expected result of 0, π_s obtains the same expected result of 0. □

A.1 Action Dependency

We start with the action dependency (AD) property. As given in section 7.2.1, we want to know if the evaluated agents behave differently depending on which agent line-up they interact with. We use $\Delta_{\mathcal{A}^+}(a, b) = 0$ if distributions a and b are equal and 1 otherwise.

Proposition 2. $General_{min}$ for the action dependency (AD) property is equal to 0 for the matching pennies environment.

Proof. To find $General_{min}$ (equation 8.1), we need to find a pair $\langle \Pi_e, \Pi_o \rangle$ that minimises the property value as much as possible. We can have this situation by selecting $\Pi_e = \{\pi_t\}$ and $\Pi_o = \{\pi_{h1}, \pi_{h2}\}$ (a π_h agent always performs Head and a π_t agent always performs Tail).

Following definition 17 we obtain the AD value for this $\langle \Pi_e, \Pi_o \rangle$. Since the multi-agent environment is not team symmetric, we need to calculate this property value for every agent slot. Also, since $|\Pi_e| = 1$ we just need to calculate this property value for one evaluated agent. Following definition 16 we calculate the AD value for the evaluated agent $\pi_t \in \Pi_e$ and each agent slot. We start with agent slot 1:

$$\begin{aligned}
AD_1(\pi_t, \Pi_o, w_{\dot{L}}, \mu) &= \eta_{\dot{L}^2} \sum_{\dot{u}, \dot{v} \in \dot{L}_{-1}^{N(\mu)}(\Pi_o) | \dot{u} \neq \dot{v}} w_{\dot{L}}(\dot{u}) w_{\dot{L}}(\dot{v}) \Delta_{\mathcal{A}^+}(\breve{A}_1(\mu[\dot{u} \overset{1}{\leftarrow} \pi_t]), \breve{A}_1(\mu[\dot{v} \overset{1}{\leftarrow} \pi_t])) = \\
&= 2 \frac{2}{1} \frac{1}{2} \frac{1}{2} \Delta_{\mathcal{A}^+}(\breve{A}_1(\mu[\pi_t, \pi_{h1}]), \breve{A}_1(\mu[\pi_t, \pi_{h2}]))
\end{aligned}$$

Note that we avoided to calculate both $\Delta_{\mathcal{A}^+}(a, b)$ and $\Delta_{\mathcal{A}^+}(b, a)$, since they provide the same result, by calculating only $\Delta_{\mathcal{A}^+}(a, b)$ and multiplying the result by 2.

The agent in both agent slots 1 (π_t) performs the same sequence of actions (always Tail) independently of the agent line-up. So:

$$AD_1(\pi_t, \Pi_o, w_{\dot{L}}, \mu) = 2\frac{2}{1}\frac{1}{2}\frac{1}{2}0 = 0$$

And for agent slot 2, the agent in both agent slots 2 (π_t) also performs the same sequence of actions (always Tail) independently of the agent line-up. So:

$$\begin{aligned}AD_2(\pi_t, \Pi_o, w_{\dot{L}}, \mu) &= \eta_{\dot{L}^2} \sum_{\dot{u},\dot{v} \in \dot{L}_{-2}^{N(\mu)}(\Pi_o) | \dot{u} \neq \dot{v}} w_{\dot{L}}(\dot{u}) w_{\dot{L}}(\dot{v}) \Delta_{\mathcal{A}^+}(\breve{A}_2(\mu[\dot{u} \overset{2}{\leftarrow} \pi_t]), \breve{A}_2(\mu[\dot{v} \overset{2}{\leftarrow} \pi_t])) = \\ &= 2\frac{2}{1}\frac{1}{2}\frac{1}{2}\Delta_{\mathcal{A}^+}(\breve{A}_2(\mu[\pi_{h1}, \pi_t]), \breve{A}_2(\mu[\pi_{h2}, \pi_t])) = \\ &= 2\frac{2}{1}\frac{1}{2}\frac{1}{2}0 = 0\end{aligned}$$

And finally, we calculate the AD value:

$$\begin{aligned}AD(\Pi_e, w_{\Pi_e}, \Pi_o, w_{\dot{L}}, \mu, w_S) &= \sum_{\pi \in \Pi_e} w_{\Pi_e}(\pi) \sum_{i=1}^{N(\mu)} w_S(i, \mu) AD_i(\pi, \Pi_o, w_{\dot{L}}, \mu) = \\ &= \frac{1}{1}\frac{1}{2}\{AD_1(\pi_t, \Pi_o, w_{\dot{L}}, \mu) + AD_2(\pi_t, \Pi_o, w_{\dot{L}}, \mu)\} = \\ &= \frac{1}{1}\frac{1}{2}\{0 + 0\} = 0\end{aligned}$$

Since 0 is the lowest possible value for the action dependency property, therefore matching pennies has $General_{min} = 0$ for this property. $\square$

Proposition 3. *$General_{max}$ for the action dependency (AD) property is equal to 1 for the matching pennies environment.*

Proof. To find $General_{max}$ (equation 8.2), we need to find a pair $\langle \Pi_e, \Pi_o \rangle$ that maximises the property value as much as possible. We can have this situation by selecting $\Pi_e = \{\pi_m\}$ and $\Pi_o = \{\pi_h, \pi_t\}$ (a π_m agent first acts randomly and from time step 2 always mimics the other agent's last action, a π_h agent always performs Head and a π_t agent always performs Tail).

Following definition 17 we obtain the AD value for this $\langle \Pi_e, \Pi_o \rangle$. Since the multi-agent environment is not team symmetric, we need to calculate this property value for every agent slot. Also, since $|\Pi_e| = 1$ we just need to calculate this property value for one evaluated agent. Following definition 16 we calculate the AD value for the evaluated agent $\pi_m \in \Pi_e$ and each agent slot. We start with agent slot 1:

$$\begin{aligned}AD_1(\pi_m, \Pi_o, w_{\dot{L}}, \mu) &= \eta_{\dot{L}^2} \sum_{\dot{u},\dot{v} \in \dot{L}_{-1}^{N(\mu)}(\Pi_o) | \dot{u} \neq \dot{v}} w_{\dot{L}}(\dot{u}) w_{\dot{L}}(\dot{v}) \Delta_{\mathcal{A}^+}(\breve{A}_1(\mu[\dot{u} \overset{1}{\leftarrow} \pi_m]), \breve{A}_1(\mu[\dot{v} \overset{1}{\leftarrow} \pi_m])) = \\ &= 2\frac{2}{1}\frac{1}{2}\frac{1}{2}\Delta_{\mathcal{A}^+}(\breve{A}_1(\mu[\pi_m, \pi_h]), \breve{A}_1(\mu[\pi_m, \pi_t]))\end{aligned}$$

Note that we avoided to calculate both $\Delta_{\mathcal{A}^+}(a, b)$ and $\Delta_{\mathcal{A}^+}(b, a)$, since they provide the same result, by calculating only $\Delta_{\mathcal{A}^+}(a, b)$ and multiplying the result by 2.

From time step 2, π_m mimics the last action of the agent in agent slot 2, so the agent in both agent slots 1 (π_m) performs different sequences of actions depending on the agent line-up. So:

$$AD_1(\pi_m, \Pi_o, w_{\dot{L}}, \mu) = 2\frac{2}{1}\frac{1}{2}\frac{1}{2}1 = 1$$

And for agent slot 2, the agent in both agent slots 2 (π_m) also performs different sequences of actions depending on the agent line-up. So:

$$\begin{aligned}AD_2(\pi_m, \Pi_o, w_{\dot{L}}, \mu) = \eta_{\dot{L}^2} &\sum_{\dot{u},\dot{v} \in \dot{L}_{-2}^{N(\mu)}(\Pi_o) | \dot{u} \neq \dot{v}} w_{\dot{L}}(\dot{u}) w_{\dot{L}}(\dot{v}) \Delta_{\mathcal{A}^+}(\breve{A}_2(\mu[\dot{u} \overset{2}{\leftarrow} \pi_m]), \breve{A}_2(\mu[\dot{v} \overset{2}{\leftarrow} \pi_m])) = \\ &= 2\frac{2}{1}\frac{1}{2}\frac{1}{2}\Delta_{\mathcal{A}^+}(\breve{A}_2(\mu[\pi_h, \pi_m]), \breve{A}_2(\mu[\pi_t, \pi_m])) = \\ &= 2\frac{2}{1}\frac{1}{2}\frac{1}{2}1 = 1\end{aligned}$$

And finally, we calculate the AD value:

$$\begin{aligned}AD(\Pi_e, w_{\Pi_e}, \Pi_o, w_{\dot{L}}, \mu, w_S) &= \sum_{\pi \in \Pi_e} w_{\Pi_e}(\pi) \sum_{i=1}^{N(\mu)} w_S(i, \mu) AD_i(\pi, \Pi_o, w_{\dot{L}}, \mu) = \\ &= \frac{1}{1}\frac{1}{2}\{AD_1(\pi_m, \Pi_o, w_{\dot{L}}, \mu) + AD_2(\pi_m, \Pi_o, w_{\dot{L}}, \mu)\} = \\ &= \frac{1}{1}\frac{1}{2}\{1 + 1\} = 1\end{aligned}$$

Since 1 is the highest possible value for the action dependency property, therefore matching pennies has $General_{max} = 1$ for this property. □

Proposition 4. $Left_{max}$ for the action dependency (AD) property is equal to 0 for the matching pennies environment.

Proof. To find $Left_{max}$ (equation 8.4), we need to find a Π_e that maximises the property value as much as possible while Π_o minimises it. Using $\Pi_o = \{\pi_{h1}, \pi_{h2}\}$ (a π_h agent always performs Head) we find this situation no matter which Π_e we use.

Following definition 17 we obtain the AD value for this $\langle \Pi_e, \Pi_o \rangle$ (where Π_e is instantiated with any permitted value). Since the multi-agent environment is not team symmetric, we need to calculate this property value for every agent slot. Also, we do not know which Π_e we have, so we use a figurative evaluated agent π from Π_e. Following definition 16 we calculate the AD value for this figurative evaluated agent $\pi \in \Pi_e$ and each agent slot. We start with agent slot 1:

$$\begin{aligned}AD_1(\pi, \Pi_o, w_{\dot{L}}, \mu) = \eta_{\dot{L}^2} &\sum_{\dot{u},\dot{v} \in \dot{L}_{-1}^{N(\mu)}(\Pi_o) | \dot{u} \neq \dot{v}} w_{\dot{L}}(\dot{u}) w_{\dot{L}}(\dot{v}) \Delta_{\mathcal{A}^+}(\breve{A}_1(\mu[\dot{u} \overset{1}{\leftarrow} \pi]), \breve{A}_1(\mu[\dot{v} \overset{1}{\leftarrow} \pi])) = \\ &= 2\frac{2}{1}\frac{1}{2}\frac{1}{2}\Delta_{\mathcal{A}^+}(\breve{A}_1(\mu[\pi, \pi_{h1}]), \breve{A}_1(\mu[\pi, \pi_{h2}]))\end{aligned}$$

Note that we avoided to calculate both $\Delta_{\mathcal{A}^+}(a,b)$ and $\Delta_{\mathcal{A}^+}(b,a)$, since they provide the same result, by calculating only $\Delta_{\mathcal{A}^+}(a,b)$ and multiplying the result by 2.

The agent in both agent slots 1 (any π) is not able to differentiate with which agent is interacting, so it is not able to change its distribution of action sequences depending on the opponent's behaviour. So, for any π we obtain the same result:

$$AD_1(\pi, \Pi_o, w_{\dot{L}}, \mu) = 2\frac{2}{1}\frac{1}{2}\frac{1}{2}0 = 0$$

And for agent slot 2, the agent in both agent slots 2 (any π) is also not able to change its distribution of action sequences depending on the opponent's behaviour. So again, for any π we obtain the same result:

$$\begin{aligned} AD_2(\pi, \Pi_o, w_{\dot{L}}, \mu) = \eta_{\dot{L}^2} & \sum_{\dot{u},\dot{v} \in \dot{L}_{-2}^{N(\mu)}(\Pi_o) | \dot{u} \neq \dot{v}} w_{\dot{L}}(\dot{u}) w_{\dot{L}}(\dot{v}) \Delta_{\mathcal{A}^+}(\breve{A}_2(\mu[\dot{u} \overset{2}{\leftarrow} \pi]), \breve{A}_2(\mu[\dot{v} \overset{2}{\leftarrow} \pi])) = \\ &= 2\frac{2}{1}\frac{1}{2}\frac{1}{2}\Delta_{\mathcal{A}^+}(\breve{A}_2(\mu[\pi_{h1}, \pi]), \breve{A}_2(\mu[\pi_{h2}, \pi])) = \\ &= 2\frac{2}{1}\frac{1}{2}\frac{1}{2}0 = 0 \end{aligned}$$

And finally, we calculate the AD value generalising for any possible evaluated agent:

$$\begin{aligned} AD(\Pi_e, w_{\Pi_e}, \Pi_o, w_{\dot{L}}, \mu, w_S) &= \sum_{\pi \in \Pi_e} w_{\Pi_e}(\pi) \sum_{i=1}^{N(\mu)} w_S(i,\mu) AD_i(\pi, \Pi_o, w_{\dot{L}}, \mu) = \\ &= \sum_{\pi \in \Pi_e} w_{\Pi_e}(\pi) \frac{1}{2} \{AD_1(\pi, \Pi_o, w_{\dot{L}}, \mu) + AD_2(\pi, \Pi_o, w_{\dot{L}}, \mu)\} = \\ &= \sum_{\pi \in \Pi_e} w_{\Pi_e}(\pi) \frac{1}{2} \{0 + 0\} = 0 \end{aligned}$$

So, for every Π_e we obtain the same result:

$$\forall \Pi_e : AD(\Pi_e, w_{\Pi_e}, \Pi_o, w_{\dot{L}}, \mu, w_S) = 0$$

Therefore, matching pennies has $Left_{max} = 0$ for this property. □

Proposition 5. *$Right_{min}$ for the action dependency (AD) property is equal to 0 for the matching pennies environment.*

Proof. To find $Right_{min}$ (equation 8.5), we need to find a Π_e that minimises the property value as much as possible while Π_o maximises it. Using $\Pi_e = \{\pi_h\}$ (a π_h agent always performs Head) we find this situation no matter which Π_o we use.

Following definition 17 we obtain the AD value for this $\langle \Pi_e, \Pi_o \rangle$ (where Π_o is instantiated with any permitted value). Since the multi-agent environment is not team symmetric, we need to calculate this property value for every agent slot. Also, since $|\Pi_e| = 1$ we just need to calculate this property value for one evaluated agent. Following definition 16 we calculate the AD value for the evaluated agent $\pi_h \in \Pi_e$ and each agent slot. We start with agent slot 1:

$$AD_1(\pi_h, \Pi_o, w_{\dot{L}}, \mu) = \eta_{\dot{L}^2} \sum_{\dot{u},\dot{v} \in \dot{L}_{-1}^{N(\mu)}(\Pi_o) | \dot{u} \neq \dot{v}} w_{\dot{L}}(\dot{u}) w_{\dot{L}}(\dot{v}) \Delta_{\mathcal{A}^+}(\breve{A}_1(\mu[\dot{u} \overset{1}{\leftarrow} \pi_h]), \breve{A}_1(\mu[\dot{v} \overset{1}{\leftarrow} \pi_h]))$$

We do not know which Π_o we have, so we use two figurative agent line-up patterns $\dot{u} = (*, \pi_1)$ and $\dot{v} = (*, \pi_2)$ from $\dot{L}_{-1}^{N(\mu)}(\Pi_o)$:

$$\Delta_{\mathcal{A}^+}(\breve{A}_1(\mu[\dot{u} \overset{1}{\leftarrow} \pi_h]), \breve{A}_1(\mu[\dot{v} \overset{1}{\leftarrow} \pi_h])) = \Delta_{\mathcal{A}^+}(\breve{A}_1(\mu[\pi_h, \pi_1]), \breve{A}_1(\mu[\pi_h, \pi_2]))$$

The agent in both agent slots 1 (π_h) performs the same sequence of actions (always Head) independently of the agent line-up. So:

$$\Delta_{\mathcal{A}^+}(\breve{A}_1(\mu[\pi_h, \pi_1]), \breve{A}_1(\mu[\pi_h, \pi_2])) = 0$$

We generalise AD_1 for any possible pair of agent line-up patterns:

$$\begin{aligned} AD_1(\pi_h, \Pi_o, w_{\dot{L}}, \mu) &= \eta_{\dot{L}^2} \sum_{\dot{u},\dot{v} \in \dot{L}_{-1}^{N(\mu)}(\Pi_o) | \dot{u} \neq \dot{v}} w_{\dot{L}}(\dot{u}) w_{\dot{L}}(\dot{v}) \Delta_{\mathcal{A}^+}(\breve{A}_1(\mu[\dot{u} \overset{1}{\leftarrow} \pi_h]), \breve{A}_1(\mu[\dot{v} \overset{1}{\leftarrow} \pi_h])) = \\ &= \eta_{\dot{L}^2} \sum_{\dot{u},\dot{v} \in \dot{L}_{-1}^{N(\mu)}(\Pi_o) | \dot{u} \neq \dot{v}} w_{\dot{L}}(\dot{u}) w_{\dot{L}}(\dot{v}) 0 = 0 \end{aligned}$$

And for agent slot 2, the agent in both agent slots 2 (π_h) also performs the same sequence of actions (always Head) independently of the agent line-up. So:

$$\begin{aligned} AD_2(\pi_h, \Pi_o, w_{\dot{L}}, \mu) &= \eta_{\dot{L}^2} \sum_{\dot{u},\dot{v} \in \dot{L}_{-2}^{N(\mu)}(\Pi_o) | \dot{u} \neq \dot{v}} w_{\dot{L}}(\dot{u}) w_{\dot{L}}(\dot{v}) \Delta_{\mathcal{A}^+}(\breve{A}_2(\mu[\dot{u} \overset{2}{\leftarrow} \pi_h]), \breve{A}_2(\mu[\dot{v} \overset{2}{\leftarrow} \pi_h])) = \\ &= \eta_{\dot{L}^2} \sum_{\dot{u},\dot{v} \in \dot{L}_{-2}^{N(\mu)}(\Pi_o) | \dot{u} \neq \dot{v}} w_{\dot{L}}(\dot{u}) w_{\dot{L}}(\dot{v}) 0 = 0 \end{aligned}$$

And finally, we calculate the AD value:

$$\begin{aligned} AD(\Pi_e, w_{\Pi_e}, \Pi_o, w_{\dot{L}}, \mu, w_S) &= \sum_{\pi \in \Pi_e} w_{\Pi_e}(\pi) \sum_{i=1}^{N(\mu)} w_S(i, \mu) AD_i(\pi, \Pi_o, w_{\dot{L}}, \mu) = \\ &= \frac{1}{1}\frac{1}{2}\{AD_1(\pi_h, \Pi_o, w_{\dot{L}}, \mu) + AD_2(\pi_h, \Pi_o, w_{\dot{L}}, \mu)\} = \\ &= \frac{1}{1}\frac{1}{2}\{0 + 0\} = 0 \end{aligned}$$

So, for every Π_o we obtain the same result:

$$\forall \Pi_o : AD(\Pi_e, w_{\Pi_e}, \Pi_o, w_{\dot{L}}, \mu, w_S) = 0$$

Therefore, matching pennies has $Right_{min} = 0$ for this property. □

A.2 Reward Dependency

We continue with the reward dependency (RD) property. As given in section 7.3.1, we want to know if the evaluated agents obtain different rewards depending on which agent line-up they interact with. We use $\Delta_{\mathbb{Q}}(a', b')$ for $\Delta_{\mathbb{Q}^+}(a, b)$ where a' and b' are the expected results of a and b respectively, and $\Delta_{\mathbb{Q}}(a', b') = 0$ if numbers a' and b' are equal and 1 otherwise. We use an average of rewards as the utility function to calculate an agent's result.

Proposition 6. *General$_{min}$* for the reward dependency (RD) property is equal to 0 for the matching pennies environment.

Proof. To find *General$_{min}$* (equation 8.1), we need to find a pair $\langle \Pi_e, \Pi_o \rangle$ that minimises the property value as much as possible. We can have this situation by selecting $\Pi_e = \{\pi_t\}$ and $\Pi_o = \{\pi_{h1}, \pi_{h2}\}$ (a π_h agent always performs Head and a π_t agent always performs Tail).

Following definition 19 we obtain the RD value for this $\langle \Pi_e, \Pi_o \rangle$. Since the multi-agent environment is not team symmetric, we need to calculate this property value for every agent slot. Also, since $|\Pi_e| = 1$ we just need to calculate this property value for one evaluated agent. Following definition 18 we calculate the RD value for the evaluated agent $\pi_t \in \Pi_e$ and each agent slot. We start with agent slot 1:

$$\begin{aligned} RD_1(\pi_t, \Pi_o, w_{\dot{L}}, \mu) &= \eta_{\dot{L}^2} \sum_{\dot{u}, \dot{v} \in \dot{L}_{-1}^{N(\mu)}(\Pi_o) | \dot{u} \neq \dot{v}} w_{\dot{L}}(\dot{u}) w_{\dot{L}}(\dot{v}) \Delta_{\mathbb{Q}}(R_1(\mu[\dot{u} \overset{1}{\leftarrow} \pi_t]), R_1(\mu[\dot{v} \overset{1}{\leftarrow} \pi_t])) = \\ &= 2 \frac{2}{1} \frac{1}{2} \frac{1}{2} \Delta_{\mathbb{Q}}(R_1(\mu[\pi_t, \pi_{h1}]), R_1(\mu[\pi_t, \pi_{h2}])) \end{aligned}$$

Note that we avoided to calculate both $\Delta_{\mathbb{Q}}(a, b)$ and $\Delta_{\mathbb{Q}}(b, a)$, since they provide the same result, by calculating only $\Delta_{\mathbb{Q}}(a, b)$ and multiplying the result by 2.

The agent in both agent slots 1 (π_t) obtains the same expected average reward (-1) independently of the agent line-up. So:

$$RD_1(\pi_t, \Pi_o, w_{\dot{L}}, \mu) = 2 \frac{2}{1} \frac{1}{2} \frac{1}{2} 0 = 0$$

And for agent slot 2, the agent in both agent slots 2 (π_t) also obtains the same expected average reward (1) independently of the agent line-up. So:

$$\begin{aligned} RD_2(\pi_t, \Pi_o, w_{\dot{L}}, \mu) &= \eta_{\dot{L}^2} \sum_{\dot{u}, \dot{v} \in \dot{L}_{-2}^{N(\mu)}(\Pi_o) | \dot{u} \neq \dot{v}} w_{\dot{L}}(\dot{u}) w_{\dot{L}}(\dot{v}) \Delta_{\mathbb{Q}}(R_2(\mu[\dot{u} \overset{2}{\leftarrow} \pi_t]), R_2(\mu[\dot{v} \overset{2}{\leftarrow} \pi_t])) = \\ &= 2 \frac{2}{1} \frac{1}{2} \frac{1}{2} \Delta_{\mathbb{Q}}(R_2(\mu[\pi_{h1}, \pi_t]), R_2(\mu[\pi_{h2}, \pi_t])) = \\ &= 2 \frac{2}{1} \frac{1}{2} \frac{1}{2} 0 = 0 \end{aligned}$$

And finally, we calculate the RD value:

$$\begin{aligned} RD(\Pi_e, w_{\Pi_e}, \Pi_o, w_{\dot{L}}, \mu, w_S) &= \sum_{\pi \in \Pi_e} w_{\Pi_e}(\pi) \sum_{i=1}^{N(\mu)} w_S(i,\mu) RD_i(\pi, \Pi_o, w_{\dot{L}}, \mu) = \\ &= \frac{1}{1}\frac{1}{2}\{RD_1(\pi_t, \Pi_o, w_{\dot{L}}, \mu) + RD_2(\pi_t, \Pi_o, w_{\dot{L}}, \mu)\} = \\ &= \frac{1}{1}\frac{1}{2}\{0+0\} = 0 \end{aligned}$$

Since 0 is the lowest possible value for the reward dependency property, therefore matching pennies has $General_{min} = 0$ for this property. □

Proposition 7. *$General_{max}$ for the reward dependency (RD) property is equal to 1 for the matching pennies environment.*

Proof. To find $General_{max}$ (equation 8.2), we need to find a pair $\langle \Pi_e, \Pi_o \rangle$ that maximises the property value as much as possible. We can have this situation by selecting $\Pi_e = \{\pi_t\}$ and $\Pi_o = \{\pi_h, \pi_t\}$ (a π_h agent always performs Head and a π_t agent always performs Tail).

Following definition 19 we obtain the RD value for this $\langle \Pi_e, \Pi_o \rangle$. Since the multi-agent environment is not team symmetric, we need to calculate this property value for every agent slot. Also, since $|\Pi_e| = 1$ we just need to calculate this property value for one evaluated agent. Following definition 18 we calculate the RD value for the evaluated agent $\pi_t \in \Pi_e$ and each agent slot. We start with agent slot 1:

$$\begin{aligned} RD_1(\pi_t, \Pi_o, w_{\dot{L}}, \mu) &= \eta_{\dot{L}^2} \sum_{\dot{u},\dot{v} \in \dot{L}_{-1}^{N(\mu)}(\Pi_o) | \dot{u} \neq \dot{v}} w_{\dot{L}}(\dot{u}) w_{\dot{L}}(\dot{v}) \Delta_{\mathbb{Q}}(R_1(\mu[\dot{u} \overset{1}{\leftarrow} \pi_t]), R_1(\mu[\dot{v} \overset{1}{\leftarrow} \pi_t])) = \\ &= 2\frac{2}{1}\frac{1}{2}\frac{1}{2}\Delta_{\mathbb{Q}}(R_1(\mu[\pi_t, \pi_h]), R_1(\mu[\pi_t, \pi_t])) \end{aligned}$$

Note that we avoided to calculate both $\Delta_{\mathbb{Q}}(a,b)$ and $\Delta_{\mathbb{Q}}(b,a)$, since they provide the same result, by calculating only $\Delta_{\mathbb{Q}}(a,b)$ and multiplying the result by 2.

In agent line-up (π_t, π_h), the agent in agent slot 1 (π_t) obtains one expected average reward (-1), while in agent line-up (π_t, π_t), the agent in agent slot 1 (π_t) obtains a different expected average reward (1). So:

$$RD_1(\pi_t, \Pi_o, w_{\dot{L}}, \mu) = 2\frac{2}{1}\frac{1}{2}\frac{1}{2}1 = 1$$

And for agent slot 2, the agent in both agent slots 2 (π_t) also obtains different expected average rewards depending on the agent line-up. So:

$$\begin{aligned} RD_2(\pi_t, \Pi_o, w_{\dot{L}}, \mu) &= \eta_{\dot{L}^2} \sum_{\dot{u},\dot{v} \in \dot{L}_{-2}^{N(\mu)}(\Pi_o) | \dot{u} \neq \dot{v}} w_{\dot{L}}(\dot{u}) w_{\dot{L}}(\dot{v}) \Delta_{\mathbb{Q}}(R_2(\mu[\dot{u} \overset{2}{\leftarrow} \pi_t]), R_2(\mu[\dot{v} \overset{2}{\leftarrow} \pi_t])) = \\ &= 2\frac{2}{1}\frac{1}{2}\frac{1}{2}\Delta_{\mathbb{Q}}(R_2(\mu[\pi_h, \pi_t]), R_2(\mu[\pi_t, \pi_t])) = \\ &= 2\frac{2}{1}\frac{1}{2}\frac{1}{2}1 = 1 \end{aligned}$$

And finally, we calculate the RD value:

$$\begin{aligned}RD(\Pi_e, w_{\Pi_e}, \Pi_o, w_{\dot{L}}, \mu, w_S) &= \sum_{\pi\in\Pi_e} w_{\Pi_e}(\pi) \sum_{i=1}^{N(\mu)} w_S(i,\mu) RD_i(\pi, \Pi_o, w_{\dot{L}}, \mu) = \\ &= \frac{1}{1}\frac{1}{2}\{RD_1(\pi_t, \Pi_o, w_{\dot{L}}, \mu) + RD_2(\pi_t, \Pi_o, w_{\dot{L}}, \mu)\} = \\ &= \frac{1}{1}\frac{1}{2}\{1+1\} = 1\end{aligned}$$

Since 1 is the highest possible value for the reward dependency property, therefore matching pennies has $General_{max} = 1$ for this property. □

Proposition 8. *$Left_{max}$ for the reward dependency (RD) property is equal to 0 for the matching pennies environment.*

Proof. To find $Left_{max}$ (equation 8.4), we need to find a Π_e that maximises the property value as much as possible while Π_o minimises it. Using $\Pi_o = \{\pi_{h1}, \pi_{h2}\}$ (a π_h agent always performs Head) we find this situation no matter which Π_e we use.

Following definition 19 we obtain the RD value for this $\langle\Pi_e, \Pi_o\rangle$ (where Π_e is instantiated with any permitted value). Since the multi-agent environment is not team symmetric, we need to calculate this property value for every agent slot. Also, we do not know which Π_e we have, so we use a figurative evaluated agent π from Π_e. Following definition 18 we calculate the RD value for this figurative evaluated agent $\pi \in \Pi_e$ and each agent slot. We start with agent slot 1:

$$\begin{aligned}RD_1(\pi, \Pi_o, w_{\dot{L}}, \mu) &= \eta_{\dot{L}^2} \sum_{\dot{u},\dot{v}\in \dot{L}_{-1}^{N(\mu)}(\Pi_o)|\dot{u}\neq\dot{v}} w_{\dot{L}}(\dot{u}) w_{\dot{L}}(\dot{v}) \Delta_{\mathbb{Q}}(R_1(\mu[\dot{u} \overset{1}{\leftarrow} \pi]), R_1(\mu[\dot{v} \overset{1}{\leftarrow} \pi])) = \\ &= 2\frac{2}{1}\frac{1}{2}\frac{1}{2}\Delta_{\mathbb{Q}}(R_1(\mu[\pi, \pi_{h1}]), R_1(\mu[\pi, \pi_{h2}]))\end{aligned}$$

Note that we avoided to calculate both $\Delta_{\mathbb{Q}}(a, b)$ and $\Delta_{\mathbb{Q}}(b, a)$, since they provide the same result, by calculating only $\Delta_{\mathbb{Q}}(a, b)$ and multiplying the result by 2.

The agent in both agent slots 1 (any π) is not able to differentiate with which agent is interacting, so it is not able to change its distribution of action sequences depending on the opponent's behaviour, obtaining the same expected average reward. So, for any π we obtain the same result:

$$RD_1(\pi, \Pi_o, w_{\dot{L}}, \mu) = 2\frac{2}{1}\frac{1}{2}\frac{1}{2}0 = 0$$

And for agent slot 2, the agent in both agent slots 2 (any π) is also not able to change its distribution of action sequences depending on the opponent's behaviour, obtaining the same expected average reward. So again, for any π we obtain the same result:

$$\begin{aligned}RD_2(\pi, \Pi_o, w_{\dot{L}}, \mu) &= \eta_{\dot{L}^2} \sum_{\dot{u},\dot{v}\in \dot{L}_{-2}^{N(\mu)}(\Pi_o)|\dot{u}\neq\dot{v}} w_{\dot{L}}(\dot{u}) w_{\dot{L}}(\dot{v}) \Delta_{\mathbb{Q}}(R_2(\mu[\dot{u} \overset{2}{\leftarrow} \pi]), R_2(\mu[\dot{v} \overset{2}{\leftarrow} \pi])) = \\ &= 2\frac{2}{1}\frac{1}{2}\frac{1}{2}\Delta_{\mathbb{Q}}(R_2(\mu[\pi_{h1}, \pi]), R_2(\mu[\pi_{h2}, \pi])) = \\ &= 2\frac{2}{1}\frac{1}{2}\frac{1}{2}0 = 0\end{aligned}$$

And finally, we calculate the RD value generalising for any possible evaluated agent:

$$\begin{aligned}RD(\Pi_e, w_{\Pi_e}, \Pi_o, w_{\dot{L}}, \mu, w_S) &= \sum_{\pi \in \Pi_e} w_{\Pi_e}(\pi) \sum_{i=1}^{N(\mu)} w_S(i,\mu) RD_i(\pi, \Pi_o, w_{\dot{L}}, \mu) = \\ &= \sum_{\pi \in \Pi_e} w_{\Pi_e}(\pi) \frac{1}{2} \{RD_1(\pi, \Pi_o, w_{\dot{L}}, \mu) + RD_2(\pi, \Pi_o, w_{\dot{L}}, \mu)\} = \\ &= \sum_{\pi \in \Pi_e} w_{\Pi_e}(\pi) \frac{1}{2} \{0 + 0\} = 0\end{aligned}$$

So, for every Π_e we obtain the same result:

$$\forall \Pi_e : RD(\Pi_e, w_{\Pi_e}, \Pi_o, w_{\dot{L}}, \mu, w_S) = 0$$

Therefore, matching pennies has $Left_{max} = 0$ for this property. □

Proposition 9. $Right_{min}$ for the reward dependency (RD) property is equal to 0 for the matching pennies environment.

Proof. To find $Right_{min}$ (equation 8.5), we need to find a Π_e that minimises the property value as much as possible while Π_o maximises it. Using $\Pi_e = \{\pi_r\}$ (a π_r agent always acts randomly) we find this situation no matter which Π_o we use.

Following definition 19 we obtain the RD value for this $\langle \Pi_e, \Pi_o \rangle$ (where Π_o is instantiated with any permitted value). Since the multi-agent environment is not team symmetric, we need to calculate this property value for every agent slot. Also, since $|\Pi_e| = 1$ we just need to calculate this property value for one evaluated agent. Following definition 18 we calculate the RD value for the evaluated agent $\pi_r \in \Pi_e$ and each agent slot. We start with agent slot 1:

$$RD_1(\pi_r, \Pi_o, w_{\dot{L}}, \mu) = \eta_{\dot{L}^2} \sum_{\dot{u},\dot{v} \in \dot{L}_{-1}^{N(\mu)}(\Pi_o) | \dot{u} \neq \dot{v}} w_{\dot{L}}(\dot{u}) w_{\dot{L}}(\dot{v}) \Delta_{\mathbb{Q}}(R_1(\mu[\dot{u} \overset{1}{\leftarrow} \pi_r]), R_1(\mu[\dot{v} \overset{1}{\leftarrow} \pi_r]))$$

We do not know which Π_o we have, so we use two figurative agent line-up patterns $\dot{u} = (*, \pi_1)$ and $\dot{v} = (*, \pi_2)$ from $\dot{L}_{-1}^{N(\mu)}(\Pi_o)$:

$$\Delta_{\mathbb{Q}}(R_1(\mu[\dot{u} \overset{1}{\leftarrow} \pi_r]), R_1(\mu[\dot{v} \overset{1}{\leftarrow} \pi_r])) = \Delta_{\mathbb{Q}}(R_1(\mu[\pi_r, \pi_1]), R_1(\mu[\pi_r, \pi_2]))$$

The agent in both agent slots 1 (π_r) obtains the same expected average reward (0 as proved in lemma 2) independently of the agent line-up. So:

$$\Delta_{\mathbb{Q}}(R_1(\mu[\pi_r, \pi_1]), R_1(\mu[\pi_r, \pi_2])) = 0$$

We generalise RD_1 for any possible pair of agent line-up patterns:

$$\begin{aligned}RD_1(\pi_r, \Pi_o, w_{\dot{L}}, \mu) &= \eta_{\dot{L}^2} \sum_{\dot{u},\dot{v} \in \dot{L}_{-1}^{N(\mu)}(\Pi_o) | \dot{u} \neq \dot{v}} w_{\dot{L}}(\dot{u}) w_{\dot{L}}(\dot{v}) \Delta_{\mathbb{Q}}(R_1(\mu[\dot{u} \overset{1}{\leftarrow} \pi_r]), R_1(\mu[\dot{v} \overset{1}{\leftarrow} \pi_r])) = \\ &= \eta_{\dot{L}^2} \sum_{\dot{u},\dot{v} \in \dot{L}_{-1}^{N(\mu)}(\Pi_o) | \dot{u} \neq \dot{v}} w_{\dot{L}}(\dot{u}) w_{\dot{L}}(\dot{v}) 0 = 0\end{aligned}$$

And for agent slot 2, the agent in both agent slots 2 (π_r) also obtains the same expected average reward (0) independently of the agent line-up. So:

$$\begin{aligned}RD_2(\pi_r, \Pi_o, w_{\dot{L}}, \mu) &= \eta_{\dot{L}^2} \sum_{\dot{u},\dot{v} \in \dot{L}_{-2}^{N(\mu)}(\Pi_o) | \dot{u} \neq \dot{v}} w_{\dot{L}}(\dot{u}) w_{\dot{L}}(\dot{v}) \Delta_{\mathbb{Q}}(R_2(\mu[\dot{u} \overset{2}{\leftarrow} \pi_r]), R_2(\mu[\dot{v} \overset{2}{\leftarrow} \pi_r])) = \\ &= \eta_{\dot{L}^2} \sum_{\dot{u},\dot{v} \in \dot{L}_{-2}^{N(\mu)}(\Pi_o) | \dot{u} \neq \dot{v}} w_{\dot{L}}(\dot{u}) w_{\dot{L}}(\dot{v}) 0 = 0\end{aligned}$$

And finally, we calculate the RD value:

$$\begin{aligned}RD(\Pi_e, w_{\Pi_e}, \Pi_o, w_{\dot{L}}, \mu, w_S) &= \sum_{\pi \in \Pi_e} w_{\Pi_e}(\pi) \sum_{i=1}^{N(\mu)} w_S(i, \mu) RD_i(\pi, \Pi_o, w_{\dot{L}}, \mu) = \\ &= \frac{1}{1}\frac{1}{2}\{RD_1(\pi_r, \Pi_o, w_{\dot{L}}, \mu) + RD_2(\pi_r, \Pi_o, w_{\dot{L}}, \mu)\} = \\ &= \frac{1}{1}\frac{1}{2}\{0 + 0\} = 0\end{aligned}$$

So, for every Π_o we obtain the same result:

$$\forall \Pi_o : RD(\Pi_e, w_{\Pi_e}, \Pi_o, w_{\dot{L}}, \mu, w_S) = 0$$

Therefore, matching pennies has $Right_{min} = 0$ for this property. □

A.3 Fine Discrimination

Now we move to the fine discrimination (FD) property. As given in section 7.5.1, we want to know if different evaluated agents obtain different expected rewards when interacting in the same agent line-up patterns. We use $\Delta_{\mathbb{Q}}(a, b) = 0$ if numbers a and b are equal and 1 otherwise. We use an average of rewards as the utility function to calculate an agent's result.

Proposition 10. *$General_{min}$ for the fine discrimination (FD) property is equal to 0 for the matching pennies environment.*

Proof. To find $General_{min}$ (equation 8.1), we need to find a pair $\langle \Pi_e, \Pi_o \rangle$ that minimises the property value as much as possible. We can have this situation by selecting $\Pi_e = \{\pi_{t1}, \pi_{t2}\}$ and $\Pi_o = \{\pi_h\}$ (a π_h agent always performs Head and a π_t agent always performs Tail).

Following definition 26 we obtain the FD value for this $\langle \Pi_e, \Pi_o \rangle$. Since the multi-agent environment is not team symmetric, we need to calculate this property value for every agent slot. Also, since $|\Pi_e| = 2$ we just need to calculate this property value for one pair of evaluated agents. Following definition 25 we calculate the FD value for the evaluated agents $\pi_{t1}, \pi_{t2} \in \Pi_e$ and each agent slot. We start with agent slot 1:

$$\begin{aligned}FD_1(\pi_{t1}, \pi_{t2}, \Pi_o, w_{\dot{L}}, \mu) &= \sum_{\dot{l} \in \dot{L}_{-1}^{N(\mu)}(\Pi_o)} w_{\dot{L}}(\dot{l}) \Delta_{\mathbb{Q}}(R_1(\mu[\dot{l} \overset{1}{\leftarrow} \pi_{t1}]), R_1(\mu[\dot{l} \overset{1}{\leftarrow} \pi_{t2}])) = \\ &= \frac{1}{1} \Delta_{\mathbb{Q}}(R_1(\mu[\pi_{t1}, \pi_h]), R_1(\mu[\pi_{t2}, \pi_h]))\end{aligned}$$

Both agents in agent slot 1 (π_{t1} and π_{t2}) obtain the same expected average reward (-1). So:

$$FD_1(\pi_{t1}, \pi_{t2}, \Pi_o, w_{\dot{L}}, \mu) = \frac{1}{1}0 = 0$$

And for agent slot 2, both agents in agent slot 2 (π_{t1} and π_{t2}) also obtain the same expected average reward (1). So:

$$\begin{aligned}FD_2(\pi_{t1}, \pi_{t2}, \Pi_o, w_{\dot{L}}, \mu) &= \sum_{\dot{l} \in \dot{L}_{-2}^{N(\mu)}(\Pi_o)} w_{\dot{L}}(\dot{l}) \Delta_{\mathbb{Q}}(R_2(\mu[\dot{l} \overset{2}{\leftarrow} \pi_{t1}]), R_2(\mu[\dot{l} \overset{2}{\leftarrow} \pi_{t2}])) = \\ &= \frac{1}{1}\Delta_{\mathbb{Q}}(R_2(\mu[\pi_h, \pi_{t1}]), R_2(\mu[\pi_h, \pi_{t2}])) = \\ &= \frac{1}{1}0 = 0\end{aligned}$$

And finally, we calculate the FD value:

$$\begin{aligned}FD(\Pi_e, w_{\Pi_e}, \Pi_o, w_{\dot{L}}, \mu, w_S) &= \eta_{\Pi^2} \sum_{\pi_1, \pi_2 \in \Pi_e | \pi_1 \neq \pi_2} w_{\Pi_e}(\pi_1) w_{\Pi_e}(\pi_2) \sum_{i=1}^{N(\mu)} w_S(i, \mu) \times \\ &\times FD_i(\pi_1, \pi_2, \Pi_o, w_{\dot{L}}, \mu) = \\ &= 2\frac{2}{1}\frac{1}{2}\frac{1}{2}\frac{1}{2}\{FD_1(\pi_{t1}, \pi_{t2}, \Pi_o, w_{\dot{L}}, \mu) + FD_2(\pi_{t1}, \pi_{t2}, \Pi_o, w_{\dot{L}}, \mu)\} = \\ &= 2\frac{2}{1}\frac{1}{2}\frac{1}{2}\frac{1}{2}\{0 + 0\} = 0\end{aligned}$$

Note that we avoided to calculate both $FD_i(a, b, \Pi_o, w_{\dot{L}}, \mu)$ and $FD_i(b, a, \Pi_o, w_{\dot{L}}, \mu)$, since they provide the same result, by calculating only $FD_i(a, b, \Pi_o, w_{\dot{L}}, \mu)$ and multiplying the result by 2.

Since 0 is the lowest possible value for the fine discrimination property, therefore matching pennies has $General_{min} = 0$ for this property. □

Proposition 11. $General_{max}$ for the fine discrimination (FD) property is equal to 1 for the matching pennies environment.

Proof. To find $General_{max}$ (equation 8.2), we need to find a pair $\langle \Pi_e, \Pi_o \rangle$ that maximises the property value as much as possible. We can have this situation by selecting $\Pi_e = \{\pi_h, \pi_t\}$ and $\Pi_o = \{\pi_h\}$ (a π_h agent always performs Head and a π_t agent always performs Tail).

Following definition 26 we obtain the FD value for this $\langle \Pi_e, \Pi_o \rangle$. Since the multi-agent environment is not team symmetric, we need to calculate this property value for every agent slot. Also, since $|\Pi_e| = 2$ we just need to calculate this property value for one pair of evaluated agents. Following definition 25 we calculate the FD value for the evaluated agents $\pi_h, \pi_t \in \Pi_e$ and each agent slot. We start with agent slot 1:

$$\begin{aligned}FD_1(\pi_h, \pi_t, \Pi_o, w_{\dot{L}}, \mu) &= \sum_{\dot{l} \in \dot{L}_{-1}^{N(\mu)}(\Pi_o)} w_{\dot{L}}(\dot{l}) \Delta_{\mathbb{Q}}(R_1(\mu[\dot{l} \overset{1}{\leftarrow} \pi_h]), R_1(\mu[\dot{l} \overset{1}{\leftarrow} \pi_t])) = \\ &= \frac{1}{1}\Delta_{\mathbb{Q}}(R_1(\mu[\pi_h, \pi_h]), R_1(\mu[\pi_t, \pi_h]))\end{aligned}$$

Both agents in agent slot 1 (π_h and π_t) obtain different expected average rewards (1 and -1 respectively). So:

$$FD_1(\pi_h, \pi_t, \Pi_o, w_{\dot{L}}, \mu) = \frac{1}{1}1 = 1$$

And for agent slot 2, both agents in agent slot 2 (π_h and π_t) also obtain different expected average rewards (-1 and 1 respectively). So:

$$\begin{aligned} FD_2(\pi_h, \pi_t, \Pi_o, w_{\dot{L}}, \mu) &= \sum_{\dot{l} \in \dot{L}_{-2}^{N(\mu)}(\Pi_o)} w_{\dot{L}}(\dot{l}) \Delta_{\mathbb{Q}}(R_2(\mu[\dot{l} \xleftarrow{2} \pi_h]), R_2(\mu[\dot{l} \xleftarrow{2} \pi_t])) = \\ &= \frac{1}{1} \Delta_{\mathbb{Q}}(R_2(\mu[\pi_h, \pi_h]), R_2(\mu[\pi_h, \pi_t])) = \\ &= \frac{1}{1}1 = 1 \end{aligned}$$

And finally, we calculate the FD value:

$$\begin{aligned} FD(\Pi_e, w_{\Pi_e}, \Pi_o, w_{\dot{L}}, \mu, w_S) &= \eta_{\Pi^2} \sum_{\pi_1, \pi_2 \in \Pi_e | \pi_1 \neq \pi_2} w_{\Pi_e}(\pi_1) w_{\Pi_e}(\pi_2) \sum_{i=1}^{N(\mu)} w_S(i, \mu) \times \\ &\times FD_i(\pi_1, \pi_2, \Pi_o, w_{\dot{L}}, \mu) = \\ &= 2 \frac{2}{1} \frac{1}{2} \frac{1}{2} \frac{1}{2} \{FD_1(\pi_h, \pi_t, \Pi_o, w_{\dot{L}}, \mu) + FD_2(\pi_h, \pi_t, \Pi_o, w_{\dot{L}}, \mu)\} = \\ &= 2 \frac{2}{1} \frac{1}{2} \frac{1}{2} \frac{1}{2} \{1 + 1\} = 1 \end{aligned}$$

Note that we avoided to calculate both $FD_i(a, b, \Pi_o, w_{\dot{L}}, \mu)$ and $FD_i(b, a, \Pi_o, w_{\dot{L}}, \mu)$, since they provide the same result, by calculating only $FD_i(a, b, \Pi_o, w_{\dot{L}}, \mu)$ and multiplying the result by 2.

Since 1 is the highest possible value for the fine discrimination property, therefore matching pennies has $General_{max} = 1$ for this property. □

Proposition 12. *$Left_{max}$ for the fine discrimination (FD) property is equal to 0 for the matching pennies environment.*

Proof. To find $Left_{max}$ (equation 8.4), we need to find a Π_e that maximises the property value as much as possible while Π_o minimises it. Using $\Pi_o = \{\pi_r\}$ (a π_r agent always acts randomly) we find this situation no matter which Π_e we use.

Following definition 26 we obtain the FD value for this $\langle \Pi_e, \Pi_o \rangle$ (where Π_e is instantiated with any permitted value). Since the multi-agent environment is not team symmetric, we need to calculate this property value for every agent slot. Also, we do not know which Π_e we have, so we use a figurative pair of evaluated agents π_1, π_2 from Π_e. Following definition 25 we calculate the FD value for these figurative evaluated agents $\pi_1, \pi_2 \in \Pi_e$ and each agent slot. We start with agent slot 1:

$$\begin{aligned} FD_1(\pi_1, \pi_2, \Pi_o, w_{\dot{L}}, \mu) &= \sum_{\dot{l} \in \dot{L}_{-1}^{N(\mu)}(\Pi_o)} w_{\dot{L}}(\dot{l}) \Delta_{\mathbb{Q}}(R_1(\mu[\dot{l} \xleftarrow{1} \pi_1]), R_1(\mu[\dot{l} \xleftarrow{1} \pi_2])) = \\ &= \frac{1}{1} \Delta_{\mathbb{Q}}(R_1(\mu[\pi_1, \pi_r]), R_1(\mu[\pi_2, \pi_r])) \end{aligned}$$

The agent in both agent slots 2 (π_r) makes its expected average reward equal to its opponent expected average reward (both 0 as proved in lemma 2). So, for any pair π_1 and π_2 we obtain the same result:

$$FD_1(\pi_1, \pi_2, \Pi_o, w_{\dot{L}}, \mu) = \frac{1}{1}0 = 0$$

And for agent slot 2, both agents in agent slot 2 (any pair π_1 and π_2) also obtain the same expected average reward (0). So again, for any pair π_1 and π_2 we obtain the same result:

$$\begin{aligned} FD_2(\pi_1, \pi_2, \Pi_o, w_{\dot{L}}, \mu) &= \sum_{\dot{l} \in \dot{L}_{-2}^{N(\mu)}(\Pi_o)} w_{\dot{L}}(\dot{l}) \Delta_{\mathbb{Q}}(R_2(\mu[\dot{l} \overset{2}{\leftarrow} \pi_1]), R_2(\mu[\dot{l} \overset{2}{\leftarrow} \pi_2])) = \\ &= \frac{1}{1}\Delta_{\mathbb{Q}}(R_2(\mu[\pi_r, \pi_1]), R_2(\mu[\pi_r, \pi_2])) = \\ &= \frac{1}{1}0 = 0 \end{aligned}$$

And finally, we calculate the FD value generalising for any possible pair of evaluated agents:

$$\begin{aligned} FD(\Pi_e, w_{\Pi_e}, \Pi_o, w_{\dot{L}}, \mu, w_S) &= \eta_{\Pi^2} \sum_{\pi_1, \pi_2 \in \Pi_e | \pi_1 \neq \pi_2} w_{\Pi_e}(\pi_1) w_{\Pi_e}(\pi_2) \sum_{i=1}^{N(\mu)} w_S(i, \mu) \times \\ &\times FD_i(\pi_1, \pi_2, \Pi_o, w_{\dot{L}}, \mu) = \\ &= 2 \times \eta_{\Pi^2} \sum_{\pi_1, \pi_2 \in \Pi_e | \pi_1 \neq \pi_2} w_{\Pi_e}(\pi_1) w_{\Pi_e}(\pi_2) \frac{1}{2} \times \\ &\times \{FD_1(\pi_1, \pi_2, \Pi_o, w_{\dot{L}}, \mu) + FD_2(\pi_1, \pi_2, \Pi_o, w_{\dot{L}}, \mu)\} = \\ &= 2 \times \eta_{\Pi^2} \sum_{\pi_1, \pi_2 \in \Pi_e | \pi_1 \neq \pi_2} w_{\Pi_e}(\pi_1) w_{\Pi_e}(\pi_2) \frac{1}{2} \{0 + 0\} = 0 \end{aligned}$$

Note that we avoided to calculate both $FD_i(a, b, \Pi_o, w_{\dot{L}}, \mu)$ and $FD_i(b, a, \Pi_o, w_{\dot{L}}, \mu)$, since they provide the same result, by calculating only $FD_i(a, b, \Pi_o, w_{\dot{L}}, \mu)$ and multiplying the result by 2.

So, for every Π_e we obtain the same result:

$$\forall \Pi_e : FD(\Pi_e, w_{\Pi_e}, \Pi_o, w_{\dot{L}}, \mu, w_S) = 0$$

Therefore, matching pennies has $Left_{max} = 0$ for this property. □

Proposition 13. *$Right_{min}$ for the fine discrimination (FD) property is equal to 0 for the matching pennies environment.*

Proof. To find $Right_{min}$ (equation 8.5), we need to find a Π_e that minimises the property value as much as possible while Π_o maximises it. Using $\Pi_e = \{\pi_{t1}, \pi_{t2}\}$ (a π_t agent always performs Tail) we find this situation no matter which Π_o we use.

Following definition 26 we obtain the FD value for this $\langle \Pi_e, \Pi_o \rangle$ (where Π_o is instantiated with any permitted value). Since the multi-agent environment is not team symmetric, we need to calculate this property value for every agent slot. Also, since $|\Pi_e| = 2$ we just need to calculate this property value for one pair of evaluated agents. Following definition 25 we calculate the FD value for the evaluated agents $\pi_{t1}, \pi_{t2} \in \Pi_e$ and each agent slot. We start with agent slot 1:

$$FD_1(\pi_{t1}, \pi_{t2}, \Pi_o, w_{\dot{L}}, \mu) = \sum_{\dot{l} \in \dot{L}_{-1}^{N(\mu)}(\Pi_o)} w_{\dot{L}}(\dot{l}) \Delta_{\mathbb{Q}}(R_1(\mu[\dot{l} \overset{1}{\leftarrow} \pi_{t1}]), R_1(\mu[\dot{l} \overset{1}{\leftarrow} \pi_{t2}]))$$

We do not know which Π_o we have, so we use a figurative agent line-up pattern $\dot{l} = (*, \pi)$ from $\dot{L}_{-1}^{N(\mu)}(\Pi_o)$:

$$\Delta_{\mathbb{Q}}(R_1(\mu[\dot{l} \overset{1}{\leftarrow} \pi_{t1}]), R_1(\mu[\dot{l} \overset{1}{\leftarrow} \pi_{t2}])) = \Delta_{\mathbb{Q}}(R_1(\mu[\pi_{t1}, \pi]), R_1(\mu[\pi_{t2}, \pi]))$$

The agent in both agent slots 2 (any π) is not able to differentiate with which agent is interacting, so it is not able to change its distribution of action sequences depending on the opponent's behaviour, obtaining both agents in agent slot 1 (π_{t1} and π_{t2}) the same expected average reward. So:

$$\Delta_{\mathbb{Q}}(R_1(\mu[\pi_{t1}, \pi]), R_1(\mu[\pi_{t2}, \pi])) = 0$$

We generalise FD_1 for any possible agent line-up pattern:

$$\begin{aligned} FD_1(\pi_{t1}, \pi_{t2}, \Pi_o, w_{\dot{L}}, \mu) &= \sum_{\dot{l} \in \dot{L}_{-1}^{N(\mu)}(\Pi_o)} w_{\dot{L}}(\dot{l}) \Delta_{\mathbb{Q}}(R_1(\mu[\dot{l} \overset{1}{\leftarrow} \pi_{t1}]), R_1(\mu[\dot{l} \overset{1}{\leftarrow} \pi_{t2}])) = \\ &= \sum_{\dot{l} \in \dot{L}_{-1}^{N(\mu)}(\Pi_o)} w_{\dot{L}}(\dot{l}) 0 = 0 \end{aligned}$$

And for agent slot 2, the agent in both agent slots 2 (any π) is also not able to differentiate with which agent is interacting, so again it is not able to change its distribution of action sequences depending on the opponent's behaviour, obtaining both agents in agent slot 2 (π_{t1} and π_{t2}) the same expected average reward. So:

$$\begin{aligned} FD_2(\pi_{t1}, \pi_{t2}, \Pi_o, w_{\dot{L}}, \mu) &= \sum_{\dot{l} \in \dot{L}_{-2}^{N(\mu)}(\Pi_o)} w_{\dot{L}}(\dot{l}) \Delta_{\mathbb{Q}}(R_2(\mu[\dot{l} \overset{2}{\leftarrow} \pi_{t1}]), R_2(\mu[\dot{l} \overset{2}{\leftarrow} \pi_{t2}])) = \\ &= \sum_{\dot{l} \in \dot{L}_{-2}^{N(\mu)}(\Pi_o)} w_{\dot{L}}(\dot{l}) 0 = 0 \end{aligned}$$

And finally, we calculate the FD value:

$$\begin{aligned} FD(\Pi_e, w_{\Pi_e}, \Pi_o, w_{\dot{L}}, \mu, w_S) = \eta_{\Pi^2} &\sum_{\pi_1, \pi_2 \in \Pi_e | \pi_1 \neq \pi_2} w_{\Pi_e}(\pi_1) w_{\Pi_e}(\pi_2) \sum_{i=1}^{N(\mu)} w_S(i, \mu) \times \\ &\times FD_i(\pi_1, \pi_2, \Pi_o, w_{\dot{L}}, \mu) = \\ &= 2 \frac{2}{1} \frac{1}{2} \frac{1}{2} \frac{1}{2} \{FD_1(\pi_{t1}, \pi_{t2}, \Pi_o, w_{\dot{L}}, \mu) + FD_2(\pi_{t1}, \pi_{t2}, \Pi_o, w_{\dot{L}}, \mu)\} = \\ &= 2 \frac{2}{1} \frac{1}{2} \frac{1}{2} \frac{1}{2} \{0 + 0\} = 0 \end{aligned}$$

Note that we avoided to calculate both $FD_i(a, b, \Pi_o, w_{\dot{L}}, \mu)$ and $FD_i(b, a, \Pi_o, w_{\dot{L}}, \mu)$, since they provide the same result, by calculating only $FD_i(a, b, \Pi_o, w_{\dot{L}}, \mu)$ and multiplying the result by 2.

So, for every Π_o we obtain the same result:

$$\forall \Pi_o : FD(\Pi_e, w_{\Pi_e}, \Pi_o, w_{\dot{L}}, \mu, w_S) = 0$$

Therefore, matching pennies has $Right_{min} = 0$ for this property. □

A.4 Strict Total Grading

We arrive to the strict total grading (STG) property. As given in section 7.5.2, we want to know if there exists a strict ordering between the evaluated agents when interacting in the multi-agent environment. We use an average of rewards as the utility function to calculate an agent's result.

To simplify the notation, we use the next table to represent the STO: $R_i(\mu[\dot{l} \overset{i,j}{\leftarrow} \pi_1, \pi_2]) < R_j(\mu[\dot{l} \overset{i,j}{\leftarrow} \pi_1, \pi_2])$, $R_i(\mu[\dot{l} \overset{i,j}{\leftarrow} \pi_2, \pi_3]) < R_j(\mu[\dot{l} \overset{i,j}{\leftarrow} \pi_2, \pi_3])$ and $R_i(\mu[\dot{l} \overset{i,j}{\leftarrow} \pi_1, \pi_3]) < R_j(\mu[\dot{l} \overset{i,j}{\leftarrow} \pi_1, \pi_3])$.

AS i		AS j
π_1	$<$	π_2
π_2	$<$	π_3
π_1	$<$	π_3

where AS stands for agent slot.

Proposition 14. *$General_{min}$ for the strict total grading (STG) property is equal to 0 for the matching pennies environment.*

Proof. To find $General_{min}$ (equation 8.1), we need to find a pair $\langle \Pi_e, \Pi_o \rangle$ that minimises the property value as much as possible. We can have this situation by selecting $\Pi_e = \{\pi_{r1}, \pi_{r2}, \pi_{r3}\}$ and $\Pi_o = \emptyset$ (a π_r agent always acts randomly).

Following definition 30 we obtain the STG value for this $\langle \Pi_e, \Pi_o \rangle$. Since the multi-agent environment is not team symmetric, we need to calculate this property value for every pair of agent slots. Also, since $|\Pi_e| = 3$ we just need to calculate this property value for one trio of evaluated agents. Following definition 29 we calculate the STG value for the evaluated agents $\pi_{r1}, \pi_{r2}, \pi_{r3} \in \Pi_e$ and each pair of agent slots. We start with agent slots 1 and 2:

$$STG_{1,2}(\pi_{r1}, \pi_{r2}, \pi_{r3}, \Pi_o, w_{\dot{L}}, \mu) = \sum_{\dot{l} \in \dot{L}_{-1,2}^{N(\mu)}(\Pi_o)} w_{\dot{L}}(\dot{l}) STO_{1,2}(\pi_{r1}, \pi_{r2}, \pi_{r3}, \dot{l}, \mu) =$$
$$= \frac{1}{1} STO_{1,2}(\pi_{r1}, \pi_{r2}, \pi_{r3}, (*, *), \mu)$$

The following table shows us $STO_{1,2}$ for all the permutations of $\pi_{r1}, \pi_{r2}, \pi_{r3}$.

AS 1		AS 2	AS 1		AS 2	AS 1		AS 2
π_{r1}	$<$	π_{r2}	π_{r1}	$<$	π_{r3}	π_{r2}	$<$	π_{r1}
π_{r2}	$<$	π_{r3}	π_{r3}	$<$	π_{r2}	π_{r1}	$<$	π_{r3}
π_{r1}	$<$	π_{r3}	π_{r1}	$<$	π_{r2}	π_{r2}	$<$	π_{r3}
AS 1		AS 2	AS 1		AS 2	AS 1		AS 2
π_{r2}	$<$	π_{r3}	π_{r3}	$<$	π_{r1}	π_{r3}	$<$	π_{r2}
π_{r3}	$<$	π_{r1}	π_{r1}	$<$	π_{r2}	π_{r2}	$<$	π_{r1}
π_{r2}	$<$	π_{r1}	π_{r3}	$<$	π_{r2}	π_{r3}	$<$	π_{r1}

It is not possible to find a STO, since for every permutation both agents obtain the same expected average reward (0 as proved in lemma 2). So:

$$STG_{1,2}(\pi_{r1}, \pi_{r2}, \pi_{r3}, \Pi_o, w_{\dot{L}}, \mu) = \frac{1}{1}0 = 0$$

And for agent slots 2 and 1, the following table shows us $STO_{2,1}$ for all the permutations of $\pi_{r1}, \pi_{r2}, \pi_{r3}$.

AS 2		AS 1	AS 2		AS 1	AS 2		AS 1
π_{r1}	$<$	π_{r2}	π_{r1}	$<$	π_{r3}	π_{r2}	$<$	π_{r1}
π_{r2}	$<$	π_{r3}	π_{r3}	$<$	π_{r2}	π_{r1}	$<$	π_{r3}
π_{r1}	$<$	π_{r3}	π_{r1}	$<$	π_{r2}	π_{r2}	$<$	π_{r3}
AS 2		AS 1	AS 2		AS 1	AS 2		AS 1
π_{r2}	$<$	π_{r3}	π_{r3}	$<$	π_{r1}	π_{r3}	$<$	π_{r2}
π_{r3}	$<$	π_{r1}	π_{r1}	$<$	π_{r2}	π_{r2}	$<$	π_{r1}
π_{r2}	$<$	π_{r1}	π_{r3}	$<$	π_{r2}	π_{r3}	$<$	π_{r1}

For every permutation both agents also obtain the same expected average reward (0). So:

$$\begin{aligned} STG_{2,1}(\pi_{r1}, \pi_{r2}, \pi_{r3}, \Pi_o, w_{\dot{L}}, \mu) &= \sum_{\dot{l} \in \dot{L}_{-2,1}^{N(\mu)}(\Pi_o)} w_{\dot{L}}(\dot{l}) STO_{2,1}(\pi_{r1}, \pi_{r2}, \pi_{r3}, \dot{l}, \mu) = \\ &= \frac{1}{1} STO_{2,1}(\pi_{r1}, \pi_{r2}, \pi_{r3}, (*,*), \mu) = \\ &= \frac{1}{1}0 = 0 \end{aligned}$$

And finally, we calculate the STG value:

$$STG(\Pi_e, w_{\Pi_e}, \Pi_o, w_{\dot{L}}, \mu, w_S) = \eta_{\Pi^3} \sum_{\pi_1,\pi_2,\pi_3 \in \Pi_e | \pi_1 \neq \pi_2 \neq \pi_3} w_{\Pi_e}(\pi_1) w_{\Pi_e}(\pi_2) w_{\Pi_e}(\pi_3) \eta_{S_1^2} \sum_{i=1}^{N(\mu)} w_S(i,\mu) \times$$

$$\times \left(\sum_{j=1}^{i-1} w_S(j,\mu) STG_{i,j}(\pi_1, \pi_2, \pi_3, \Pi_o, w_{\dot{L}}, \mu) + \sum_{j=i+1}^{N(\mu)} w_S(j,\mu) STG_{i,j}(\pi_1, \pi_2, \pi_3, \Pi_o, w_{\dot{L}}, \mu) \right) =$$

$$\begin{aligned} &= 6 \frac{9}{2} \frac{1}{3} \frac{1}{3} \frac{1}{3} \frac{2}{1} \frac{1}{2} \frac{1}{2} \{ STG_{1,2}(\pi_{r1}, \pi_{r2}, \pi_{r3}, \Pi_o, w_{\dot{L}}, \mu) + \\ &+ STG_{2,1}(\pi_{r1}, \pi_{r2}, \pi_{r3}, \Pi_o, w_{\dot{L}}, \mu) \} = \\ &= 6 \frac{9}{2} \frac{1}{3} \frac{1}{3} \frac{1}{3} \frac{2}{1} \frac{1}{2} \frac{1}{2} \{0 + 0\} = 0 \end{aligned}$$

Note that we avoided to calculate all the permutations of π_1, π_2, π_3 for $STG_{i,j}(\pi_1, \pi_2, \pi_3, \Pi_o, w_{\dot{L}}, \mu)$, since they provide the same result, by calculating only one permutation and multiplying the result by 6 (the number of permutations).

Since 0 is the lowest possible value for the strict total grading property, therefore matching pennies has $General_{min} = 0$ for this property. □

Proposition 15. *$General_{max}$ for the strict total grading (STG) property is equal to 1 for the matching pennies environment.*

Proof. To find $General_{max}$ (equation 8.2), we need to find a pair $\langle \Pi_e, \Pi_o \rangle$ that maximises the property value as much as possible. We can have this situation by selecting $\Pi_e = \{\pi_h, \pi_{h/t}, \pi_{m/o}\}$ and $\Pi_o = \emptyset$ (a π_h agent always performs Head, a $\pi_{h/t}$ agent always performs Head when interacting in agent slot 1 and always performs Tail when interacting in agent slot 2, and a $\pi_{m/o}$ agent first acts randomly and from time step 2 always mimics the other agent's last action when interacting in agent slot 1 and always performs the opposite of the other agent's last action when interacting in agent slot 2)[1].

Following definition 30 we obtain the STG value for this $\langle \Pi_e, \Pi_o \rangle$. Since the multi-agent environment is not team symmetric, we need to calculate this property value for every pair of agent slots. Also, since $|\Pi_e| = 3$ we just need to calculate this property value for one trio of evaluated agents. Following definition 29 we calculate the STG value for the evaluated agents $\pi_h, \pi_{h/t}, \pi_{m/o} \in \Pi_e$ and each pair of agent slots. We start with agent slots 1 and 2:

$$STG_{1,2}(\pi_h, \pi_{h/t}, \pi_{m/o}, \Pi_o, w_{\dot{L}}, \mu) = \sum_{\dot{l} \in \dot{L}_{-1,2}^{N(\mu)}(\Pi_o)} w_{\dot{L}}(\dot{l}) STO_{1,2}(\pi_h, \pi_{h/t}, \pi_{m/o}, \dot{l}, \mu) =$$
$$= \frac{1}{1} STO_{1,2}(\pi_h, \pi_{h/t}, \pi_{m/o}, (*,*), \mu)$$

The following table shows us $STO_{1,2}$ for all the permutations of $\pi_h, \pi_{h/t}, \pi_{m/o}$.

AS 1		AS 2	AS 1		AS 2	AS 1		AS 2
π_h	$<$	$\pi_{h/t}$	π_h	$<$	$\pi_{m/o}$	$\pi_{h/t}$	$<$	π_h
$\pi_{h/t}$	$<$	$\pi_{m/o}$	$\pi_{m/o}$	$<$	$\pi_{h/t}$	π_h	$<$	$\pi_{m/o}$
π_h	$<$	$\pi_{m/o}$	π_h	$<$	$\pi_{h/t}$	$\pi_{h/t}$	$<$	$\pi_{m/o}$
AS 1		AS 2	AS 1		AS 2	AS 1		AS 2
$\pi_{h/t}$	$<$	$\pi_{m/o}$	$\pi_{m/o}$	$<$	π_h	$\pi_{m/o}$	$<$	$\pi_{h/t}$
$\pi_{m/o}$	$<$	π_h	π_h	$<$	$\pi_{h/t}$	$\pi_{h/t}$	$<$	π_h
$\pi_{h/t}$	$<$	π_h	$\pi_{m/o}$	$<$	$\pi_{h/t}$	$\pi_{m/o}$	$<$	π_h

It is possible to find a STO for the first permutation. In agent line-up $(\pi_h, \pi_{h/t})$, π_h always performs Head and $\pi_{h/t}$ always performs Tail, obtaining the expected average rewards of -1 and 1 respectively. In agent line-up $(\pi_{h/t}, \pi_{m/o})$, $\pi_{h/t}$ always performs Head and $\pi_{m/o}$ always performs Tail, obtaining the expected average rewards of -1 and 1 respectively. In agent line-up $(\pi_h, \pi_{m/o})$, π_h always performs Head and $\pi_{m/o}$ always performs Tail, obtaining the expected average rewards of -1 and 1 respectively. So:

[1]Note that $\pi_{h/t}$ and $\pi_{m/o}$ have to know in which agent slot they are interacting. To infer this, they start with a random action at the first time step and then they look at the other agent's action and the reward they obtain.

$$STG_{1,2}(\pi_h, \pi_{h/t}, \pi_{m/o}, \Pi_o, w_{\dot{L}}, \mu) = \frac{1}{1}1 = 1$$

And for agent slots 2 and 1, the following table shows us $STO_{2,1}$ for all the permutations of $\pi_h, \pi_{h/t}, \pi_{m/o}$.

AS 2		AS 1	AS 2		AS 1	AS 2		AS 1
π_h	$<$	$\pi_{h/t}$	π_h	$<$	$\pi_{m/o}$	$\pi_{h/t}$	$<$	π_h
$\pi_{h/t}$	$<$	$\pi_{m/o}$	$\pi_{m/o}$	$<$	$\pi_{h/t}$	π_h	$<$	$\pi_{m/o}$
π_h	$<$	$\pi_{m/o}$	π_h	$<$	$\pi_{h/t}$	$\pi_{h/t}$	$<$	$\pi_{m/o}$
AS 2		AS 1	AS 2		AS 1	AS 2		AS 1
$\pi_{h/t}$	$<$	$\pi_{m/o}$	$\pi_{m/o}$	$<$	π_h	$\pi_{m/o}$	$<$	$\pi_{h/t}$
$\pi_{m/o}$	$<$	π_h	π_h	$<$	$\pi_{h/t}$	$\pi_{h/t}$	$<$	π_h
$\pi_{h/t}$	$<$	π_h	$\pi_{m/o}$	$<$	$\pi_{h/t}$	$\pi_{m/o}$	$<$	π_h

Again, it is possible to find a STO for the first permutation. In agent line-up $(\pi_{h/t}, \pi_h)$, π_h always performs Head and $\pi_{h/t}$ always performs Head, obtaining the expected average rewards of -1 and 1 respectively. In agent line-up $(\pi_{m/o}, \pi_{h/t})$, $\pi_{h/t}$ always performs Tail and $\pi_{m/o}$ always performs Tail, obtaining the expected average rewards of -1 and 1 respectively. In agent line-up $(\pi_{m/o}, \pi_h)$, π_h always performs Head and $\pi_{m/o}$ always performs Head, obtaining the expected average rewards of -1 and 1 respectively. So:

$$\begin{aligned} STG_{2,1}(\pi_h, \pi_{h/t}, \pi_{m/o}, \Pi_o, w_{\dot{L}}, \mu) &= \sum_{\dot{l} \in \dot{L}^{N(\mu)}_{-2,1}(\Pi_o)} w_{\dot{L}}(\dot{l}) STO_{2,1}(\pi_h, \pi_{h/t}, \pi_{m/o}, \dot{l}, \mu) = \\ &= \frac{1}{1} STO_{2,1}(\pi_h, \pi_{h/t}, \pi_{m/o}, (*,*), \mu) = \\ &= \frac{1}{1}1 = 1 \end{aligned}$$

And finally, we calculate the STG value:

$$\begin{aligned} STG(\Pi_e, w_{\Pi_e}, \Pi_o, w_{\dot{L}}, \mu, w_S) = \eta_{\Pi^3} \sum_{\pi_1,\pi_2,\pi_3 \in \Pi_e | \pi_1 \neq \pi_2 \neq \pi_3} w_{\Pi_e}(\pi_1) w_{\Pi_e}(\pi_2) w_{\Pi_e}(\pi_3) \eta_{S_1^2} \sum_{i=1}^{N(\mu)} w_S(i,\mu) \times \\ \times \left(\sum_{j=1}^{i-1} w_S(j,\mu) STG_{i,j}(\pi_1,\pi_2,\pi_3,\Pi_o,w_{\dot{L}},\mu) + \sum_{j=i+1}^{N(\mu)} w_S(j,\mu) STG_{i,j}(\pi_1,\pi_2,\pi_3,\Pi_o,w_{\dot{L}},\mu) \right) = \\ = 6\frac{9}{2}\frac{1}{3}\frac{1}{3}\frac{1}{3}\frac{2}{1}\frac{1}{2}\frac{1}{2} \{ STG_{1,2}(\pi_h, \pi_{h/t}, \pi_{m/o}, \Pi_o, w_{\dot{L}}, \mu) + \\ + STG_{2,1}(\pi_h, \pi_{h/t}, \pi_{m/o}, \Pi_o, w_{\dot{L}}, \mu) \} = \\ = 6\frac{9}{2}\frac{1}{3}\frac{1}{3}\frac{1}{3}\frac{2}{1}\frac{1}{2}\frac{1}{2} \{1+1\} = 1 \end{aligned}$$

Note that we avoided to calculate all the permutations of π_1, π_2, π_3 for $STG_{i,j}(\pi_1, \pi_2, \pi_3, \Pi_o, w_{\dot{L}}, \mu)$, since they provide the same result, by calculating only one permutation and multiplying the result by 6 (the number of permutations).

Since 1 is the highest possible value for the strict total grading property, therefore matching pennies has $General_{max} = 1$ for this property. □

Proposition 16. $Left_{max}$ for the strict total grading (STG) property is equal to 1 for the matching pennies environment.

Proof. To find $Left_{max}$ (equation 8.4), we need to find a Π_e that maximises the property value as much as possible while Π_o minimises it. Using $\Pi_e = \{\pi_h, \pi_{h/t}, \pi_{m/o}\}$ (a π_h agent always performs Head, a $\pi_{h/t}$ agent always performs Head when interacting in agent slot 1 and always performs Tail when interacting in agent slot 2, and a $\pi_{m/o}$ agent first acts randomly and from time step 2 always mimics the other agent's last action when interacting in agent slot 1 and always performs the opposite of the other agent's last action when interacting in agent slot 2)[2] we find this situation no matter which Π_o we use.

Following definition 30 we obtain the STG value for this $\langle \Pi_e, \Pi_o \rangle$ (where Π_o is instantiated with any permitted value). Since the multi-agent environment is not team symmetric, we need to calculate this property value for every pair of agent slots. Also, since $|\Pi_e| = 3$ we just need to calculate this property value for one trio of evaluated agents. Following definition 29 we calculate the STG value for the evaluated agents $\pi_h, \pi_{h/t}, \pi_{m/o} \in \Pi_e$ and each pair of agent slots. We start with agent slots 1 and 2:

$$STG_{1,2}(\pi_h, \pi_{h/t}, \pi_{m/o}, \Pi_o, w_{\dot{L}}, \mu) = \sum_{\dot{l} \in \dot{L}^{N(\mu)}_{-1,2}(\Pi_o)} w_{\dot{L}}(\dot{l}) STO_{1,2}(\pi_h, \pi_{h/t}, \pi_{m/o}, \dot{l}, \mu) =$$

$$= \frac{1}{1} STO_{1,2}(\pi_h, \pi_{h/t}, \pi_{m/o}, (*,*), \mu)$$

Note that the choice of Π_o does not affect the result of $STG_{1,2}$.

The following table shows us $STO_{1,2}$ for all the permutations of $\pi_h, \pi_{h/t}, \pi_{m/o}$.

AS 1		AS 2	AS 1		AS 2	AS 1		AS 2
π_h	$<$	$\pi_{h/t}$	π_h	$<$	$\pi_{m/o}$	$\pi_{h/t}$	$<$	π_h
$\pi_{h/t}$	$<$	$\pi_{m/o}$	$\pi_{m/o}$	$<$	$\pi_{h/t}$	π_h	$<$	$\pi_{m/o}$
π_h	$<$	$\pi_{m/o}$	π_h	$<$	$\pi_{h/t}$	$\pi_{h/t}$	$<$	$\pi_{m/o}$
$\pi_{h/t}$	$<$	$\pi_{m/o}$	$\pi_{m/o}$	$<$	π_h	$\pi_{m/o}$	$<$	$\pi_{h/t}$
$\pi_{m/o}$	$<$	π_h	π_h	$<$	$\pi_{h/t}$	$\pi_{h/t}$	$<$	π_h
$\pi_{h/t}$	$<$	π_h	$\pi_{m/o}$	$<$	$\pi_{h/t}$	$\pi_{m/o}$	$<$	π_h

It is possible to find a STO for the first permutation. In agent line-up $(\pi_h, \pi_{h/t})$, π_h always performs Head and $\pi_{h/t}$ always performs Tail, obtaining the expected average rewards of -1 and 1 respectively. In agent line-up $(\pi_{h/t}, \pi_{m/o})$, $\pi_{h/t}$ always performs Head and $\pi_{m/o}$ always performs Tail, obtaining the expected average rewards of -1 and 1 respectively. In agent line-up $(\pi_h, \pi_{m/o})$, π_h always performs Head and $\pi_{m/o}$ always performs Tail, obtaining the expected average rewards of -1 and 1 respectively. So:

$$STG_{1,2}(\pi_h, \pi_{h/t}, \pi_{m/o}, \Pi_o, w_{\dot{L}}, \mu) = \frac{1}{1} 1 = 1$$

And for agent slots 2 and 1, the following table shows us $STO_{2,1}$ for all the permutations of $\pi_h, \pi_{h/t}, \pi_{m/o}$.

[2]Note that $\pi_{h/t}$ and $\pi_{m/o}$ have to know in which agent slot they are interacting. To infer this, they start with a random action at the first time step and then they look at the other agent's action and the reward they obtain.

AS 2		AS 1	AS 2		AS 1	AS 2		AS 1
π_h	$<$	$\pi_{h/t}$	π_h	$<$	$\pi_{m/o}$	$\pi_{h/t}$	$<$	π_h
$\pi_{h/t}$	$<$	$\pi_{m/o}$	$\pi_{m/o}$	$<$	$\pi_{h/t}$	π_h	$<$	$\pi_{m/o}$
π_h	$<$	$\pi_{m/o}$	π_h	$<$	$\pi_{h/t}$	$\pi_{h/t}$	$<$	$\pi_{m/o}$
AS 2		AS 1	AS 2		AS 1	AS 2		AS 1
$\pi_{h/t}$	$<$	$\pi_{m/o}$	$\pi_{m/o}$	$<$	π_h	$\pi_{m/o}$	$<$	$\pi_{h/t}$
$\pi_{m/o}$	$<$	π_h	π_h	$<$	$\pi_{h/t}$	$\pi_{h/t}$	$<$	π_h
$\pi_{h/t}$	$<$	π_h	$\pi_{m/o}$	$<$	$\pi_{h/t}$	$\pi_{m/o}$	$<$	π_h

Again, it is possible to find a STO for the first permutation. In agent line-up $(\pi_{h/t}, \pi_h)$, π_h always performs Head and $\pi_{h/t}$ always performs Head, obtaining the expected average rewards of -1 and 1 respectively. In agent line-up $(\pi_{m/o}, \pi_{h/t})$, $\pi_{h/t}$ always performs Tail and $\pi_{m/o}$ always performs Tail, obtaining the expected average rewards of -1 and 1 respectively. In agent line-up $(\pi_{m/o}, \pi_h)$, π_h always performs Head and $\pi_{m/o}$ always performs Head, obtaining the expected average rewards of -1 and 1 respectively. So:

$$\begin{aligned}
STG_{2,1}(\pi_h, \pi_{h/t}, \pi_{m/o}, \Pi_o, w_{\dot{L}}, \mu) &= \sum_{\dot{l} \in \dot{L}^{N(\mu)}_{-2,1}(\Pi_o)} w_{\dot{L}}(\dot{l}) STO_{2,1}(\pi_h, \pi_{h/t}, \pi_{m/o}, \dot{l}, \mu) = \\
&= \frac{1}{1} STO_{2,1}(\pi_h, \pi_{h/t}, \pi_{m/o}, (*,*), \mu) = \\
&= \frac{1}{1} 1 = 1
\end{aligned}$$

Note again that the choice of Π_o does not affect the result of $STG_{2,1}$.

And finally, we calculate the STG value:

$$\begin{aligned}
STG(\Pi_e, w_{\Pi_e}, \Pi_o, w_{\dot{L}}, \mu, w_S) = \eta_{\Pi^3} \sum_{\pi_1,\pi_2,\pi_3 \in \Pi_e | \pi_1 \neq \pi_2 \neq \pi_3} w_{\Pi_e}(\pi_1) w_{\Pi_e}(\pi_2) w_{\Pi_e}(\pi_3) \eta_{S_1^2} \sum_{i=1}^{N(\mu)} w_S(i, \mu) \times \\
\times \left(\sum_{j=1}^{i-1} w_S(j, \mu) STG_{i,j}(\pi_1, \pi_2, \pi_3, \Pi_o, w_{\dot{L}}, \mu) + \sum_{j=i+1}^{N(\mu)} w_S(j, \mu) STG_{i,j}(\pi_1, \pi_2, \pi_3, \Pi_o, w_{\dot{L}}, \mu) \right) = \\
= 6 \frac{9}{2}\frac{1}{3}\frac{1}{3}\frac{1}{3}\frac{2}{1}\frac{1}{2}\frac{1}{2} \{STG_{1,2}(\pi_h, \pi_{h/t}, \pi_{m/o}, \Pi_o, w_{\dot{L}}, \mu) + \\
+ STG_{2,1}(\pi_h, \pi_{h/t}, \pi_{m/o}, \Pi_o, w_{\dot{L}}, \mu)\} = \\
= 6 \frac{9}{2}\frac{1}{3}\frac{1}{3}\frac{1}{3}\frac{2}{1}\frac{1}{2}\frac{1}{2} \{1 + 1\} = 1
\end{aligned}$$

Note that we avoided to calculate all the permutations of π_1, π_2, π_3 for $STG_{i,j}(\pi_1, \pi_2, \pi_3, \Pi_o, w_{\dot{L}}, \mu)$, since they provide the same result, by calculating only one permutation and multiplying the result by 6 (the number of permutations).

So, for every Π_o we obtain the same result:

$$\forall \Pi_o : STG(\Pi_e, w_{\Pi_e}, \Pi_o, w_{\dot{L}}, \mu, w_S) = 1$$

Therefore, matching pennies has $Left_{max} = 1$ for this property. $\square$

Proposition 17. *Right$_{min}$* for the strict total grading (STG) property is equal to 0 for the matching pennies environment.

Proof. To find *Right$_{min}$* (equation 8.5), we need to find a Π_e that minimises the property value as much as possible while Π_o maximises it. Using $\Pi_e = \{\pi_{r1}, \pi_{r2}, \pi_{r3}\}$ (a π_r agent always acts randomly) we find this situation no matter which Π_o we use.

Following definition 30 we obtain the STG value for this $\langle \Pi_e, \Pi_o \rangle$ (where Π_o is instantiated with any permitted value). Since the multi-agent environment is not team symmetric, we need to calculate this property value for every pair of agent slots. Also, since $|\Pi_e| = 3$ we just need to calculate this property value for one trio of evaluated agents. Following definition 29 we calculate the STG value for the evaluated agents $\pi_{r1}, \pi_{r2}, \pi_{r3} \in \Pi_e$ and each pair of agent slots. We start with agent slots 1 and 2:

$$STG_{1,2}(\pi_{r1}, \pi_{r2}, \pi_{r3}, \Pi_o, w_{\dot{L}}, \mu) = \sum_{\dot{l} \in \dot{L}^{N(\mu)}_{-1,2}(\Pi_o)} w_{\dot{L}}(\dot{l}) STO_{1,2}(\pi_{r1}, \pi_{r2}, \pi_{r3}, \dot{l}, \mu) =$$
$$= \frac{1}{1} STO_{1,2}(\pi_{r1}, \pi_{r2}, \pi_{r3}, (*, *), \mu)$$

Note that the choice of Π_o does not affect the result of $STG_{1,2}$.

The following table shows us $STO_{1,2}$ for all the permutations of $\pi_{r1}, \pi_{r2}, \pi_{r3}$.

AS 1		AS 2	AS 1		AS 2	AS 1		AS 2
π_{r1}	$<$	π_{r2}	π_{r1}	$<$	π_{r3}	π_{r2}	$<$	π_{r1}
π_{r2}	$<$	π_{r3}	π_{r3}	$<$	π_{r2}	π_{r1}	$<$	π_{r3}
π_{r1}	$<$	π_{r3}	π_{r1}	$<$	π_{r2}	π_{r2}	$<$	π_{r3}
AS 1		AS 2	AS 1		AS 2	AS 1		AS 2
π_{r2}	$<$	π_{r3}	π_{r3}	$<$	π_{r1}	π_{r3}	$<$	π_{r2}
π_{r3}	$<$	π_{r1}	π_{r1}	$<$	π_{r2}	π_{r2}	$<$	π_{r1}
π_{r2}	$<$	π_{r1}	π_{r3}	$<$	π_{r2}	π_{r3}	$<$	π_{r1}

It is not possible to find a STO, since for every permutation both agents obtain the same expected average reward (0 as proved in lemma 2). So:

$$STG_{1,2}(\pi_{r1}, \pi_{r2}, \pi_{r3}, \Pi_o, w_{\dot{L}}, \mu) = \frac{1}{1} 0 = 0$$

And for agent slots 2 and 1, the following table shows us $STO_{2,1}$ for all the permutations of $\pi_{r1}, \pi_{r2}, \pi_{r3}$.

AS 2		AS 1	AS 2		AS 1	AS 2		AS 1
π_{r1}	$<$	π_{r2}	π_{r1}	$<$	π_{r3}	π_{r2}	$<$	π_{r1}
π_{r2}	$<$	π_{r3}	π_{r3}	$<$	π_{r2}	π_{r1}	$<$	π_{r3}
π_{r1}	$<$	π_{r3}	π_{r1}	$<$	π_{r2}	π_{r2}	$<$	π_{r3}
AS 2		AS 1	AS 2		AS 1	AS 2		AS 1
π_{r2}	$<$	π_{r3}	π_{r3}	$<$	π_{r1}	π_{r3}	$<$	π_{r2}
π_{r3}	$<$	π_{r1}	π_{r1}	$<$	π_{r2}	π_{r2}	$<$	π_{r1}
π_{r2}	$<$	π_{r1}	π_{r3}	$<$	π_{r2}	π_{r3}	$<$	π_{r1}

For every permutation both agents also obtain the same expected average reward (0). So:

$$\begin{aligned} STG_{2,1}(\pi_{r1}, \pi_{r2}, \pi_{r3}, \Pi_o, w_{\dot{L}}, \mu) &= \sum_{\dot{l} \in \dot{L}^{N(\mu)}_{-2,1}(\Pi_o)} w_{\dot{L}}(\dot{l}) STO_{2,1}(\pi_{r1}, \pi_{r2}, \pi_{r3}, \dot{l}, \mu) = \\ &= \frac{1}{1} STO_{2,1}(\pi_{r1}, \pi_{r2}, \pi_{r3}, (*,*), \mu) = \\ &= \frac{1}{1} 0 = 0 \end{aligned}$$

Note again that the choice of Π_o does not affect the result of $STG_{2,1}$.

And finally, we calculate the STG value:

$$\begin{aligned} STG(\Pi_e, w_{\Pi_e}, \Pi_o, w_{\dot{L}}, \mu, w_S) = \eta_{\Pi^3} \sum_{\pi_1,\pi_2,\pi_3 \in \Pi_e | \pi_1 \neq \pi_2 \neq \pi_3} w_{\Pi_e}(\pi_1) w_{\Pi_e}(\pi_2) w_{\Pi_e}(\pi_3) \eta_{S_1^2} \sum_{i=1}^{N(\mu)} w_S(i,\mu) \times \\ \times \left(\sum_{j=1}^{i-1} w_S(j,\mu) STG_{i,j}(\pi_1, \pi_2, \pi_3, \Pi_o, w_{\dot{L}}, \mu) + \sum_{j=i+1}^{N(\mu)} w_S(j,\mu) STG_{i,j}(\pi_1, \pi_2, \pi_3, \Pi_o, w_{\dot{L}}, \mu) \right) = \\ = 6 \frac{9}{2} \frac{1}{3} \frac{1}{3} \frac{1}{3} \frac{2}{1} \frac{1}{2} \frac{1}{2} \{ STG_{1,2}(\pi_{r1}, \pi_{r2}, \pi_{r3}, \Pi_o, w_{\dot{L}}, \mu) + \\ + STG_{2,1}(\pi_{r1}, \pi_{r2}, \pi_{r3}, \Pi_o, w_{\dot{L}}, \mu) \} = \\ = 6 \frac{9}{2} \frac{1}{3} \frac{1}{3} \frac{1}{3} \frac{2}{1} \frac{1}{2} \frac{1}{2} \{0 + 0\} = 0 \end{aligned}$$

Note that we avoided to calculate all the permutations of π_1, π_2, π_3 for $STG_{i,j}(\pi_1, \pi_2, \pi_3, \Pi_o, w_{\dot{L}}, \mu)$, since they provide the same result, by calculating only one permutation and multiplying the result by 6 (the number of permutations).

So, for every Π_o we obtain the same result:

$$\forall \Pi_o : STG(\Pi_e, w_{\Pi_e}, \Pi_o, w_{\dot{L}}, \mu, w_S) = 0$$

Therefore, matching pennies has $Right_{min} = 0$ for this property. □

A.5 Partial Grading

Now we arrive to the partial grading (PG) property. As given in section 7.5.2, we want to know if there exists a partial ordering between the evaluated agents when interacting in the multi-agent environment. We use an average of rewards as the utility function to calculate an agent's result.

To simplify the notation, we use the next table to represent the PO: $R_i(\mu[\dot{l} \overset{i,j}{\leftarrow} \pi_1, \pi_2]) \leq R_j(\mu[\dot{l} \overset{i,j}{\leftarrow} \pi_1, \pi_2])$, $R_i(\mu[\dot{l} \overset{i,j}{\leftarrow} \pi_2, \pi_3]) \leq R_j(\mu[\dot{l} \overset{i,j}{\leftarrow} \pi_2, \pi_3])$ and $R_i(\mu[\dot{l} \overset{i,j}{\leftarrow} \pi_1, \pi_3]) \leq R_j(\mu[\dot{l} \overset{i,j}{\leftarrow} \pi_1, \pi_3])$.

AS i		AS j
π_1	$\leq$	π_2
π_2	$\leq$	π_3
π_1	$\leq$	π_3

where AS stands for agent slot.

Proposition 18. *$General_{min}$* for the partial grading (PG) property is equal to 0 for the matching pennies environment.

Proof. To find $General_{min}$ (equation 8.1), we need to find a pair $\langle \Pi_e, \Pi_o \rangle$ that minimises the property value as much as possible. We can have this situation by selecting $\Pi_e = \{\pi_{h1}, \pi_{h2}, \pi_t\}$ and $\Pi_o = \emptyset$ (a π_h agent always performs Head and a π_t agent always performs Tail).

Following definition 31 we obtain the PG value for this $\langle \Pi_e, \Pi_o \rangle$. Since the multi-agent environment is not team symmetric, we need to calculate this property value for every pair of agent slots. Also, since $|\Pi_e| = 3$ we just need to calculate this property value for one trio of evaluated agents. Following definition 29 (for PG) we calculate the PG value for the evaluated agents $\pi_{h1}, \pi_{h2}, \pi_t \in \Pi_e$ and each pair of agent slots. We start with agent slots 1 and 2:

$$PG_{1,2}(\pi_{h1}, \pi_{h2}, \pi_t, \Pi_o, w_{\dot{L}}, \mu) = \sum_{\dot{l} \in \dot{L}^{N(\mu)}_{-1,2}(\Pi_o)} w_{\dot{L}}(\dot{l}) PO_{1,2}(\pi_{h1}, \pi_{h2}, \pi_t, \dot{l}, \mu) =$$

$$= \frac{1}{1} PO_{1,2}(\pi_{h1}, \pi_{h2}, \pi_t, (*, *), \mu)$$

The following table shows us $PO_{1,2}$ for all the permutations of $\pi_{h1}, \pi_{h2}, \pi_t$.

AS 1		AS 2	AS 1		AS 2	AS 1		AS 2
π_{h1}	$\leq$	π_{h2}	π_{h1}	$\leq$	π_t	π_{h2}	$\leq$	π_{h1}
π_{h2}	$\leq$	π_t	π_t	$\leq$	π_{h2}	π_{h1}	$\leq$	π_t
π_{h1}	$\leq$	π_t	π_{h1}	$\leq$	π_{h2}	π_{h2}	$\leq$	π_t
AS 1		AS 2	AS 1		AS 2	AS 1		AS 2
π_{h2}	$\leq$	π_t	π_t	$\leq$	π_{h1}	π_t	$\leq$	π_{h2}
π_t	$\leq$	π_{h1}	π_{h1}	$\leq$	π_{h2}	π_{h2}	$\leq$	π_{h1}
π_{h2}	$\leq$	π_{h1}	π_t	$\leq$	π_{h2}	π_t	$\leq$	π_{h1}

It is not possible to find a PO, since for every permutation we have either the agent line-up (π_{h1}, π_{h2}) or (π_{h2}, π_{h1}). In both cases, the agents obtain the expected average rewards of 1 and -1 respectively. So:

$$PG_{1,2}(\pi_{h1}, \pi_{h2}, \pi_t, \Pi_o, w_{\dot{L}}, \mu) = \frac{1}{1} 0 = 0$$

And for agent slots 2 and 1, the following table shows us $PO_{2,1}$ for all the permutations of $\pi_{h1}, \pi_{h2}, \pi_t$.

AS 2		AS 1	AS 2		AS 1	AS 2		AS 1
π_{h1}	$\leq$	π_{h2}	π_{h1}	$\leq$	π_t	π_{h2}	$\leq$	π_{h1}
π_{h2}	$\leq$	π_t	π_t	$\leq$	π_{h2}	π_{h1}	$\leq$	π_t
π_{h1}	$\leq$	π_t	π_{h1}	$\leq$	π_{h2}	π_{h2}	$\leq$	π_t
AS 2		AS 1	AS 2		AS 1	AS 2		AS 1
π_{h2}	$\leq$	π_t	π_t	$\leq$	π_{h1}	π_t	$\leq$	π_{h2}
π_t	$\leq$	π_{h1}	π_{h1}	$\leq$	π_{h2}	π_{h2}	$\leq$	π_{h1}
π_{h2}	$\leq$	π_{h1}	π_t	$\leq$	π_{h2}	π_t	$\leq$	π_{h1}

It is also not possible to find a PO, since for every permutation we have either the agent line-up (π_t, π_{h1}) or (π_{h1}, π_t). In both cases, the agents obtain the expected average rewards of -1 and 1 respectively. So:

$$\begin{aligned} PG_{2,1}(\pi_{h1}, \pi_{h2}, \pi_t, \Pi_o, w_{\dot{L}}, \mu) &= \sum_{\dot{l} \in \dot{L}_{-2,1}^{N(\mu)}(\Pi_o)} w_{\dot{L}}(\dot{l}) PO_{2,1}(\pi_{h1}, \pi_{h2}, \pi_t, \dot{l}, \mu) = \\ &= \frac{1}{1} PO_{2,1}(\pi_{h1}, \pi_{h2}, \pi_t, (*, *), \mu) = \\ &= \frac{1}{1} 0 = 0 \end{aligned}$$

And finally, we calculate the PG value:

$$\begin{aligned} PG(\Pi_e, w_{\Pi_e}, \Pi_o, w_{\dot{L}}, \mu, w_S) = \eta_{\Pi^3} \sum_{\pi_1,\pi_2,\pi_3 \in \Pi_e | \pi_1 \neq \pi_2 \neq \pi_3} w_{\Pi_e}(\pi_1) w_{\Pi_e}(\pi_2) w_{\Pi_e}(\pi_3) \eta_{S_1^2} \sum_{i=1}^{N(\mu)} w_S(i, \mu) \times \\ \times \left(\sum_{j=1}^{i-1} w_S(j, \mu) PG_{i,j}(\pi_1, \pi_2, \pi_3, \Pi_o, w_{\dot{L}}, \mu) + \sum_{j=i+1}^{N(\mu)} w_S(j, \mu) PG_{i,j}(\pi_1, \pi_2, \pi_3, \Pi_o, w_{\dot{L}}, \mu) \right) = \\ = 6 \frac{9}{2} \frac{1}{3} \frac{1}{3} \frac{1}{3} \frac{2}{1} \frac{1}{2} \frac{1}{2} \{ PG_{1,2}(\pi_{h1}, \pi_{h2}, \pi_t, \Pi_o, w_{\dot{L}}, \mu) + \\ + PG_{2,1}(\pi_{h1}, \pi_{h2}, \pi_t, \Pi_o, w_{\dot{L}}, \mu) \} = \\ = 6 \frac{9}{2} \frac{1}{3} \frac{1}{3} \frac{1}{3} \frac{2}{1} \frac{1}{2} \frac{1}{2} \{0 + 0\} = 0 \end{aligned}$$

Note that we avoided to calculate all the permutations of π_1, π_2, π_3 for $PG_{i,j}(\pi_1, \pi_2, \pi_3, \Pi_o, w_{\dot{L}}, \mu)$, since they provide the same result, by calculating only one permutation and multiplying the result by 6 (the number of permutations).

Since 0 is the lowest possible value for the partial grading property, therefore matching pennies has $General_{min} = 0$ for this property. □

Proposition 19. *$General_{max}$ for the partial grading (PG) property is equal to 1 for the matching pennies environment.*

Proof. To find $General_{max}$ (equation 8.2), we need to find a pair $\langle \Pi_e, \Pi_o \rangle$ that maximises the property value as much as possible. We can have this situation by selecting $\Pi_e = \{\pi_{r1}, \pi_{r2}, \pi_{r3}\}$ and $\Pi_o = \emptyset$ (a π_r agent always acts randomly).

Following definition 31 we obtain the PG value for this $\langle \Pi_e, \Pi_o \rangle$. Since the multi-agent environment is not team symmetric, we need to calculate this property value for every pair of agent slots. Also, since $|\Pi_e| = 3$ we just need to calculate this property value for one trio of evaluated agents. Following definition 29 (for PG) we calculate the PG value for the evaluated agents $\pi_{r1}, \pi_{r2}, \pi_{r3} \in \Pi_e$ and each pair of agent slots. We start with agent slots 1 and 2:

$$\begin{aligned} PG_{1,2}(\pi_{r1}, \pi_{r2}, \pi_{r3}, \Pi_o, w_{\dot{L}}, \mu) &= \sum_{\dot{l} \in \dot{L}_{-1,2}^{N(\mu)}(\Pi_o)} w_{\dot{L}}(\dot{l}) PO_{1,2}(\pi_{r1}, \pi_{r2}, \pi_{r3}, \dot{l}, \mu) = \\ &= \frac{1}{1} PO_{1,2}(\pi_{r1}, \pi_{r2}, \pi_{r3}, (*, *), \mu) \end{aligned}$$

The following table shows us $PO_{1,2}$ for all the permutations of $\pi_{r1}, \pi_{r2}, \pi_{r3}$.

AS 1		AS 2	AS 1		AS 2	AS 1		AS 2
π_{r1}	$\leq$	π_{r2}	π_{r1}	$\leq$	π_{r3}	π_{r2}	$\leq$	π_{r1}
π_{r2}	$\leq$	π_{r3}	π_{r3}	$\leq$	π_{r2}	π_{r1}	$\leq$	π_{r3}
π_{r1}	$\leq$	π_{r3}	π_{r1}	$\leq$	π_{r2}	π_{r2}	$\leq$	π_{r3}
π_{r2}	$\leq$	π_{r3}	π_{r3}	$\leq$	π_{r1}	π_{r3}	$\leq$	π_{r2}
π_{r3}	$\leq$	π_{r1}	π_{r1}	$\leq$	π_{r2}	π_{r2}	$\leq$	π_{r1}
π_{r2}	$\leq$	π_{r1}	π_{r3}	$\leq$	π_{r2}	π_{r3}	$\leq$	π_{r1}

It is possible to find a PO for every permutation, since both agents obtain the same expected average reward (0 as proved in lemma 2). So:

$$PG_{1,2}(\pi_{r1},\pi_{r2},\pi_{r3},\Pi_o,w_{\dot{L}},\mu) = \frac{1}{1}1 = 1$$

And for agent slots 2 and 1, the following table shows us $PO_{2,1}$ for all the permutations of $\pi_{r1},\pi_{r2},\pi_{r3}$.

AS 2		AS 1	AS 2		AS 1	AS 2		AS 1
π_{r1}	$\leq$	π_{r2}	π_{r1}	$\leq$	π_{r3}	π_{r2}	$\leq$	π_{r1}
π_{r2}	$\leq$	π_{r3}	π_{r3}	$\leq$	π_{r2}	π_{r1}	$\leq$	π_{r3}
π_{r1}	$\leq$	π_{r3}	π_{r1}	$\leq$	π_{r2}	π_{r2}	$\leq$	π_{r3}
π_{r2}	$\leq$	π_{r3}	π_{r3}	$\leq$	π_{r1}	π_{r3}	$\leq$	π_{r2}
π_{r3}	$\leq$	π_{r1}	π_{r1}	$\leq$	π_{r2}	π_{r2}	$\leq$	π_{r1}
π_{r2}	$\leq$	π_{r1}	π_{r3}	$\leq$	π_{r2}	π_{r3}	$\leq$	π_{r1}

For every permutation both agents also obtain the same expected average reward (0). So:

$$\begin{aligned} PG_{2,1}(\pi_{r1},\pi_{r2},\pi_{r3},\Pi_o,w_{\dot{L}},\mu) &= \sum_{\dot{l}\in\dot{L}^{N(\mu)}_{-2,1}(\Pi_o)} w_{\dot{L}}(\dot{l})PO_{2,1}(\pi_{r1},\pi_{r2},\pi_{r3},\dot{l},\mu) = \\ &= \frac{1}{1}PO_{2,1}(\pi_{r1},\pi_{r2},\pi_{r3},(*,*),\mu) = \\ &= \frac{1}{1}1 = 1 \end{aligned}$$

And finally, we calculate the PG value:

$$\begin{aligned} PG(\Pi_e,w_{\Pi_e},\Pi_o,w_{\dot{L}},\mu,w_S) &= \eta_{\Pi^3}\sum_{\pi_1,\pi_2,\pi_3\in\Pi_e|\pi_1\neq\pi_2\neq\pi_3} w_{\Pi_e}(\pi_1)w_{\Pi_e}(\pi_2)w_{\Pi_e}(\pi_3)\eta_{S_1^2}\sum_{i=1}^{N(\mu)} w_S(i,\mu)\times \\ &\times\left(\sum_{j=1}^{i-1} w_S(j,\mu)PG_{i,j}(\pi_1,\pi_2,\pi_3,\Pi_o,w_{\dot{L}},\mu) + \sum_{j=i+1}^{N(\mu)} w_S(j,\mu)PG_{i,j}(\pi_1,\pi_2,\pi_3,\Pi_o,w_{\dot{L}},\mu)\right) = \\ &= 6\frac{9}{2}\frac{1}{3}\frac{1}{3}\frac{1}{3}\frac{2}{1}\frac{1}{2}\frac{1}{2}\{PG_{1,2}(\pi_{r1},\pi_{r2},\pi_{r3},\Pi_o,w_{\dot{L}},\mu)+ \\ &+ PG_{2,1}(\pi_{r1},\pi_{r2},\pi_{r3},\Pi_o,w_{\dot{L}},\mu)\} = \\ &= 6\frac{9}{2}\frac{1}{3}\frac{1}{3}\frac{1}{3}\frac{2}{1}\frac{1}{2}\frac{1}{2}\{1+1\} = 1 \end{aligned}$$

Note that we avoided to calculate all the permutations of π_1, π_2, π_3 for $PG_{i,j}(\pi_1, \pi_2, \pi_3, \Pi_o, w_{\dot{L}}, \mu)$, since they provide the same result, by calculating only one permutation and multiplying the result by 6 (the number of permutations).

Since 1 is the highest possible value for the partial grading property, therefore matching pennies has $General_{max} = 1$ for this property. □

Proposition 20. *$Left_{max}$ for the partial grading (PG) property is equal to 1 for the matching pennies environment.*

Proof. To find $Left_{max}$ (equation 8.4), we need to find a Π_e that maximises the property value as much as possible while Π_o minimises it. Using $\Pi_e = \{\pi_{r1}, \pi_{r2}, \pi_{r3}\}$ (a π_r agent always acts randomly) we find this situation no matter which Π_o we use.

Following definition 31 we obtain the PG value for this $\langle \Pi_e, \Pi_o \rangle$ (where Π_o is instantiated with any permitted value). Since the multi-agent environment is not team symmetric, we need to calculate this property value for every pair of agent slots. Also, since $|\Pi_e| = 3$ we just need to calculate this property value for one trio of evaluated agents. Following definition 29 (for PG) we calculate the PG value for the evaluated agents $\pi_{r1}, \pi_{r2}, \pi_{r3} \in \Pi_e$ and each pair of agent slots. We start with agent slots 1 and 2:

$$PG_{1,2}(\pi_{r1}, \pi_{r2}, \pi_{r3}, \Pi_o, w_{\dot{L}}, \mu) = \sum_{\dot{l} \in \dot{L}_{-1,2}^{N(\mu)}(\Pi_o)} w_{\dot{L}}(\dot{l}) PO_{1,2}(\pi_{r1}, \pi_{r2}, \pi_{r3}, \dot{l}, \mu) =$$
$$= \frac{1}{1} PO_{1,2}(\pi_{r1}, \pi_{r2}, \pi_{r3}, (*, *), \mu)$$

Note that the choice of Π_o does not affect the result of $PG_{1,2}$.

The following table shows us $PO_{1,2}$ for all the permutations of $\pi_{r1}, \pi_{r2}, \pi_{r3}$.

AS 1		AS 2	AS 1		AS 2	AS 1		AS 2
π_{r1}	$\leq$	π_{r2}	π_{r1}	$\leq$	π_{r3}	π_{r2}	$\leq$	π_{r1}
π_{r2}	$\leq$	π_{r3}	π_{r3}	$\leq$	π_{r2}	π_{r1}	$\leq$	π_{r3}
π_{r1}	$\leq$	π_{r3}	π_{r1}	$\leq$	π_{r2}	π_{r2}	$\leq$	π_{r3}
AS 1		AS 2	AS 1		AS 2	AS 1		AS 2
π_{r2}	$\leq$	π_{r3}	π_{r3}	$\leq$	π_{r1}	π_{r3}	$\leq$	π_{r2}
π_{r3}	$\leq$	π_{r1}	π_{r1}	$\leq$	π_{r2}	π_{r2}	$\leq$	π_{r1}
π_{r2}	$\leq$	π_{r1}	π_{r3}	$\leq$	π_{r2}	π_{r3}	$\leq$	π_{r1}

It is possible to find a PO for every permutation, since both agents obtain the same expected average reward (0 as proved in lemma 2). So:

$$PG_{1,2}(\pi_{r1}, \pi_{r2}, \pi_{r3}, \Pi_o, w_{\dot{L}}, \mu) = \frac{1}{1} 1 = 1$$

And for agent slots 2 and 1, the following table shows us $PO_{2,1}$ for all the permutations of $\pi_{r1}, \pi_{r2}, \pi_{r3}$.

AS 2		AS 1	AS 2		AS 1	AS 2		AS 1
π_{r1}	$\leq$	π_{r2}	π_{r1}	$\leq$	π_{r3}	π_{r2}	$\leq$	π_{r1}
π_{r2}	$\leq$	π_{r3}	π_{r3}	$\leq$	π_{r2}	π_{r1}	$\leq$	π_{r3}
π_{r1}	$\leq$	π_{r3}	π_{r1}	$\leq$	π_{r2}	π_{r2}	$\leq$	π_{r3}

AS 2		AS 1	AS 2		AS 1	AS 2		AS 1
π_{r2}	$\leq$	π_{r3}	π_{r3}	$\leq$	π_{r1}	π_{r3}	$\leq$	π_{r2}
π_{r3}	$\leq$	π_{r1}	π_{r1}	$\leq$	π_{r2}	π_{r2}	$\leq$	π_{r1}
π_{r2}	$\leq$	π_{r1}	π_{r3}	$\leq$	π_{r2}	π_{r3}	$\leq$	π_{r1}

For every permutation both agents also obtain the same expected average reward (0). So:

$$PG_{2,1}(\pi_{r1},\pi_{r2},\pi_{r3},\Pi_o,w_{\dot{L}},\mu) = \sum_{\dot{l}\in \dot{L}_{-2,1}^{N(\mu)}(\Pi_o)} w_{\dot{L}}(\dot{l})PO_{2,1}(\pi_{r1},\pi_{r2},\pi_{r3},\dot{l},\mu) =$$
$$= \frac{1}{1}PO_{2,1}(\pi_{r1},\pi_{r2},\pi_{r3},(*,*),\mu) =$$
$$= \frac{1}{1}1 = 1$$

Note again that the choice of Π_o does not affect the result of $PG_{2,1}$.

And finally, we calculate the PG value:

$$PG(\Pi_e,w_{\Pi_e},\Pi_o,w_{\dot{L}},\mu,w_S) = \eta_{\Pi^3}\sum_{\pi_1,\pi_2,\pi_3\in\Pi_e|\pi_1\neq\pi_2\neq\pi_3} w_{\Pi_e}(\pi_1)w_{\Pi_e}(\pi_2)w_{\Pi_e}(\pi_3)\eta_{S_1^2}\sum_{i=1}^{N(\mu)} w_S(i,\mu)\times$$
$$\times\left(\sum_{j=1}^{i-1} w_S(j,\mu)PG_{i,j}(\pi_1,\pi_2,\pi_3,\Pi_o,w_{\dot{L}},\mu) + \sum_{j=i+1}^{N(\mu)} w_S(j,\mu)PG_{i,j}(\pi_1,\pi_2,\pi_3,\Pi_o,w_{\dot{L}},\mu)\right) =$$
$$= 6\frac{9}{2}\frac{1}{3}\frac{1}{3}\frac{1}{3}\frac{2}{1}\frac{1}{2}\frac{1}{2}\{PG_{1,2}(\pi_{r1},\pi_{r2},\pi_{r3},\Pi_o,w_{\dot{L}},\mu)+$$
$$+PG_{2,1}(\pi_{r1},\pi_{r2},\pi_{r3},\Pi_o,w_{\dot{L}},\mu)\} =$$
$$= 6\frac{9}{2}\frac{1}{3}\frac{1}{3}\frac{1}{3}\frac{2}{1}\frac{1}{2}\frac{1}{2}\{1+1\} = 1$$

Note that we avoided to calculate all the permutations of π_1,π_2,π_3 for $PG_{i,j}(\pi_1,\pi_2,\pi_3,\Pi_o,w_{\dot{L}},\mu)$, since they provide the same result, by calculating only one permutation and multiplying the result by 6 (the number of permutations).

So, for every Π_o we obtain the same result:

$$\forall\Pi_o : PG(\Pi_e,w_{\Pi_e},\Pi_o,w_{\dot{L}},\mu,w_S) = 1$$

Therefore, matching pennies has $Left_{max} = 1$ for this property. □

Proposition 21. *$Right_{min}$* for the partial grading (PG) property is equal to 0 for the matching pennies environment.

Proof. To find $Right_{min}$ (equation 8.5), we need to find a Π_e that minimises the property value as much as possible while Π_o maximises it. Using $\Pi_e = \{\pi_{h1},\pi_{h2},\pi_t\}$ (a π_h agent always performs Head and a π_t agent always performs Tail) we find this situation no matter which Π_o we use.

Following definition 31 we obtain the PG value for this $\langle\Pi_e,\Pi_o\rangle$ (where Π_o is instantiated with any permitted value). Since the multi-agent environment is not team symmetric, we need to calculate this property value for every pair of agent slots. Also, since $|\Pi_e| = 3$ we just need to calculate this property value for one trio of evaluated agents. Following definition 29 (for

PG) we calculate the PG value for the evaluated agents $\pi_{h1}, \pi_{h2}, \pi_t \in \Pi_e$ and each pair of agent slots. We start with agent slots 1 and 2:

$$
\begin{aligned}
PG_{1,2}(\pi_{h1}, \pi_{h2}, \pi_t, \Pi_o, w_{\dot{L}}, \mu) &= \sum_{\dot{l} \in \dot{L}_{-1,2}^{N(\mu)}(\Pi_o)} w_{\dot{L}}(\dot{l}) PO_{1,2}(\pi_{h1}, \pi_{h2}, \pi_t, \dot{l}, \mu) = \\
&= \frac{1}{1} PO_{1,2}(\pi_{h1}, \pi_{h2}, \pi_t, (*, *), \mu)
\end{aligned}
$$

Note that the choice of Π_o does not affect the result of $PG_{1,2}$.

The following table shows us $PO_{1,2}$ for all the permutations of $\pi_{h1}, \pi_{h2}, \pi_t$.

AS 1		AS 2	AS 1		AS 2	AS 1		AS 2
π_{h1}	$\leq$	π_{h2}	π_{h1}	$\leq$	π_t	π_{h2}	$\leq$	π_{h1}
π_{h2}	$\leq$	π_t	π_t	$\leq$	π_{h2}	π_{h1}	$\leq$	π_t
π_{h1}	$\leq$	π_t	π_{h1}	$\leq$	π_{h2}	π_{h2}	$\leq$	π_t
AS 1		AS 2	AS 1		AS 2	AS 1		AS 2
π_{h2}	$\leq$	π_t	π_t	$\leq$	π_{h1}	π_t	$\leq$	π_{h2}
π_t	$\leq$	π_{h1}	π_{h1}	$\leq$	π_{h2}	π_{h2}	$\leq$	π_{h1}
π_{h2}	$\leq$	π_{h1}	π_t	$\leq$	π_{h2}	π_t	$\leq$	π_{h1}

It is not possible to find a PO, since for every permutation we have either the agent line-up (π_{h1}, π_{h2}) or (π_{h2}, π_{h1}). In both cases, the agents obtain the expected average rewards of 1 and -1 respectively. So:

$$
PG_{1,2}(\pi_{h1}, \pi_{h2}, \pi_t, \Pi_o, w_{\dot{L}}, \mu) = \frac{1}{1} 0 = 0
$$

And for agent slots 2 and 1, the following table shows us $PO_{2,1}$ for all the permutations of $\pi_{h1}, \pi_{h2}, \pi_t$.

AS 2		AS 1	AS 2		AS 1	AS 2		AS 1
π_{h1}	$\leq$	π_{h2}	π_{h1}	$\leq$	π_t	π_{h2}	$\leq$	π_{h1}
π_{h2}	$\leq$	π_t	π_t	$\leq$	π_{h2}	π_{h1}	$\leq$	π_t
π_{h1}	$\leq$	π_t	π_{h1}	$\leq$	π_{h2}	π_{h2}	$\leq$	π_t
AS 2		AS 1	AS 2		AS 1	AS 2		AS 1
π_{h2}	$\leq$	π_t	π_t	$\leq$	π_{h1}	π_t	$\leq$	π_{h2}
π_t	$\leq$	π_{h1}	π_{h1}	$\leq$	π_{h2}	π_{h2}	$\leq$	π_{h1}
π_{h2}	$\leq$	π_{h1}	π_t	$\leq$	π_{h2}	π_t	$\leq$	π_{h1}

It is also not possible to find a PO, since for every permutation we have either the agent line-up (π_t, π_{h1}) or (π_{h1}, π_t). In both cases, the agents obtain the expected average rewards of -1 and 1 respectively. So:

$$
\begin{aligned}
PG_{2,1}(\pi_{h1}, \pi_{h2}, \pi_t, \Pi_o, w_{\dot{L}}, \mu) &= \sum_{\dot{l} \in \dot{L}_{-2,1}^{N(\mu)}(\Pi_o)} w_{\dot{L}}(\dot{l}) PO_{2,1}(\pi_{h1}, \pi_{h2}, \pi_t, \dot{l}, \mu) = \\
&= \frac{1}{1} PO_{2,1}(\pi_{h1}, \pi_{h2}, \pi_t, (*, *), \mu) = \\
&= \frac{1}{1} 0 = 0
\end{aligned}
$$

Note again that the choice of Π_o does not affect the result of $PG_{2,1}$.

And finally, we calculate the PG value:

$$PG(\Pi_e, w_{\Pi_e}, \Pi_o, w_{\dot{L}}, \mu, w_S) = \eta_{\Pi^3} \sum_{\pi_1,\pi_2,\pi_3 \in \Pi_e | \pi_1 \neq \pi_2 \neq \pi_3} w_{\Pi_e}(\pi_1) w_{\Pi_e}(\pi_2) w_{\Pi_e}(\pi_3) \eta_{S_1^2} \sum_{i=1}^{N(\mu)} w_S(i,\mu) \times$$
$$\times \left(\sum_{j=1}^{i-1} w_S(j,\mu) PG_{i,j}(\pi_1,\pi_2,\pi_3,\Pi_o,w_{\dot{L}},\mu) + \sum_{j=i+1}^{N(\mu)} w_S(j,\mu) PG_{i,j}(\pi_1,\pi_2,\pi_3,\Pi_o,w_{\dot{L}},\mu) \right) =$$
$$= 6 \frac{9}{2}\frac{1}{3}\frac{1}{3}\frac{1}{3}\frac{2}{1}\frac{1}{2}\frac{1}{2} \{PG_{1,2}(\pi_{h1},\pi_{h2},\pi_t,\Pi_o,w_{\dot{L}},\mu) +$$
$$+ PG_{2,1}(\pi_{h1},\pi_{h2},\pi_t,\Pi_o,w_{\dot{L}},\mu)\} =$$
$$= 6 \frac{9}{2}\frac{1}{3}\frac{1}{3}\frac{1}{3}\frac{2}{1}\frac{1}{2}\frac{1}{2} \{0+0\} = 0$$

Note that we avoided to calculate all the permutations of π_1, π_2, π_3 for $PG_{i,j}(\pi_1,\pi_2,\pi_3,\Pi_o,w_{\dot{L}},\mu)$, since they provide the same result, by calculating only one permutation and multiplying the result by 6 (the number of permutations).

So, for every Π_o we obtain the same result:

$$\forall \Pi_o : PG(\Pi_e, w_{\Pi_e}, \Pi_o, w_{\dot{L}}, \mu, w_S) = 0$$

Therefore, matching pennies has $Right_{min} = 0$ for this property. □

A.6 Slot Result Dependency

Next we see the slot result dependency (SRD) property. As given in section 7.3.2, we want to know how much competitive or cooperative the multi-agent environment is. We use an average of rewards as the utility function to calculate an agent's result.

Proposition 22. *General* range for the slot result dependency (SRD) property is equal to $[-1, -1]$ for the matching pennies environment.

Proof. Following definition 21 we obtain the SRD value for $\langle \Pi_e, \Pi_o \rangle$ (where Π_e and Π_o are instantiated with any permitted values). Since the multi-agent environment is not team symmetric, we need to calculate this property value for every pair of agent slots. Also, we do not know which Π_e we have, so we use a figurative evaluated agent π_1 from Π_e. Following definition 20 we calculate the SRD value for this figurative evaluated agent $\pi_1 \in \Pi_e$ and each pair of agent slots. We start with agent slots 1 and 2:

$$SRD_{1,2}(\pi_1, \Pi_o, w_{\dot{L}}, \mu) = corr_{\dot{l} \in \dot{L}_{-1}^{N(\mu)}(\Pi_o)}[w_{\dot{L}}(\dot{l})](R_1(\mu[\dot{l} \xleftarrow{1} \pi_1]), R_2(\mu[\dot{l} \xleftarrow{1} \pi_1]))$$

We do not know which Π_o we have, so we use a figurative agent line-up pattern $\dot{l} = (*, \pi_2)$ from $\dot{L}_{-1}^{N(\mu)}(\Pi_o)$:

$$corr(R_1(\mu[\dot{l} \xleftarrow{1} \pi_1]), R_2(\mu[\dot{l} \xleftarrow{1} \pi_1])) = corr(R_1(\mu[\pi_1,\pi_2]), R_2(\mu[\pi_1,\pi_2]))$$

This game is a zero-sum game with two agents. That means that, in every game, the sum of both agents' rewards is always zero or, in other words, when the agent in agent slot 1 (any

π_1) obtains a reward (r) the agent in agent slot 2 (any π_2) obtains the opposite reward ($-r$), and this relation is propagated to expected average rewards as well. Since we use a correlation function between expected average rewards, and the agents in agent slots 1 and 2 always obtain opposite expected average rewards, then the correlation function always obtains the same value[3] of -1. So, for any π_1 we obtain the same result:

$$corr(R_1(\mu[\pi_1,\pi_2]), R_2(\mu[\pi_1,\pi_2])) = -1$$

We generalise $SRD_{1,2}$ for any possible agent line-up pattern:

$$\begin{aligned} SRD_{1,2}(\pi_1,\Pi_o,w_{\dot{L}},\mu) &= corr_{\dot{l}\in\dot{L}_{-1}^{N(\mu)}(\Pi_o)}[w_{\dot{L}}(\dot{l})](R_1(\mu[\dot{l}\overset{1}{\leftarrow}\pi_1]), R_2(\mu[\dot{l}\overset{1}{\leftarrow}\pi_1])) = \\ &= corr_{\dot{l}\in\dot{L}_{-1}^{N(\mu)}(\Pi_o)}[w_{\dot{L}}(\dot{l})]\,(-1) = -1 \end{aligned}$$

And for agent slots 2 and 1, the correlation function also always obtains the same value of -1. So:

$$\begin{aligned} SRD_{2,1}(\pi_1,\Pi_o,w_{\dot{L}},\mu) &= corr_{\dot{l}\in\dot{L}_{-2}^{N(\mu)}(\Pi_o)}[w_{\dot{L}}(\dot{l})](R_2(\mu[\dot{l}\overset{2}{\leftarrow}\pi_1]), R_1(\mu[\dot{l}\overset{2}{\leftarrow}\pi_1])) = \\ &= corr_{\dot{l}\in\dot{L}_{-2}^{N(\mu)}(\Pi_o)}[w_{\dot{L}}(\dot{l})]\,(-1) = -1 \end{aligned}$$

And finally, we calculate the SRD value generalising for any possible evaluated agent:

$$\begin{aligned} SRD(\Pi_e,w_{\Pi_e},\Pi_o,w_{\dot{L}},\mu,w_S) &= \sum_{\pi\in\Pi_e} w_{\Pi_e}(\pi)\eta_{S_1^2}\sum_{i=1}^{N(\mu)} w_S(i,\mu)\times \\ &\times\left(\sum_{j=1}^{i-1} w_S(j,\mu)SRD_{i,j}(\pi,\Pi_o,w_{\dot{L}},\mu) + \sum_{j=i+1}^{N(\mu)} w_S(j,\mu)SRD_{i,j}(\pi,\Pi_o,w_{\dot{L}},\mu)\right) = \\ &= \sum_{\pi\in\Pi_e} w_{\Pi_e}(\pi)\frac{2}{1}\frac{1}{2}\frac{1}{2}\{SRD_{1,2}(\pi,\Pi_o,w_{\dot{L}},\mu)+ \\ &+ SRD_{2,1}(\pi,\Pi_o,w_{\dot{L}},\mu)\} = \\ &= \sum_{\pi\in\Pi_e} w_{\Pi_e}(\pi)\frac{2}{1}\frac{1}{2}\frac{1}{2}\{-1+(-1)\} = -1 \end{aligned}$$

So, for every pair $\langle\Pi_e,\Pi_o\rangle$ we obtain the same result:

$$\forall\Pi_e,\Pi_o : SRD(\Pi_e,w_{\Pi_e},\Pi_o,w_{\dot{L}},\mu,w_S) = -1$$

Therefore, matching pennies has *General* $= [-1,-1]$ for this property. □

A.7 Competitive Anticipation

Finally, we follow with the competitive anticipation (AComp) property. As given in section 7.4.1, we want to know how much benefit the evaluated agents obtain when they anticipate competing agents. We use an average of rewards as the utility function to calculate an agent's result.

[3]Provided there is at least one game which is not a tie.

Proposition 23. *$General_{min}$* for the competitive anticipation (AComp) property is equal to $-\frac{1}{2}$ for the matching pennies environment.

Proof. To find $General_{min}$ (equation 8.1), we need to find a pair $\langle \Pi_e, \Pi_o \rangle$ that minimises the property value as much as possible. We can have this situation by selecting $\Pi_e = \{\pi_{t/h}\}$ and $\Pi_o = \{\pi_h\}$ (a π_h agent always performs Head and a $\pi_{t/h}$ agent always performs Tail when interacting in agent slot 1 and always performs Head when interacting in agent slot 2)[4].

Following definition 23 we obtain the AComp value for this $\langle \Pi_e, \Pi_o \rangle$. Since the multi-agent environment is not team symmetric, we need to calculate this property value for every pair of agent slots in different teams. Also, since $|\Pi_e| = 1$ we just need to calculate this property value for one evaluated agent. Following definition 22 we calculate the Ant value for the evaluated agent $\pi_{t/h} \in \Pi_e$ and each pair of agent slots in different teams. We start with agent slots 1 and 2:

$$\begin{aligned} Ant_{1,2}(\pi_{t/h}, \Pi_o, w_{\dot{L}}, \mu) &= \sum_{\dot{l} \in \dot{L}_{-1,2}^{N(\mu)}(\Pi_o)} \sum_{\pi \in \Pi_o} w_{\dot{L}}(\dot{l} \overset{2}{\leftarrow} \pi) \times \\ &\times \frac{1}{2} \left(R_1(\mu[\dot{l} \overset{1,2}{\leftarrow} \pi_{t/h}, \pi]) - R_1(\mu[\dot{l} \overset{1,2}{\leftarrow} \pi_{t/h}, \pi_r]) \right) = \\ &= \frac{1}{1}\frac{1}{2} \left(R_1(\mu[\pi_{t/h}, \pi_h]) - R_1(\mu[\pi_{t/h}, \pi_r]) \right) \end{aligned}$$

In agent line-up $(\pi_{t/h}, \pi_h)$, the agent in agent slot 1 $(\pi_{t/h})$ obtains the expected average reward of -1, while in agent line-up $(\pi_{t/h}, \pi_r)$, the agent in agent slot 1 $(\pi_{t/h})$ obtains the expected average reward of 0 (as proved in lemma 2). So:

$$Ant_{1,2}(\pi_{t/h}, \Pi_o, w_{\dot{L}}, \mu) = \frac{1}{1}\frac{1}{2}\left[(-1) - 0\right] = -\frac{1}{2}$$

And for agent slots 2 and 1, in agent line-up $(\pi_h, \pi_{t/h})$, the agent in agent slot 2 $(\pi_{t/h})$ obtains the expected average reward of -1, while in agent line-up $(\pi_r, \pi_{t/h})$, the agent in agent slot 2 $(\pi_{t/h})$ obtains the expected average reward of 0. So:

$$\begin{aligned} Ant_{2,1}(\pi_{t/h}, \Pi_o, w_{\dot{L}}, \mu) &= \sum_{\dot{l} \in \dot{L}_{-2,1}^{N(\mu)}(\Pi_o)} \sum_{\pi \in \Pi_o} w_{\dot{L}}(\dot{l} \overset{1}{\leftarrow} \pi) \times \\ &\times \frac{1}{2} \left(R_2(\mu[\dot{l} \overset{2,1}{\leftarrow} \pi_{t/h}, \pi]) - R_2(\mu[\dot{l} \overset{2,1}{\leftarrow} \pi_{t/h}, \pi_r]) \right) = \\ &= \frac{1}{1}\frac{1}{2} \left(R_2(\mu[\pi_h, \pi_{t/h}]) - R_2(\mu[\pi_r, \pi_{t/h}]) \right) = \\ &= \frac{1}{1}\frac{1}{2}\left[(-1) - 0\right] = -\frac{1}{2} \end{aligned}$$

And finally, we calculate the AComp value:

[4]Note that $\pi_{t/h}$ has to know in which agent slot it is interacting. To infer this, it starts with a random action at the first time step and then it looks at the other agent's action and the reward it obtains.

$$\begin{aligned}
AComp(\Pi_e, w_{\Pi_e}, \Pi_o, w_{\dot{L}}, \mu, w_S) &= \sum_{\pi \in \Pi_e} w_{\Pi_e}(\pi) \eta_{S_2^2} \sum_{t_1, t_2 \in \tau | t_1 \neq t_2} \sum_{i \in t_1} w_S(i, \mu) \sum_{j \in t_2} w_S(j, \mu) \times \\
&\times Ant_{i,j}(\pi, \Pi_o, w_{\dot{L}}, \mu) = \\
&= \frac{1}{1}\frac{2}{1}\frac{1}{2}\frac{1}{2}\{Ant_{1,2}(\pi_{t/h}, \Pi_o, w_{\dot{L}}, \mu) + Ant_{2,1}(\pi_{t/h}, \Pi_o, w_{\dot{L}}, \mu)\} = \\
&= \frac{1}{1}\frac{2}{1}\frac{1}{2}\frac{1}{2}\left\{-\frac{1}{2} + \left(-\frac{1}{2}\right)\right\} = -\frac{1}{2}
\end{aligned}$$

Since $-\frac{1}{2}$ is the lowest possible value that we can obtain for the competitive anticipation property, therefore matching pennies has $General_{min} = -\frac{1}{2}$ for this property. □

Proposition 24. *$General_{max}$ for the competitive anticipation (AComp) property is equal to $\frac{1}{2}$ for the matching pennies environment.*

Proof. To find $General_{max}$ (equation 8.2), we need to find a pair $\langle \Pi_e, \Pi_o \rangle$ that maximises the property value as much as possible. We can have this situation by selecting $\Pi_e = \{\pi_{h/t}\}$ and $\Pi_o = \{\pi_h\}$ (a π_h agent always performs Head and a $\pi_{h/t}$ agent always performs Head when interacting in agent slot 1 and always performs Tail when interacting in agent slot 2)[5].

Following definition 23 we obtain the AComp value for this $\langle \Pi_e, \Pi_o \rangle$. Since the multi-agent environment is not team symmetric, we need to calculate this property value for every pair of agent slots in different teams. Also, since $|\Pi_e| = 1$ we just need to calculate this property value for one evaluated agent. Following definition 22 we calculate the Ant value for the evaluated agent $\pi_{h/t} \in \Pi_e$ and each pair of agent slots in different teams. We start with agent slots 1 and 2:

$$\begin{aligned}
Ant_{1,2}(\pi_{h/t}, \Pi_o, w_{\dot{L}}, \mu) &= \sum_{\dot{l} \in \dot{L}_{-1,2}^{N(\mu)}(\Pi_o)} \sum_{\pi \in \Pi_o} w_{\dot{L}}(\dot{l} \overset{2}{\leftarrow} \pi) \times \\
&\times \frac{1}{2}\left(R_1(\mu[\dot{l} \overset{1,2}{\leftarrow} \pi_{h/t}, \pi]) - R_1(\mu[\dot{l} \overset{1,2}{\leftarrow} \pi_{h/t}, \pi_r])\right) = \\
&= \frac{1}{1}\frac{1}{2}\left(R_1(\mu[\pi_{h/t}, \pi_h]) - R_1(\mu[\pi_{h/t}, \pi_r])\right)
\end{aligned}$$

In agent line-up $(\pi_{h/t}, \pi_h)$, the agent in agent slot 1 $(\pi_{h/t})$ obtains the expected average reward of 1, while in agent line-up $(\pi_{h/t}, \pi_r)$, the agent in agent slot 1 $(\pi_{h/t})$ obtains the expected average reward of 0 (as proved in lemma 2). So:

$$Ant_{1,2}(\pi_{h/t}, \Pi_o, w_{\dot{L}}, \mu) = \frac{1}{1}\frac{1}{2}(1 - 0) = \frac{1}{2}$$

And for agent slots 2 and 1, in agent line-up $(\pi_h, \pi_{h/t})$, the agent in agent slot 2 $(\pi_{h/t})$ obtains the expected average reward of 1, while in agent line-up $(\pi_r, \pi_{h/t})$, the agent in agent slot 2 $(\pi_{h/t})$ obtains the expected average reward of 0. So:

[5]Note that $\pi_{h/t}$ has to know in which agent slot it is interacting. To infer this, it starts with a random action at the first time step and then it looks at the other agent's action and the reward it obtains.

$$
\begin{aligned}
Ant_{2,1}(\pi_{h/t}, \Pi_o, w_{\dot{L}}, \mu) &= \sum_{\dot{l} \in \dot{L}^{N(\mu)}_{-2,1}(\Pi_o)} \sum_{\pi \in \Pi_o} w_{\dot{L}}(\dot{l} \overset{1}{\leftarrow} \pi) \times \\
&\times \frac{1}{2}\left(R_2(\mu[\dot{l} \overset{2,1}{\leftarrow} \pi_{h/t}, \pi]) - R_2(\mu[\dot{l} \overset{2,1}{\leftarrow} \pi_{h/t}, \pi_r])\right) = \\
&= \frac{1}{1}\frac{1}{2}\left(R_2(\mu[\pi_h, \pi_{h/t}]) - R_2(\mu[\pi_r, \pi_{h/t}])\right) = \\
&= \frac{1}{1}\frac{1}{2}(1-0) = \frac{1}{2}
\end{aligned}
$$

And finally, we calculate the AComp value:

$$
\begin{aligned}
AComp(\Pi_e, w_{\Pi_e}, \Pi_o, w_{\dot{L}}, \mu, w_S) &= \sum_{\pi \in \Pi_e} w_{\Pi_e}(\pi) \eta_{S_2^2} \sum_{t_1, t_2 \in \tau | t_1 \neq t_2} \sum_{i \in t_1} w_S(i, \mu) \sum_{j \in t_2} w_S(j, \mu) \times \\
&\times Ant_{i,j}(\pi, \Pi_o, w_{\dot{L}}, \mu) = \\
&= \frac{1}{1}\frac{2}{1}\frac{1}{2}\frac{1}{2}\{Ant_{1,2}(\pi_{h/t}, \Pi_o, w_{\dot{L}}, \mu) + Ant_{2,1}(\pi_{h/t}, \Pi_o, w_{\dot{L}}, \mu)\} = \\
&= \frac{1}{1}\frac{2}{1}\frac{1}{2}\frac{1}{2}\left\{\frac{1}{2} + \frac{1}{2}\right\} = \frac{1}{2}
\end{aligned}
$$

Since $\frac{1}{2}$ is the highest possible value that we can obtain for the competitive anticipation property, therefore matching pennies has $General_{max} = \frac{1}{2}$ for this property. $\square$

Appendix B

Prisoner's Dilemma Properties

In this section we prove how we obtained the values for the properties for the prisoner's dilemma environment (section 2.1.2). We use uniform unit weights for w_{Π_e}, $w_{\dot{L}}$ and w_S.

B.1 Action Dependency

We start with the action dependency (AD) property. As given in section 7.2.1, we want to know if the evaluated agents behave differently depending on which agent line-up they interact with. We use $\Delta_{\mathcal{A}^+}(a,b) = 0$ if distributions a and b are equal and 1 otherwise.

Proposition 25. *General*$_{min}$ for the action dependency (AD) property is equal to 0 for the prisoner's dilemma environment.

Proof. To find *General*$_{min}$ (equation 8.1), we need to find a pair $\langle \Pi_e, \Pi_o \rangle$ that minimises the property value as much as possible. We can have this situation by selecting $\Pi_e = \{\pi_b\}$ and $\Pi_o = \{\pi_{s1}, \pi_{s2}\}$ (a π_s agent always performs Silent and a π_b agent always performs Betray).

Following definition 17 we obtain the AD value for this $\langle \Pi_e, \Pi_o \rangle$. Since the multi-agent environment is team symmetric, we just need to calculate this property value for one agent slot and generalise it to all agent slots. Also, since $|\Pi_e| = 1$ we just need to calculate this property value for one evaluated agent. Following definition 16 we calculate the AD value for the evaluated agent $\pi_b \in \Pi_e$ and agent slot 1:

$$AD_1(\pi_b, \Pi_o, w_{\dot{L}}, \mu) = \eta_{\dot{L}^2} \sum_{\dot{u},\dot{v} \in \dot{L}_{-1}^{N(\mu)}(\Pi_o) | \dot{u} \neq \dot{v}} w_{\dot{L}}(\dot{u}) w_{\dot{L}}(\dot{v}) \Delta_{\mathcal{A}^+}(\breve{A}_1(\mu[\dot{u} \xleftarrow{1} \pi_b]), \breve{A}_1(\mu[\dot{v} \xleftarrow{1} \pi_b])) =$$

$$= 2\frac{2}{1}\frac{1}{2}\frac{1}{2}\Delta_{\mathcal{A}^+}(\breve{A}_1(\mu[\pi_b, \pi_{s1}]), \breve{A}_1(\mu[\pi_b, \pi_{s2}]))$$

Note that we avoided to calculate both $\Delta_{\mathcal{A}^+}(a,b)$ and $\Delta_{\mathcal{A}^+}(b,a)$, since they provide the same result, by calculating only $\Delta_{\mathcal{A}^+}(a,b)$ and multiplying the result by 2.

The agent in both agent slots 1 (π_b) performs the same sequence of actions (always Betray) independently of the agent line-up. So:

$$AD_1(\pi_b, \Pi_o, w_{\dot{L}}, \mu) = 2\frac{2}{1}\frac{1}{2}\frac{1}{2}0 = 0$$

And finally, we calculate the AD value generalising for every agent slot:

$$AD(\Pi_e, w_{\Pi_e}, \Pi_o, w_{\dot{L}}, \mu, w_S) = \sum_{\pi \in \Pi_e} w_{\Pi_e}(\pi) \sum_{i=1}^{N(\mu)} w_S(i,\mu) AD_i(\pi, \Pi_o, w_{\dot{L}}, \mu) =$$
$$= \frac{1}{1}\sum_{i=1}^{N(\mu)} w_S(i,\mu)0 = 0$$

Since 0 is the lowest possible value for the action dependency property, therefore prisoner's dilemma has $General_{min} = 0$ for this property. □

Proposition 26. $General_{max}$ for the action dependency (AD) property is equal to 1 for the prisoner's dilemma environment.

Proof. To find $General_{max}$ (equation 8.2), we need to find a pair $\langle \Pi_e, \Pi_o \rangle$ that maximises the property value as much as possible. We can have this situation by selecting $\Pi_e = \{\pi_m\}$ and $\Pi_o = \{\pi_s, \pi_b\}$ (a π_m agent first acts randomly and from time step 2 always mimics the other agent's last action, a π_s agent always performs Silent and a π_b agent always performs Betray).

Following definition 17 we obtain the AD value for this $\langle \Pi_e, \Pi_o \rangle$. Since the multi-agent environment is team symmetric, we just need to calculate this property value for one agent slot and generalise it to all agent slots. Also, since $|\Pi_e| = 1$ we just need to calculate this property value for one evaluated agent. Following definition 16 we calculate the AD value for the evaluated agent $\pi_m \in \Pi_e$ and agent slot 1:

$$AD_1(\pi_m, \Pi_o, w_{\dot{L}}, \mu) = \eta_{\dot{L}^2} \sum_{\dot{u},\dot{v} \in \dot{L}_{-1}^{N(\mu)}(\Pi_o) | \dot{u} \neq \dot{v}} w_{\dot{L}}(\dot{u}) w_{\dot{L}}(\dot{v}) \Delta_{\mathcal{A}^+}(\breve{A}_1(\mu[\dot{u} \overset{1}{\leftarrow} \pi_m]), \breve{A}_1(\mu[\dot{v} \overset{1}{\leftarrow} \pi_m])) =$$
$$= 2\frac{2}{1}\frac{1}{2}\frac{1}{2}\Delta_{\mathcal{A}^+}(\breve{A}_1(\mu[\pi_m, \pi_s]), \breve{A}_1(\mu[\pi_m, \pi_b]))$$

Note that we avoided to calculate both $\Delta_{\mathcal{A}^+}(a,b)$ and $\Delta_{\mathcal{A}^+}(b,a)$, since they provide the same result, by calculating only $\Delta_{\mathcal{A}^+}(a,b)$ and multiplying the result by 2.

From time step 2, π_m mimics the last action of the agent in agent slot 2, so the agent in both agent slots 1 (π_m) performs different sequences of actions depending on the agent line-up. So:

$$AD_1(\pi_m, \Pi_o, w_{\dot{L}}, \mu) = 2\frac{2}{1}\frac{1}{2}\frac{1}{2}1 = 1$$

And finally, we calculate the AD value generalising for every agent slot:

$$AD(\Pi_e, w_{\Pi_e}, \Pi_o, w_{\dot{L}}, \mu, w_S) = \sum_{\pi \in \Pi_e} w_{\Pi_e}(\pi) \sum_{i=1}^{N(\mu)} w_S(i,\mu) AD_i(\pi, \Pi_o, w_{\dot{L}}, \mu) =$$
$$= \frac{1}{1}\sum_{i=1}^{N(\mu)} w_S(i,\mu)1 = 1$$

Since 1 is the highest possible value for the action dependency property, therefore prisoner's dilemma has $General_{max} = 1$ for this property. □

Proposition 27. *$Left_{max}$ for the action dependency (AD) property is equal to 0 for the prisoner's dilemma environment.*

Proof. To find $Left_{max}$ (equation 8.4), we need to find a Π_e that maximises the property value as much as possible while Π_o minimises it. Using $\Pi_o = \{\pi_{s1}, \pi_{s2}\}$ (a π_s agent always performs Silent) we find this situation no matter which Π_e we use.

Following definition 17 we obtain the AD value for this $\langle\Pi_e, \Pi_o\rangle$ (where Π_e is instantiated with any permitted value). Since the multi-agent environment is team symmetric, we just need to calculate this property value for one agent slot and generalise it to all agent slots. Also, we do not know which Π_e we have, so we use a figurative evaluated agent π from Π_e. Following definition 16 we calculate the AD value for this figurative evaluated agent $\pi \in \Pi_e$ and agent slot 1:

$$AD_1(\pi, \Pi_o, w_{\dot{L}}, \mu) = \eta_{\dot{L}^2} \sum_{\dot{u},\dot{v} \in \dot{L}_{-1}^{N(\mu)}(\Pi_o) | \dot{u} \neq \dot{v}} w_{\dot{L}}(\dot{u}) w_{\dot{L}}(\dot{v}) \Delta_{\mathcal{A}^+}(\breve{A}_1(\mu[\dot{u} \overset{1}{\leftarrow} \pi]), \breve{A}_1(\mu[\dot{v} \overset{1}{\leftarrow} \pi])) =$$
$$= 2\frac{2}{1}\frac{1}{2}\frac{1}{2}\Delta_{\mathcal{A}^+}(\breve{A}_1(\mu[\pi, \pi_{s1}]), \breve{A}_1(\mu[\pi, \pi_{s2}]))$$

Note that we avoided to calculate both $\Delta_{\mathcal{A}^+}(a, b)$ and $\Delta_{\mathcal{A}^+}(b, a)$, since they provide the same result, by calculating only $\Delta_{\mathcal{A}^+}(a, b)$ and multiplying the result by 2.

The agent in both agent slots 1 (any π) is not able to differentiate with which agent is interacting, so it is not able to change its distribution of action sequences depending on the opponent's behaviour. So, for any π we obtain the same result:

$$AD_1(\pi, \Pi_o, w_{\dot{L}}, \mu) = 2\frac{2}{1}\frac{1}{2}\frac{1}{2}0 = 0$$

And finally, we calculate the AD value generalising for every agent slot and any possible evaluated agent:

$$AD(\Pi_e, w_{\Pi_e}, \Pi_o, w_{\dot{L}}, \mu, w_S) = \sum_{\pi \in \Pi_e} w_{\Pi_e}(\pi) \sum_{i=1}^{N(\mu)} w_S(i, \mu) AD_i(\pi, \Pi_o, w_{\dot{L}}, \mu) =$$
$$= \sum_{\pi \in \Pi_e} w_{\Pi_e}(\pi) \sum_{i=1}^{N(\mu)} w_S(i, \mu) 0 = 0$$

So, for every Π_e we obtain the same result:

$$\forall \Pi_e : AD(\Pi_e, w_{\Pi_e}, \Pi_o, w_{\dot{L}}, \mu, w_S) = 0$$

Therefore, prisoner's dilemma has $Left_{max} = 0$ for this property. □

Proposition 28. *$Right_{min}$ for the action dependency (AD) property is equal to 0 for the prisoner's dilemma environment.*

Proof. To find $Right_{min}$ (equation 8.5), we need to find a Π_e that minimises the property value as much as possible while Π_o maximises it. Using $\Pi_e = \{\pi_s\}$ (a π_s agent always performs Silent) we find this situation no matter which Π_o we use.

Following definition 17 we obtain the AD value for this $\langle \Pi_e, \Pi_o \rangle$ (where Π_o is instantiated with any permitted value). Since the multi-agent environment is team symmetric, we just need to calculate this property value for one agent slot and generalise it to all agent slots. Also, since $|\Pi_e| = 1$ we just need to calculate this property value for one evaluated agent. Following definition 16 we calculate the AD value for the evaluated agent $\pi_s \in \Pi_e$ and agent slot 1:

$$AD_1(\pi_s, \Pi_o, w_{\dot{L}}, \mu) = \eta_{\dot{L}^2} \sum_{\dot{u},\dot{v} \in \dot{L}_{-1}^{N(\mu)}(\Pi_o) | \dot{u} \neq \dot{v}} w_{\dot{L}}(\dot{u}) w_{\dot{L}}(\dot{v}) \Delta_{\mathcal{A}^+}(\breve{A}_1(\mu[\dot{u} \overset{1}{\leftarrow} \pi_s]), \breve{A}_1(\mu[\dot{v} \overset{1}{\leftarrow} \pi_s]))$$

We do not know which Π_o we have, so we use two figurative agent line-up patterns $\dot{u} = (*, \pi_1)$ and $\dot{v} = (*, \pi_2)$ from $\dot{L}_{-1}^{N(\mu)}(\Pi_o)$:

$$\Delta_{\mathcal{A}^+}(\breve{A}_1(\mu[\dot{u} \overset{1}{\leftarrow} \pi_s]), \breve{A}_1(\mu[\dot{v} \overset{1}{\leftarrow} \pi_s])) = \Delta_{\mathcal{A}^+}(\breve{A}_1(\mu[\pi_s, \pi_1]), \breve{A}_1(\mu[\pi_s, \pi_2]))$$

The agent in both agent slots 1 (π_s) performs the same sequence of actions (always Silent) independently of the agent line-up. So:

$$\Delta_{\mathcal{A}^+}(\breve{A}_1(\mu[\pi_s, \pi_1]), \breve{A}_1(\mu[\pi_s, \pi_2])) = 0$$

We generalise AD_1 for any possible pair of agent line-up patterns:

$$\begin{aligned} AD_1(\pi_s, \Pi_o, w_{\dot{L}}, \mu) &= \eta_{\dot{L}^2} \sum_{\dot{u},\dot{v} \in \dot{L}_{-1}^{N(\mu)}(\Pi_o) | \dot{u} \neq \dot{v}} w_{\dot{L}}(\dot{u}) w_{\dot{L}}(\dot{v}) \Delta_{\mathcal{A}^+}(\breve{A}_1(\mu[\dot{u} \overset{1}{\leftarrow} \pi_s]), \breve{A}_1(\mu[\dot{v} \overset{1}{\leftarrow} \pi_s])) = \\ &= \eta_{\dot{L}^2} \sum_{\dot{u},\dot{v} \in \dot{L}_{-1}^{N(\mu)}(\Pi_o) | \dot{u} \neq \dot{v}} w_{\dot{L}}(\dot{u}) w_{\dot{L}}(\dot{v}) 0 = 0 \end{aligned}$$

And finally, we calculate the AD value generalising for every agent slot:

$$\begin{aligned} AD(\Pi_e, w_{\Pi_e}, \Pi_o, w_{\dot{L}}, \mu, w_S) &= \sum_{\pi \in \Pi_e} w_{\Pi_e}(\pi) \sum_{i=1}^{N(\mu)} w_S(i, \mu) AD_i(\pi, \Pi_o, w_{\dot{L}}, \mu) = \\ &= \frac{1}{1} \sum_{i=1}^{N(\mu)} w_S(i, \mu) 0 = 0 \end{aligned}$$

So, for every Π_o we obtain the same result:

$$\forall \Pi_o : AD(\Pi_e, w_{\Pi_e}, \Pi_o, w_{\dot{L}}, \mu, w_S) = 0$$

Therefore, prisoner's dilemma has $Right_{min} = 0$ for this property. □

B.2 Reward Dependency

We continue with the reward dependency (RD) property. As given in section 7.3.1, we want to know if the evaluated agents obtain different rewards depending on which agent line-up they interact with. We use $\Delta_{\mathbb{Q}}(a', b')$ for $\Delta_{\mathbb{Q}^+}(a, b)$ where a' and b' are the expected results of a and b respectively, and $\Delta_{\mathbb{Q}}(a', b') = 0$ if numbers a' and b' are equal and 1 otherwise. We use an average of rewards as the utility function to calculate an agent's result.

Proposition 29. *General$_{min}$* for the reward dependency (RD) property is equal to 0 for the prisoner's dilemma environment.

Proof. To find *General$_{min}$* (equation 8.1), we need to find a pair $\langle \Pi_e, \Pi_o \rangle$ that minimises the property value as much as possible. We can have this situation by selecting $\Pi_e = \{\pi_b\}$ and $\Pi_o = \{\pi_{s1}, \pi_{s2}\}$ (a π_s agent always performs Silent and a π_b agent always performs Betray).

Following definition 19 we obtain the RD value for this $\langle \Pi_e, \Pi_o \rangle$. Since the multi-agent environment is team symmetric, we just need to calculate this property value for one agent slot and generalise it to all agent slots. Also, since $|\Pi_e| = 1$ we just need to calculate this property value for one evaluated agent. Following definition 18 we calculate the RD value for the evaluated agent $\pi_b \in \Pi_e$ and agent slot 1:

$$RD_1(\pi_b, \Pi_o, w_{\dot{L}}, \mu) = \eta_{\dot{L}^2} \sum_{\dot{u}, \dot{v} \in \dot{L}_{-1}^{N(\mu)}(\Pi_o) | \dot{u} \neq \dot{v}} w_{\dot{L}}(\dot{u}) w_{\dot{L}}(\dot{v}) \Delta_{\mathbb{Q}}(R_1(\mu[\dot{u} \overset{1}{\leftarrow} \pi_b]), R_1(\mu[\dot{v} \overset{1}{\leftarrow} \pi_b])) =$$
$$= 2 \frac{2}{1} \frac{1}{2} \frac{1}{2} \Delta_{\mathbb{Q}}(R_1(\mu[\pi_b, \pi_{s1}]), R_1(\mu[\pi_b, \pi_{s2}]))$$

Note that we avoided to calculate both $\Delta_{\mathbb{Q}}(a, b)$ and $\Delta_{\mathbb{Q}}(b, a)$, since they provide the same result, by calculating only $\Delta_{\mathbb{Q}}(a, b)$ and multiplying the result by 2.

The agent in both agent slots 1 (π_b) obtains the same expected average reward (1) independently of the agent line-up. So:

$$RD_1(\pi_b, \Pi_o, w_{\dot{L}}, \mu) = 2 \frac{2}{1} \frac{1}{2} \frac{1}{2} 0 = 0$$

And finally, we calculate the RD value generalising for every agent slot:

$$RD(\Pi_e, w_{\Pi_e}, \Pi_o, w_{\dot{L}}, \mu, w_S) = \sum_{\pi \in \Pi_e} w_{\Pi_e}(\pi) \sum_{i=1}^{N(\mu)} w_S(i, \mu) RD_i(\pi, \Pi_o, w_{\dot{L}}, \mu) =$$
$$= \frac{1}{1} \sum_{i=1}^{N(\mu)} w_S(i, \mu) 0 = 0$$

Since 0 is the lowest possible value for the reward dependency property, therefore prisoner's dilemma has *General$_{min}$* $= 0$ for this property. $\square$

Proposition 30. *General$_{max}$* for the reward dependency (RD) property is equal to 1 for the prisoner's dilemma environment.

Proof. To find $General_{max}$ (equation 8.2), we need to find a pair $\langle \Pi_e, \Pi_o \rangle$ that maximises the property value as much as possible. We can have this situation by selecting $\Pi_e = \{\pi_b\}$ and $\Pi_o = \{\pi_s, \pi_b\}$ (a π_s agent always performs Silent and a π_b agent always performs Betray).

Following definition 19 we obtain the RD value for this $\langle \Pi_e, \Pi_o \rangle$. Since the multi-agent environment is team symmetric, we just need to calculate this property value for one agent slot and generalise it to all agent slots. Also, since $|\Pi_e| = 1$ we just need to calculate this property value for one evaluated agent. Following definition 18 we calculate the RD value for the evaluated agent $\pi_b \in \Pi_e$ and agent slot 1:

$$RD_1(\pi_b, \Pi_o, w_{\dot{L}}, \mu) = \eta_{\dot{L}^2} \sum_{\dot{u},\dot{v} \in \dot{L}_{-1}^{N(\mu)}(\Pi_o) | \dot{u} \neq \dot{v}} w_{\dot{L}}(\dot{u}) w_{\dot{L}}(\dot{v}) \Delta_{\mathbb{Q}}(R_1(\mu[\dot{u} \overset{1}{\leftarrow} \pi_b]), R_1(\mu[\dot{v} \overset{1}{\leftarrow} \pi_b])) =$$
$$= 2\frac{2}{1}\frac{1}{2}\frac{1}{2}\Delta_{\mathbb{Q}}(R_1(\mu[\pi_b, \pi_s]), R_1(\mu[\pi_b, \pi_b]))$$

Note that we avoided to calculate both $\Delta_{\mathbb{Q}}(a, b)$ and $\Delta_{\mathbb{Q}}(b, a)$, since they provide the same result, by calculating only $\Delta_{\mathbb{Q}}(a, b)$ and multiplying the result by 2.

In agent line-up (π_b, π_s), the agent in agent slot 1 (π_b) obtains one expected average reward (1), while in agent line-up (π_b, π_b), the agent in agent slot 1 (π_b) obtains a different expected average reward $(-\frac{1}{3})$. So:

$$RD_1(\pi_b, \Pi_o, w_{\dot{L}}, \mu) = 2\frac{2}{1}\frac{1}{2}\frac{1}{2}1 = 1$$

And finally, we calculate the RD value generalising for every agent slot:

$$RD(\Pi_e, w_{\Pi_e}, \Pi_o, w_{\dot{L}}, \mu, w_S) = \sum_{\pi \in \Pi_e} w_{\Pi_e}(\pi) \sum_{i=1}^{N(\mu)} w_S(i, \mu) RD_i(\pi, \Pi_o, w_{\dot{L}}, \mu) =$$
$$= \frac{1}{1} \sum_{i=1}^{N(\mu)} w_S(i, \mu) 1 = 1$$

Since 1 is the highest possible value for the reward dependency property, therefore prisoner's dilemma has $General_{max} = 1$ for this property. □

Proposition 31. *$Left_{max}$ for the reward dependency (RD) property is equal to 0 for the prisoner's dilemma environment.*

Proof. To find $Left_{max}$ (equation 8.4), we need to find a Π_e that maximises the property value as much as possible while Π_o minimises it. Using $\Pi_o = \{\pi_{s1}, \pi_{s2}\}$ (a π_s agent always performs Silent) we find this situation no matter which Π_e we use.

Following definition 19 we obtain the RD value for this $\langle \Pi_e, \Pi_o \rangle$ (where Π_e is instantiated with any permitted value). Since the multi-agent environment is team symmetric, we just need to calculate this property value for one agent slot and generalise it to all agent slots. Also, we do not know which Π_e we have, so we use a figurative evaluated agent π from Π_e. Following definition 18 we calculate the RD value for this figurative evaluated agent $\pi \in \Pi_e$ and agent slot 1:

$$RD_1(\pi, \Pi_o, w_{\dot{L}}, \mu) = \eta_{\dot{L}^2} \sum_{\dot{u},\dot{v} \in \dot{L}_{-1}^{N(\mu)}(\Pi_o) | \dot{u} \neq \dot{v}} w_{\dot{L}}(\dot{u}) w_{\dot{L}}(\dot{v}) \Delta_{\mathbb{Q}}(R_1(\mu[\dot{u} \overset{1}{\leftarrow} \pi]), R_1(\mu[\dot{v} \overset{1}{\leftarrow} \pi])) =$$
$$= 2\frac{2}{1}\frac{1}{2}\frac{1}{2}\Delta_{\mathbb{Q}}(R_1(\mu[\pi, \pi_{s1}]), R_1(\mu[\pi, \pi_{s2}]))$$

Note that we avoided to calculate both $\Delta_{\mathbb{Q}}(a, b)$ and $\Delta_{\mathbb{Q}}(b, a)$, since they provide the same result, by calculating only $\Delta_{\mathbb{Q}}(a, b)$ and multiplying the result by 2.

The agent in both agent slots 1 (any π) is not able to differentiate with which agent is interacting, so it is not able to change its distribution of action sequences depending on the opponent's behaviour, obtaining the same expected average reward. So, for any π we obtain the same result:

$$RD_1(\pi, \Pi_o, w_{\dot{L}}, \mu) = 2\frac{2}{1}\frac{1}{2}\frac{1}{2}0 = 0$$

And finally, we calculate the RD value generalising for every agent slot and any possible evaluated agent:

$$RD(\Pi_e, w_{\Pi_e}, \Pi_o, w_{\dot{L}}, \mu, w_S) = \sum_{\pi \in \Pi_e} w_{\Pi_e}(\pi) \sum_{i=1}^{N(\mu)} w_S(i, \mu) RD_i(\pi, \Pi_o, w_{\dot{L}}, \mu) =$$
$$= \sum_{\pi \in \Pi_e} w_{\Pi_e}(\pi) \sum_{i=1}^{N(\mu)} w_S(i, \mu) 0 = 0$$

So, for every Π_e we obtain the same result:

$$\forall \Pi_e : RD(\Pi_e, w_{\Pi_e}, \Pi_o, w_{\dot{L}}, \mu, w_S) = 0$$

Therefore, prisoner's dilemma has $Left_{max} = 0$ for this property. □

Proposition 32. $Right_{min}$ for the reward dependency (RD) property is equal to 1 for the prisoner's dilemma environment.

Proof. To find $Right_{min}$ (equation 8.5), we need to find a Π_e that minimises the property value as much as possible while Π_o maximises it. Using $\Pi_o = \{\pi_s, \pi_b\}$ (a π_s agent always performs Silent and a π_b agent always performs Betray) we find this situation no matter which Π_e we use.

Following definition 19 we obtain the RD value for this $\langle \Pi_e, \Pi_o \rangle$ (where Π_e is instantiated with any permitted value). Since the multi-agent environment is team symmetric, we just need to calculate this property value for one agent slot and generalise it to all agent slots. Also, we do not know which Π_e we have, so we use a figurative evaluated agent π from Π_e. Following definition 18 we calculate the RD value for this figurative evaluated agent $\pi \in \Pi_e$ and agent slot 1:

$$RD_1(\pi, \Pi_o, w_{\dot{L}}, \mu) = \eta_{\dot{L}^2} \sum_{\dot{u},\dot{v} \in \dot{L}_{-1}^{N(\mu)}(\Pi_o) | \dot{u} \neq \dot{v}} w_{\dot{L}}(\dot{u}) w_{\dot{L}}(\dot{v}) \Delta_{\mathbb{Q}}(R_1(\mu[\dot{u} \overset{1}{\leftarrow} \pi]), R_1(\mu[\dot{v} \overset{1}{\leftarrow} \pi])) =$$
$$= 2\frac{2}{1}\frac{1}{2}\frac{1}{2}\Delta_{\mathbb{Q}}(R_1(\mu[\pi, \pi_s]), R_1(\mu[\pi, \pi_b]))$$

Note that we avoided to calculate both $\Delta_{\mathbb{Q}}(a, b)$ and $\Delta_{\mathbb{Q}}(b, a)$, since they provide the same result, by calculating only $\Delta_{\mathbb{Q}}(a, b)$ and multiplying the result by 2.

In agent line-up (π, π_s), the agent in agent slot 1 (any π) obtains an expected average reward (between $\frac{1}{3}$ and 1), while in agent line-up (π, π_b), the agent in agent slot 1 (any π) obtains another different expected average reward (between -1 and $-\frac{1}{3}$). So, for any π we obtain the same result:

$$RD_1(\pi, \Pi_o, w_{\dot{L}}, \mu) = 2\frac{2}{1}\frac{1}{2}\frac{1}{2}1 = 1$$

And finally, we calculate the RD value generalising for every agent slot and any possible evaluated agent:

$$\begin{aligned} RD(\Pi_e, w_{\Pi_e}, \Pi_o, w_{\dot{L}}, \mu, w_S) &= \sum_{\pi \in \Pi_e} w_{\Pi_e}(\pi) \sum_{i=1}^{N(\mu)} w_S(i, \mu) RD_i(\pi, \Pi_o, w_{\dot{L}}, \mu) = \\ &= \sum_{\pi \in \Pi_e} w_{\Pi_e}(\pi) \sum_{i=1}^{N(\mu)} w_S(i, \mu) 1 = 1 \end{aligned}$$

So, for every Π_e we obtain the same result:

$$\forall \Pi_e : RD(\Pi_e, w_{\Pi_e}, \Pi_o, w_{\dot{L}}, \mu, w_S) = 1$$

Therefore, prisoner's dilemma has $Right_{min} = 1$ for this property. □

B.3 Fine Discrimination

Now we move to the fine discrimination (FD) property. As given in section 7.5.1, we want to know if different evaluated agents obtain different expected rewards when interacting in the same agent line-up patterns. We use $\Delta_{\mathbb{Q}}(a, b) = 0$ if numbers a and b are equal and 1 otherwise. We use an average of rewards as the utility function to calculate an agent's result.

Proposition 33. *General$_{min}$* for the fine discrimination (FD) property is equal to 0 for the prisoner's dilemma environment.

Proof. To find $General_{min}$ (equation 8.1), we need to find a pair $\langle \Pi_e, \Pi_o \rangle$ that minimises the property value as much as possible. We can have this situation by selecting $\Pi_e = \{\pi_{b1}, \pi_{b2}\}$ and $\Pi_o = \{\pi_s\}$ (a π_s agent always performs Silent and a π_b agent always performs Betray).

Following definition 26 we obtain the FD value for this $\langle \Pi_e, \Pi_o \rangle$. Since the multi-agent environment is team symmetric, we just need to calculate this property value for one agent slot and generalise it to all agent slots. Also, since $|\Pi_e| = 2$ we just need to calculate this property value for one pair of evaluated agents. Following definition 25 we calculate the FD value for the evaluated agents $\pi_{b1}, \pi_{b2} \in \Pi_e$ and agent slot 1:

$$\begin{aligned} FD_1(\pi_{b1}, \pi_{b2}, \Pi_o, w_{\dot{L}}, \mu) &= \sum_{\dot{l} \in \dot{L}_{-1}^{N(\mu)}(\Pi_o)} w_{\dot{L}}(\dot{l}) \Delta_{\mathbb{Q}}(R_1(\mu[\dot{l} \overset{1}{\leftarrow} \pi_{b1}]), R_1(\mu[\dot{l} \overset{1}{\leftarrow} \pi_{b2}])) = \\ &= \frac{1}{1} \Delta_{\mathbb{Q}}(R_1(\mu[\pi_{b1}, \pi_s]), R_1(\mu[\pi_{b2}, \pi_s])) \end{aligned}$$

Both agents in agent slot 1 (π_{b1} and π_{b2}) obtain the same expected average reward (1). So:

$$FD_1(\pi_{b1}, \pi_{b2}, \Pi_o, w_{\dot{L}}, \mu) = \frac{1}{1}0 = 0$$

And finally, we calculate the FD value generalising for every agent slot:

$$\begin{aligned} FD(\Pi_e, w_{\Pi_e}, \Pi_o, w_{\dot{L}}, \mu, w_S) = \eta_{\Pi^2} \sum_{\pi_1,\pi_2 \in \Pi_e | \pi_1 \neq \pi_2} w_{\Pi_e}(\pi_1) w_{\Pi_e}(\pi_2) \sum_{i=1}^{N(\mu)} w_S(i,\mu) \times \\ \times FD_i(\pi_1, \pi_2, \Pi_o, w_{\dot{L}}, \mu) = \\ = 2\frac{2}{1}\frac{1}{2}\frac{1}{2} \sum_{i=1}^{N(\mu)} w_S(i,\mu) 0 = 0 \end{aligned}$$

Note that we avoided to calculate both $FD_i(a, b, \Pi_o, w_{\dot{L}}, \mu)$ and $FD_i(b, a, \Pi_o, w_{\dot{L}}, \mu)$, since they provide the same result, by calculating only $FD_i(a, b, \Pi_o, w_{\dot{L}}, \mu)$ and multiplying the result by 2.

Since 0 is the lowest possible value for the fine discrimination property, therefore prisoner's dilemma has $General_{min} = 0$ for this property. □

Proposition 34. *$General_{max}$ for the fine discrimination (FD) property is equal to 1 for the prisoner's dilemma environment.*

Proof. To find $General_{max}$ (equation 8.2), we need to find a pair $\langle \Pi_e, \Pi_o \rangle$ that maximises the property value as much as possible. We can have this situation by selecting $\Pi_e = \{\pi_s, \pi_b\}$ and $\Pi_o = \{\pi_s\}$ (a π_s agent always performs Silent and a π_b agent always performs Betray).

Following definition 26 we obtain the FD value for this $\langle \Pi_e, \Pi_o \rangle$. Since the multi-agent environment is team symmetric, we just need to calculate this property value for one agent slot and generalise it to all agent slots. Also, since $|\Pi_e| = 2$ we just need to calculate this property value for one pair of evaluated agents. Following definition 25 we calculate the FD value for the evaluated agents $\pi_s, \pi_b \in \Pi_e$ and agent slot 1:

$$\begin{aligned} FD_1(\pi_s, \pi_b, \Pi_o, w_{\dot{L}}, \mu) = \sum_{\dot{l} \in \dot{L}_{-1}^{N(\mu)}(\Pi_o)} w_{\dot{L}}(\dot{l}) \Delta_{\mathbb{Q}}(R_1(\mu[\dot{l} \xleftarrow{1} \pi_s]), R_1(\mu[\dot{l} \xleftarrow{1} \pi_b])) = \\ = \frac{1}{1} \Delta_{\mathbb{Q}}(R_1(\mu[\pi_s, \pi_s]), R_1(\mu[\pi_b, \pi_s])) \end{aligned}$$

Both agents in agent slot 1 (π_s and π_b) obtain different expected average rewards ($\frac{1}{3}$ and 1 respectively). So:

$$FD_1(\pi_s, \pi_b, \Pi_o, w_{\dot{L}}, \mu) = \frac{1}{1}1 = 1$$

And finally, we calculate the FD value generalising for every agent slot:

$$\begin{aligned} FD(\Pi_e, w_{\Pi_e}, \Pi_o, w_{\dot{L}}, \mu, w_S) = \eta_{\Pi^2} \sum_{\pi_1,\pi_2 \in \Pi_e | \pi_1 \neq \pi_2} w_{\Pi_e}(\pi_1) w_{\Pi_e}(\pi_2) \sum_{i=1}^{N(\mu)} w_S(i,\mu) \times \\ \times FD_i(\pi_1, \pi_2, \Pi_o, w_{\dot{L}}, \mu) = \\ = 2\frac{2}{1}\frac{1}{2}\frac{1}{2} \sum_{i=1}^{N(\mu)} w_S(i,\mu) 1 = 1 \end{aligned}$$

Note that we avoided to calculate both $FD_i(a, b, \Pi_o, w_{\dot{L}}, \mu)$ and $FD_i(b, a, \Pi_o, w_{\dot{L}}, \mu)$, since they provide the same result, by calculating only $FD_i(a, b, \Pi_o, w_{\dot{L}}, \mu)$ and multiplying the result by 2.

Since 1 is the highest possible value for the fine discrimination property, therefore prisoner's dilemma has $General_{max} = 1$ for this property. □

Conjecture 1. *$Left_{max}$ for the fine discrimination (FD) property is equal to 0 for the prisoner's dilemma environment.*

An agent $\pi \in \Pi_o$ can force every evaluated agent to obtain an expected average reward equal to 0 (in the limit). The procedure is simple. While the evaluated agent has an expected average reward lower than 0, π performs Silent forcing the evaluated agent to increase its expected average reward. Analogously, while the evaluated agent has an expected average reward greater than 0, π performs Betray forcing the evaluated agent to decrease its expected average reward. If this procedure is repeated indefinitely, the expected average reward of any evaluated agent tends to 0. So:

$$\forall \Pi_e \exists \Pi_o : FD(\Pi_e, w_{\Pi_e}, \Pi_o, w_{\dot{L}}, \mu, w_S) = 0$$

Therefore, prisoner's dilemma has $Left_{max} = 0$ for this property.

Proposition 35. *$Right_{min}$ for the fine discrimination (FD) property is equal to 0 for the prisoner's dilemma environment.*

Proof. To find $Right_{min}$ (equation 8.5), we need to find a Π_e that minimises the property value as much as possible while Π_o maximises it. Using $\Pi_e = \{\pi_{b1}, \pi_{b2}\}$ (a π_b agent always performs Betray) we find this situation no matter which Π_o we use.

Following definition 26 we obtain the FD value for this $\langle \Pi_e, \Pi_o \rangle$ (where Π_o is instantiated with any permitted value). Since the multi-agent environment is team symmetric, we just need to calculate this property value for one agent slot and generalise it to all agent slots. Also, since $|\Pi_e| = 2$ we just need to calculate this property value for one pair of evaluated agents. Following definition 25 we calculate the FD value for the evaluated agents $\pi_{b1}, \pi_{b2} \in \Pi_e$ and agent slot 1:

$$FD_1(\pi_{b1}, \pi_{b2}, \Pi_o, w_{\dot{L}}, \mu) = \sum_{\dot{l} \in \dot{L}_{-1}^{N(\mu)}(\Pi_o)} w_{\dot{L}}(\dot{l}) \Delta_{\mathbb{Q}}(R_1(\mu[\dot{l} \overset{1}{\leftarrow} \pi_{b1}]), R_1(\mu[\dot{l} \overset{1}{\leftarrow} \pi_{b2}]))$$

We do not know which Π_o we have, so we use a figurative agent line-up pattern $\dot{l} = (*, \pi)$ from $\dot{L}_{-1}^{N(\mu)}(\Pi_o)$:

$$\Delta_{\mathbb{Q}}(R_1(\mu[\dot{l} \overset{1}{\leftarrow} \pi_{b1}]), R_1(\mu[\dot{l} \overset{1}{\leftarrow} \pi_{b2}])) = \Delta_{\mathbb{Q}}(R_1(\mu[\pi_{b1}, \pi]), R_1(\mu[\pi_{b2}, \pi]))$$

The agent in both agent slots 2 (any π) is not able to differentiate with which agent is interacting, so it is not able to change its distribution of action sequences depending on the opponent's behaviour, obtaining both agents in agent slot 1 (π_{b1} and π_{b2}) the same expected average reward. So:

$$\Delta_{\mathbb{Q}}(R_1(\mu[\pi_{b1}, \pi]), R_1(\mu[\pi_{b2}, \pi])) = 0$$

We generalise FD_1 for any possible agent line-up pattern:

$$FD_1(\pi_{b1}, \pi_{b2}, \Pi_o, w_{\dot{L}}, \mu) = \sum_{\dot{l} \in \dot{L}_{-1}^{N(\mu)}(\Pi_o)} w_{\dot{L}}(\dot{l}) \Delta_{\mathbb{Q}}(R_1(\mu[\dot{l} \overset{1}{\leftarrow} \pi_{b1}]), R_1(\mu[\dot{l} \overset{1}{\leftarrow} \pi_{b2}])) =$$
$$= \sum_{\dot{l} \in \dot{L}_{-1}^{N(\mu)}(\Pi_o)} w_{\dot{L}}(\dot{l}) 0 = 0$$

And finally, we calculate the FD value generalising for every agent slot:

$$FD(\Pi_e, w_{\Pi_e}, \Pi_o, w_{\dot{L}}, \mu, w_S) = \eta_{\Pi^2} \sum_{\pi_1, \pi_2 \in \Pi_e | \pi_1 \neq \pi_2} w_{\Pi_e}(\pi_1) w_{\Pi_e}(\pi_2) \sum_{i=1}^{N(\mu)} w_S(i, \mu) \times$$
$$\times FD_i(\pi_1, \pi_2, \Pi_o, w_{\dot{L}}, \mu) =$$
$$= 2 \frac{2}{1} \frac{1}{2} \frac{1}{2} \sum_{i=1}^{N(\mu)} w_S(i, \mu) 0 = 0$$

Note that we avoided to calculate both $FD_i(a, b, \Pi_o, w_{\dot{L}}, \mu)$ and $FD_i(b, a, \Pi_o, w_{\dot{L}}, \mu)$, since they provide the same result, by calculating only $FD_i(a, b, \Pi_o, w_{\dot{L}}, \mu)$ and multiplying the result by 2.

So, for every Π_o we obtain the same result:

$$\forall \Pi_o : FD(\Pi_e, w_{\Pi_e}, \Pi_o, w_{\dot{L}}, \mu, w_S) = 0$$

Therefore, prisoner's dilemma has $Right_{min} = 0$ for this property. □

B.4 Strict Total Grading

We arrive to the strict total grading (STG) property. As given in section 7.5.2, we want to know if there exists a strict ordering between the evaluated agents when interacting in the multi-agent environment. We use an average of rewards as the utility function to calculate an agent's result.

To simplify the notation, we use the next table to represent the STO: $R_i(\mu[\dot{l} \overset{i,j}{\leftarrow} \pi_1, \pi_2]) < R_j(\mu[\dot{l} \overset{i,j}{\leftarrow} \pi_1, \pi_2])$, $R_i(\mu[\dot{l} \overset{i,j}{\leftarrow} \pi_2, \pi_3]) < R_j(\mu[\dot{l} \overset{i,j}{\leftarrow} \pi_2, \pi_3])$ and $R_i(\mu[\dot{l} \overset{i,j}{\leftarrow} \pi_1, \pi_3]) < R_j(\mu[\dot{l} \overset{i,j}{\leftarrow} \pi_1, \pi_3])$.

AS i		AS j
π_1	$<$	π_2
π_2	$<$	π_3
π_1	$<$	π_3

where AS stands for agent slot.

Proposition 36. *General*$_{min}$ for the strict total grading (STG) property is equal to 0 for the prisoner's dilemma environment.

Proof. To find $General_{min}$ (equation 8.1), we need to find a pair $\langle \Pi_e, \Pi_o \rangle$ that minimises the property value as much as possible. We can have this situation by selecting $\Pi_e = \{\pi_{b1}, \pi_{b2}, \pi_{b3}\}$ and $\Pi_o = \emptyset$ (a π_b agent always performs Betray).

Following definition 30 we obtain the STG value for this $\langle \Pi_e, \Pi_o \rangle$. Since the multi-agent environment is team symmetric, we just need to calculate this property value for one pair of agent slots and generalise it to all pairs of agent slots. Also, since $|\Pi_e| = 3$ we just need to calculate this property value for one trio of evaluated agents. Following definition 29 we calculate the STG value for the evaluated agents $\pi_{b1}, \pi_{b2}, \pi_{b3} \in \Pi_e$ and the pair of agent slots 1 and 2:

$$STG_{1,2}(\pi_{b1}, \pi_{b2}, \pi_{b3}, \Pi_o, w_{\dot{L}}, \mu) = \sum_{\dot{l} \in \dot{L}^{N(\mu)}_{-1,2}(\Pi_o)} w_{\dot{L}}(\dot{l}) STO_{1,2}(\pi_{b1}, \pi_{b2}, \pi_{b3}, \dot{l}, \mu) =$$
$$= \frac{1}{1} STO_{1,2}(\pi_{b1}, \pi_{b2}, \pi_{b3}, (*, *), \mu)$$

The following table shows us $STO_{1,2}$ for all the permutations of $\pi_{b1}, \pi_{b2}, \pi_{b3}$.

AS 1		AS 2	AS 1		AS 2	AS 1		AS 2
π_{b1}	$<$	π_{b2}	π_{b1}	$<$	π_{b3}	π_{b2}	$<$	π_{b1}
π_{b2}	$<$	π_{b3}	π_{b3}	$<$	π_{b2}	π_{b1}	$<$	π_{b3}
π_{b1}	$<$	π_{b3}	π_{b1}	$<$	π_{b2}	π_{b2}	$<$	π_{b3}
AS 1		AS 2	AS 1		AS 2	AS 1		AS 2
π_{b2}	$<$	π_{b3}	π_{b3}	$<$	π_{b1}	π_{b3}	$<$	π_{b2}
π_{b3}	$<$	π_{b1}	π_{b1}	$<$	π_{b2}	π_{b2}	$<$	π_{b1}
π_{b2}	$<$	π_{b1}	π_{b3}	$<$	π_{b2}	π_{b3}	$<$	π_{b1}

It is not possible to find a STO, since for every permutation both agents obtain the same expected average reward ($-\frac{1}{3}$). So:

$$STG_{1,2}(\pi_{b1}, \pi_{b2}, \pi_{b3}, \Pi_o, w_{\dot{L}}, \mu) = \frac{1}{1} 0 = 0$$

And finally, we calculate the STG value generalising for every pair of agent slots:

$$STG(\Pi_e, w_{\Pi_e}, \Pi_o, w_{\dot{L}}, \mu, w_S) = \eta_{\Pi^3} \sum_{\pi_1,\pi_2,\pi_3 \in \Pi_e | \pi_1 \neq \pi_2 \neq \pi_3} w_{\Pi_e}(\pi_1) w_{\Pi_e}(\pi_2) w_{\Pi_e}(\pi_3) \eta_{S_1^2} \sum_{i=1}^{N(\mu)} w_S(i, \mu) \times$$
$$\times \left(\sum_{j=1}^{i-1} w_S(j, \mu) STG_{i,j}(\pi_1, \pi_2, \pi_3, \Pi_o, w_{\dot{L}}, \mu) + \sum_{j=i+1}^{N(\mu)} w_S(j, \mu) STG_{i,j}(\pi_1, \pi_2, \pi_3, \Pi_o, w_{\dot{L}}, \mu) \right) =$$
$$= 6 \frac{9}{2} \frac{1}{3} \frac{1}{3} \frac{1}{3} \eta_{S_1^2} \sum_{i=1}^{N(\mu)} w_S(i, \mu) \left(\sum_{j=1}^{i-1} w_S(j, \mu) 0 + \sum_{j=i+1}^{N(\mu)} w_S(j, \mu) 0 \right) = 0$$

Note that we avoided to calculate all the permutations of π_1, π_2, π_3 for $STG_{i,j}(\pi_1, \pi_2, \pi_3, \Pi_o, w_{\dot{L}}, \mu)$, since they provide the same result, by calculating only one permutation and multiplying the result by 6 (the number of permutations).

Since 0 is the lowest possible value for the strict total grading property, therefore prisoner's dilemma has $General_{min} = 0$ for this property. □

Proposition 37. *$General_{max}$* for the strict total grading (STG) property is equal to 1 for the prisoner's dilemma environment.

Proof. To find $General_{max}$ (equation 8.2), we need to find a pair $\langle \Pi_e, \Pi_o \rangle$ that maximises the property value as much as possible. We can have this situation by selecting $\Pi_e = \{\pi_s, \pi_b, \pi_r\}$ and $\Pi_o = \emptyset$ (a π_s agent always performs Silent, a π_b agent always performs Betray and a π_r agent always acts randomly).

Following definition 30 we obtain the STG value for this $\langle \Pi_e, \Pi_o \rangle$. Since the multi-agent environment is team symmetric, we just need to calculate this property value for one pair of agent slots and generalise it to all pairs of agent slots. Also, since $|\Pi_e| = 3$ we just need to calculate this property value for one trio of evaluated agents. Following definition 29 we calculate the STG value for the evaluated agents $\pi_s, \pi_b, \pi_r \in \Pi_e$ and the pair of agent slots 1 and 2:

$$STG_{1,2}(\pi_s, \pi_b, \pi_r, \Pi_o, w_{\dot{L}}, \mu) = \sum_{\dot{l} \in \dot{L}_{-1,2}^{N(\mu)}(\Pi_o)} w_{\dot{L}}(\dot{l}) STO_{1,2}(\pi_s, \pi_b, \pi_r, \dot{l}, \mu) =$$

$$= \frac{1}{1} STO_{1,2}(\pi_s, \pi_b, \pi_r, (*, *), \mu)$$

The following table shows us $STO_{1,2}$ for all the permutations of π_s, π_b, π_r.

AS 1		AS 2	AS 1		AS 2	AS 1		AS 2
π_s	$<$	π_b	π_s	$<$	π_r	π_b	$<$	π_s
π_b	$<$	π_r	π_r	$<$	π_b	π_s	$<$	π_r
π_s	$<$	π_r	π_s	$<$	π_b	π_b	$<$	π_r
AS 1		AS 2	AS 1		AS 2	AS 1		AS 2
π_b	$<$	π_r	π_r	$<$	π_s	π_r	$<$	π_b
π_r	$<$	π_s	π_s	$<$	π_b	π_b	$<$	π_s
π_b	$<$	π_s	π_r	$<$	π_b	π_r	$<$	π_s

It is possible to find a STO for the second permutation. In agent line-up (π_s, π_r), π_s always performs Silent and π_r always acts randomly, obtaining the expected average rewards of $-\frac{1}{3}$ and $\frac{2}{3}$ respectively. In agent line-up (π_r, π_b), π_r always acts randomly and π_b always performs Betray, obtaining the expected average rewards of $-\frac{2}{3}$ and $\frac{1}{3}$ respectively. In agent line-up (π_s, π_b), π_s always performs Silent and π_b always performs Betray, obtaining the expected average rewards of -1 and 1 respectively. So:

$$STG_{1,2}(\pi_s, \pi_b, \pi_r, \Pi_o, w_{\dot{L}}, \mu) = \frac{1}{1} 1 = 1$$

And finally, we calculate the STG value generalising for every pair of agent slots:

$$STG(\Pi_e, w_{\Pi_e}, \Pi_o, w_{\dot{L}}, \mu, w_S) = \eta_{\Pi^3} \sum_{\pi_1,\pi_2,\pi_3 \in \Pi_e | \pi_1 \neq \pi_2 \neq \pi_3} w_{\Pi_e}(\pi_1) w_{\Pi_e}(\pi_2) w_{\Pi_e}(\pi_3) \eta_{S_1^2} \sum_{i=1}^{N(\mu)} w_S(i,\mu) \times$$
$$\times \left(\sum_{j=1}^{i-1} w_S(j,\mu) STG_{i,j}(\pi_1,\pi_2,\pi_3,\Pi_o,w_{\dot{L}},\mu) + \sum_{j=i+1}^{N(\mu)} w_S(j,\mu) STG_{i,j}(\pi_1,\pi_2,\pi_3,\Pi_o,w_{\dot{L}},\mu) \right) =$$
$$= 6 \frac{9}{2} \frac{1}{3} \frac{1}{3} \frac{1}{3} \eta_{S_1^2} \sum_{i=1}^{N(\mu)} w_S(i,\mu) \left(\sum_{j=1}^{i-1} w_S(j,\mu) 1 + \sum_{j=i+1}^{N(\mu)} w_S(j,\mu) 1 \right) = 1$$

Note that we avoided to calculate all the permutations of π_1, π_2, π_3 for $STG_{i,j}(\pi_1, \pi_2, \pi_3, \Pi_o, w_{\dot{L}}, \mu)$, since they provide the same result, by calculating only one permutation and multiplying the result by 6 (the number of permutations).

Since 1 is the highest possible value for the strict total grading property, therefore prisoner's dilemma has $General_{max} = 1$ for this property. □

Proposition 38. *$Left_{max}$ for the strict total grading (STG) property is equal to 1 for the prisoner's dilemma environment.*

Proof. To find $Left_{max}$ (equation 8.4), we need to find a Π_e that maximises the property value as much as possible while Π_o minimises it. Using $\Pi_e = \{\pi_s, \pi_b, \pi_r\}$ (a π_s agent always performs Silent, a π_b agent always performs Betray and a π_r agent always acts randomly) we find this situation no matter which Π_o we use.

Following definition 30 we obtain the STG value for this $\langle \Pi_e, \Pi_o \rangle$ (where Π_o is instantiated with any permitted value). Since the multi-agent environment is team symmetric, we just need to calculate this property value for one pair of agent slots and generalise it to all pairs of agent slots. Also, since $|\Pi_e| = 3$ we just need to calculate this property value for one trio of evaluated agents. Following definition 29 we calculate the STG value for the evaluated agents $\pi_s, \pi_b, \pi_r \in \Pi_e$ and the pair of agent slots 1 and 2:

$$STG_{1,2}(\pi_s, \pi_b, \pi_r, \Pi_o, w_{\dot{L}}, \mu) = \sum_{\dot{l} \in \dot{L}_{-1,2}^{N(\mu)}(\Pi_o)} w_{\dot{L}}(\dot{l}) STO_{1,2}(\pi_s, \pi_b, \pi_r, \dot{l}, \mu) =$$
$$= \frac{1}{1} STO_{1,2}(\pi_s, \pi_b, \pi_r, (*,*), \mu)$$

Note that the choice of Π_o does not affect the result of $STG_{1,2}$.

The following table shows us $STO_{1,2}$ for all the permutations of π_s, π_b, π_r.

AS 1		AS 2	AS 1		AS 2	AS 1		AS 2
π_s	$<$	π_b	π_s	$<$	π_r	π_b	$<$	π_s
π_b	$<$	π_r	π_r	$<$	π_b	π_s	$<$	π_r
π_s	$<$	π_r	π_s	$<$	π_b	π_b	$<$	π_r
AS 1		AS 2	AS 1		AS 2	AS 1		AS 2
π_b	$<$	π_r	π_r	$<$	π_s	π_r	$<$	π_b
π_r	$<$	π_s	π_s	$<$	π_b	π_b	$<$	π_s
π_b	$<$	π_s	π_r	$<$	π_b	π_r	$<$	π_s

It is possible to find a STO for the second permutation. In agent line-up (π_s, π_r), π_s always performs Silent and π_r always acts randomly, obtaining the expected average rewards of $-\frac{1}{3}$ and $\frac{2}{3}$ respectively. In agent line-up (π_r, π_b), π_r always acts randomly and π_b always performs Betray, obtaining the expected average rewards of $-\frac{2}{3}$ and $\frac{1}{3}$ respectively. In agent line-up (π_s, π_b), π_s always performs Silent and π_b always performs Betray, obtaining the expected average rewards of -1 and 1 respectively. So:

$$STG_{1,2}(\pi_s, \pi_b, \pi_r, \Pi_o, w_{\dot{L}}, \mu) = \frac{1}{1}1 = 1$$

And finally, we calculate the STG value generalising for every pair of agent slots:

$$STG(\Pi_e, w_{\Pi_e}, \Pi_o, w_{\dot{L}}, \mu, w_S) = \eta_{\Pi^3} \sum_{\pi_1,\pi_2,\pi_3 \in \Pi_e | \pi_1 \neq \pi_2 \neq \pi_3} w_{\Pi_e}(\pi_1) w_{\Pi_e}(\pi_2) w_{\Pi_e}(\pi_3) \eta_{S_1^2} \sum_{i=1}^{N(\mu)} w_S(i,\mu) \times$$
$$\times \left(\sum_{j=1}^{i-1} w_S(j,\mu) STG_{i,j}(\pi_1, \pi_2, \pi_3, \Pi_o, w_{\dot{L}}, \mu) + \sum_{j=i+1}^{N(\mu)} w_S(j,\mu) STG_{i,j}(\pi_1, \pi_2, \pi_3, \Pi_o, w_{\dot{L}}, \mu) \right) =$$
$$= 6 \frac{9}{2} \frac{1}{3} \frac{1}{3} \frac{1}{3} \eta_{S_1^2} \sum_{i=1}^{N(\mu)} w_S(i,\mu) \left(\sum_{j=1}^{i-1} w_S(j,\mu) 1 + \sum_{j=i+1}^{N(\mu)} w_S(j,\mu) 1 \right) = 1$$

Note that we avoided to calculate all the permutations of π_1, π_2, π_3 for $STG_{i,j}(\pi_1, \pi_2, \pi_3, \Pi_o, w_{\dot{L}}, \mu)$, since they provide the same result, by calculating only one permutation and multiplying the result by 6 (the number of permutations).

So, for every Π_o we obtain the same result:

$$\forall \Pi_o : STG(\Pi_e, w_{\Pi_e}, \Pi_o, w_{\dot{L}}, \mu, w_S) = 1$$

Therefore, prisoner's dilemma has $Left_{max} = 1$ for this property. □

Proposition 39. *$Right_{min}$ for the strict total grading (STG) property is equal to 0 for the prisoner's dilemma environment.*

Proof. To find $Right_{min}$ (equation 8.5), we need to find a Π_e that minimises the property value as much as possible while Π_o maximises it. Using $\Pi_e = \{\pi_{b1}, \pi_{b2}, \pi_{b3}\}$ (a π_b agent always performs Betray) we find this situation no matter which Π_o we use.

Following definition 30 we obtain the STG value for this $\langle \Pi_e, \Pi_o \rangle$ (where Π_o is instantiated with any permitted value). Since the multi-agent environment is team symmetric, we just need to calculate this property value for one pair of agent slots and generalise it to all pairs of agent slots. Also, since $|\Pi_e| = 3$ we just need to calculate this property value for one trio of evaluated agents. Following definition 29 we calculate the STG value for the evaluated agents $\pi_{b1}, \pi_{b2}, \pi_{b3} \in \Pi_e$ and the pair of agent slots 1 and 2:

$$STG_{1,2}(\pi_{b1}, \pi_{b2}, \pi_{b3}, \Pi_o, w_{\dot{L}}, \mu) = \sum_{\dot{l} \in \dot{L}_{-1,2}^{N(\mu)}(\Pi_o)} w_{\dot{L}}(\dot{l}) STO_{1,2}(\pi_{b1}, \pi_{b2}, \pi_{b3}, \dot{l}, \mu) =$$
$$= \frac{1}{1} STO_{1,2}(\pi_{b1}, \pi_{b2}, \pi_{b3}, (*,*), \mu)$$

Note that the choice of Π_o does not affect the result of $STG_{1,2}$.

The following table shows us $STO_{1,2}$ for all the permutations of $\pi_{b1}, \pi_{b2}, \pi_{b3}$.

AS 1		AS 2	AS 1		AS 2	AS 1		AS 2
π_{b1}	$<$	π_{b2}	π_{b1}	$<$	π_{b3}	π_{b2}	$<$	π_{b1}
π_{b2}	$<$	π_{b3}	π_{b3}	$<$	π_{b2}	π_{b1}	$<$	π_{b3}
π_{b1}	$<$	π_{b3}	π_{b1}	$<$	π_{b2}	π_{b2}	$<$	π_{b3}
AS 1		AS 2	AS 1		AS 2	AS 1		AS 2
π_{b2}	$<$	π_{b3}	π_{b3}	$<$	π_{b1}	π_{b3}	$<$	π_{b2}
π_{b3}	$<$	π_{b1}	π_{b1}	$<$	π_{b2}	π_{b2}	$<$	π_{b1}
π_{b2}	$<$	π_{b1}	π_{b3}	$<$	π_{b2}	π_{b3}	$<$	π_{b1}

It is not possible to find a STO, since for every permutation both agents obtain the same expected average reward ($-\frac{1}{3}$). So:

$$STG_{1,2}(\pi_{b1}, \pi_{b2}, \pi_{b3}, \Pi_o, w_{\dot{L}}, \mu) = \frac{1}{1}0 = 0$$

And finally, we calculate the STG value generalising for every pair of agent slots:

$$STG(\Pi_e, w_{\Pi_e}, \Pi_o, w_{\dot{L}}, \mu, w_S) = \eta_{\Pi^3} \sum_{\pi_1,\pi_2,\pi_3 \in \Pi_e | \pi_1 \neq \pi_2 \neq \pi_3} w_{\Pi_e}(\pi_1) w_{\Pi_e}(\pi_2) w_{\Pi_e}(\pi_3) \eta_{S_1^2} \sum_{i=1}^{N(\mu)} w_S(i,\mu) \times$$
$$\times \left(\sum_{j=1}^{i-1} w_S(j,\mu) STG_{i,j}(\pi_1, \pi_2, \pi_3, \Pi_o, w_{\dot{L}}, \mu) + \sum_{j=i+1}^{N(\mu)} w_S(j,\mu) STG_{i,j}(\pi_1, \pi_2, \pi_3, \Pi_o, w_{\dot{L}}, \mu) \right) =$$
$$= 6 \frac{9}{2} \frac{1}{3} \frac{1}{3} \frac{1}{3} \eta_{S_1^2} \sum_{i=1}^{N(\mu)} w_S(i,\mu) \left(\sum_{j=1}^{i-1} w_S(j,\mu) 0 + \sum_{j=i+1}^{N(\mu)} w_S(j,\mu) 0 \right) = 0$$

Note that we avoided to calculate all the permutations of π_1, π_2, π_3 for $STG_{i,j}(\pi_1, \pi_2, \pi_3, \Pi_o, w_{\dot{L}}, \mu)$, since they provide the same result, by calculating only one permutation and multiplying the result by 6 (the number of permutations).

So, for every Π_o we obtain the same result:

$$\forall \Pi_o : STG(\Pi_e, w_{\Pi_e}, \Pi_o, w_{\dot{L}}, \mu, w_S) = 0$$

Therefore, prisoner's dilemma has $Right_{min} = 0$ for this property. □

B.5 Partial Grading

Now we arrive to the partial grading (PG) property. As given in section 7.5.2, we want to know if there exists a partial ordering between the evaluated agents when interacting in the multi-agent environment. We use an average of rewards as the utility function to calculate an agent's result.

To simplify the notation, we use the next table to represent the PO: $R_i(\mu[l \overset{i,j}{\leftarrow} \pi_1, \pi_2]) \leq R_j(\mu[l \overset{i,j}{\leftarrow} \pi_1, \pi_2])$, $R_i(\mu[l \overset{i,j}{\leftarrow} \pi_2, \pi_3]) \leq R_j(\mu[l \overset{i,j}{\leftarrow} \pi_2, \pi_3])$ and $R_i(\mu[l \overset{i,j}{\leftarrow} \pi_1, \pi_3]) \leq R_j(\mu[l \overset{i,j}{\leftarrow} \pi_1, \pi_3])$.

AS i		AS j
π_1	$\leq$	π_2
π_2	$\leq$	π_3
π_1	$\leq$	π_3

where AS stands for agent slot.

Proposition 40. *$General_{min}$ for the partial grading (PG) property is equal to 0 for the prisoner's dilemma environment.*

Proof. To find $General_{min}$ (equation 8.1), we need to find a pair $\langle \Pi_e, \Pi_o \rangle$ that minimises the property value as much as possible. We can have this situation by selecting $\Pi_e = \{\pi_b^{-ssb}, \pi_b^{-bbs}, \pi_m\}$ and $\Pi_o = \emptyset$ (a π_m agent first acts randomly and from time step 2 always mimics the other agent's last action, a π_b^{-ssb} agent always performs Betray except for the last three actions where it performs Silent twice and finalises performing Betray, and a π_b^{-bbs} agent always performs Betray except for the last three actions where it performs Betray twice and finalises performing Silent).

Following definition 31 we obtain the PG value for this $\langle \Pi_e, \Pi_o \rangle$. Since the multi-agent environment is team symmetric, we just need to calculate this property value for one pair of agent slots and generalise it to all pairs of agent slots. Also, since $|\Pi_e| = 3$ we just need to calculate this property value for one trio of evaluated agents. Following definition 29 (for PG) we calculate the PG value for the evaluated agents $\pi_b^{-ssb}, \pi_b^{-bbs}, \pi_m \in \Pi_e$ and the pair of agent slots 1 and 2:

$$PG_{1,2}(\pi_b^{-ssb}, \pi_b^{-bbs}, \pi_m, \Pi_o, w_{\dot{L}}, \mu) = \sum_{\dot{l} \in \dot{L}_{-1,2}^{N(\mu)}(\Pi_o)} w_{\dot{L}}(\dot{l}) PO_{1,2}(\pi_b^{-ssb}, \pi_b^{-bbs}, \pi_m, \dot{l}, \mu) =$$

$$= \frac{1}{1} PO_{1,2}(\pi_b^{-ssb}, \pi_b^{-bbs}, \pi_m, (*,*), \mu)$$

The following table shows us $PO_{1,2}$ for all the permutations of $\pi_b^{-ssb}, \pi_b^{-bbs}, \pi_m$.

AS 1		AS 2	AS 1		AS 2	AS 1		AS 2
π_b^{-ssb}	$\leq$	π_b^{-bbs}	π_b^{-ssb}	$\leq$	π_m	π_b^{-bbs}	$\leq$	π_b^{-ssb}
π_b^{-bbs}	$\leq$	π_m	π_m	$\leq$	π_b^{-bbs}	π_b^{-ssb}	$\leq$	π_m
π_b^{-ssb}	$\leq$	π_m	π_b^{-ssb}	$\leq$	π_b^{-bbs}	π_b^{-bbs}	$\leq$	π_m
AS 1		AS 2	AS 1		AS 2	AS 1		AS 2
π_b^{-bbs}	$\leq$	π_m	π_m	$\leq$	π_b^{-ssb}	π_m	$\leq$	π_b^{-bbs}
π_m	$\leq$	π_b^{-ssb}	π_b^{-ssb}	$\leq$	π_b^{-bbs}	π_b^{-bbs}	$\leq$	π_b^{-ssb}
π_b^{-bbs}	$\leq$	π_b^{-ssb}	π_m	$\leq$	π_b^{-bbs}	π_m	$\leq$	π_b^{-ssb}

It is not possible to find a PO, since for every permutation we have either the agent line-up (π_b^{-ssb}, π_m), $(\pi_b^{-bbs}, \pi_b^{-ssb})$ or (π_m, π_b^{-bbs}). In the three cases, the agent in agent slot 1 obtains a slightly better expected average reward than the agent in agent slot 2. So:

$$PG_{1,2}(\pi_b^{-ssb}, \pi_b^{-bbs}, \pi_m, \Pi_o, w_{\dot{L}}, \mu) = \frac{1}{1} 0 = 0$$

And finally, we calculate the PG value generalising for every pair of agent slots:

$$PG(\Pi_e, w_{\Pi_e}, \Pi_o, w_{\dot{L}}, \mu, w_S) = \eta_{\Pi^3} \sum_{\pi_1,\pi_2,\pi_3 \in \Pi_e | \pi_1 \neq \pi_2 \neq \pi_3} w_{\Pi_e}(\pi_1) w_{\Pi_e}(\pi_2) w_{\Pi_e}(\pi_3) \eta_{S_1^2} \sum_{i=1}^{N(\mu)} w_S(i,\mu) \times$$
$$\times \left(\sum_{j=1}^{i-1} w_S(j,\mu) PG_{i,j}(\pi_1,\pi_2,\pi_3,\Pi_o,w_{\dot{L}},\mu) + \sum_{j=i+1}^{N(\mu)} w_S(j,\mu) PG_{i,j}(\pi_1,\pi_2,\pi_3,\Pi_o,w_{\dot{L}},\mu) \right) =$$
$$= 6\frac{9}{2}\frac{1}{3}\frac{1}{3}\frac{1}{3}\eta_{S_1^2} \sum_{i=1}^{N(\mu)} w_S(i,\mu) \left(\sum_{j=1}^{i-1} w_S(j,\mu)0 + \sum_{j=i+1}^{N(\mu)} w_S(j,\mu)0 \right) = 0$$

Note that we avoided to calculate all the permutations of π_1, π_2, π_3 for $PG_{i,j}(\pi_1, \pi_2, \pi_3, \Pi_o, w_{\dot{L}}, \mu)$, since they provide the same result, by calculating only one permutation and multiplying the result by 6 (the number of permutations).

Since 0 is the lowest possible value for the partial grading property, therefore prisoner's dilemma has $General_{min} = 0$ for this property. □

Proposition 41. *$General_{max}$ for the partial grading (PG) property is equal to 1 for the prisoner's dilemma environment.*

Proof. To find $General_{max}$ (equation 8.2), we need to find a pair $\langle \Pi_e, \Pi_o \rangle$ that maximises the property value as much as possible. We can have this situation by selecting $\Pi_e = \{\pi_{b1}, \pi_{b2}, \pi_{b3}\}$ and $\Pi_o = \emptyset$ (a π_b agent always performs Betray).

Following definition 31 we obtain the PG value for this $\langle \Pi_e, \Pi_o \rangle$. Since the multi-agent environment is team symmetric, we just need to calculate this property value for one pair of agent slots and generalise it to all pairs of agent slots. Also, since $|\Pi_e| = 3$ we just need to calculate this property value for one trio of evaluated agents. Following definition 29 (for PG) we calculate the PG value for the evaluated agents $\pi_{b1}, \pi_{b2}, \pi_{b3} \in \Pi_e$ and the pair of agent slots 1 and 2:

$$PG_{1,2}(\pi_{b1}, \pi_{b2}, \pi_{b3}, \Pi_o, w_{\dot{L}}, \mu) = \sum_{\dot{l} \in \dot{L}_{-1,2}^{N(\mu)}(\Pi_o)} w_{\dot{L}}(\dot{l}) PO_{1,2}(\pi_{b1}, \pi_{b2}, \pi_{b3}, \dot{l}, \mu) =$$
$$= \frac{1}{1} PO_{1,2}(\pi_{b1}, \pi_{b2}, \pi_{b3}, (*,*), \mu)$$

The following table shows us $PO_{1,2}$ for all the permutations of $\pi_{b1}, \pi_{b2}, \pi_{b3}$.

AS 1		AS 2	AS 1		AS 2	AS 1		AS 2
π_{b1}	$\leq$	π_{b2}	π_{b1}	$\leq$	π_{b3}	π_{b2}	$\leq$	π_{b1}
π_{b2}	$\leq$	π_{b3}	π_{b3}	$\leq$	π_{b2}	π_{b1}	$\leq$	π_{b3}
π_{b1}	$\leq$	π_{b3}	π_{b1}	$\leq$	π_{b2}	π_{b2}	$\leq$	π_{b3}
AS 1		AS 2	AS 1		AS 2	AS 1		AS 2
π_{b2}	$\leq$	π_{b3}	π_{b3}	$\leq$	π_{b1}	π_{b3}	$\leq$	π_{b2}
π_{b3}	$\leq$	π_{b1}	π_{b1}	$\leq$	π_{b2}	π_{b2}	$\leq$	π_{b1}
π_{b2}	$\leq$	π_{b1}	π_{b3}	$\leq$	π_{b2}	π_{b3}	$\leq$	π_{b1}

It is possible to find a PO for every permutation, since both agents obtain the same expected average reward ($-\frac{1}{3}$). So:

$$PG_{1,2}(\pi_{b1}, \pi_{b2}, \pi_{b3}, \Pi_o, w_{\dot{L}}, \mu) = \frac{1}{1}1 = 1$$

And finally, we calculate the PG value generalising for every pair of agent slots:

$$PG(\Pi_e, w_{\Pi_e}, \Pi_o, w_{\dot{L}}, \mu, w_S) = \eta_{\Pi^3} \sum_{\pi_1,\pi_2,\pi_3 \in \Pi_e | \pi_1 \neq \pi_2 \neq \pi_3} w_{\Pi_e}(\pi_1) w_{\Pi_e}(\pi_2) w_{\Pi_e}(\pi_3) \eta_{S_1^2} \sum_{i=1}^{N(\mu)} w_S(i,\mu) \times$$

$$\times \left(\sum_{j=1}^{i-1} w_S(j,\mu) PG_{i,j}(\pi_1, \pi_2, \pi_3, \Pi_o, w_{\dot{L}}, \mu) + \sum_{j=i+1}^{N(\mu)} w_S(j,\mu) PG_{i,j}(\pi_1, \pi_2, \pi_3, \Pi_o, w_{\dot{L}}, \mu) \right) =$$

$$= 6 \frac{9}{2} \frac{1}{3} \frac{1}{3} \frac{1}{3} \eta_{S_1^2} \sum_{i=1}^{N(\mu)} w_S(i,\mu) \left(\sum_{j=1}^{i-1} w_S(j,\mu) 1 + \sum_{j=i+1}^{N(\mu)} w_S(j,\mu) 1 \right) = 1$$

Note that we avoided to calculate all the permutations of π_1, π_2, π_3 for $PG_{i,j}(\pi_1, \pi_2, \pi_3, \Pi_o, w_{\dot{L}}, \mu)$, since they provide the same result, by calculating only one permutation and multiplying the result by 6 (the number of permutations).

Since 1 is the highest possible value for the partial grading property, therefore prisoner's dilemma has $General_{max} = 1$ for this property. $\square$

Proposition 42. *$Left_{max}$ for the partial grading (PG) property is equal to 1 for the prisoner's dilemma environment.*

Proof. To find $Left_{max}$ (equation 8.4), we need to find a Π_e that maximises the property value as much as possible while Π_o minimises it. Using $\Pi_e = \{\pi_{b1}, \pi_{b2}, \pi_{b3}\}$ (a π_b agent always performs Betray) we find this situation no matter which Π_o we use.

Following definition 31 we obtain the PG value for this $\langle \Pi_e, \Pi_o \rangle$ (where Π_o is instantiated with any permitted value). Since the multi-agent environment is team symmetric, we just need to calculate this property value for one pair of agent slots and generalise it to all pairs of agent slots. Also, since $|\Pi_e| = 3$ we just need to calculate this property value for one trio of evaluated agents. Following definition 29 (for PG) we calculate the PG value for the evaluated agents $\pi_{b1}, \pi_{b2}, \pi_{b3} \in \Pi_e$ and each pair of agent slots. We start with agent slots 1 and 2:

$$PG_{1,2}(\pi_{b1}, \pi_{b2}, \pi_{b3}, \Pi_o, w_{\dot{L}}, \mu) = \sum_{\dot{l} \in \dot{L}^{N(\mu)}_{-1,2}(\Pi_o)} w_{\dot{L}}(\dot{l}) PO_{1,2}(\pi_{b1}, \pi_{b2}, \pi_{b3}, \dot{l}, \mu) =$$

$$= \frac{1}{1} PO_{1,2}(\pi_{b1}, \pi_{b2}, \pi_{b3}, (*,*), \mu)$$

Note that the choice of Π_o does not affect the result of $PG_{1,2}$.

The following table shows us $PO_{1,2}$ for all the permutations of $\pi_{b1}, \pi_{b2}, \pi_{b3}$.

AS 1		AS 2	AS 1		AS 2	AS 1		AS 2
π_{b1}	$\leq$	π_{b2}	π_{b1}	$\leq$	π_{b3}	π_{b2}	$\leq$	π_{b1}
π_{b2}	$\leq$	π_{b3}	π_{b3}	$\leq$	π_{b2}	π_{b1}	$\leq$	π_{b3}
π_{b1}	$\leq$	π_{b3}	π_{b1}	$\leq$	π_{b2}	π_{b2}	$\leq$	π_{b3}
AS 1		AS 2	AS 1		AS 2	AS 1		AS 2
π_{b2}	$\leq$	π_{b3}	π_{b3}	$\leq$	π_{b1}	π_{b3}	$\leq$	π_{b2}
π_{b3}	$\leq$	π_{b1}	π_{b1}	$\leq$	π_{b2}	π_{b2}	$\leq$	π_{b1}
π_{b2}	$\leq$	π_{b1}	π_{b3}	$\leq$	π_{b2}	π_{b3}	$\leq$	π_{b1}

It is possible to find a PO for every permutation, since both agents obtain the same expected average reward ($-\frac{1}{3}$). So:

$$PG_{1,2}(\pi_{b1}, \pi_{b2}, \pi_{b3}, \Pi_o, w_{\dot{L}}, \mu) = \frac{1}{1}1 = 1$$

And finally, we calculate the PG value generalising for every pair of agent slots:

$$PG(\Pi_e, w_{\Pi_e}, \Pi_o, w_{\dot{L}}, \mu, w_S) = \eta_{\Pi^3} \sum_{\pi_1,\pi_2,\pi_3 \in \Pi_e | \pi_1 \neq \pi_2 \neq \pi_3} w_{\Pi_e}(\pi_1) w_{\Pi_e}(\pi_2) w_{\Pi_e}(\pi_3) \eta_{S_1^2} \sum_{i=1}^{N(\mu)} w_S(i,\mu) \times$$
$$\times \left(\sum_{j=1}^{i-1} w_S(j,\mu) PG_{i,j}(\pi_1, \pi_2, \pi_3, \Pi_o, w_{\dot{L}}, \mu) + \sum_{j=i+1}^{N(\mu)} w_S(j,\mu) PG_{i,j}(\pi_1, \pi_2, \pi_3, \Pi_o, w_{\dot{L}}, \mu) \right) =$$
$$= 6\frac{9}{2}\frac{1}{3}\frac{1}{3}\frac{1}{3}\eta_{S_1^2} \sum_{i=1}^{N(\mu)} w_S(i,\mu) \left(\sum_{j=1}^{i-1} w_S(j,\mu)1 + \sum_{j=i+1}^{N(\mu)} w_S(j,\mu)1 \right) = 1$$

So, for every Π_o we obtain the same result:

$$\forall \Pi_o : PG(\Pi_e, w_{\Pi_e}, \Pi_o, w_{\dot{L}}, \mu, w_S) = 1$$

Therefore, prisoner's dilemma has $Left_{max} = 1$ for this property. □

Proposition 43. *$Right_{min}$ for the partial grading (PG) property is equal to 0 for the prisoner's dilemma environment.*

Proof. To find $Right_{min}$ (equation 8.5), we need to find a Π_e that minimises the property value as much as possible while Π_o maximises it. Using $\Pi_e = \{\pi_b^{-ssb}, \pi_b^{-bbs}, \pi_m\}$ and $\Pi_o = \emptyset$ (a π_m agent first acts randomly and from time step 2 always mimics the other agent's last action, a π_b^{-ssb} agent always performs Betray except for the last three actions where it performs Silent twice and finalises performing Betray, and a π_b^{-bbs} agent always performs Betray except for the last three actions where it performs Betray twice and finalises performing Silent) we find this situation no matter which Π_o we use.

Following definition 31 we obtain the PG value for this $\langle \Pi_e, \Pi_o \rangle$ (where Π_o is instantiated with any permitted value). Since the multi-agent environment is team symmetric, we just need to calculate this property value for one pair of agent slots and generalise it to all pairs of agent slots. Also, since $|\Pi_e| = 3$ we just need to calculate this property value for one trio of evaluated agents. Following definition 29 (for PG) we calculate the PG value for the evaluated agents $\pi_b^{-ssb}, \pi_b^{-bbs}, \pi_m \in \Pi_e$ and the pair of agent slots 1 and 2:

$$PG_{1,2}(\pi_b^{-ssb}, \pi_b^{-bbs}, \pi_m, \Pi_o, w_{\dot{L}}, \mu) = \sum_{\dot{l} \in \dot{L}_{-1,2}^{N(\mu)}(\Pi_o)} w_{\dot{L}}(\dot{l}) PO_{1,2}(\pi_b^{-ssb}, \pi_b^{-bbs}, \pi_m, \dot{l}, \mu) =$$
$$= \frac{1}{1} PO_{1,2}(\pi_b^{-ssb}, \pi_b^{-bbs}, \pi_m, (*,*), \mu)$$

Note that the choice of Π_o does not affect the result of $PG_{1,2}$.

The following table shows us $PO_{1,2}$ for all the permutations of $\pi_b^{-ssb}, \pi_b^{-bbs}, \pi_m$.

AS 1		AS 2	AS 1		AS 2	AS 1		AS 2
π_b^{-ssb}	$\leq$	π_b^{-bbs}	π_b^{-ssb}	$\leq$	π_m	π_b^{-bbs}	$\leq$	π_b^{-ssb}
π_b^{-bbs}	$\leq$	π_m	π_m	$\leq$	π_b^{-bbs}	π_b^{-ssb}	$\leq$	π_m
π_b^{-ssb}	$\leq$	π_m	π_b^{-ssb}	$\leq$	π_b^{-bbs}	π_b^{-bbs}	$\leq$	π_m
AS 1		AS 2	AS 1		AS 2	AS 1		AS 2
π_b^{-bbs}	$\leq$	π_m	π_m	$\leq$	π_b^{-ssb}	π_m	$\leq$	π_b^{-bbs}
π_m	$\leq$	π_b^{-ssb}	π_b^{-ssb}	$\leq$	π_b^{-bbs}	π_b^{-bbs}	$\leq$	π_b^{-ssb}
π_b^{-bbs}	$\leq$	π_b^{-ssb}	π_m	$\leq$	π_b^{-bbs}	π_m	$\leq$	π_b^{-ssb}

It is not possible to find a PO, since for every permutation we have either the agent line-up (π_b^{-ssb}, π_m), $(\pi_b^{-bbs}, \pi_b^{-ssb})$ or (π_m, π_b^{-bbs}). In the three cases, the agent in agent slot 1 obtains a slightly better expected average reward than the agent in agent slot 2. So:

$$PG_{1,2}(\pi_b^{-ssb}, \pi_b^{-bbs}, \pi_m, \Pi_o, w_{\dot{L}}, \mu) = \frac{1}{1}0 = 0$$

And finally, we calculate the PG value generalising for every pair of agent slots:

$$PG(\Pi_e, w_{\Pi_e}, \Pi_o, w_{\dot{L}}, \mu, w_S) = \eta_{\Pi^3} \sum_{\pi_1,\pi_2,\pi_3 \in \Pi_e | \pi_1 \neq \pi_2 \neq \pi_3} w_{\Pi_e}(\pi_1) w_{\Pi_e}(\pi_2) w_{\Pi_e}(\pi_3) \eta_{S_1^2} \sum_{i=1}^{N(\mu)} w_S(i,\mu) \times$$
$$\times \left(\sum_{j=1}^{i-1} w_S(j,\mu) PG_{i,j}(\pi_1,\pi_2,\pi_3,\Pi_o,w_{\dot{L}},\mu) + \sum_{j=i+1}^{N(\mu)} w_S(j,\mu) PG_{i,j}(\pi_1,\pi_2,\pi_3,\Pi_o,w_{\dot{L}},\mu) \right) =$$
$$= 6\frac{9}{2}\frac{1}{3}\frac{1}{3}\frac{1}{3}\eta_{S_1^2} \sum_{i=1}^{N(\mu)} w_S(i,\mu) \left(\sum_{j=1}^{i-1} w_S(j,\mu)0 + \sum_{j=i+1}^{N(\mu)} w_S(j,\mu)0 \right) = 0$$

Note that we avoided to calculate all the permutations of π_1, π_2, π_3 for $PG_{i,j}(\pi_1, \pi_2, \pi_3, \Pi_o, w_{\dot{L}}, \mu)$, since they provide the same result, by calculating only one permutation and multiplying the result by 6 (the number of permutations).

So, for every Π_o we obtain the same result:

$$\forall \Pi_o : PG(\Pi_e, w_{\Pi_e}, \Pi_o, w_{\dot{L}}, \mu, w_S) = 0$$

Therefore, prisoner's dilemma has $Right_{min} = 0$ for this property. $\square$

B.6 Slot Result Dependency

Next we see the slot result dependency (SRD) property. As given in section 7.3.2, we want to know how much competitive or cooperative the multi-agent environment is. We use an average of rewards as the utility function to calculate an agent's result.

Proposition 44. *$General_{min}$ for the slot result dependency (SRD) property is equal to -1 for the prisoner's dilemma environment.*

Proof. To find $General_{min}$ (equation 8.1), we need to find a pair $\langle \Pi_e, \Pi_o \rangle$ that minimises the property value as much as possible. We can have this situation by selecting $\Pi_e = \{\pi_b\}$ and $\Pi_o = \{\pi_s\}$ (a π_s agent always performs Silent and a π_b agent always performs Betray).

Following definition 21 we obtain the SRD value for this $\langle \Pi_e, \Pi_o \rangle$. Since the multi-agent environment is team symmetric, we just need to calculate this property value for one pair of agent slots and generalise it to all pairs of agent slots. Also, since $|\Pi_e| = 1$ we just need to calculate this property value for one evaluated agent. Following definition 20 we calculate the SRD value for the evaluated agent $\pi_b \in \Pi_e$ and the pair of agent slots 1 and 2:

$$SRD_{1,2}(\pi_b, \Pi_o, w_{\dot{L}}, \mu) = corr_{\dot{l} \in \dot{L}_{-1}^{N(\mu)}(\Pi_o)}[w_{\dot{L}}(\dot{l})](R_1(\mu[\dot{l} \xleftarrow{1} \pi_b]), R_2(\mu[\dot{l} \xleftarrow{1} \pi_b])) =$$
$$= corr(R_1(\mu[\pi_b, \pi_s]), R_2(\mu[\pi_b, \pi_s]))$$

The agent in agent slot 1 (π_b) obtains the expected average reward of 1, while the agent in agent slot 2 (π_s) obtains the expected average reward of -1. Since we use a correlation function between expected average rewards, and the agents in agent slots 1 and 2 obtain opposite expected average rewards, then the correlation function obtains the value of -1. So:

$$SRD_{1,2}(\pi_b, \Pi_o, w_{\dot{L}}, \mu) = -1$$

And finally, we calculate the SRD value generalising for every pair of agent slots:

$$SRD(\Pi_e, w_{\Pi_e}, \Pi_o, w_{\dot{L}}, \mu, w_S) = \sum_{\pi \in \Pi_e} w_{\Pi_e}(\pi) \eta_{S_1^2} \sum_{i=1}^{N(\mu)} w_S(i, \mu) \times$$
$$\times \left(\sum_{j=1}^{i-1} w_S(j, \mu) SRD_{i,j}(\pi, \Pi_o, w_{\dot{L}}, \mu) + \sum_{j=i+1}^{N(\mu)} w_S(j, \mu) SRD_{i,j}(\pi, \Pi_o, w_{\dot{L}}, \mu) \right) =$$
$$= \frac{1}{1} \eta_{S_1^2} \sum_{i=1}^{N(\mu)} w_S(i, \mu) \left[\sum_{j=1}^{i-1} w_S(j, \mu)\,(-1) + \sum_{j=i+1}^{N(\mu)} w_S(j, \mu)\,(-1) \right] = -1$$

Since -1 is the lowest possible value for the slot result dependency property, therefore prisoner's dilemma has $General_{min} = -1$ for this property. □

Proposition 45. *$General_{max}$ for the slot result dependency (SRD) property is equal to 1 for the prisoner's dilemma environment.*

Proof. To find $General_{max}$ (equation 8.2), we need to find a pair $\langle \Pi_e, \Pi_o \rangle$ that maximises the property value as much as possible. We can have this situation by selecting $\Pi_e = \{\pi_b\}$ and $\Pi_o = \{\pi_b\}$ (a π_b agent always performs Betray).

Following definition 21 we obtain the SRD value for this $\langle \Pi_e, \Pi_o \rangle$. Since the multi-agent environment is team symmetric, we just need to calculate this property value for one pair of agent slots and generalise it to all pairs of agent slots. Also, since $|\Pi_e| = 1$ we just need to calculate this property value for one evaluated agent. Following definition 20 we calculate the SRD value for the evaluated agent $\pi_b \in \Pi_e$ and the pair of agent slots 1 and 2:

$$SRD_{1,2}(\pi_b, \Pi_o, w_{\dot{L}}, \mu) = corr_{\dot{l} \in \dot{L}_{-1}^{N(\mu)}(\Pi_o)}[w_{\dot{L}}(\dot{l})](R_1(\mu[\dot{l} \xleftarrow{1} \pi_b]), R_2(\mu[\dot{l} \xleftarrow{1} \pi_b])) =$$
$$= corr(R_1(\mu[\pi_b, \pi_b]), R_2(\mu[\pi_b, \pi_b]))$$

The agents in agent slot 1 (π_b) and agent slot 2 (π_b) obtain the same expected average reward ($-\frac{1}{3}$). Since we use a correlation function between the expected average rewards, and the agents in agent slots 1 and 2 obtain the same expected average reward, then the correlation function obtains the value of 1. So:

$$SRD_{1,2}(\pi_b, \Pi_o, w_{\dot{L}}, \mu) = 1$$

And finally, we calculate the SRD value generalising for every pair of agent slots:

$$SRD(\Pi_e, w_{\Pi_e}, \Pi_o, w_{\dot{L}}, \mu, w_S) = \sum_{\pi \in \Pi_e} w_{\Pi_e}(\pi) \eta_{S_1^2} \sum_{i=1}^{N(\mu)} w_S(i,\mu) \times$$
$$\times \left(\sum_{j=1}^{i-1} w_S(j,\mu) SRD_{i,j}(\pi, \Pi_o, w_{\dot{L}}, \mu) + \sum_{j=i+1}^{N(\mu)} w_S(j,\mu) SRD_{i,j}(\pi, \Pi_o, w_{\dot{L}}, \mu) \right) =$$
$$= \frac{1}{1} \eta_{S_1^2} \sum_{i=1}^{N(\mu)} w_S(i,\mu) \left(\sum_{j=1}^{i-1} w_S(j,\mu) 1 + \sum_{j=i+1}^{N(\mu)} w_S(j,\mu) 1 \right) = 1$$

Since 1 is the highest possible value for the slot result dependency property, therefore prisoner's dilemma has $General_{max} = 1$ for this property. □

B.7 Competitive Anticipation

Finally, we follow with the competitive anticipation (AComp) property. As given in section 7.4.1, we want to know how much benefit the evaluated agents obtain when they anticipate competing agents. We use an average of rewards as the utility function to calculate an agent's result.

Proposition 46. *$General_{min}$ for the competitive anticipation (AComp) property is equal to $-\frac{2}{3}$ for the prisoner's dilemma environment.*

Proof. To find $General_{min}$ (equation 8.1), we need to find a pair $\langle \Pi_e, \Pi_o \rangle$ that minimises the property value as much as possible. We can have this situation by selecting $\Pi_e = \{\pi_{s/b}\}$ and $\Pi_o = \{\pi_b\}$ (a π_b agent always performs Betray and a $\pi_{s/b}$ agent performs Silent until the other agent also performs Silent, then it starts to perform Betray).

Following definition 23 we obtain the AComp value for this $\langle \Pi_e, \Pi_o \rangle$. Since the multi-agent environment is team symmetric, we just need to calculate this property value for one pair of agent slots in different teams and generalise it to all pairs of agent slots in different teams. Also, since $|\Pi_e| = 1$ we just need to calculate this property value for one evaluated agent. Following definition 22 we calculate the Ant value for the evaluated agent $\pi_{s/b} \in \Pi_e$ and the pair of agent slots 1 and 2:

$$Ant_{1,2}(\pi_{s/b}, \Pi_o, w_{\dot{L}}, \mu) = \sum_{\dot{l} \in \dot{L}_{-1,2}^{N(\mu)}(\Pi_o)} \sum_{\pi \in \Pi_o} w_{\dot{L}}(\dot{l} \overset{2}{\leftarrow} \pi) \times$$
$$\times \frac{1}{2} \left(R_1(\mu[\dot{l} \overset{1,2}{\leftarrow} \pi_{s/b}, \pi]) - R_1(\mu[\dot{l} \overset{1,2}{\leftarrow} \pi_{s/b}, \pi_r]) \right) =$$
$$= \frac{1}{1} \frac{1}{2} \left(R_1(\mu[\pi_{s/b}, \pi_b]) - R_1(\mu[\pi_{s/b}, \pi_r]) \right)$$

In agent line-up $(\pi_{s/b}, \pi_b)$, the agent in agent slot 1 $(\pi_{s/b})$ obtains the expected average reward of -1, while in agent line-up $(\pi_{s/b}, \pi_r)$, the agent in agent slot 1 $(\pi_{s/b})$ starts performing Silent and then it continues performing Betray once π_r performs Silent, and the agent in agent slot 2 (π_r) always acts randomly, obtaining the agent in agent slot 1 $(\pi_{s/b})$ the expected average reward of $\frac{1}{3}$ (in the limit). So:

$$Ant_{1,2}(\pi_{s/b}, \Pi_o, w_{\dot{L}}, \mu) = \frac{1}{1}\frac{1}{2}\left[(-1) - \frac{1}{3}\right] = -\frac{2}{3}$$

And finally, we calculate the AComp value generalising for every pair of agent slots in different teams:

$$\begin{aligned} ACfavorite &= \end{aligned}$$

$$\begin{aligned} AComp(\Pi_e, w_{\Pi_e}, \Pi_o, w_{\dot{L}}, \mu, w_S) &= \sum_{\pi \in \Pi_e} w_{\Pi_e}(\pi) \eta_{S_2^2} \sum_{t_1, t_2 \in \tau | t_1 \neq t_2} \sum_{i \in t_1} w_S(i, \mu) \sum_{j \in t_2} w_S(j, \mu) \times \\ &\times Ant_{i,j}(\pi, \Pi_o, w_{\dot{L}}, \mu) = \\ &= \frac{1}{1} \eta_{S_2^2} \sum_{t_1, t_2 \in \tau | t_1 \neq t_2} \sum_{i \in t_1} w_S(i, \mu) \sum_{j \in t_2} w_S(j, \mu) \left(-\frac{2}{3}\right) = -\frac{2}{3} \end{aligned}$$

Since $-\frac{2}{3}$ is the lowest possible value that we can obtain for the competitive anticipation property, therefore prisoner's dilemma has $General_{min} = -\frac{2}{3}$ for this property. □

Proposition 47. *$General_{max}$ for the competitive anticipation (AComp) property is equal to $\frac{2}{3}$ for the prisoner's dilemma environment.*

Proof. To find $General_{max}$ (equation 8.2), we need to find a pair $\langle \Pi_e, \Pi_o \rangle$ that maximises the property value as much as possible. We can have this situation by selecting $\Pi_e = \{\pi_{b/s}\}$ and $\Pi_o = \{\pi_s\}$ (a π_s agent always performs Silent and a $\pi_{b/s}$ agent performs Betray until the other agent also performs Betray, then it starts to perform Silent).

Following definition 23 we obtain the AComp value for this $\langle \Pi_e, \Pi_o \rangle$. Since the multi-agent environment is team symmetric, we just need to calculate this property value for one pair of agent slots in different teams and generalise it to all pairs of agent slots in different teams. Also, since $|\Pi_e| = 1$ we just need to calculate this property value for one evaluated agent. Following definition 22 we calculate the Ant value for the evaluated agent $\pi_{b/s} \in \Pi_e$ and the pair of agent slots 1 and 2:

$$\begin{aligned} Ant_{1,2}(\pi_{b/s}, \Pi_o, w_{\dot{L}}, \mu) &= \sum_{\dot{l} \in \dot{L}_{-1,2}^{N(\mu)}(\Pi_o)} \sum_{\pi \in \Pi_o} w_{\dot{L}}(\dot{l} \overset{2}{\leftarrow} \pi) \times \\ &\times \frac{1}{2} \left(R_1(\mu[\dot{l} \overset{1,2}{\leftarrow} \pi_{b/s}, \pi]) - R_1(\mu[\dot{l} \overset{1,2}{\leftarrow} \pi_{b/s}, \pi_r]) \right) = \\ &= \frac{1}{1}\frac{1}{2} \left(R_1(\mu[\pi_{b/s}, \pi_s]) - R_1(\mu[\pi_{b/s}, \pi_r]) \right) \end{aligned}$$

In agent line-up $(\pi_{b/s}, \pi_s)$, the agent in agent slot 1 $(\pi_{b/s})$ obtains the expected average reward of 1, while in agent line-up $(\pi_{b/s}, \pi_r)$, the agent in agent slot 1 $(\pi_{b/s})$ starts performing Betray and then it continues performing Silent once π_r performs Betray, and the agent in agent slot 2 (π_r) always acts randomly, obtaining the agent in agent slot 1 $(\pi_{b/s})$ the expected average reward of $-\frac{1}{3}$ (in the limit). So:

$$Ant_{1,2}(\pi_{b/s}, \Pi_o, w_{\dot{L}}, \mu) = \frac{1}{1}\frac{1}{2}\left[1 - \left(-\frac{1}{3}\right)\right] = \frac{2}{3}$$

And finally, we calculate the AComp value generalising for every pair of agent slots in different teams:

$$\begin{aligned} AComp(\Pi_e, w_{\Pi_e}, \Pi_o, w_{\dot{L}}, \mu, w_S) &= \sum_{\pi \in \Pi_e} w_{\Pi_e}(\pi)\eta_{S_2^2} \sum_{t_1,t_2 \in \tau | t_1 \neq t_2} \sum_{i \in t_1} w_S(i,\mu) \sum_{j \in t_2} w_S(j,\mu) \times \\ &\times Ant_{i,j}(\pi, \Pi_o, w_{\dot{L}}, \mu) = \\ &= \frac{1}{1}\eta_{S_2^2} \sum_{t_1,t_2 \in \tau | t_1 \neq t_2} \sum_{i \in t_1} w_S(i,\mu) \sum_{j \in t_2} w_S(j,\mu) \frac{2}{3} = \frac{2}{3} \end{aligned}$$

Since $\frac{2}{3}$ is the highest possible value that we can obtain for the competitive anticipation property, therefore prisoner's dilemma has $General_{max} = \frac{2}{3}$ for this property. □

Appendix C

Predator-Prey Properties

In this section we prove how we obtained the values for the properties for the predator-prey environment (section 2.1.3). We use uniform unit weights for w_{Π_e}, $w_{\dot{L}}$ and w_S. To calculate some of the values for the properties, we make use of lemma 3.

Lemma 3. When three perfectly coordinated predators are trying to chase the prey, it is always chased in 5 time steps or less no matter the behaviour of the prey.

Since there exists a lot of variants to chase the prey, we cannot show them all. Instead, here we show one of the largest sequences of actions to chase the prey in 5 time steps when the prey is trying to escape and the predators are well coordinated.

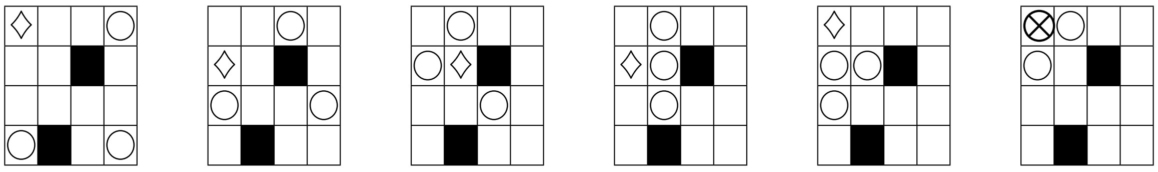


Figure C.1: One of the largest sequences of actions to chase the prey in 5 time steps with three well coordinated predators. $\otimes$ represents a chased prey.

Other behaviours of the prey will lead it closer to the boundaries, where the predators would chase it more easily.

C.1 Action Dependency

We start with the action dependency (AD) property. As given in section 7.2.1, we want to know if the evaluated agents behave differently depending on which agent line-up they interact with. We use $\Delta_{\mathcal{A}^+}(a, b) = 0$ if distributions a and b are equal and 1 otherwise.

Proposition 48. $General_{min}$ for the action dependency (AD) property is equal to 0 for the predator-prey environment.

Proof. To find $General_{min}$ (equation 8.1), we need to find a pair $\langle \Pi_e, \Pi_o \rangle$ that minimises the property value as much as possible. We can have this situation by selecting $\Pi_e = \{\pi_u\}$ and $\Pi_o = \{\pi_{d1}, \pi_{d2}\}$ (a π_d agent always performs Down and a π_u agent always performs Up).

Following definition 17 we obtain the AD value for this $\langle \Pi_e, \Pi_o \rangle$. Since the multi-agent environment is not team symmetric, we need to calculate this property value for every agent slot. Also, since $|\Pi_e| = 1$ we just need to calculate this property value for one evaluated agent. Following definition 16 we calculate the AD value for the evaluated agent $\pi_u \in \Pi_e$ and each agent slot. We start with agent slot 1:

$$\begin{aligned} AD_1(\pi_u, \Pi_o, w_{\dot{L}}, \mu) &= \eta_{\dot{L}^2} \sum_{\dot{u},\dot{v} \in \dot{L}_{-1}^{N(\mu)}(\Pi_o) | \dot{u} \neq \dot{v}} w_{\dot{L}}(\dot{u}) w_{\dot{L}}(\dot{v}) \Delta_{\mathcal{A}^+}(\breve{A}_1(\mu[\dot{u} \xleftarrow{1} \pi_u]), \breve{A}_1(\mu[\dot{v} \xleftarrow{1} \pi_u])) = \\ &= 2\frac{8}{7}\frac{1}{8}\frac{1}{8}\{\Delta_{\mathcal{A}^+}(\breve{A}_1(\mu[\pi_u, \pi_{d1}, \pi_{d1}, \pi_{d1}]), \breve{A}_1(\mu[\pi_u, \pi_{d1}, \pi_{d1}, \pi_{d2}]))+ \\ &+ \Delta_{\mathcal{A}^+}(\breve{A}_1(\mu[\pi_u, \pi_{d1}, \pi_{d1}, \pi_{d1}]), \breve{A}_1(\mu[\pi_u, \pi_{d1}, \pi_{d2}, \pi_{d1}]))+ \\ &\vdots \\ &+ \Delta_{\mathcal{A}^+}(\breve{A}_1(\mu[\pi_u, \pi_{d2}, \pi_{d2}, \pi_{d1}]), \breve{A}_1(\mu[\pi_u, \pi_{d2}, \pi_{d2}, \pi_{d2}]))\} \end{aligned}$$

Note that we avoided to calculate both $\Delta_{\mathcal{A}^+}(a, b)$ and $\Delta_{\mathcal{A}^+}(b, a)$, since they provide the same result, by calculating only $\Delta_{\mathcal{A}^+}(a, b)$ and multiplying the result by 2.

We have 28 possible pairs of agent line-ups, where the agent in both agent slots 1 (π_u) performs the same sequence of actions (always Up) independently of the agent line-up. So:

$$AD_1(\pi_u, \Pi_o, w_{\dot{L}}, \mu) = 2\frac{8}{7}\frac{1}{8}\frac{1}{8}\{28 \times 0\} = 0$$

For agent slot 2, the agent in both agent slots 2 (π_u) also performs the same sequence of actions (always Up) independently of the agent line-up. So:

$$\begin{aligned} AD_2(\pi_u, \Pi_o, w_{\dot{L}}, \mu) &= \eta_{\dot{L}^2} \sum_{\dot{u},\dot{v} \in \dot{L}_{-2}^{N(\mu)}(\Pi_o) | \dot{u} \neq \dot{v}} w_{\dot{L}}(\dot{u}) w_{\dot{L}}(\dot{v}) \Delta_{\mathcal{A}^+}(\breve{A}_2(\mu[\dot{u} \xleftarrow{2} \pi_u]), \breve{A}_2(\mu[\dot{v} \xleftarrow{2} \pi_u])) = \\ &= 2\frac{8}{7}\frac{1}{8}\frac{1}{8}\{\Delta_{\mathcal{A}^+}(\breve{A}_2(\mu[\pi_{d1}, \pi_u, \pi_{d1}, \pi_{d1}]), \breve{A}_2(\mu[\pi_{d1}, \pi_u, \pi_{d1}, \pi_{d2}]))+ \\ &+ \Delta_{\mathcal{A}^+}(\breve{A}_2(\mu[\pi_{d1}, \pi_u, \pi_{d1}, \pi_{d1}]), \breve{A}_2(\mu[\pi_{d1}, \pi_u, \pi_{d2}, \pi_{d1}]))+ \\ &\vdots \\ &+ \Delta_{\mathcal{A}^+}(\breve{A}_2(\mu[\pi_{d2}, \pi_u, \pi_{d2}, \pi_{d1}]), \breve{A}_2(\mu[\pi_{d2}, \pi_u, \pi_{d2}, \pi_{d2}]))\} = \\ &= 2\frac{8}{7}\frac{1}{8}\frac{1}{8}\{28 \times 0\} = 0 \end{aligned}$$

For agent slot 3, the agent in both agent slots 3 (π_u) also performs the same sequence of actions (always Up) independently of the agent line-up. So:

$$\begin{aligned} AD_3(\pi_u,\Pi_o,w_{\dot{L}},\mu) &= \eta_{\dot{L}^2} \sum_{\dot{u},\dot{v}\in \dot{L}_{-3}^{N(\mu)}(\Pi_o)|\dot{u}\neq\dot{v}} w_{\dot{L}}(\dot{u})w_{\dot{L}}(\dot{v})\Delta_{\mathcal{A}^+}(\breve{A}_3(\mu[\dot{u}\overset{3}{\leftarrow}\pi_u]),\breve{A}_3(\mu[\dot{v}\overset{3}{\leftarrow}\pi_u])) = \\ &= 2\frac{8}{7}\frac{1}{8}\frac{1}{8}\{\Delta_{\mathcal{A}^+}(\breve{A}_3(\mu[\pi_{d1},\pi_{d1},\pi_u,\pi_{d1}]),\breve{A}_3(\mu[\pi_{d1},\pi_{d1},\pi_u,\pi_{d2}]))+ \\ &+ \Delta_{\mathcal{A}^+}(\breve{A}_3(\mu[\pi_{d1},\pi_{d1},\pi_u,\pi_{d1}]),\breve{A}_3(\mu[\pi_{d1},\pi_{d2},\pi_u,\pi_{d1}]))+ \\ &\vdots \\ &+ \Delta_{\mathcal{A}^+}(\breve{A}_3(\mu[\pi_{d2},\pi_{d2},\pi_u,\pi_{d1}]),\breve{A}_3(\mu[\pi_{d2},\pi_{d2},\pi_u,\pi_{d2}]))\} = \\ &= 2\frac{8}{7}\frac{1}{8}\frac{1}{8}\{28\times 0\} = 0 \end{aligned}$$

And for agent slot 4, the agent in both agent slots 4 (π_u) also performs the same sequence of actions (always Up) independently of the agent line-up. So:

$$\begin{aligned} AD_4(\pi_u,\Pi_o,w_{\dot{L}},\mu) &= \eta_{\dot{L}^2} \sum_{\dot{u},\dot{v}\in \dot{L}_{-4}^{N(\mu)}(\Pi_o)|\dot{u}\neq\dot{v}} w_{\dot{L}}(\dot{u})w_{\dot{L}}(\dot{v})\Delta_{\mathcal{A}^+}(\breve{A}_4(\mu[\dot{u}\overset{4}{\leftarrow}\pi_u]),\breve{A}_4(\mu[\dot{v}\overset{4}{\leftarrow}\pi_u])) = \\ &= 2\frac{8}{7}\frac{1}{8}\frac{1}{8}\{\Delta_{\mathcal{A}^+}(\breve{A}_4(\mu[\pi_{d1},\pi_{d1},\pi_{d1},\pi_u]),\breve{A}_4(\mu[\pi_{d1},\pi_{d1},\pi_{d2},\pi_u]))+ \\ &+ \Delta_{\mathcal{A}^+}(\breve{A}_4(\mu[\pi_{d1},\pi_{d1},\pi_{d1},\pi_u]),\breve{A}_4(\mu[\pi_{d1},\pi_{d2},\pi_{d1},\pi_u]))+ \\ &\vdots \\ &+ \Delta_{\mathcal{A}^+}(\breve{A}_4(\mu[\pi_{d2},\pi_{d2},\pi_{d1},\pi_u]),\breve{A}_4(\mu[\pi_{d2},\pi_{d2},\pi_{d2},\pi_u]))\} = \\ &= 2\frac{8}{7}\frac{1}{8}\frac{1}{8}\{28\times 0\} = 0 \end{aligned}$$

And finally, we calculate the AD value:

$$\begin{aligned} AD(\Pi_e,w_{\Pi_e},\Pi_o,w_{\dot{L}},\mu,w_S) &= \sum_{\pi\in\Pi_e} w_{\Pi_e}(\pi)\sum_{i=1}^{N(\mu)} w_S(i,\mu)AD_i(\pi,\Pi_o,w_{\dot{L}},\mu) = \\ &= \frac{1}{1}\frac{1}{4}\{AD_1(\pi_u,\Pi_o,w_{\dot{L}},\mu)+AD_2(\pi_u,\Pi_o,w_{\dot{L}},\mu)+ \\ &+ AD_3(\pi_u,\Pi_o,w_{\dot{L}},\mu)+AD_4(\pi_u,\Pi_o,w_{\dot{L}},\mu)\} = \\ &= \frac{1}{1}\frac{1}{4}\{0+0+0+0\} = 0 \end{aligned}$$

Since 0 is the lowest possible value for the action dependency property, therefore predator-prey has $General_{min} = 0$ for this property. □

Proposition 49. *$General_{max}$ for the action dependency (AD) property is equal to 1 for the predator-prey environment.*

Proof. To find $General_{max}$ (equation 8.2), we need to find a pair $\langle\Pi_e,\Pi_o\rangle$ that maximises the property value as much as possible. We can have this situation by selecting $\Pi_e = \{\pi_m\}$ and $\Pi_o = \{\pi_u,\pi_d\}$ (a π_m agent first acts randomly and from time step 2 always mimics sequentially the other agents' last action, a π_u agent always performs Up and a π_d agent always performs Down).

Following definition 17 we obtain the AD value for this $\langle \Pi_e, \Pi_o \rangle$. Since the multi-agent environment is not team symmetric, we need to calculate this property value for every agent slot. Also, since $|\Pi_e| = 1$ we just need to calculate this property value for one evaluated agent. Following definition 16 we calculate the AD value for the evaluated agent $\pi_m \in \Pi_e$ and each agent slot. We start with agent slot 1:

$$\begin{aligned}
AD_1(\pi_m, \Pi_o, w_{\dot{L}}, \mu) = \eta_{\dot{L}^2} & \sum_{\dot{u},\dot{v} \in \dot{L}_{-1}^{N(\mu)}(\Pi_o) | \dot{u} \neq \dot{v}} w_{\dot{L}}(\dot{u}) w_{\dot{L}}(\dot{v}) \Delta_{\mathcal{A}^+}(\breve{A}_1(\mu[\dot{u} \overset{1}{\leftarrow} \pi_m]), \breve{A}_1(\mu[\dot{v} \overset{1}{\leftarrow} \pi_m])) = \\
&= 2\frac{8}{7}\frac{1}{8}\frac{1}{8}\{\Delta_{\mathcal{A}^+}(\breve{A}_1(\mu[\pi_m, \pi_u, \pi_u, \pi_u]), \breve{A}_1(\mu[\pi_m, \pi_u, \pi_u, \pi_d]))+ \\
&+ \Delta_{\mathcal{A}^+}(\breve{A}_1(\mu[\pi_m, \pi_u, \pi_u, \pi_u]), \breve{A}_1(\mu[\pi_m, \pi_u, \pi_d, \pi_u]))+ \\
&\vdots \\
&+ \Delta_{\mathcal{A}^+}(\breve{A}_1(\mu[\pi_m, \pi_d, \pi_d, \pi_u]), \breve{A}_1(\mu[\pi_m, \pi_d, \pi_d, \pi_d]))\}
\end{aligned}$$

Note that we avoided to calculate both $\Delta_{\mathcal{A}^+}(a, b)$ and $\Delta_{\mathcal{A}^+}(b, a)$, since they provide the same result, by calculating only $\Delta_{\mathcal{A}^+}(a, b)$ and multiplying the result by 2.

We have 28 possible pairs of agent line-ups, where the agent in both agent slots 1 (π_m) performs different sequences of actions depending on the agent line-up. So:

$$AD_1(\pi_m, \Pi_o, w_{\dot{L}}, \mu) = 2\frac{8}{7}\frac{1}{8}\frac{1}{8}\{28 \times 1\} = 1$$

For slot 2, the agent in both agent slots 2 (π_m) also performs different sequences of actions depending on the agent line-up. So:

$$\begin{aligned}
AD_2(\pi_m, \Pi_o, w_{\dot{L}}, \mu) = \eta_{\dot{L}^2} & \sum_{\dot{u},\dot{v} \in \dot{L}_{-2}^{N(\mu)}(\Pi_o) | \dot{u} \neq \dot{v}} w_{\dot{L}}(\dot{u}) w_{\dot{L}}(\dot{v}) \Delta_{\mathcal{A}^+}(\breve{A}_2(\mu[\dot{u} \overset{2}{\leftarrow} \pi_m]), \breve{A}_2(\mu[\dot{v} \overset{2}{\leftarrow} \pi_m])) = \\
&= 2\frac{8}{7}\frac{1}{8}\frac{1}{8}\{\Delta_{\mathcal{A}^+}(\breve{A}_2(\mu[\pi_u, \pi_m, \pi_u, \pi_u]), \breve{A}_2(\mu[\pi_u, \pi_m, \pi_u, \pi_d]))+ \\
&+ \Delta_{\mathcal{A}^+}(\breve{A}_2(\mu[\pi_u, \pi_m, \pi_u, \pi_u]), \breve{A}_2(\mu[\pi_u, \pi_m, \pi_d, \pi_u]))+ \\
&\vdots \\
&+ \Delta_{\mathcal{A}^+}(\breve{A}_2(\mu[\pi_d, \pi_m, \pi_d, \pi_u]), \breve{A}_2(\mu[\pi_d, \pi_m, \pi_d, \pi_d]))\} = \\
&= 2\frac{8}{7}\frac{1}{8}\frac{1}{8}\{28 \times 1\} = 1
\end{aligned}$$

For slot 3, the agent in both agent slots 3 (π_m) also performs different sequences of actions depending on the agent line-up. So:

$$\begin{aligned}AD_3(\pi_m, \Pi_o, w_{\dot{L}}, \mu) &= \eta_{\dot{L}^2} \sum_{\dot{u},\dot{v} \in \dot{L}_{-3}^{N(\mu)}(\Pi_o) | \dot{u} \neq \dot{v}} w_{\dot{L}}(\dot{u}) w_{\dot{L}}(\dot{v}) \Delta_{\mathcal{A}^+}(\breve{A}_3(\mu[\dot{u} \overset{3}{\leftarrow} \pi_m]), \breve{A}_3(\mu[\dot{v} \overset{3}{\leftarrow} \pi_m])) = \\ &= 2\frac{8}{7}\frac{1}{8}\frac{1}{8}\{\Delta_{\mathcal{A}^+}(\breve{A}_3(\mu[\pi_u, \pi_u, \pi_m, \pi_u]), \breve{A}_3(\mu[\pi_u, \pi_u, \pi_m, \pi_d])) + \\ &+ \Delta_{\mathcal{A}^+}(\breve{A}_3(\mu[\pi_u, \pi_u, \pi_m, \pi_u]), \breve{A}_3(\mu[\pi_u, \pi_d, \pi_m, \pi_u])) + \\ &\vdots \\ &+ \Delta_{\mathcal{A}^+}(\breve{A}_3(\mu[\pi_d, \pi_d, \pi_m, \pi_u]), \breve{A}_3(\mu[\pi_d, \pi_d, \pi_m, \pi_d]))\} = \\ &= 2\frac{8}{7}\frac{1}{8}\frac{1}{8}\{28 \times 1\} = 1\end{aligned}$$

And for slot 4, the agent in both agent slots 4 (π_m) also performs different sequences of actions depending on the agent line-up. So:

$$\begin{aligned}AD_4(\pi_m, \Pi_o, w_{\dot{L}}, \mu) &= \eta_{\dot{L}^2} \sum_{\dot{u},\dot{v} \in \dot{L}_{-4}^{N(\mu)}(\Pi_o) | \dot{u} \neq \dot{v}} w_{\dot{L}}(\dot{u}) w_{\dot{L}}(\dot{v}) \Delta_{\mathcal{A}^+}(\breve{A}_4(\mu[\dot{u} \overset{4}{\leftarrow} \pi_m]), \breve{A}_4(\mu[\dot{v} \overset{4}{\leftarrow} \pi_m])) = \\ &= 2\frac{8}{7}\frac{1}{8}\frac{1}{8}\{\Delta_{\mathcal{A}^+}(\breve{A}_4(\mu[\pi_u, \pi_u, \pi_u, \pi_m]), \breve{A}_4(\mu[\pi_u, \pi_u, \pi_d, \pi_m])) + \\ &+ \Delta_{\mathcal{A}^+}(\breve{A}_4(\mu[\pi_u, \pi_u, \pi_u, \pi_m]), \breve{A}_4(\mu[\pi_u, \pi_d, \pi_u, \pi_m])) + \\ &\vdots \\ &+ \Delta_{\mathcal{A}^+}(\breve{A}_4(\mu[\pi_d, \pi_d, \pi_u, \pi_m]), \breve{A}_4(\mu[\pi_d, \pi_d, \pi_d, \pi_m]))\} = \\ &= 2\frac{8}{7}\frac{1}{8}\frac{1}{8}\{28 \times 1\} = 1\end{aligned}$$

And finally, we calculate the AD value:

$$\begin{aligned}AD(\Pi_e, w_{\Pi_e}, \Pi_o, w_{\dot{L}}, \mu, w_S) &= \sum_{\pi \in \Pi_e} w_{\Pi_e}(\pi) \sum_{i=1}^{N(\mu)} w_S(i, \mu) AD_i(\pi, \Pi_o, w_{\dot{L}}, \mu) = \\ &= \frac{1}{1}\frac{1}{4}\{AD_1(\pi_m, \Pi_o, w_{\dot{L}}, \mu) + AD_2(\pi_m, \Pi_o, w_{\dot{L}}, \mu) + \\ &+ AD_3(\pi_m, \Pi_o, w_{\dot{L}}, \mu) + AD_4(\pi_m, \Pi_o, w_{\dot{L}}, \mu)\} = \\ &= \frac{1}{1}\frac{1}{4}\{1 + 1 + 1 + 1\} = 1\end{aligned}$$

Since 1 is the highest possible value for the action dependency property, therefore predator-prey has $General_{max} = 1$ for this property. □

Proposition 50. *$Left_{max}$ for the action dependency (AD) property is equal to 0 for the predator-prey environment.*

Proof. To find $Left_{max}$ (equation 8.4), we need to find a Π_e that maximises the property value as much as possible while Π_o minimises it. Using $\Pi_o = \{\pi_{d1}, \pi_{d2}\}$ (a π_d agent always performs Down) we find this situation no matter which Π_e we use.

Following definition 17 we obtain the AD value for this $\langle \Pi_e, \Pi_o \rangle$ (where Π_e is instantiated with any permitted value). Since the multi-agent environment is not team symmetric, we need to calculate this property value for every agent slot. Also, we do not know which Π_e we have, so we use a figurative evaluated agent π from Π_e. Following definition 16 we calculate the AD value for this figurative evaluated agent $\pi \in \Pi_e$ and each agent slot. We start with agent slot 1:

$$
\begin{aligned}
AD_1(\pi, \Pi_o, w_{\dot{L}}, \mu) &= \eta_{\dot{L}^2} \sum_{\dot{u},\dot{v} \in \dot{L}_{-1}^{N(\mu)}(\Pi_o) | \dot{u} \neq \dot{v}} w_{\dot{L}}(\dot{u}) w_{\dot{L}}(\dot{v}) \Delta_{\mathcal{A}^+}(\breve{A}_1(\mu[\dot{u} \xleftarrow{1} \pi]), \breve{A}_1(\mu[\dot{v} \xleftarrow{1} \pi])) = \\
&= 2\frac{8}{7}\frac{1}{8}\frac{1}{8}\{\Delta_{\mathcal{A}^+}(\breve{A}_1(\mu[\pi, \pi_{d1}, \pi_{d1}, \pi_{d1}]), \breve{A}_1(\mu[\pi, \pi_{d1}, \pi_{d1}, \pi_{d2}]))+ \\
&+ \Delta_{\mathcal{A}^+}(\breve{A}_1(\mu[\pi, \pi_{d1}, \pi_{d1}, \pi_{d1}]), \breve{A}_1(\mu[\pi, \pi_{d1}, \pi_{d2}, \pi_{d1}]))+ \\
&\vdots \\
&+ \Delta_{\mathcal{A}^+}(\breve{A}_1(\mu[\pi, \pi_{d2}, \pi_{d2}, \pi_{d1}]), \breve{A}_1(\mu[\pi, \pi_{d2}, \pi_{d2}, \pi_{d2}]))\}
\end{aligned}
$$

Note that we avoided to calculate both $\Delta_{\mathcal{A}^+}(a, b)$ and $\Delta_{\mathcal{A}^+}(b, a)$, since they provide the same result, by calculating only $\Delta_{\mathcal{A}^+}(a, b)$ and multiplying the result by 2.

We have 28 possible pairs of agent line-ups, where the agent in both agent slots 1 (any π) is not able to differentiate with which agents is interacting, so it is not able to change its distribution of action sequences depending on the agent line-up. So, for any π we obtain the same result:

$$AD_1(\pi, \Pi_o, w_{\dot{L}}, \mu) = 2\frac{8}{7}\frac{1}{8}\frac{1}{8}\{28 \times 0\} = 0$$

For agent slot 2, the agent in both agent slots 2 (any π) is also not able to change its distribution of action sequences depending on the agent line-up. So again, for any π we obtain the same result:

$$
\begin{aligned}
AD_2(\pi, \Pi_o, w_{\dot{L}}, \mu) &= \eta_{\dot{L}^2} \sum_{\dot{u},\dot{v} \in \dot{L}_{-2}^{N(\mu)}(\Pi_o) | \dot{u} \neq \dot{v}} w_{\dot{L}}(\dot{u}) w_{\dot{L}}(\dot{v}) \Delta_{\mathcal{A}^+}(\breve{A}_2(\mu[\dot{u} \xleftarrow{2} \pi]), \breve{A}_2(\mu[\dot{v} \xleftarrow{2} \pi])) = \\
&= 2\frac{8}{7}\frac{1}{8}\frac{1}{8}\{\Delta_{\mathcal{A}^+}(\breve{A}_2(\mu[\pi_{d1}, \pi, \pi_{d1}, \pi_{d1}]), \breve{A}_2(\mu[\pi_{d1}, \pi, \pi_{d1}, \pi_{d2}]))+ \\
&+ \Delta_{\mathcal{A}^+}(\breve{A}_2(\mu[\pi_{d1}, \pi, \pi_{d1}, \pi_{d1}]), \breve{A}_2(\mu[\pi_{d1}, \pi, \pi_{d2}, \pi_{d1}]))+ \\
&\vdots \\
&+ \Delta_{\mathcal{A}^+}(\breve{A}_2(\mu[\pi_{d2}, \pi, \pi_{d2}, \pi_{d1}]), \breve{A}_2(\mu[\pi_{d2}, \pi, \pi_{d2}, \pi_{d2}]))\} = \\
&= 2\frac{8}{7}\frac{1}{8}\frac{1}{8}\{28 \times 0\} = 0
\end{aligned}
$$

For agent slot 3, the agent in both agent slots 3 (any π) is also not able to change its distribution of action sequences depending on the agent line-up. So again, for any π we obtain the same result:

$$\begin{aligned}AD_3(\pi,\Pi_o,w_{\dot{L}},\mu) &= \eta_{\dot{L}^2} \sum_{\dot{u},\dot{v}\in \dot{L}_{-3}^{N(\mu)}(\Pi_o)|\dot{u}\neq\dot{v}} w_{\dot{L}}(\dot{u})w_{\dot{L}}(\dot{v})\Delta_{\mathcal{A}^+}(\breve{A}_3(\mu[\dot{u}\xleftarrow{3}\pi]),\breve{A}_3(\mu[\dot{v}\xleftarrow{3}\pi])) =\\
&= 2\frac{8}{7}\frac{1}{8}\frac{1}{8}\{\Delta_{\mathcal{A}^+}(\breve{A}_3(\mu[\pi_{d1},\pi_{d1},\pi,\pi_{d1}]),\breve{A}_3(\mu[\pi_{d1},\pi_{d1},\pi,\pi_{d2}]))+\\
&+\Delta_{\mathcal{A}^+}(\breve{A}_3(\mu[\pi_{d1},\pi_{d1},\pi,\pi_{d1}]),\breve{A}_3(\mu[\pi_{d1},\pi_{d2},\pi,\pi_{d1}]))+\\
&\vdots\\
&+\Delta_{\mathcal{A}^+}(\breve{A}_3(\mu[\pi_{d2},\pi_{d2},\pi,\pi_{d1}]),\breve{A}_3(\mu[\pi_{d2},\pi_{d2},\pi,\pi_{d2}]))\} =\\
&= 2\frac{8}{7}\frac{1}{8}\frac{1}{8}\{28\times 0\} = 0\end{aligned}$$

And for agent slot 4, the agent in both agent slots 4 (any π) is also not able to change its distribution of action sequences depending on the agent line-up. So again, for any π we obtain the same result:

$$\begin{aligned}AD_4(\pi,\Pi_o,w_{\dot{L}},\mu) &= \eta_{\dot{L}^2} \sum_{\dot{u},\dot{v}\in \dot{L}_{-4}^{N(\mu)}(\Pi_o)|\dot{u}\neq\dot{v}} w_{\dot{L}}(\dot{u})w_{\dot{L}}(\dot{v})\Delta_{\mathcal{A}^+}(\breve{A}_4(\mu[\dot{u}\xleftarrow{4}\pi]),\breve{A}_4(\mu[\dot{v}\xleftarrow{4}\pi])) =\\
&= 2\frac{8}{7}\frac{1}{8}\frac{1}{8}\{\Delta_{\mathcal{A}^+}(\breve{A}_4(\mu[\pi_{d1},\pi_{d1},\pi_{d1},\pi]),\breve{A}_4(\mu[\pi_{d1},\pi_{d1},\pi_{d2},\pi]))+\\
&+\Delta_{\mathcal{A}^+}(\breve{A}_4(\mu[\pi_{d1},\pi_{d1},\pi_{d1},\pi]),\breve{A}_4(\mu[\pi_{d1},\pi_{d2},\pi_{d1},\pi]))+\\
&\vdots\\
&+\Delta_{\mathcal{A}^+}(\breve{A}_4(\mu[\pi_{d2},\pi_{d2},\pi_{d1},\pi]),\breve{A}_4(\mu[\pi_{d2},\pi_{d2},\pi_{d2},\pi]))\} =\\
&= 2\frac{8}{7}\frac{1}{8}\frac{1}{8}\{28\times 0\} = 0\end{aligned}$$

And finally, we calculate the AD value generalising for any possible evaluated agent:

$$\begin{aligned}AD(\Pi_e,w_{\Pi_e},\Pi_o,w_{\dot{L}},\mu,w_S) &= \sum_{\pi\in\Pi_e} w_{\Pi_e}(\pi)\sum_{i=1}^{N(\mu)} w_S(i,\mu)AD_i(\pi,\Pi_o,w_{\dot{L}},\mu) =\\
&= \sum_{\pi\in\Pi_e} w_{\Pi_e}(\pi)\frac{1}{4}\{AD_1(\pi,\Pi_o,w_{\dot{L}},\mu)+AD_2(\pi,\Pi_o,w_{\dot{L}},\mu)+\\
&+AD_3(\pi,\Pi_o,w_{\dot{L}},\mu)+AD_4(\pi,\Pi_o,w_{\dot{L}},\mu)\} =\\
&= \sum_{\pi\in\Pi_e} w_{\Pi_e}(\pi)\frac{1}{4}\{0+0+0+0\} = 0\end{aligned}$$

So, for every Π_e we obtain the same result:

$$\forall\Pi_e : AD(\Pi_e,w_{\Pi_e},\Pi_o,w_{\dot{L}},\mu,w_S) = 0$$

Therefore, predator-prey has $Left_{max} = 0$ for this property. □

Proposition 51. *$Right_{min}$* for the action dependency (AD) property is equal to 0 for the predator-prey environment.

Proof. To find $Right_{min}$ (equation 8.5), we need to find a Π_e that minimises the property value as much as possible while Π_o maximises it. Using $\Pi_e = \{\pi_u\}$ (a π_u agent always performs Up) we find this situation no matter which Π_o we use.

Following definition 17 we obtain the AD value for this $\langle \Pi_e, \Pi_o \rangle$ (where Π_o is instantiated with any permitted value). Since the multi-agent environment is not team symmetric, we need to calculate this property value for every agent slot. Also, since $|\Pi_e| = 1$ we just need to calculate this property value for one evaluated agent. Following definition 16 we calculate the AD value for the evaluated agent $\pi_u \in \Pi_e$ and each agent slot. We start with agent slot 1:

$$AD_1(\pi_u, \Pi_o, w_{\dot{L}}, \mu) = \eta_{\dot{L}^2} \sum_{\dot{u},\dot{v} \in \dot{L}_{-1}^{N(\mu)}(\Pi_o) | \dot{u} \neq \dot{v}} w_{\dot{L}}(\dot{u}) w_{\dot{L}}(\dot{v}) \Delta_{\mathcal{A}^+}(\breve{A}_1(\mu[\dot{u} \overset{1}{\leftarrow} \pi_u]), \breve{A}_1(\mu[\dot{v} \overset{1}{\leftarrow} \pi_u]))$$

We do not know which Π_o we have, so we use two figurative agent line-up patterns $\dot{u} = (*, \pi_1, \pi_2, \pi_3)$ and $\dot{v} = (*, \pi_4, \pi_5, \pi_6)$ from $\dot{L}_{-1}^{N(\mu)}(\Pi_o)$:

$$\Delta_{\mathcal{A}^+}(\breve{A}_1(\mu[\dot{u} \overset{1}{\leftarrow} \pi_u]), \breve{A}_1(\mu[\dot{v} \overset{1}{\leftarrow} \pi_u])) = \Delta_{\mathcal{A}^+}(\breve{A}_1(\mu[\pi_u, \pi_1, \pi_2, \pi_3]), \breve{A}_1(\mu[\pi_u, \pi_4, \pi_5, \pi_6]))$$

The agent in both agent slots 1 (π_u) performs the same sequence of actions (always Up) independently of the agent line-up. So:

$$\Delta_{\mathcal{A}^+}(\breve{A}_1(\mu[\pi_u, \pi_1, \pi_2, \pi_3]), \breve{A}_1(\mu[\pi_u, \pi_4, \pi_5, \pi_6])) = 0$$

We generalise AD_1 for any possible pair of agent line-up patterns:

$$AD_1(\pi_u, \Pi_o, w_{\dot{L}}, \mu) = \eta_{\dot{L}^2} \sum_{\dot{u},\dot{v} \in \dot{L}_{-1}^{N(\mu)}(\Pi_o) | \dot{u} \neq \dot{v}} w_{\dot{L}}(\dot{u}) w_{\dot{L}}(\dot{v}) \Delta_{\mathcal{A}^+}(\breve{A}_1(\mu[\dot{u} \overset{1}{\leftarrow} \pi_u]), \breve{A}_1(\mu[\dot{v} \overset{1}{\leftarrow} \pi_u])) =$$
$$= \eta_{\dot{L}^2} \sum_{\dot{u},\dot{v} \in \dot{L}_{-1}^{N(\mu)}(\Pi_o) | \dot{u} \neq \dot{v}} w_{\dot{L}}(\dot{u}) w_{\dot{L}}(\dot{v}) 0 = 0$$

For agent slot 2, the agent in both agent slots 2 (π_u) also performs the same sequence of actions (always Up) independently of the agent line-up. So:

$$AD_2(\pi_u, \Pi_o, w_{\dot{L}}, \mu) = \eta_{\dot{L}^2} \sum_{\dot{u},\dot{v} \in \dot{L}_{-2}^{N(\mu)}(\Pi_o) | \dot{u} \neq \dot{v}} w_{\dot{L}}(\dot{u}) w_{\dot{L}}(\dot{v}) \Delta_{\mathcal{A}^+}(\breve{A}_2(\mu[\dot{u} \overset{2}{\leftarrow} \pi_u]), \breve{A}_2(\mu[\dot{v} \overset{2}{\leftarrow} \pi_u])) =$$
$$= \eta_{\dot{L}^2} \sum_{\dot{u},\dot{v} \in \dot{L}_{-2}^{N(\mu)}(\Pi_o) | \dot{u} \neq \dot{v}} w_{\dot{L}}(\dot{u}) w_{\dot{L}}(\dot{v}) 0 = 0$$

For agent slot 3, the agent in both agent slots 3 (π_u) also performs the same sequence of actions (always Up) independently of the agent line-up. So:

$$AD_3(\pi_u, \Pi_o, w_{\dot{L}}, \mu) = \eta_{\dot{L}^2} \sum_{\dot{u},\dot{v} \in \dot{L}_{-3}^{N(\mu)}(\Pi_o) | \dot{u} \neq \dot{v}} w_{\dot{L}}(\dot{u}) w_{\dot{L}}(\dot{v}) \Delta_{\mathcal{A}^+}(\breve{A}_3(\mu[\dot{u} \overset{3}{\leftarrow} \pi_u]), \breve{A}_3(\mu[\dot{v} \overset{3}{\leftarrow} \pi_u])) =$$
$$= \eta_{\dot{L}^2} \sum_{\dot{u},\dot{v} \in \dot{L}_{-3}^{N(\mu)}(\Pi_o) | \dot{u} \neq \dot{v}} w_{\dot{L}}(\dot{u}) w_{\dot{L}}(\dot{v}) 0 = 0$$

And for agent slot 4, the agent in both agent slots 4 (π_u) also performs the same sequence of actions (always Up) independently of the agent line-up. So:

$$\begin{aligned}AD_4(\pi_u, \Pi_o, w_{\dot{L}}, \mu) &= \eta_{\dot{L}^2} \sum_{\dot{u},\dot{v}\in \dot{L}_{-4}^{N(\mu)}(\Pi_o)|\dot{u}\neq\dot{v}} w_{\dot{L}}(\dot{u})w_{\dot{L}}(\dot{v})\Delta_{\mathcal{A}^+}(\breve{A}_4(\mu[\dot{u} \overset{4}{\leftarrow} \pi_u]), \breve{A}_4(\mu[\dot{v} \overset{4}{\leftarrow} \pi_u])) = \\ &= \eta_{\dot{L}^2} \sum_{\dot{u},\dot{v}\in \dot{L}_{-4}^{N(\mu)}(\Pi_o)|\dot{u}\neq\dot{v}} w_{\dot{L}}(\dot{u})w_{\dot{L}}(\dot{v})0 = 0\end{aligned}$$

And finally, we calculate the AD value:

$$\begin{aligned}AD(\Pi_e, w_{\Pi_e}, \Pi_o, w_{\dot{L}}, \mu, w_S) &= \sum_{\pi\in\Pi_e} w_{\Pi_e}(\pi) \sum_{i=1}^{N(\mu)} w_S(i,\mu) AD_i(\pi, \Pi_o, w_{\dot{L}}, \mu) = \\ &= \frac{1}{1}\frac{1}{4}\{AD_1(\pi_u, \Pi_o, w_{\dot{L}}, \mu) + AD_2(\pi_u, \Pi_o, w_{\dot{L}}, \mu) + \\ &+ AD_3(\pi_u, \Pi_o, w_{\dot{L}}, \mu) + AD_4(\pi_u, \Pi_o, w_{\dot{L}}, \mu)\} = \\ &= \frac{1}{1}\frac{1}{4}\{0+0+0+0\} = 0\end{aligned}$$

So, for every Π_o we obtain the same result:

$$\forall \Pi_o : AD(\Pi_e, w_{\Pi_e}, \Pi_o, w_{\dot{L}}, \mu, w_S) = 0$$

Therefore, predator-prey has $Right_{min} = 0$ for this property. □

C.2 Reward Dependency

We continue with the reward dependency (RD) property. As given in section 7.3.1, we want to know if the evaluated agents obtain different rewards depending on which agent line-up they interact with. We use $\Delta_{\mathbb{Q}}(a', b')$ for $\Delta_{\mathbb{Q}^+}(a, b)$ where a' and b' are the expected results of a and b respectively, and $\Delta_{\mathbb{Q}}(a', b') = 0$ if numbers a' and b' are equal and 1 otherwise. We use an average of rewards as the utility function to calculate an agent's result.

Proposition 52. $General_{min}$ for the reward dependency (RD) property is equal to 0 for the predator-prey environment.

Proof. To find $General_{min}$ (equation 8.1), we need to find a pair $\langle \Pi_e, \Pi_o \rangle$ that minimises the property value as much as possible. We can have this situation by selecting $\Pi_e = \{\pi_u\}$ and $\Pi_o = \{\pi_{d1}, \pi_{d2}\}$ (a π_d agent always performs Down and a π_u agent always performs Up).

Following definition 19 we obtain the RD value for this $\langle \Pi_e, \Pi_o \rangle$. Since the multi-agent environment is not team symmetric, we need to calculate this property value for every agent slot. Also, since $|\Pi_e| = 1$ we just need to calculate this property value for one evaluated agent. Following definition 18 we calculate the RD value for the evaluated agent $\pi_u \in \Pi_e$ and each agent slot. We start with agent slot 1:

$$\begin{aligned}
RD_1(\pi_u, \Pi_o, w_{\dot{L}}, \mu) &= \eta_{\dot{L}^2} \sum_{\dot{u},\dot{v} \in \dot{L}_{-1}^{N(\mu)}(\Pi_o) | \dot{u} \neq \dot{v}} w_{\dot{L}}(\dot{u}) w_{\dot{L}}(\dot{v}) \Delta_{\mathbb{Q}}(R_1(\mu[\dot{u} \xleftarrow{1} \pi_u]), R_1(\mu[\dot{v} \xleftarrow{1} \pi_u])) = \\
&= 2\frac{8}{7}\frac{1}{8}\frac{1}{8}\{\Delta_{\mathbb{Q}}(R_1(\mu[\pi_u, \pi_{d1}, \pi_{d1}, \pi_{d1}]), R_1(\mu[\pi_u, \pi_{d1}, \pi_{d1}, \pi_{d2}]))+ \\
&+ \Delta_{\mathbb{Q}}(R_1(\mu[\pi_u, \pi_{d1}, \pi_{d1}, \pi_{d1}]), R_1(\mu[\pi_u, \pi_{d1}, \pi_{d2}, \pi_{d1}]))+ \\
&\vdots \\
&+ \Delta_{\mathbb{Q}}(R_1(\mu[\pi_u, \pi_{d2}, \pi_{d2}, \pi_{d1}]), R_1(\mu[\pi_u, \pi_{d2}, \pi_{d2}, \pi_{d2}]))\}
\end{aligned}$$

Note that we avoided to calculate both $\Delta_{\mathbb{Q}}(a, b)$ and $\Delta_{\mathbb{Q}}(b, a)$, since they provide the same result, by calculating only $\Delta_{\mathbb{Q}}(a, b)$ and multiplying the result by 2.

We have 28 possible pairs of agent line-ups, where the agent in both agent slots 1 (π_u) obtains the same expected average reward (1) independently of the agent line-up. So:

$$RD_1(\pi_u, \Pi_o, w_{\dot{L}}, \mu) = 2\frac{8}{7}\frac{1}{8}\frac{1}{8}\{28 \times 0\} = 0$$

For agent slot 2, the agent in both agent slots 2 (π_u) also obtains the same expected average reward (1) independently of the agent line-up. So:

$$\begin{aligned}
RD_2(\pi_u, \Pi_o, w_{\dot{L}}, \mu) &= \eta_{\dot{L}^2} \sum_{\dot{u},\dot{v} \in \dot{L}_{-2}^{N(\mu)}(\Pi_o) | \dot{u} \neq \dot{v}} w_{\dot{L}}(\dot{u}) w_{\dot{L}}(\dot{v}) \Delta_{\mathbb{Q}}(R_2(\mu[\dot{u} \xleftarrow{2} \pi_u]), R_2(\mu[\dot{v} \xleftarrow{2} \pi_u])) = \\
&= 2\frac{8}{7}\frac{1}{8}\frac{1}{8}\{\Delta_{\mathbb{Q}}(R_2(\mu[\pi_{d1}, \pi_u, \pi_{d1}, \pi_{d1}]), R_2(\mu[\pi_{d1}, \pi_u, \pi_{d1}, \pi_{d2}]))+ \\
&+ \Delta_{\mathbb{Q}}(R_2(\mu[\pi_{d1}, \pi_u, \pi_{d1}, \pi_{d1}]), R_2(\mu[\pi_{d1}, \pi_u, \pi_{d2}, \pi_{d1}]))+ \\
&\vdots \\
&+ \Delta_{\mathbb{Q}}(R_2(\mu[\pi_{d2}, \pi_u, \pi_{d2}, \pi_{d1}]), R_2(\mu[\pi_{d2}, \pi_u, \pi_{d2}, \pi_{d2}]))\} = \\
&= 2\frac{8}{7}\frac{1}{8}\frac{1}{8}\{28 \times 0\} = 0
\end{aligned}$$

For agent slot 3, the agent in both agent slots 3 (π_u) also obtains the same expected average reward (-1) independently of the agent line-up. So:

$$\begin{aligned}
RD_3(\pi_u, \Pi_o, w_{\dot{L}}, \mu) &= \eta_{\dot{L}^2} \sum_{\dot{u},\dot{v} \in \dot{L}_{-3}^{N(\mu)}(\Pi_o) | \dot{u} \neq \dot{v}} w_{\dot{L}}(\dot{u}) w_{\dot{L}}(\dot{v}) \Delta_{\mathbb{Q}}(R_3(\mu[\dot{u} \xleftarrow{3} \pi_u]), R_3(\mu[\dot{v} \xleftarrow{3} \pi_u])) = \\
&= 2\frac{8}{7}\frac{1}{8}\frac{1}{8}\{\Delta_{\mathbb{Q}}(R_3(\mu[\pi_{d1}, \pi_{d1}, \pi_u, \pi_{d1}]), R_3(\mu[\pi_{d1}, \pi_{d1}, \pi_u, \pi_{d2}]))+ \\
&+ \Delta_{\mathbb{Q}}(R_3(\mu[\pi_{d1}, \pi_{d1}, \pi_u, \pi_{d1}]), R_3(\mu[\pi_{d1}, \pi_{d2}, \pi_u, \pi_{d1}]))+ \\
&\vdots \\
&+ \Delta_{\mathbb{Q}}(R_3(\mu[\pi_{d2}, \pi_{d2}, \pi_u, \pi_{d1}]), R_3(\mu[\pi_{d2}, \pi_{d2}, \pi_u, \pi_{d2}]))\} = \\
&= 2\frac{8}{7}\frac{1}{8}\frac{1}{8}\{28 \times 0\} = 0
\end{aligned}$$

And for agent slot 4, the agent in both agent slots 4 (π_u) also obtains the same expected average reward (1) independently of the agent line-up. So:

$$\begin{aligned}
RD_4(\pi_u, \Pi_o, w_{\dot{L}}, \mu) = \eta_{\dot{L}^2} & \sum_{\dot{u},\dot{v} \in \dot{L}_{-4}^{N(\mu)}(\Pi_o) | \dot{u} \neq \dot{v}} w_{\dot{L}}(\dot{u}) w_{\dot{L}}(\dot{v}) \Delta_{\mathbb{Q}}(R_4(\mu[\dot{u} \overset{4}{\leftarrow} \pi_u]), R_4(\mu[\dot{v} \overset{4}{\leftarrow} \pi_u])) = \\
& = 2\frac{8}{7}\frac{1}{8}\frac{1}{8}\{\Delta_{\mathbb{Q}}(R_4(\mu[\pi_{d1}, \pi_{d1}, \pi_{d1}, \pi_u]), R_4(\mu[\pi_{d1}, \pi_{d1}, \pi_{d2}, \pi_u]))+ \\
& + \Delta_{\mathbb{Q}}(R_4(\mu[\pi_{d1}, \pi_{d1}, \pi_{d1}, \pi_u]), R_4(\mu[\pi_{d1}, \pi_{d2}, \pi_{d1}, \pi_u]))+ \\
& \qquad \vdots \\
& + \Delta_{\mathbb{Q}}(R_4(\mu[\pi_{d2}, \pi_{d2}, \pi_{d1}, \pi_u]), R_4(\mu[\pi_{d2}, \pi_{d2}, \pi_{d2}, \pi_u]))\} = \\
& = 2\frac{8}{7}\frac{1}{8}\frac{1}{8}\{28 \times 0\} = 0
\end{aligned}$$

And finally, we calculate the RD value:

$$\begin{aligned}
RD(\Pi_e, w_{\Pi_e}, \Pi_o, w_{\dot{L}}, \mu, w_S) = & \sum_{\pi \in \Pi_e} w_{\Pi_e}(\pi) \sum_{i=1}^{N(\mu)} w_S(i, \mu) RD_i(\pi, \Pi_o, w_{\dot{L}}, \mu) = \\
& = \frac{1}{1}\frac{1}{4}\{RD_1(\pi_u, \Pi_o, w_{\dot{L}}, \mu) + RD_2(\pi_u, \Pi_o, w_{\dot{L}}, \mu)+ \\
& + RD_3(\pi_u, \Pi_o, w_{\dot{L}}, \mu) + RD_4(\pi_u, \Pi_o, w_{\dot{L}}, \mu)\} = \\
& = \frac{1}{1}\frac{1}{4}\{0 + 0 + 0 + 0\} = 0
\end{aligned}$$

Since 0 is the lowest possible value for the reward dependency property, therefore predator-prey has $General_{min} = 0$ for this property. □

Conjecture 2. $General_{max}$ for the reward dependency (RD) property is equal to 1 for the predator-prey environment.

To find $General_{max}$ (equation 8.2), we need to find a pair $\langle \Pi_e, \Pi_o \rangle$ that maximises the property value as much as possible. We can have this situation by selecting $\Pi_e = \{\pi_r\}$ and $\Pi_o = \{\pi_s, \pi_r\}$ (a π_r agent always acts randomly and a π_s agent always stays in the same cell[1]).

Following definition 19 we obtain the RD value for this $\langle \Pi_e, \Pi_o \rangle$. Since the multi-agent environment is not team symmetric, we need to calculate this property value for every agent slot. Also, since $|\Pi_e| = 1$ we just need to calculate this property value for one evaluated agent. Following definition 18 we calculate the RD value for the evaluated agent $\pi_r \in \Pi_e$ and each agent slot. We start with agent slot 1:

[1]Note that every cell has an action that leads to a block or a boundary, therefore an agent performing this action stays at its current cell.

$$
\begin{aligned}
RD_1(\pi_r, \Pi_o, w_{\dot{L}}, \mu) &= \eta_{\dot{L}^2} \sum_{\dot{u},\dot{v} \in \dot{L}_{-1}^{N(\mu)}(\Pi_o) | \dot{u} \neq \dot{v}} w_{\dot{L}}(\dot{u}) w_{\dot{L}}(\dot{v}) \Delta_{\mathbb{Q}}(R_1(\mu[\dot{u} \xleftarrow{1} \pi_r]), R_1(\mu[\dot{v} \xleftarrow{1} \pi_r])) = \\
&= 2 \frac{8}{7} \frac{1}{8} \frac{1}{8} \{ \Delta_{\mathbb{Q}}(R_1(\mu[\pi_r, \pi_s, \pi_s, \pi_s]), R_1(\mu[\pi_r, \pi_s, \pi_s, \pi_r])) + \\
&+ \Delta_{\mathbb{Q}}(R_1(\mu[\pi_r, \pi_s, \pi_s, \pi_s]), R_1(\mu[\pi_r, \pi_s, \pi_r, \pi_s])) + \\
&\vdots \\
&+ \Delta_{\mathbb{Q}}(R_1(\mu[\pi_r, \pi_r, \pi_r, \pi_s]), R_1(\mu[\pi_r, \pi_r, \pi_r, \pi_r])) \}
\end{aligned}
$$

Note that we avoided to calculate both $\Delta_{\mathbb{Q}}(a, b)$ and $\Delta_{\mathbb{Q}}(b, a)$, since they provide the same result, by calculating only $\Delta_{\mathbb{Q}}(a, b)$ and multiplying the result by 2.

The expected average reward that agents obtain in these agent line-ups highly depends on the positions of the agents. The expected average reward of the agent in agent slot 1 (π_r) ranges from -1 to 1 (both values exclusive, since it always exists some probability that the prey is either chased or not). One reason is the stochastic behaviour of the π_r agents, which makes that, for every pair of agent line-ups, the agents in the same agent slot obtain different expected average rewards. Another reason is that the starting positions of the agent slots (in particular those for the predator team) do not have a symmetric place in the space (blocks are not symmetrically located in the space) which, for each π_r in a different agent slot, provides (most likely) different probabilities to chase the prey[2]. This makes that, for every pair of agent line-ups, the agents in the same agent slot obtain different expected average rewards, making its reward dependency equal to 1.

$$
RD_1(\pi_r, \Pi_o, w_{\dot{L}}, \mu) = 2 \frac{8}{7} \frac{1}{8} \frac{1}{8} \{28 \times 1\} = 1
$$

For agent slot 2, the expected results of agent line-ups highly depend on the positions of the agents too. So:

$$
\begin{aligned}
RD_2(\pi_r, \Pi_o, w_{\dot{L}}, \mu) &= \eta_{\dot{L}^2} \sum_{\dot{u},\dot{v} \in \dot{L}_{-2}^{N(\mu)}(\Pi_o) | \dot{u} \neq \dot{v}} w_{\dot{L}}(\dot{u}) w_{\dot{L}}(\dot{v}) \Delta_{\mathbb{Q}}(R_2(\mu[\dot{u} \xleftarrow{2} \pi_r]), R_2(\mu[\dot{v} \xleftarrow{2} \pi_r])) = \\
&= 2 \frac{8}{7} \frac{1}{8} \frac{1}{8} \{ \Delta_{\mathbb{Q}}(R_2(\mu[\pi_s, \pi_r, \pi_s, \pi_s]), R_2(\mu[\pi_s, \pi_r, \pi_s, \pi_r])) + \\
&+ \Delta_{\mathbb{Q}}(R_2(\mu[\pi_s, \pi_r, \pi_s, \pi_s]), R_2(\mu[\pi_s, \pi_r, \pi_r, \pi_s])) + \\
&\vdots \\
&+ \Delta_{\mathbb{Q}}(R_2(\mu[\pi_r, \pi_r, \pi_r, \pi_s]), R_2(\mu[\pi_r, \pi_r, \pi_r, \pi_r])) \} = \\
&= 2 \frac{8}{7} \frac{1}{8} \frac{1}{8} \{28 \times 1\} = 1
\end{aligned}
$$

For agent slot 3, the expected results of agent line-ups highly depend on the positions of the agents too. So:

[2]It is more likely that the prey is chased by the predator in agent slot 3 (which starting position is located in the lower left corner) than the predator in agent slot 2 (which starting position is located in the upper right corner), and the predator in agent slot 4 (which starting position is located in the lower right corner) has the lowest probability to chase the prey.

$$\begin{aligned}
RD_3(\pi_r, \Pi_o, w_{\dot{L}}, \mu) &= \eta_{\dot{L}^2} \sum_{\dot{u},\dot{v} \in \dot{L}_{-3}^{N(\mu)}(\Pi_o) | \dot{u} \neq \dot{v}} w_{\dot{L}}(\dot{u}) w_{\dot{L}}(\dot{v}) \Delta_{\mathbb{Q}}(R_3(\mu[\dot{u} \overset{3}{\leftarrow} \pi_r]), R_3(\mu[\dot{v} \overset{3}{\leftarrow} \pi_r])) = \\
&= 2\frac{8}{7}\frac{1}{8}\frac{1}{8}\{\Delta_{\mathbb{Q}}(R_3(\mu[\pi_s, \pi_s, \pi_r, \pi_s]), R_3(\mu[\pi_s, \pi_s, \pi_r, \pi_r]))+ \\
&+ \Delta_{\mathbb{Q}}(R_3(\mu[\pi_s, \pi_s, \pi_r, \pi_s]), R_3(\mu[\pi_s, \pi_r, \pi_r, \pi_s]))+ \\
&\vdots \\
&+ \Delta_{\mathbb{Q}}(R_3(\mu[\pi_r, \pi_r, \pi_r, \pi_s]), R_3(\mu[\pi_r, \pi_r, \pi_r, \pi_r]))\} = \\
&= 2\frac{8}{7}\frac{1}{8}\frac{1}{8}\{28 \times 1\} = 1
\end{aligned}$$

And for agent slot 4, the expected results of agent line-ups highly depend on the positions of the agents too. So:

$$\begin{aligned}
RD_4(\pi_r, \Pi_o, w_{\dot{L}}, \mu) &= \eta_{\dot{L}^2} \sum_{\dot{u},\dot{v} \in \dot{L}_{-4}^{N(\mu)}(\Pi_o) | \dot{u} \neq \dot{v}} w_{\dot{L}}(\dot{u}) w_{\dot{L}}(\dot{v}) \Delta_{\mathbb{Q}}(R_4(\mu[\dot{u} \overset{4}{\leftarrow} \pi_r]), R_4(\mu[\dot{v} \overset{4}{\leftarrow} \pi_r])) = \\
&= 2\frac{8}{7}\frac{1}{8}\frac{1}{8}\{\Delta_{\mathbb{Q}}(R_4(\mu[\pi_s, \pi_s, \pi_s, \pi_r]), R_4(\mu[\pi_s, \pi_s, \pi_r, \pi_r]))+ \\
&+ \Delta_{\mathbb{Q}}(R_4(\mu[\pi_s, \pi_s, \pi_s, \pi_r]), R_4(\mu[\pi_s, \pi_r, \pi_s, \pi_r]))+ \\
&\vdots \\
&+ \Delta_{\mathbb{Q}}(R_4(\mu[\pi_r, \pi_r, \pi_s, \pi_r]), R_4(\mu[\pi_r, \pi_r, \pi_r, \pi_r]))\} = \\
&= 2\frac{8}{7}\frac{1}{8}\frac{1}{8}\{28 \times 1\} = 1
\end{aligned}$$

And finally, we calculate the RD value:

$$\begin{aligned}
RD(\Pi_e, w_{\Pi_e}, \Pi_o, w_{\dot{L}}, \mu, w_S) &= \sum_{\pi \in \Pi_e} w_{\Pi_e}(\pi) \sum_{i=1}^{N(\mu)} w_S(i, \mu) RD_i(\pi, \Pi_o, w_{\dot{L}}, \mu) = \\
&= \frac{1}{1}\frac{1}{4}\{RD_1(\pi_r, \Pi_o, w_{\dot{L}}, \mu) + RD_2(\pi_r, \Pi_o, w_{\dot{L}}, \mu)+ \\
&+ RD_3(\pi_r, \Pi_o, w_{\dot{L}}, \mu) + RD_4(\pi_r, \Pi_o, w_{\dot{L}}, \mu)\} = \\
&= \frac{1}{1}\frac{1}{4}\{1 + 1 + 1 + 1\} = 1
\end{aligned}$$

Since 1 is the highest possible value for the reward dependency property, therefore predator-prey has $General_{max} = 1$ for this property.

Proposition 53. $Left_{max}$ for the reward dependency (RD) property is equal to 0 for the predator-prey environment.

Proof. To find $Left_{max}$ (equation 8.4), we need to find a Π_e that maximises the property value as much as possible while Π_o minimises it. Using $\Pi_o = \{\pi_{d1}, \pi_{d2}\}$ (a π_d agent always performs Down) we find this situation no matter which Π_e we use.

Following definition 19 we obtain the RD value for this $\langle \Pi_e, \Pi_o \rangle$ (where Π_e is instantiated with any permitted value). Since the multi-agent environment is not team symmetric, we need to calculate this property value for every agent slot. Also, we do not know which Π_e we have, so we use a figurative evaluated agent π from Π_e. Following definition 18 we calculate the RD value for this figurative evaluated agent $\pi \in \Pi_e$ and each agent slot. We start with agent slot 1:

$$\begin{aligned}
RD_1(\pi, \Pi_o, w_{\dot{L}}, \mu) = \eta_{\dot{L}^2} & \sum_{\dot{u},\dot{v} \in \dot{L}_{-1}^{N(\mu)}(\Pi_o) | \dot{u} \neq \dot{v}} w_{\dot{L}}(\dot{u}) w_{\dot{L}}(\dot{v}) \Delta_{\mathbb{Q}}(R_1(\mu[\dot{u} \overset{1}{\leftarrow} \pi]), R_1(\mu[\dot{v} \overset{1}{\leftarrow} \pi])) = \\
&= 2\frac{8}{7}\frac{1}{8}\frac{1}{8}\{\Delta_{\mathbb{Q}}(R_1(\mu[\pi, \pi_{d1}, \pi_{d1}, \pi_{d1}]), R_1(\mu[\pi, \pi_{d1}, \pi_{d1}, \pi_{d2}]))+ \\
&+ \Delta_{\mathbb{Q}}(R_1(\mu[\pi, \pi_{d1}, \pi_{d1}, \pi_{d1}]), R_1(\mu[\pi, \pi_{d1}, \pi_{d2}, \pi_{d1}]))+ \\
&\vdots \\
&+ \Delta_{\mathbb{Q}}(R_1(\mu[\pi, \pi_{d2}, \pi_{d2}, \pi_{d1}]), R_1(\mu[\pi, \pi_{d2}, \pi_{d2}, \pi_{d2}]))\}
\end{aligned}$$

Note that we avoided to calculate both $\Delta_{\mathbb{Q}}(a, b)$ and $\Delta_{\mathbb{Q}}(b, a)$, since they provide the same result, by calculating only $\Delta_{\mathbb{Q}}(a, b)$ and multiplying the result by 2.

We have 28 possible pairs of agent line-ups, where the agent in both agent slots 1 (any π) is not able to differentiate with which agents is interacting, so it is not able to change its distribution of action sequences depending on the agent line-up, obtaining the same expected average reward. So, for any π we obtain the same result:

$$RD_1(\pi, \Pi_o, w_{\dot{L}}, \mu) = 2\frac{8}{7}\frac{1}{8}\frac{1}{8}\{28 \times 0\} = 0$$

For agent slot 2, the agent in both agent slots 2 (any π) is also not able to change its distribution of action sequences depending on the agent line-up, obtaining the same expected average reward. So again, for any π we obtain the same result:

$$\begin{aligned}
RD_2(\pi, \Pi_o, w_{\dot{L}}, \mu) = \eta_{\dot{L}^2} & \sum_{\dot{u},\dot{v} \in \dot{L}_{-2}^{N(\mu)}(\Pi_o) | \dot{u} \neq \dot{v}} w_{\dot{L}}(\dot{u}) w_{\dot{L}}(\dot{v}) \Delta_{\mathbb{Q}}(R_2(\mu[\dot{u} \overset{2}{\leftarrow} \pi]), R_2(\mu[\dot{v} \overset{2}{\leftarrow} \pi])) = \\
&= 2\frac{8}{7}\frac{1}{8}\frac{1}{8}\{\Delta_{\mathbb{Q}}(R_2(\mu[\pi_{d1}, \pi, \pi_{d1}, \pi_{d1}]), R_2(\mu[\pi_{d1}, \pi, \pi_{d1}, \pi_{d2}]))+ \\
&+ \Delta_{\mathbb{Q}}(R_2(\mu[\pi_{d1}, \pi, \pi_{d1}, \pi_{d1}]), R_2(\mu[\pi_{d1}, \pi, \pi_{d2}, \pi_{d1}]))+ \\
&\vdots \\
&+ \Delta_{\mathbb{Q}}(R_2(\mu[\pi_{d2}, \pi, \pi_{d2}, \pi_{d1}]), R_2(\mu[\pi_{d2}, \pi, \pi_{d2}, \pi_{d2}]))\} = \\
&= 2\frac{8}{7}\frac{1}{8}\frac{1}{8}\{28 \times 0\} = 0
\end{aligned}$$

For agent slot 3, the agent in both agent slots 3 (any π) is also not able to change its distribution of action sequences depending on the agent line-up, obtaining the same expected average reward. So again, for any π we obtain the same result:

$$\begin{aligned}
RD_3(\pi, \Pi_o, w_{\dot{L}}, \mu) &= \eta_{\dot{L}^2} \sum_{\dot{u},\dot{v} \in \dot{L}_{-3}^{N(\mu)}(\Pi_o) | \dot{u} \neq \dot{v}} w_{\dot{L}}(\dot{u}) w_{\dot{L}}(\dot{v}) \Delta_{\mathbb{Q}}(R_3(\mu[\dot{u} \overset{3}{\leftarrow} \pi]), R_3(\mu[\dot{v} \overset{3}{\leftarrow} \pi])) = \\
&= 2\frac{8}{7}\frac{1}{8}\frac{1}{8}\{\Delta_{\mathbb{Q}}(R_3(\mu[\pi_{d1}, \pi_{d1}, \pi, \pi_{d1}]), R_3(\mu[\pi_{d1}, \pi_{d1}, \pi, \pi_{d2}]))+ \\
&+ \Delta_{\mathbb{Q}}(R_3(\mu[\pi_{d1}, \pi_{d1}, \pi, \pi_{d1}]), R_3(\mu[\pi_{d1}, \pi_{d2}, \pi, \pi_{d1}]))+ \\
&\vdots \\
&+ \Delta_{\mathbb{Q}}(R_3(\mu[\pi_{d2}, \pi_{d2}, \pi, \pi_{d1}]), R_3(\mu[\pi_{d2}, \pi_{d2}, \pi, \pi_{d2}]))\} = \\
&= 2\frac{8}{7}\frac{1}{8}\frac{1}{8}\{28 \times 0\} = 0
\end{aligned}$$

And for agent slot 4, the agent in both agent slots 4 (any π) is also not able to change its distribution of action sequences depending on the agent line-up, obtaining the same expected average reward. So again, for any π we obtain the same result:

$$\begin{aligned}
RD_4(\pi, \Pi_o, w_{\dot{L}}, \mu) &= \eta_{\dot{L}^2} \sum_{\dot{u},\dot{v} \in \dot{L}_{-4}^{N(\mu)}(\Pi_o) | \dot{u} \neq \dot{v}} w_{\dot{L}}(\dot{u}) w_{\dot{L}}(\dot{v}) \Delta_{\mathbb{Q}}(R_4(\mu[\dot{u} \overset{4}{\leftarrow} \pi]), R_4(\mu[\dot{v} \overset{4}{\leftarrow} \pi])) = \\
&= 2\frac{8}{7}\frac{1}{8}\frac{1}{8}\{\Delta_{\mathbb{Q}}(R_4(\mu[\pi_{d1}, \pi_{d1}, \pi_{d1}, \pi]), R_4(\mu[\pi_{d1}, \pi_{d1}, \pi_{d2}, \pi]))+ \\
&+ \Delta_{\mathbb{Q}}(R_4(\mu[\pi_{d1}, \pi_{d1}, \pi_{d1}, \pi]), R_4(\mu[\pi_{d1}, \pi_{d2}, \pi_{d1}, \pi]))+ \\
&\vdots \\
&+ \Delta_{\mathbb{Q}}(R_4(\mu[\pi_{d2}, \pi_{d2}, \pi_{d1}, \pi]), R_4(\mu[\pi_{d2}, \pi_{d2}, \pi_{d2}, \pi]))\} = \\
&= 2\frac{8}{7}\frac{1}{8}\frac{1}{8}\{28 \times 0\} = 0
\end{aligned}$$

And finally, we calculate the RD value generalising for any possible evaluated agent:

$$\begin{aligned}
RD(\Pi_e, w_{\Pi_e}, \Pi_o, w_{\dot{L}}, \mu, w_S) &= \sum_{\pi \in \Pi_e} w_{\Pi_e}(\pi) \sum_{i=1}^{N(\mu)} w_S(i, \mu) RD_i(\pi, \Pi_o, w_{\dot{L}}, \mu) = \\
&= \sum_{\pi \in \Pi_e} w_{\Pi_e}(\pi) \frac{1}{4}\{RD_1(\pi, \Pi_o, w_{\dot{L}}, \mu) + RD_2(\pi, \Pi_o, w_{\dot{L}}, \mu)+ \\
&+ RD_3(\pi, \Pi_o, w_{\dot{L}}, \mu) + RD_4(\pi, \Pi_o, w_{\dot{L}}, \mu)\} = \\
&= \sum_{\pi \in \Pi_e} w_{\Pi_e}(\pi) \frac{1}{4}\{0 + 0 + 0 + 0\} = 0
\end{aligned}$$

So, for every Π_e we obtain the same result:

$$\forall \Pi_e : RD(\Pi_e, w_{\Pi_e}, \Pi_o, w_{\dot{L}}, \mu, w_S) = 0$$

Therefore, predator-prey has $Left_{max} = 0$ for this property. □

Approximation 1. *Right$_{min}$* for the reward dependency (RD) property is equal to $\frac{13}{28}$ (as a *lower* approximation) for the predator-prey environment.

Proof. To find $Right_{min}$ (equation 8.5), we need to find a Π_e that minimises the property value as much as possible while Π_o maximises it. Using $\Pi_e = \{\pi_{chase}\}$ and $\Pi_o = \{\pi_{lose}, \pi_{win}\}$ (a π_{chase} agent tries to be chased when interacting as the prey and tries to perfectly coordinate with the other predators to chase the prey when interacting as a predator, a π_{lose} agent tries to be chased when interacting as the prey and avoids to chase the prey when interacting as a predator, and a π_{win} agent tries not to be chased when interacting as the prey and tries to perfectly coordinate with the other predators to chase the prey when interacting as a predator) we find a *lower* approximation.

Following definition 19 we obtain the RD value for this $\langle \Pi_e, \Pi_o \rangle$. Since the multi-agent environment is not team symmetric, we need to calculate this property value for every agent slot. Also, since $|\Pi_e| = 1$ we just need to calculate this property value for one evaluated agent. Following definition 18 we calculate the RD value for the evaluated agent $\pi_{chase} \in \Pi_e$ and each agent slot. We start with agent slot 1:

$$\begin{aligned} RD_1(\pi_{chase}, \Pi_o, w_{\dot{L}}, \mu) &= \eta_{\dot{L}^2} \sum_{\dot{u},\dot{v} \in \dot{L}_{-1}^{N(\mu)}(\Pi_o) | \dot{u} \neq \dot{v}} w_{\dot{L}}(\dot{u}) w_{\dot{L}}(\dot{v}) \Delta_{\mathbb{Q}}(R_1(\mu[\dot{u} \overset{1}{\leftarrow} \pi_{chase}]), R_1(\mu[\dot{v} \overset{1}{\leftarrow} \pi_{chase}])) = \\ &= 2\frac{8}{7}\frac{1}{8}\frac{1}{8}\{\Delta_{\mathbb{Q}}(R_1(\mu[\pi_{chase}, \pi_{lose}, \pi_{lose}, \pi_{lose}]), R_1(\mu[\pi_{chase}, \pi_{lose}, \pi_{lose}, \pi_{win}]))+ \\ &+ \Delta_{\mathbb{Q}}(R_1(\mu[\pi_{chase}, \pi_{lose}, \pi_{lose}, \pi_{lose}]), R_1(\mu[\pi_{chase}, \pi_{lose}, \pi_{win}, \pi_{lose}]))+ \\ &\vdots \\ &+ \Delta_{\mathbb{Q}}(R_1(\mu[\pi_{chase}, \pi_{win}, \pi_{win}, \pi_{lose}]), R_1(\mu[\pi_{chase}, \pi_{win}, \pi_{win}, \pi_{win}]))\} \end{aligned}$$

Note that we avoided to calculate both $\Delta_{\mathbb{Q}}(a, b)$ and $\Delta_{\mathbb{Q}}(b, a)$, since they provide the same result, by calculating only $\Delta_{\mathbb{Q}}(a, b)$ and multiplying the result by 2.

We have 28 possible pairs of agent line-ups, where π_{chase} from Π_e tries to make in the maximum pairs of agent line-ups, the agents in the same agent slot obtain the same expected average rewards while π_{win} and π_{lose} from Π_o try to make in the maximum pairs of agent line-ups, the agents in the same agent slot obtain different expected average rewards. The agents from Π_o can only assure that two agent line-ups ($(\pi_{chase}, \pi_{lose}, \pi_{lose}, \pi_{lose})$ and $(\pi_{chase}, \pi_{win}, \pi_{win}, \pi_{win})$) have different results ('prey not chased' and 'prey chased' respectively), therefore the agent from Π_e can obtain the same expected average reward on seven of the eight agent line-ups[3] (and therefore obtain the same expected average reward on twenty one pairs of agent line-ups). So:

$$RD_1(\pi_{chase}, \Pi_o, w_{\dot{L}}, \mu) = 2\frac{8}{7}\frac{1}{8}\frac{1}{8}\{21 \times 0 + 7 \times 1\} = \frac{1}{4}$$

For agent slot 2, the agents from Π_o can assure that three agent line-ups ($(\pi_{lose}, \pi_{chase}, \pi_{lose}, \pi_{win})$, $(\pi_{lose}, \pi_{chase}, \pi_{win}, \pi_{lose})$ and $(\pi_{lose}, \pi_{chase}, \pi_{win}, \pi_{win})$) have the same result ('prey chased') and other three agent line-ups ($(\pi_{win}, \pi_{chase}, \pi_{lose}, \pi_{lose})$, $(\pi_{win}, \pi_{chase}, \pi_{lose}, \pi_{win})$ and $(\pi_{win}, \pi_{chase}, \pi_{win}, \pi_{lose})$) have a different result ('prey not chased'), therefore the agent from Π_e can only obtain the same expected average reward on five of the eight agent line-ups (and therefore obtain the same expected average reward on thirteen pairs of agent line-ups). So:

[3]Note that only one predator trying to win is enough to chase a prey which wants to be chased.

$$RD_2(\pi_{chase}, \Pi_o, w_{\dot{L}}, \mu) = \eta_{\dot{L}^2} \sum_{\dot{u},\dot{v} \in \dot{L}_{-2}^{N(\mu)}(\Pi_o) | \dot{u} \neq \dot{v}} w_{\dot{L}}(\dot{u}) w_{\dot{L}}(\dot{v}) \Delta_{\mathbb{Q}}(R_2(\mu[\dot{u} \overset{2}{\leftarrow} \pi_{chase}]), R_2(\mu[\dot{v} \overset{2}{\leftarrow} \pi_{chase}])) =$$
$$= 2\frac{8}{7}\frac{1}{8}\frac{1}{8}\{\Delta_{\mathbb{Q}}(R_2(\mu[\pi_{lose}, \pi_{chase}, \pi_{lose}, \pi_{lose}]), R_2(\mu[\pi_{lose}, \pi_{chase}, \pi_{lose}, \pi_{win}]))+$$
$$+ \Delta_{\mathbb{Q}}(R_2(\mu[\pi_{lose}, \pi_{chase}, \pi_{lose}, \pi_{lose}]), R_2(\mu[\pi_{lose}, \pi_{chase}, \pi_{win}, \pi_{lose}]))+$$
$$\vdots$$
$$+ \Delta_{\mathbb{Q}}(R_2(\mu[\pi_{win}, \pi_{chase}, \pi_{win}, \pi_{lose}]), R_2(\mu[\pi_{win}, \pi_{chase}, \pi_{win}, \pi_{win}]))\} =$$
$$= 2\frac{8}{7}\frac{1}{8}\frac{1}{8}\{13 \times 0 + 15 \times 1\} = \frac{15}{28}$$

For agent slot 3, the agents from Π_o can also assure that three agent line-ups $((\pi_{lose}, \pi_{lose}, \pi_{chase}, \pi_{win}), (\pi_{lose}, \pi_{win}, \pi_{chase}, \pi_{lose})$ and $(\pi_{lose}, \pi_{win}, \pi_{chase}, \pi_{win}))$ have the same result ('prey chased') and other three agent line-ups $((\pi_{win}, \pi_{lose}, \pi_{chase}, \pi_{lose}), (\pi_{win}, \pi_{lose}, \pi_{chase}, \pi_{win})$ and $(\pi_{win}, \pi_{win}, \pi_{chase}, \pi_{lose}))$ have a different result ('prey not chased'), therefore the agent from Π_e can again only obtain the same expected average reward on five of the eight agent line-ups (and therefore obtain the same expected average reward on thirteen pairs of agent line-ups). So:

$$RD_3(\pi_{chase}, \Pi_o, w_{\dot{L}}, \mu) = \eta_{\dot{L}^2} \sum_{\dot{u},\dot{v} \in \dot{L}_{-3}^{N(\mu)}(\Pi_o) | \dot{u} \neq \dot{v}} w_{\dot{L}}(\dot{u}) w_{\dot{L}}(\dot{v}) \Delta_{\mathbb{Q}}(R_3(\mu[\dot{u} \overset{3}{\leftarrow} \pi_{chase}]), R_3(\mu[\dot{v} \overset{3}{\leftarrow} \pi_{chase}])) =$$
$$= 2\frac{8}{7}\frac{1}{8}\frac{1}{8}\{\Delta_{\mathbb{Q}}(R_3(\mu[\pi_{lose}, \pi_{lose}, \pi_{chase}, \pi_{lose}]), R_3(\mu[\pi_{lose}, \pi_{lose}, \pi_{chase}, \pi_{win}]))+$$
$$+ \Delta_{\mathbb{Q}}(R_3(\mu[\pi_{lose}, \pi_{lose}, \pi_{chase}, \pi_{lose}]), R_3(\mu[\pi_{lose}, \pi_{win}, \pi_{chase}, \pi_{lose}]))+$$
$$\vdots$$
$$+ \Delta_{\mathbb{Q}}(R_3(\mu[\pi_{win}, \pi_{win}, \pi_{chase}, \pi_{lose}]), R_3(\mu[\pi_{win}, \pi_{win}, \pi_{chase}, \pi_{win}]))\} =$$
$$= 2\frac{8}{7}\frac{1}{8}\frac{1}{8}\{13 \times 0 + 15 \times 1\} = \frac{15}{28}$$

And for agent slot 4, the agents from Π_o can also assure that three agent line-ups $((\pi_{lose}, \pi_{lose}, \pi_{win}, \pi_{chase}), (\pi_{lose}, \pi_{win}, \pi_{lose}, \pi_{chase})$ and $(\pi_{lose}, \pi_{win}, \pi_{win}, \pi_{chase}))$ have the same result ('prey chased') and other three agent line-ups $((\pi_{win}, \pi_{lose}, \pi_{lose}, \pi_{chase}), (\pi_{win}, \pi_{lose}, \pi_{win}, \pi_{chase})$ and $(\pi_{win}, \pi_{win}, \pi_{lose}, \pi_{chase}))$ have a different result ('prey not chased'), therefore the agent from Π_e can again only obtain the same expected average reward on five of the eight agent line-ups (and therefore obtain the same expected average reward on thirteen pairs of agent line-ups). So:

$$
\begin{aligned}
RD_4(\pi_{chase}, \Pi_o, w_{\dot{L}}, \mu) = \eta_{\dot{L}^2} \sum_{\dot{u},\dot{v}\in \dot{L}_{-4}^{N(\mu)}(\Pi_o)|\dot{u}\neq\dot{v}} & w_{\dot{L}}(\dot{u}) w_{\dot{L}}(\dot{v}) \Delta_{\mathbb{Q}}(R_4(\mu[\dot{u} \overset{4}{\leftarrow} \pi_{chase}]), R_4(\mu[\dot{v} \overset{4}{\leftarrow} \pi_{chase}])) = \\
&= 2\frac{8}{7}\frac{1}{8}\frac{1}{8}\{\Delta_{\mathbb{Q}}(R_4(\mu[\pi_{lose}, \pi_{lose}, \pi_{lose}, \pi_{chase}]), R_4(\mu[\pi_{lose}, \pi_{lose}, \pi_{win}, \pi_{chase}]))+ \\
&+ \Delta_{\mathbb{Q}}(R_4(\mu[\pi_{lose}, \pi_{lose}, \pi_{lose}, \pi_{chase}]), R_4(\mu[\pi_{lose}, \pi_{win}, \pi_{lose}, \pi_{chase}]))+ \\
&\qquad\vdots \\
&+ \Delta_{\mathbb{Q}}(R_4(\mu[\pi_{win}, \pi_{win}, \pi_{lose}, \pi_{chase}]), R_4(\mu[\pi_{win}, \pi_{win}, \pi_{win}, \pi_{chase}]))\} = \\
&= 2\frac{8}{7}\frac{1}{8}\frac{1}{8}\{13 \times 0 + 15 \times 1\} = \frac{15}{28}
\end{aligned}
$$

And finally, we calculate the RD value:

$$
\begin{aligned}
RD(\Pi_e, w_{\Pi_e}, \Pi_o, w_{\dot{L}}, \mu, w_S) &= \sum_{\pi\in\Pi_e} w_{\Pi_e}(\pi) \sum_{i=1}^{N(\mu)} w_S(i,\mu) RD_i(\pi, \Pi_o, w_{\dot{L}}, \mu) = \\
&= \frac{1}{1}\frac{1}{4}\{RD_1(\pi_{chase}, \Pi_o, w_{\dot{L}}, \mu) + RD_2(\pi_{chase}, \Pi_o, w_{\dot{L}}, \mu)+ \\
&+ RD_3(\pi_{chase}, \Pi_o, w_{\dot{L}}, \mu) + RD_4(\pi_{chase}, \Pi_o, w_{\dot{L}}, \mu)\} = \\
&= \frac{1}{1}\frac{1}{4}\left\{\frac{1}{4} + \frac{15}{28} + \frac{15}{28} + \frac{15}{28}\right\} = \frac{13}{28}
\end{aligned}
$$

Therefore, predator-prey has $Right_{min} = \frac{13}{28}$ (as a *lower* approximation) for this property. □

C.3 Fine Discrimination

Now we move to the fine discrimination (FD) property. As given in section 7.5.1, we want to know if different evaluated agents obtain different expected rewards when interacting in the same agent line-up patterns. We use $\Delta_{\mathbb{Q}}(a, b) = 0$ if numbers a and b are equal and 1 otherwise. We use an average of rewards as the utility function to calculate an agent's result.

Proposition 54. *$General_{min}$ for the fine discrimination (FD) property is equal to 0 for the predator-prey environment.*

Proof. To find $General_{min}$ (equation 8.1), we need to find a pair $\langle\Pi_e, \Pi_o\rangle$ that minimises the property value as much as possible. We can have this situation by selecting $\Pi_e = \{\pi_{u1}, \pi_{u2}\}$ and $\Pi_o = \{\pi_d\}$ (a π_u agent always performs Up and a π_d agent always performs Down).

Following definition 26 we obtain the FD value for this $\langle\Pi_e, \Pi_o\rangle$. Since the multi-agent environment is not team symmetric, we need to calculate this property value for every agent slot. Also, since $|\Pi_e| = 2$ we just need to calculate this property value for one pair of evaluated agents. Following definition 25 we calculate the FD value for the evaluated agents $\pi_{u1}, \pi_{u2} \in \Pi_e$ and each agent slot. We start with agent slot 1:

$$\begin{aligned} FD_1(\pi_{u1}, \pi_{u2}, \Pi_o, w_{\dot{L}}, \mu) &= \sum_{\dot{l} \in \dot{L}_{-1}^{N(\mu)}(\Pi_o)} w_{\dot{L}}(\dot{l}) \Delta_{\mathbb{Q}}(R_1(\mu[\dot{l} \xleftarrow{1} \pi_{u1}]), R_1(\mu[\dot{l} \xleftarrow{1} \pi_{u2}])) = \\ &= \frac{1}{1} \Delta_{\mathbb{Q}}(R_1(\mu[\pi_{u1}, \pi_d, \pi_d, \pi_d]), R_1(\mu[\pi_{u2}, \pi_d, \pi_d, \pi_d])) \end{aligned}$$

Both agents in agent slot 1 (π_{u1} and π_{u2}) obtain the same expected average reward (1). So:

$$FD_1(\pi_{u1}, \pi_{u2}, \Pi_o, w_{\dot{L}}, \mu) = \frac{1}{1} 0 = 0$$

For agent slot 2, both agents in agent slot 2 (π_{u1} and π_{u2}) also obtain the same expected average reward (1). So:

$$\begin{aligned} FD_2(\pi_{u1}, \pi_{u2}, \Pi_o, w_{\dot{L}}, \mu) &= \sum_{\dot{l} \in \dot{L}_{-2}^{N(\mu)}(\Pi_o)} w_{\dot{L}}(\dot{l}) \Delta_{\mathbb{Q}}(R_2(\mu[\dot{l} \xleftarrow{2} \pi_{u1}]), R_2(\mu[\dot{l} \xleftarrow{2} \pi_{u2}])) = \\ &= \frac{1}{1} \Delta_{\mathbb{Q}}(R_2(\mu[\pi_d, \pi_{u1}, \pi_d, \pi_d]), R_2(\mu[\pi_d, \pi_{u2}, \pi_d, \pi_d])) = \\ &= \frac{1}{1} 0 = 0 \end{aligned}$$

For agent slot 3, both agents in agent slot 3 (π_{u1} and π_{u2}) also obtain the same expected average reward (-1). So:

$$\begin{aligned} FD_3(\pi_{u1}, \pi_{u2}, \Pi_o, w_{\dot{L}}, \mu) &= \sum_{\dot{l} \in \dot{L}_{-3}^{N(\mu)}(\Pi_o)} w_{\dot{L}}(\dot{l}) \Delta_{\mathbb{Q}}(R_3(\mu[\dot{l} \xleftarrow{3} \pi_{u1}]), R_3(\mu[\dot{l} \xleftarrow{3} \pi_{u2}])) = \\ &= \frac{1}{1} \Delta_{\mathbb{Q}}(R_3(\mu[\pi_d, \pi_d, \pi_{u1}, \pi_d]), R_3(\mu[\pi_d, \pi_d, \pi_{u2}, \pi_d])) = \\ &= \frac{1}{1} 0 = 0 \end{aligned}$$

And for agent slot 4, both agents in agent slot 4 (π_{u1} and π_{u2}) also obtain the same expected average reward (1). So:

$$\begin{aligned} FD_4(\pi_{u1}, \pi_{u2}, \Pi_o, w_{\dot{L}}, \mu) &= \sum_{\dot{l} \in \dot{L}_{-4}^{N(\mu)}(\Pi_o)} w_{\dot{L}}(\dot{l}) \Delta_{\mathbb{Q}}(R_4(\mu[\dot{l} \xleftarrow{4} \pi_{u1}]), R_4(\mu[\dot{l} \xleftarrow{4} \pi_{u2}])) = \\ &= \frac{1}{1} \Delta_{\mathbb{Q}}(R_4(\mu[\pi_d, \pi_d, \pi_d, \pi_{u1}]), R_4(\mu[\pi_d, \pi_d, \pi_d, \pi_{u2}])) = \\ &= \frac{1}{1} 0 = 0 \end{aligned}$$

And finally, we calculate the FD value:

$$\begin{aligned}
FD(\Pi_e, w_{\Pi_e}, \Pi_o, w_{\dot{L}}, \mu, w_S) = \eta_{\Pi^2} & \sum_{\pi_1,\pi_2 \in \Pi_e | \pi_1 \neq \pi_2} w_{\Pi_e}(\pi_1) w_{\Pi_e}(\pi_2) \sum_{i=1}^{N(\mu)} w_S(i,\mu) \times \\
& \times FD_i(\pi_1, \pi_2, \Pi_o, w_{\dot{L}}, \mu) = \\
& = 2 \frac{2}{1} \frac{1}{2} \frac{1}{2} \frac{1}{4} \{ FD_1(\pi_{u1}, \pi_{u2}, \Pi_o, w_{\dot{L}}, \mu) + FD_2(\pi_{u1}, \pi_{u2}, \Pi_o, w_{\dot{L}}, \mu) + \\
& + FD_3(\pi_{u1}, \pi_{u2}, \Pi_o, w_{\dot{L}}, \mu) + FD_4(\pi_{u1}, \pi_{u2}, \Pi_o, w_{\dot{L}}, \mu) \} = \\
& = 2 \frac{2}{1} \frac{1}{2} \frac{1}{2} \frac{1}{4} \{0 + 0 + 0 + 0\} = 0
\end{aligned}$$

Note that we avoided to calculate both $FD_i(a, b, \Pi_o, w_{\dot{L}}, \mu)$ and $FD_i(b, a, \Pi_o, w_{\dot{L}}, \mu)$, since they provide the same result, by calculating only $FD_i(a, b, \Pi_o, w_{\dot{L}}, \mu)$ and multiplying the result by 2.

Since 0 is the lowest possible value for the fine discrimination property, therefore predator-prey has $General_{min} = 0$ for this property. □

Proposition 55. $General_{max}$ for the fine discrimination (FD) property is equal to 1 for the predator-prey environment.

Proof. To find $General_{max}$ (equation 8.2), we need to find a pair $\langle \Pi_e, \Pi_o \rangle$ that maximises the property value as much as possible. We can have this situation by selecting $\Pi_e = \{\pi_s, \pi_r\}$ and $\Pi_o = \{\pi_s\}$ (a π_r agent always acts randomly and a π_s agent always stays in the same cell[4]).

Following definition 26 we obtain the FD value for this $\langle \Pi_e, \Pi_o \rangle$. Since the multi-agent environment is not team symmetric, we need to calculate this property value for every agent slot. Also, since $|\Pi_e| = 2$ we just need to calculate this property value for one pair of evaluated agents. Following definition 25 we calculate the FD value for the evaluated agents $\pi_s, \pi_r \in \Pi_e$ and each agent slot. We start with agent slot 1:

$$\begin{aligned}
FD_1(\pi_s, \pi_r, \Pi_o, w_{\dot{L}}, \mu) = & \sum_{\dot{l} \in \dot{L}_{-1}^{N(\mu)}(\Pi_o)} w_{\dot{L}}(\dot{l}) \Delta_{\mathbb{Q}}(R_1(\mu[\dot{l} \xleftarrow{1} \pi_s]), R_1(\mu[\dot{l} \xleftarrow{1} \pi_r])) = \\
& = \frac{1}{1} \Delta_{\mathbb{Q}}(R_1(\mu[\pi_s, \pi_s, \pi_s, \pi_s]), R_1(\mu[\pi_r, \pi_s, \pi_s, \pi_s]))
\end{aligned}$$

Both agents in agent slot 1 (π_s and π_r) obtain different expected average rewards (1 and slightly less than 1 respectively). So:

$$FD_1(\pi_s, \pi_r, \Pi_o, w_{\dot{L}}, \mu) = \frac{1}{1} 1 = 1$$

For agent slot 2, both agents in agent slot 2 (π_s and π_r) also obtain different expected average rewards (-1 and slightly more than -1 respectively). So:

[4]Note that every cell has an action that leads to a block or a boundary, therefore an agent performing this action stays at its current cell.

$$\begin{aligned} FD_2(\pi_s, \pi_r, \Pi_o, w_{\dot{L}}, \mu) &= \sum_{\dot{l} \in \dot{L}_{-2}^{N(\mu)}(\Pi_o)} w_{\dot{L}}(\dot{l}) \Delta_{\mathbb{Q}}(R_2(\mu[\dot{l} \overset{2}{\leftarrow} \pi_s]), R_2(\mu[\dot{l} \overset{2}{\leftarrow} \pi_r])) = \\ &= \frac{1}{1} \Delta_{\mathbb{Q}}(R_2(\mu[\pi_s, \pi_s, \pi_s, \pi_s]), R_2(\mu[\pi_s, \pi_r, \pi_s, \pi_s])) = \\ &= \frac{1}{1} 1 = 1 \end{aligned}$$

For agent slot 3, both agents in agent slot 3 (π_s and π_r) also obtain different expected average rewards (-1 and slightly more than -1 respectively). So:

$$\begin{aligned} FD_3(\pi_s, \pi_r, \Pi_o, w_{\dot{L}}, \mu) &= \sum_{\dot{l} \in \dot{L}_{-3}^{N(\mu)}(\Pi_o)} w_{\dot{L}}(\dot{l}) \Delta_{\mathbb{Q}}(R_3(\mu[\dot{l} \overset{3}{\leftarrow} \pi_s]), R_3(\mu[\dot{l} \overset{3}{\leftarrow} \pi_r])) = \\ &= \frac{1}{1} \Delta_{\mathbb{Q}}(R_3(\mu[\pi_s, \pi_s, \pi_s, \pi_s]), R_3(\mu[\pi_s, \pi_s, \pi_r, \pi_s])) = \\ &= \frac{1}{1} 1 = 1 \end{aligned}$$

And for agent slot 4, both agents in agent slot 4 (π_s and π_r) also obtain different expected average rewards (-1 and slightly more than -1 respectively). So:

$$\begin{aligned} FD_4(\pi_s, \pi_r, \Pi_o, w_{\dot{L}}, \mu) &= \sum_{\dot{l} \in \dot{L}_{-4}^{N(\mu)}(\Pi_o)} w_{\dot{L}}(\dot{l}) \Delta_{\mathbb{Q}}(R_4(\mu[\dot{l} \overset{4}{\leftarrow} \pi_s]), R_4(\mu[\dot{l} \overset{4}{\leftarrow} \pi_r])) = \\ &= \frac{1}{1} \Delta_{\mathbb{Q}}(R_4(\mu[\pi_s, \pi_s, \pi_s, \pi_s]), R_4(\mu[\pi_s, \pi_s, \pi_s, \pi_r])) = \\ &= \frac{1}{1} 1 = 1 \end{aligned}$$

And finally, we calculate the FD value:

$$\begin{aligned} FD(\Pi_e, w_{\Pi_e}, \Pi_o, w_{\dot{L}}, \mu, w_S) &= \eta_{\Pi^2} \sum_{\pi_1, \pi_2 \in \Pi_e | \pi_1 \neq \pi_2} w_{\Pi_e}(\pi_1) w_{\Pi_e}(\pi_2) \sum_{i=1}^{N(\mu)} w_S(i, \mu) \times \\ &\times FD_i(\pi_1, \pi_2, \Pi_o, w_{\dot{L}}, \mu) = \\ &= 2 \frac{2}{1} \frac{1}{2} \frac{1}{2} \frac{1}{4} \{FD_1(\pi_s, \pi_r, \Pi_o, w_{\dot{L}}, \mu) + FD_2(\pi_s, \pi_r, \Pi_o, w_{\dot{L}}, \mu) + \\ &+ FD_3(\pi_s, \pi_r, \Pi_o, w_{\dot{L}}, \mu) + FD_4(\pi_s, \pi_r, \Pi_o, w_{\dot{L}}, \mu)\} = \\ &= 2 \frac{2}{1} \frac{1}{2} \frac{1}{2} \frac{1}{4} \{1 + 1 + 1 + 1\} = 1 \end{aligned}$$

Note that we avoided to calculate both $FD_i(a, b, \Pi_o, w_{\dot{L}}, \mu)$ and $FD_i(b, a, \Pi_o, w_{\dot{L}}, \mu)$, since they provide the same result, by calculating only $FD_i(a, b, \Pi_o, w_{\dot{L}}, \mu)$ and multiplying the result by 2.

Since 1 is the highest possible value for the fine discrimination property, therefore predator-prey has $General_{max} = 1$ for this property. □

Proposition 56. *$Left_{max}$ for the fine discrimination (FD) property is equal to 0 for the predator-prey environment.*

Proof. To find $Left_{max}$ (equation 8.4), we need to find a Π_e that maximises the property value as much as possible while Π_o minimises it. Using $\Pi_o = \{\pi_{chase}\}$ (a π_{chase} agent tries to be chased when interacting as the prey and tries to perfectly coordinate with the other predators to chase the prey when interacting as a predator) we find this situation no matter which Π_e we use.

Following definition 26 we obtain the FD value for this $\langle\Pi_e, \Pi_o\rangle$ (where Π_e is instantiated with any permitted value). Since the multi-agent environment is not team symmetric, we need to calculate this property value for every agent slot. Also, we do not know which Π_e we have, so we use a figurative pair of evaluated agents π_1, π_2 from Π_e. Following definition 25 we calculate the FD value for these figurative evaluated agents $\pi_1, \pi_2 \in \Pi_e$ and each agent slot. We start with agent slot 1:

$$
\begin{aligned}
FD_1(\pi_1, \pi_2, \Pi_o, w_{\dot{L}}, \mu) &= \sum_{\dot{l} \in \dot{L}_{-1}^{N(\mu)}(\Pi_o)} w_{\dot{L}}(\dot{l}) \Delta_{\mathbb{Q}}(R_1(\mu[\dot{l} \overset{1}{\leftarrow} \pi_1]), R_1(\mu[\dot{l} \overset{1}{\leftarrow} \pi_2])) = \\
&= \frac{1}{1} \Delta_{\mathbb{Q}}(R_1(\mu[\pi_1, \pi_{chase}, \pi_{chase}, \pi_{chase}]), R_1(\mu[\pi_2, \pi_{chase}, \pi_{chase}, \pi_{chase}]))
\end{aligned}
$$

The predators perfectly coordinate to always chase the prey (as seen in lemma 3), obtaining both agents in agent slot 1 (any pair π_1 and π_2) the same expected average reward (-1). So, for any pair π_1 and π_2 we obtain the same result:

$$
FD_1(\pi_1, \pi_2, \Pi_o, w_{\dot{L}}, \mu) = \frac{1}{1} 0 = 0
$$

For agent slot 2, the prey is always chased by the two perfectly coordinated predators[5], obtaining both agents in agent slot 2 (any pair π_1 and π_2) the same expected average reward (1). So again, for any pair π_1 and π_2 we obtain the same result:

$$
\begin{aligned}
FD_2(\pi_1, \pi_2, \Pi_o, w_{\dot{L}}, \mu) &= \sum_{\dot{l} \in \dot{L}_{-2}^{N(\mu)}(\Pi_o)} w_{\dot{L}}(\dot{l}) \Delta_{\mathbb{Q}}(R_2(\mu[\dot{l} \overset{2}{\leftarrow} \pi_1]), R_2(\mu[\dot{l} \overset{2}{\leftarrow} \pi_2])) = \\
&= \frac{1}{1} \Delta_{\mathbb{Q}}(R_2(\mu[\pi_{chase}, \pi_1, \pi_{chase}, \pi_{chase}]), R_2(\mu[\pi_{chase}, \pi_2, \pi_{chase}, \pi_{chase}])) = \\
&= \frac{1}{1} 0 = 0
\end{aligned}
$$

For agent slot 3, both agents in agent slot 3 (any pair π_1 and π_2) also obtain the same expected average reward (1). So again, for any pair π_1 and π_2 we obtain the same result:

$$
\begin{aligned}
FD_3(\pi_1, \pi_2, \Pi_o, w_{\dot{L}}, \mu) &= \sum_{\dot{l} \in \dot{L}_{-3}^{N(\mu)}(\Pi_o)} w_{\dot{L}}(\dot{l}) \Delta_{\mathbb{Q}}(R_3(\mu[\dot{l} \overset{3}{\leftarrow} \pi_1]), R_3(\mu[\dot{l} \overset{3}{\leftarrow} \pi_2])) = \\
&= \frac{1}{1} \Delta_{\mathbb{Q}}(R_3(\mu[\pi_{chase}, \pi_{chase}, \pi_1, \pi_{chase}]), R_3(\mu[\pi_{chase}, \pi_{chase}, \pi_2, \pi_{chase}])) = \\
&= \frac{1}{1} 0 = 0
\end{aligned}
$$

[5]Note that two predator trying to chase the prey is enough to chase a prey which wants to be chased.

And for agent slot 4, both agents in agent slot 4 (any pair π_1 and π_2) also obtain the same expected average reward (1). So again, for any pair π_1 and π_2 we obtain the same result:

$$\begin{aligned} FD_4(\pi_1, \pi_2, \Pi_o, w_{\dot{L}}, \mu) = & \sum_{\dot{l} \in \dot{L}_{-4}^{N(\mu)}(\Pi_o)} w_{\dot{L}}(\dot{l}) \Delta_{\mathbb{Q}}(R_4(\mu[\dot{l} \overset{4}{\leftarrow} \pi_1]), R_4(\mu[\dot{l} \overset{4}{\leftarrow} \pi_2])) = \\ = & \frac{1}{1} \Delta_{\mathbb{Q}}(R_4(\mu[\pi_{chase}, \pi_{chase}, \pi_{chase}, \pi_1]), R_4(\mu[\pi_{chase}, \pi_{chase}, \pi_{chase}, \pi_2])) = \\ = & \frac{1}{1} 0 = 0 \end{aligned}$$

And finally, we calculate the FD value generalising for any possible pair of evaluated agents:

$$\begin{aligned} FD(\Pi_e, w_{\Pi_e}, \Pi_o, w_{\dot{L}}, \mu, w_S) = \eta_{\Pi^2} & \sum_{\pi_1, \pi_2 \in \Pi_e | \pi_1 \neq \pi_2} w_{\Pi_e}(\pi_1) w_{\Pi_e}(\pi_2) \sum_{i=1}^{N(\mu)} w_S(i, \mu) \times \\ & \times FD_i(\pi_1, \pi_2, \Pi_o, w_{\dot{L}}, \mu) = \\ & = 2 \times \eta_{\Pi^2} \sum_{\pi_1, \pi_2 \in \Pi_e | \pi_1 \neq \pi_2} w_{\Pi_e}(\pi_1) w_{\Pi_e}(\pi_2) \frac{1}{4} \times \\ & \times \{FD_1(\pi_1, \pi_2, \Pi_o, w_{\dot{L}}, \mu) + FD_2(\pi_1, \pi_2, \Pi_o, w_{\dot{L}}, \mu) + \\ & + FD_3(\pi_1, \pi_2, \Pi_o, w_{\dot{L}}, \mu) + FD_4(\pi_1, \pi_2, \Pi_o, w_{\dot{L}}, \mu)\} = \\ & = 2 \times \eta_{\Pi^2} \sum_{\pi_1, \pi_2 \in \Pi_e | \pi_1 \neq \pi_2} w_{\Pi_e}(\pi_1) w_{\Pi_e}(\pi_2) \frac{1}{4} \{0 + 0 + 0 + 0\} = 0 \end{aligned}$$

Note that we avoided to calculate both $FD_i(a, b, \Pi_o, w_{\dot{L}}, \mu)$ and $FD_i(b, a, \Pi_o, w_{\dot{L}}, \mu)$, since they provide the same result, by calculating only $FD_i(a, b, \Pi_o, w_{\dot{L}}, \mu)$ and multiplying the result by 2.

So, for every Π_e we obtain the same result:

$$\forall \Pi_e : FD(\Pi_e, w_{\Pi_e}, \Pi_o, w_{\dot{L}}, \mu, w_S) = 0$$

Therefore, predator-prey has $Left_{max} = 0$ for this property. □

Proposition 57. *$Right_{min}$ for the fine discrimination (FD) property is equal to 0 for the predator-prey environment.*

Proof. To find $Right_{min}$ (equation 8.5), we need to find a Π_e that minimises the property value as much as possible while Π_o maximises it. Using $\Pi_e = \{\pi_{u1}, \pi_{u2}\}$ (a π_u agent always performs Up) we find this situation no matter which Π_o we use.

Following definition 26 we obtain the FD value for this $\langle \Pi_e, \Pi_o \rangle$ (where Π_o is instantiated with any permitted value). Since the multi-agent environment is not team symmetric, we need to calculate this property value for every agent slot. Also, since $|\Pi_e| = 2$ we just need to calculate this property value for one pair of evaluated agents. Following definition 25 we calculate the FD value for the evaluated agents $\pi_{u1}, \pi_{u2} \in \Pi_e$ and each agent slot. We start with agent slot 1:

$$FD_1(\pi_{u1}, \pi_{u2}, \Pi_o, w_{\dot{L}}, \mu) = \sum_{\dot{l} \in \dot{L}_{-1}^{N(\mu)}(\Pi_o)} w_{\dot{L}}(\dot{l}) \Delta_{\mathbb{Q}}(R_1(\mu[\dot{l} \overset{1}{\leftarrow} \pi_{u1}]), R_1(\mu[\dot{l} \overset{1}{\leftarrow} \pi_{u2}]))$$

We do not know which Π_o we have, so we use a figurative agent line-up pattern $\dot{l} = (*, \pi_1, \pi_2, \pi_3)$ from $\dot{L}_{-1}^{N(\mu)}(\Pi_o)$:

$$\Delta_{\mathbb{Q}}(R_1(\mu[\dot{l} \overset{1}{\leftarrow} \pi_{u1}]), R_1(\mu[\dot{l} \overset{1}{\leftarrow} \pi_{u2}])) = \Delta_{\mathbb{Q}}(R_1(\mu[\pi_{u1}, \pi_1, \pi_2, \pi_3]), R_1(\mu[\pi_{u2}, \pi_1, \pi_2, \pi_3]))$$

The other agents (any trio π_1,π_2 and π_3) are not able to differentiate with which agent they are interacting, so they are not able to change their distribution of action sequences depending on the behaviour of the agent in agent slot 1, obtaining both agents in agent slot 1 (π_{u1} and π_{u2}) the same expected average reward. So:

$$\Delta_{\mathbb{Q}}(R_1(\mu[\pi_{u1}, \pi_1, \pi_2, \pi_3]), R_1(\mu[\pi_{u2}, \pi_1, \pi_2, \pi_3])) = 0$$

We generalise FD_1 for any possible agent line-up pattern:

$$\begin{aligned} FD_1(\pi_{u1}, \pi_{u2}, \Pi_o, w_{\dot{L}}, \mu) &= \sum_{\dot{l} \in \dot{L}_{-1}^{N(\mu)}(\Pi_o)} w_{\dot{L}}(\dot{l}) \Delta_{\mathbb{Q}}(R_1(\mu[\dot{l} \overset{1}{\leftarrow} \pi_{u1}]), R_1(\mu[\dot{l} \overset{1}{\leftarrow} \pi_{u2}])) = \\ &= \sum_{\dot{l} \in \dot{L}_{-1}^{N(\mu)}(\Pi_o)} w_{\dot{L}}(\dot{l}) 0 = 0 \end{aligned}$$

For agent slot 2, the other agents (any trio π_1,π_2 and π_3) are also not able to change their distribution of action sequences depending on the behaviour of the agent in agent slot 2, obtaining both agents in agent slot 2 (π_{u1} and π_{u2}) the same expected average reward. So:

$$\begin{aligned} FD_2(\pi_{u1}, \pi_{u2}, \Pi_o, w_{\dot{L}}, \mu) &= \sum_{\dot{l} \in \dot{L}_{-2}^{N(\mu)}(\Pi_o)} w_{\dot{L}}(\dot{l}) \Delta_{\mathbb{Q}}(R_2(\mu[\dot{l} \overset{2}{\leftarrow} \pi_{u1}]), R_2(\mu[\dot{l} \overset{2}{\leftarrow} \pi_{u2}])) = \\ &= \sum_{\dot{l} \in \dot{L}_{-2}^{N(\mu)}(\Pi_o)} w_{\dot{L}}(\dot{l}) 0 = 0 \end{aligned}$$

For agent slot 3, the other agents (any trio π_1,π_2 and π_3) are also not able to change their distribution of action sequences depending on the behaviour of the agent in agent slot 3, obtaining both agents in agent slot 3 (π_{u1} and π_{u2}) the same expected average reward. So:

$$\begin{aligned} FD_3(\pi_{u1}, \pi_{u2}, \Pi_o, w_{\dot{L}}, \mu) &= \sum_{\dot{l} \in \dot{L}_{-3}^{N(\mu)}(\Pi_o)} w_{\dot{L}}(\dot{l}) \Delta_{\mathbb{Q}}(R_3(\mu[\dot{l} \overset{3}{\leftarrow} \pi_{u1}]), R_3(\mu[\dot{l} \overset{3}{\leftarrow} \pi_{u2}])) = \\ &= \sum_{\dot{l} \in \dot{L}_{-3}^{N(\mu)}(\Pi_o)} w_{\dot{L}}(\dot{l}) 0 = 0 \end{aligned}$$

And for agent slot 4, the other agents (any trio π_1,π_2 and π_3) are also not able to change their distribution of action sequences depending on the behaviour of the agent in agent slot 4, obtaining both agents in agent slot 4 (π_{u1} and π_{u2}) the same expected average reward. So:

$$\begin{aligned} FD_4(\pi_{u1}, \pi_{u2}, \Pi_o, w_{\dot{L}}, \mu) &= \sum_{\dot{l} \in \dot{L}_{-4}^{N(\mu)}(\Pi_o)} w_{\dot{L}}(\dot{l}) \Delta_{\mathbb{Q}}(R_4(\mu[\dot{l} \overset{4}{\leftarrow} \pi_{u1}]), R_4(\mu[\dot{l} \overset{4}{\leftarrow} \pi_{u2}])) = \\ &= \sum_{\dot{l} \in \dot{L}_{-4}^{N(\mu)}(\Pi_o)} w_{\dot{L}}(\dot{l}) 0 = 0 \end{aligned}$$

And finally, we calculate the FD value:

$$\begin{aligned}FD(\Pi_e, w_{\Pi_e}, \Pi_o, w_{\dot{L}}, \mu, w_S) &= \eta_{\Pi^2} \sum_{\pi_1,\pi_2 \in \Pi_e | \pi_1 \neq \pi_2} w_{\Pi_e}(\pi_1) w_{\Pi_e}(\pi_2) \sum_{i=1}^{N(\mu)} w_S(i,\mu) \times \\ &\times FD_i(\pi_1, \pi_2, \Pi_o, w_{\dot{L}}, \mu) = \\ &= 2\frac{2}{1}\frac{1}{2}\frac{1}{2}\frac{1}{4}\{FD_1(\pi_{u1}, \pi_{u2}, \Pi_o, w_{\dot{L}}, \mu) + FD_2(\pi_{u1}, \pi_{u2}, \Pi_o, w_{\dot{L}}, \mu) + \\ &+ FD_3(\pi_{u1}, \pi_{u2}, \Pi_o, w_{\dot{L}}, \mu) + FD_4(\pi_{u1}, \pi_{u2}, \Pi_o, w_{\dot{L}}, \mu)\} = \\ &= 2\frac{2}{1}\frac{1}{2}\frac{1}{2}\frac{1}{4}\{0+0+0+0\} = 0\end{aligned}$$

Note that we avoided to calculate both $FD_i(a, b, \Pi_o, w_{\dot{L}}, \mu)$ and $FD_i(b, a, \Pi_o, w_{\dot{L}}, \mu)$, since they provide the same result, by calculating only $FD_i(a, b, \Pi_o, w_{\dot{L}}, \mu)$ and multiplying the result by 2.

So, for every Π_o we obtain the same result:

$$\forall \Pi_o : FD(\Pi_e, w_{\Pi_e}, \Pi_o, w_{\dot{L}}, \mu, w_S) = 0$$

Therefore, predator-prey has $Right_{min} = 0$ for this property. □

C.4 Strict Total Grading

We arrive to the strict total grading (STG) property. As given in section 7.5.2, we want to know if there exists a strict ordering between the evaluated agents when interacting in the multi-agent environment. We use an average of rewards as the utility function to calculate an agent's result.

To simplify the notation, we use the next table to represent the STO: $R_i(\mu[\dot{l} \overset{i,j}{\leftarrow} \pi_1, \pi_2]) < R_j(\mu[\dot{l} \overset{i,j}{\leftarrow} \pi_1, \pi_2])$, $R_i(\mu[\dot{l} \overset{i,j}{\leftarrow} \pi_2, \pi_3]) < R_j(\mu[\dot{l} \overset{i,j}{\leftarrow} \pi_2, \pi_3])$ and $R_i(\mu[\dot{l} \overset{i,j}{\leftarrow} \pi_1, \pi_3]) < R_j(\mu[\dot{l} \overset{i,j}{\leftarrow} \pi_1, \pi_3])$.

AS i		AS j
π_1	$<$	π_2
π_2	$<$	π_3
π_1	$<$	π_3

where AS stands for agent slot.

Proposition 58. *$General_{min}$ for the strict total grading (STG) property is equal to 0 for the predator-prey environment.*

Proof. To find $General_{min}$ (equation 8.1), we need to find a pair $\langle \Pi_e, \Pi_o \rangle$ that minimises the property value as much as possible. We can have this situation by selecting $\Pi_e = \{\pi_{s1}, \pi_{s2}, \pi_{s3}\}$ and $\Pi_o = \{\pi_x\}$ (a π_s agent always stays in the same cell[6], and a π_x agent acts stochastically

[6]Note that every cell has an action that leads to a block or a boundary, therefore an agent performing this action stays at its current cell.

with a probability of $1/\sqrt{2}$ to do not reach the upper left corner and a probability of $1 - 1/\sqrt{2}$ to reach this corner).

Following definition 30 we obtain the STG value for this $\langle \Pi_e, \Pi_o \rangle$. Since the multi-agent environment is not team symmetric, we need to calculate this property value for every pair of agent slots. Also, since $|\Pi_e| = 3$ we just need to calculate this property value for one trio of evaluated agents. Following definition 29 we calculate the STG value for the evaluated agents $\pi_{s1}, \pi_{s2}, \pi_{s3} \in \Pi_e$ and each pair of agent slots. We start with agent slots 1 and 2:

$$STG_{1,2}(\pi_{s1}, \pi_{s2}, \pi_{s3}, \Pi_o, w_{\dot{L}}, \mu) = \sum_{\dot{l} \in \dot{L}_{-1,2}^{N(\mu)}(\Pi_o)} w_{\dot{L}}(\dot{l}) STO_{1,2}(\pi_{s1}, \pi_{s2}, \pi_{s3}, \dot{l}, \mu) =$$

$$= \frac{1}{1} STO_{1,2}(\pi_{s1}, \pi_{s2}, \pi_{s3}, (*, *, \pi_x, \pi_x), \mu)$$

The following table shows us $STO_{1,2}$ for all the permutations of $\pi_{s1}, \pi_{s2}, \pi_{s3}$.

AS 1		AS 2	AS 1		AS 2	AS 1		AS 2
π_{s1}	$<$	π_{s2}	π_{s1}	$<$	π_{s3}	π_{s2}	$<$	π_{s1}
π_{s2}	$<$	π_{s3}	π_{s3}	$<$	π_{s2}	π_{s1}	$<$	π_{s3}
π_{s1}	$<$	π_{s3}	π_{s1}	$<$	π_{s2}	π_{s2}	$<$	π_{s3}
AS 1		AS 2	AS 1		AS 2	AS 1		AS 2
π_{s2}	$<$	π_{s3}	π_{s3}	$<$	π_{s1}	π_{s3}	$<$	π_{s2}
π_{s3}	$<$	π_{s1}	π_{s1}	$<$	π_{s2}	π_{s2}	$<$	π_{s1}
π_{s2}	$<$	π_{s1}	π_{s3}	$<$	π_{s2}	π_{s3}	$<$	π_{s1}

It is not possible to find a STO, since for every permutation the agents in agent slots 3 (π_x) and 4 (π_x) have a probability of $(1/\sqrt{2}) \times (1/\sqrt{2}) = 1/2$ to do not chase the prey (any π_s) and the same probability $1 - 1/2 = 1/2$ to chase the prey (any π_s), making the agents in agent slots 1 (any π_s) and 2 (any π_s) to obtain the same expected average reward (0). So:

$$STG_{1,2}(\pi_{s1}, \pi_{s2}, \pi_{s3}, \Pi_o, w_{\dot{L}}, \mu) = \frac{1}{1} 0 = 0$$

For agent slots 1 and 3, the following table shows us $STO_{1,3}$ for all the permutations of $\pi_{s1}, \pi_{s2}, \pi_{s3}$.

AS 1		AS 3	AS 1		AS 3	AS 1		AS 3
π_{s1}	$<$	π_{s2}	π_{s1}	$<$	π_{s3}	π_{s2}	$<$	π_{s1}
π_{s2}	$<$	π_{s3}	π_{s3}	$<$	π_{s2}	π_{s1}	$<$	π_{s3}
π_{s1}	$<$	π_{s3}	π_{s1}	$<$	π_{s2}	π_{s2}	$<$	π_{s3}
AS 1		AS 3	AS 1		AS 3	AS 1		AS 3
π_{s2}	$<$	π_{s3}	π_{s3}	$<$	π_{s1}	π_{s3}	$<$	π_{s2}
π_{s3}	$<$	π_{s1}	π_{s1}	$<$	π_{s2}	π_{s2}	$<$	π_{s1}
π_{s2}	$<$	π_{s1}	π_{s3}	$<$	π_{s2}	π_{s3}	$<$	π_{s1}

It is also not possible to find a STO for any permutation, making again the agents in agent slots 1 (any π_s) and 3 (any π_s) to obtain the same expected average reward (0). So:

$$\begin{aligned} STG_{1,3}(\pi_{s1}, \pi_{s2}, \pi_{s3}, \Pi_o, w_{\dot{L}}, \mu) &= \sum_{\dot{l} \in \dot{L}_{-1,3}^{N(\mu)}(\Pi_o)} w_{\dot{L}}(\dot{l}) STO_{1,3}(\pi_{s1}, \pi_{s2}, \pi_{s3}, \dot{l}, \mu) = \\ &= \frac{1}{1} STO_{1,3}(\pi_{s1}, \pi_{s2}, \pi_{s3}, (*, \pi_x, *, \pi_x), \mu) = \\ &= \frac{1}{1} 0 = 0 \end{aligned}$$

For agent slots 1 and 4, the following table shows us $STO_{1,4}$ for all the permutations of $\pi_{s1}, \pi_{s2}, \pi_{s3}$.

AS 1		AS 4	AS 1		AS 4	AS 1		AS 4
π_{s1}	$<$	π_{s2}	π_{s1}	$<$	π_{s3}	π_{s2}	$<$	π_{s1}
π_{s2}	$<$	π_{s3}	π_{s3}	$<$	π_{s2}	π_{s1}	$<$	π_{s3}
π_{s1}	$<$	π_{s3}	π_{s1}	$<$	π_{s2}	π_{s2}	$<$	π_{s3}
AS 1		AS 4	AS 1		AS 4	AS 1		AS 4
π_{s2}	$<$	π_{s3}	π_{s3}	$<$	π_{s1}	π_{s3}	$<$	π_{s2}
π_{s3}	$<$	π_{s1}	π_{s1}	$<$	π_{s2}	π_{s2}	$<$	π_{s1}
π_{s2}	$<$	π_{s1}	π_{s3}	$<$	π_{s2}	π_{s3}	$<$	π_{s1}

It is also not possible to find a STO for any permutation, making again the agents in agent slots 1 (any π_s) and 4 (any π_s) to obtain the same expected average reward (0). So:

$$\begin{aligned} STG_{1,4}(\pi_{s1}, \pi_{s2}, \pi_{s3}, \Pi_o, w_{\dot{L}}, \mu) &= \sum_{\dot{l} \in \dot{L}_{-1,4}^{N(\mu)}(\Pi_o)} w_{\dot{L}}(\dot{l}) STO_{1,4}(\pi_{s1}, \pi_{s2}, \pi_{s3}, \dot{l}, \mu) = \\ &= \frac{1}{1} STO_{1,4}(\pi_{s1}, \pi_{s2}, \pi_{s3}, (*, \pi_x, \pi_x, *), \mu) = \\ &= \frac{1}{1} 0 = 0 \end{aligned}$$

For agent slots 2 and 1, the following table shows us $STO_{2,1}$ for all the permutations of $\pi_{s1}, \pi_{s2}, \pi_{s3}$.

AS 2		AS 1	AS 2		AS 1	AS 2		AS 1
π_{s1}	$<$	π_{s2}	π_{s1}	$<$	π_{s3}	π_{s2}	$<$	π_{s1}
π_{s2}	$<$	π_{s3}	π_{s3}	$<$	π_{s2}	π_{s1}	$<$	π_{s3}
π_{s1}	$<$	π_{s3}	π_{s1}	$<$	π_{s2}	π_{s2}	$<$	π_{s3}
AS 2		AS 1	AS 2		AS 1	AS 2		AS 1
π_{s2}	$<$	π_{s3}	π_{s3}	$<$	π_{s1}	π_{s3}	$<$	π_{s2}
π_{s3}	$<$	π_{s1}	π_{s1}	$<$	π_{s2}	π_{s2}	$<$	π_{s1}
π_{s2}	$<$	π_{s1}	π_{s3}	$<$	π_{s2}	π_{s3}	$<$	π_{s1}

It is also not possible to find a STO for any permutation, making again the agents in agent slots 2 (any π_s) and 1 (any π_s) to obtain the same expected average reward (0). So:

$$
\begin{aligned}
STG_{2,1}(\pi_{s1}, \pi_{s2}, \pi_{s3}, \Pi_o, w_{\dot{L}}, \mu) &= \sum_{\dot{l} \in \dot{L}_{-2,1}^{N(\mu)}(\Pi_o)} w_{\dot{L}}(\dot{l}) STO_{2,1}(\pi_{s1}, \pi_{s2}, \pi_{s3}, \dot{l}, \mu) = \\
&= \frac{1}{1} STO_{2,1}(\pi_{s1}, \pi_{s2}, \pi_{s3}, (*, *, \pi_x, \pi_x), \mu) = \\
&= \frac{1}{1} 0 = 0
\end{aligned}
$$

For agent slots 2 and 3, it is not possible to find a STO, since both agents obtain the same expected average reward due they are in the same team. So:

$$
\begin{aligned}
STG_{2,3}(\pi_{s1}, \pi_{s2}, \pi_{s3}, \Pi_o, w_{\dot{L}}, \mu) &= \sum_{\dot{l} \in \dot{L}_{-2,3}^{N(\mu)}(\Pi_o)} w_{\dot{L}}(\dot{l}) STO_{2,3}(\pi_{s1}, \pi_{s2}, \pi_{s3}, \dot{l}, \mu) = \\
&= \frac{1}{1} STO_{2,3}(\pi_{s1}, \pi_{s2}, \pi_{s3}, (\pi_x, *, *, \pi_x), \mu) = \\
&= \frac{1}{1} 0 = 0
\end{aligned}
$$

For agent slots 2 and 4, it is also not possible to find a STO, since again both agents obtain the same expected average reward due they are in the same team. So:

$$
\begin{aligned}
STG_{2,4}(\pi_{s1}, \pi_{s2}, \pi_{s3}, \Pi_o, w_{\dot{L}}, \mu) &= \sum_{\dot{l} \in \dot{L}_{-2,4}^{N(\mu)}(\Pi_o)} w_{\dot{L}}(\dot{l}) STO_{2,4}(\pi_{s1}, \pi_{s2}, \pi_{s3}, \dot{l}, \mu) = \\
&= \frac{1}{1} STO_{2,4}(\pi_{s1}, \pi_{s2}, \pi_{s3}, (\pi_x, *, \pi_x, *), \mu) = \\
&= \frac{1}{1} 0 = 0
\end{aligned}
$$

For agent slots 3 and 1, the following table shows us $STO_{3,1}$ for all the permutations of $\pi_{s1}, \pi_{s2}, \pi_{s3}$.

AS 3		AS 1	AS 3		AS 1	AS 3		AS 1
π_{s1}	$<$	π_{s2}	π_{s1}	$<$	π_{s3}	π_{s2}	$<$	π_{s1}
π_{s2}	$<$	π_{s3}	π_{s3}	$<$	π_{s2}	π_{s1}	$<$	π_{s3}
π_{s1}	$<$	π_{s3}	π_{s1}	$<$	π_{s2}	π_{s2}	$<$	π_{s3}
AS 3		AS 1	AS 3		AS 1	AS 3		AS 1
π_{s2}	$<$	π_{s3}	π_{s3}	$<$	π_{s1}	π_{s3}	$<$	π_{s2}
π_{s3}	$<$	π_{s1}	π_{s1}	$<$	π_{s2}	π_{s2}	$<$	π_{s1}
π_{s2}	$<$	π_{s1}	π_{s3}	$<$	π_{s2}	π_{s3}	$<$	π_{s1}

It is also not possible to find a STO for any permutation, making the agents in agent slots 3 (any π_s) and 1 (any π_s) to obtain the same expected average reward (0). So:

$$
\begin{aligned}
STG_{3,1}(\pi_{s1}, \pi_{s2}, \pi_{s3}, \Pi_o, w_{\dot{L}}, \mu) &= \sum_{\dot{l} \in \dot{L}_{-3,1}^{N(\mu)}(\Pi_o)} w_{\dot{L}}(\dot{l}) STO_{3,1}(\pi_{s1}, \pi_{s2}, \pi_{s3}, \dot{l}, \mu) = \\
&= \frac{1}{1} STO_{3,1}(\pi_{s1}, \pi_{s2}, \pi_{s3}, (*, \pi_x, *, \pi_x), \mu) = \\
&= \frac{1}{1} 0 = 0
\end{aligned}
$$

For agent slots 3 and 2, it is also not possible to find a STO, since both agents obtain the same expected average reward due they are in the same team. So:

$$\begin{aligned} STG_{3,2}(\pi_{s1},\pi_{s2},\pi_{s3},\Pi_o,w_{\dot{L}},\mu) &= \sum_{\dot{l}\in\dot{L}^{N(\mu)}_{-3,2}(\Pi_o)} w_{\dot{L}}(\dot{l})STO_{3,2}(\pi_{s1},\pi_{s2},\pi_{s3},\dot{l},\mu) = \\ &= \frac{1}{1}STO_{3,2}(\pi_{s1},\pi_{s2},\pi_{s3},(\pi_x,*,*,\pi_x),\mu) = \\ &= \frac{1}{1}0 = 0 \end{aligned}$$

For agent slots 3 and 4, it is also not possible to find a STO, since again both agents obtain the same expected average reward due they are in the same team. So:

$$\begin{aligned} STG_{3,4}(\pi_{s1},\pi_{s2},\pi_{s3},\Pi_o,w_{\dot{L}},\mu) &= \sum_{\dot{l}\in\dot{L}^{N(\mu)}_{-3,4}(\Pi_o)} w_{\dot{L}}(\dot{l})STO_{3,4}(\pi_{s1},\pi_{s2},\pi_{s3},\dot{l},\mu) = \\ &= \frac{1}{1}STO_{3,4}(\pi_{s1},\pi_{s2},\pi_{s3},(\pi_x,\pi_x,*,*),\mu) = \\ &= \frac{1}{1}0 = 0 \end{aligned}$$

For agent slots 4 and 1, the following table shows us $STO_{4,1}$ for all the permutations of $\pi_{s1},\pi_{s2},\pi_{s3}$.

AS 4		AS 1	AS 4		AS 1	AS 4		AS 1
π_{s1}	$<$	π_{s2}	π_{s1}	$<$	π_{s3}	π_{s2}	$<$	π_{s1}
π_{s2}	$<$	π_{s3}	π_{s3}	$<$	π_{s2}	π_{s1}	$<$	π_{s3}
π_{s1}	$<$	π_{s3}	π_{s1}	$<$	π_{s2}	π_{s2}	$<$	π_{s3}
AS 4		AS 1	AS 4		AS 1	AS 4		AS 1
π_{s2}	$<$	π_{s3}	π_{s3}	$<$	π_{s1}	π_{s3}	$<$	π_{s2}
π_{s3}	$<$	π_{s1}	π_{s1}	$<$	π_{s2}	π_{s2}	$<$	π_{s1}
π_{s2}	$<$	π_{s1}	π_{s3}	$<$	π_{s2}	π_{s3}	$<$	π_{s1}

It is also not possible to find a STO for any permutation, making the agents in agent slots 4 (any π_s) and 1 (any π_s) to obtain the same expected average reward (0). So:

$$\begin{aligned} STG_{4,1}(\pi_{s1},\pi_{s2},\pi_{s3},\Pi_o,w_{\dot{L}},\mu) &= \sum_{\dot{l}\in\dot{L}^{N(\mu)}_{-4,1}(\Pi_o)} w_{\dot{L}}(\dot{l})STO_{4,1}(\pi_{s1},\pi_{s2},\pi_{s3},\dot{l},\mu) = \\ &= \frac{1}{1}STO_{4,1}(\pi_{s1},\pi_{s2},\pi_{s3},(*,\pi_x,\pi_x,*),\mu) = \\ &= \frac{1}{1}0 = 0 \end{aligned}$$

For agent slots 4 and 2, it is also not possible to find a STO, since both agents obtain the same expected average reward due they are in the same team. So:

$$\begin{aligned}STG_{4,2}(\pi_{s1},\pi_{s2},\pi_{s3},\Pi_o,w_{\dot{L}},\mu) &= \sum_{\dot{l}\in\dot{L}^{N(\mu)}_{-4,2}(\Pi_o)} w_{\dot{L}}(\dot{l})STO_{4,2}(\pi_{s1},\pi_{s2},\pi_{s3},\dot{l},\mu) =\\ &= \frac{1}{1}STO_{4,2}(\pi_{s1},\pi_{s2},\pi_{s3},(\pi_x,*,\pi_x,*),\mu) =\\ &= \frac{1}{1}0 = 0\end{aligned}$$

And for agent slots 4 and 3, it is also not possible to find a STO, since again both agents obtain the same expected average reward due they are in the same team. So:

$$\begin{aligned}STG_{4,3}(\pi_{s1},\pi_{s2},\pi_{s3},\Pi_o,w_{\dot{L}},\mu) &= \sum_{\dot{l}\in\dot{L}^{N(\mu)}_{-4,3}(\Pi_o)} w_{\dot{L}}(\dot{l})STO_{4,3}(\pi_{s1},\pi_{s2},\pi_{s3},\dot{l},\mu) =\\ &= \frac{1}{1}STO_{4,3}(\pi_{s1},\pi_{s2},\pi_{s3},(\pi_x,\pi_x,*,*),\mu) =\\ &= \frac{1}{1}0 = 0\end{aligned}$$

And finally, we calculate the STG value:

$$\begin{aligned}STG(\Pi_e,w_{\Pi_e},\Pi_o,w_{\dot{L}},\mu,w_S) &= \eta_{\Pi^3}\sum_{\pi_1,\pi_2,\pi_3\in\Pi_e|\pi_1\neq\pi_2\neq\pi_3} w_{\Pi_e}(\pi_1)w_{\Pi_e}(\pi_2)w_{\Pi_e}(\pi_3)\eta_{S^2_1}\sum_{i=1}^{N(\mu)} w_S(i,\mu)\times\\ \times\left(\sum_{j=1}^{i-1} w_S(j,\mu)STG_{i,j}(\pi_1,\pi_2,\pi_3,\Pi_o,w_{\dot{L}},\mu) + \sum_{j=i+1}^{N(\mu)} w_S(j,\mu)STG_{i,j}(\pi_1,\pi_2,\pi_3,\Pi_o,w_{\dot{L}},\mu)\right) &=\\ &= 6\frac{9}{2}\frac{1}{3}\frac{1}{3}\frac{1}{3}\frac{4}{3}\frac{1}{4}\frac{1}{4}\{STG_{1,2}(\pi_{s1},\pi_{s2},\pi_{s3},\Pi_o,w_{\dot{L}},\mu)+\\ &+ STG_{1,3}(\pi_{s1},\pi_{s2},\pi_{s3},\Pi_o,w_{\dot{L}},\mu)+\\ &+ STG_{1,4}(\pi_{s1},\pi_{s2},\pi_{s3},\Pi_o,w_{\dot{L}},\mu)+\\ &+ STG_{2,1}(\pi_{s1},\pi_{s2},\pi_{s3},\Pi_o,w_{\dot{L}},\mu)+\\ &+ STG_{2,3}(\pi_{s1},\pi_{s2},\pi_{s3},\Pi_o,w_{\dot{L}},\mu)+\\ &+ STG_{2,4}(\pi_{s1},\pi_{s2},\pi_{s3},\Pi_o,w_{\dot{L}},\mu)+\\ &+ STG_{3,1}(\pi_{s1},\pi_{s2},\pi_{s3},\Pi_o,w_{\dot{L}},\mu)+\\ &+ STG_{3,2}(\pi_{s1},\pi_{s2},\pi_{s3},\Pi_o,w_{\dot{L}},\mu)+\\ &+ STG_{3,4}(\pi_{s1},\pi_{s2},\pi_{s3},\Pi_o,w_{\dot{L}},\mu)+\\ &+ STG_{4,1}(\pi_{s1},\pi_{s2},\pi_{s3},\Pi_o,w_{\dot{L}},\mu)+\\ &+ STG_{4,2}(\pi_{s1},\pi_{s2},\pi_{s3},\Pi_o,w_{\dot{L}},\mu)+\\ &+ STG_{4,3}(\pi_{s1},\pi_{s2},\pi_{s3},\Pi_o,w_{\dot{L}},\mu)\} =\\ &= 6\frac{9}{2}\frac{1}{3}\frac{1}{3}\frac{1}{3}\frac{4}{3}\frac{1}{4}\frac{1}{4}\{12\times 0\} = 0\end{aligned}$$

Note that we avoided to calculate all the permutations of π_1,π_2,π_3 for $STG_{i,j}(\pi_1,\pi_2,\pi_3,\Pi_o,w_{\dot{L}},\mu)$, since they provide the same result, by calculating only one permutation and multiplying the result by 6 (the number of permutations).

Since 0 is the lowest possible value for the strict total grading property, therefore predator-prey has $General_{min} = 0$ for this property. □

Proposition 59. *$General_{max}$ for the strict total grading (STG) property is equal to $\frac{1}{2}$ for the predator-prey environment.*

Proof. To find $General_{max}$ (equation 8.2), we need to find a pair $\langle \Pi_e, \Pi_o \rangle$ that maximises the property value as much as possible. We can have this situation by selecting $\Pi_e = \{\pi_x, \pi_y, \pi_z\}$ and $\Pi_o = \{\pi_s\}$ (a π_s agent always stays in the same cell[7] and π_x, π_y and π_z behave as shown in figures C.2, C.3 and C.4 respectively).

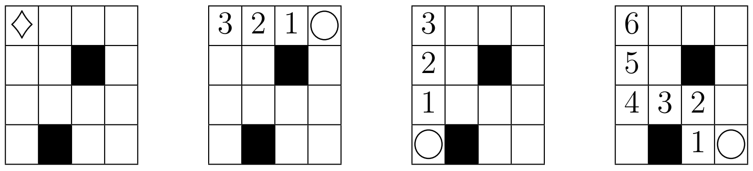


Figure C.2: Behaviour of π_x when interacting on each of the agent slots. Numbers represent the position of π_x on each time step.

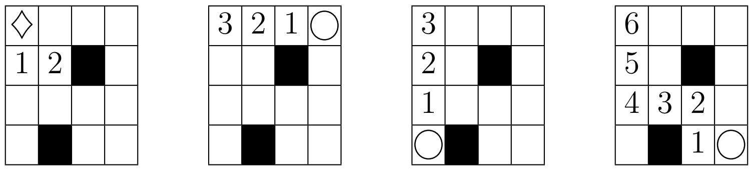


Figure C.3: Behaviour of π_y when interacting on each of the agent slots. Numbers represent the position of π_y on each time step.

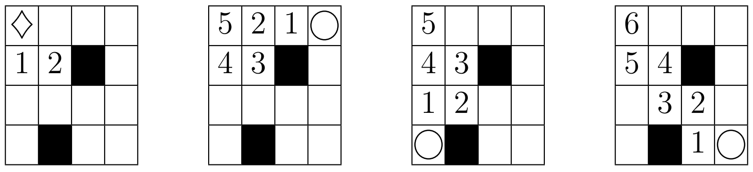


Figure C.4: Behaviour of π_z when interacting on each of the agent slots. Numbers represent the position of π_z on each time step.

Following definition 30 we obtain the STG value for this $\langle \Pi_e, \Pi_o \rangle$. Since the multi-agent environment is not team symmetric, we need to calculate this property value for every pair of agent slots. Also, since $|\Pi_e| = 3$ we just need to calculate this property value for one trio of evaluated agents. Following definition 29 we calculate the STG value for the evaluated agents $\pi_x, \pi_y, \pi_z \in \Pi_e$ and each pair of agent slots. We start with agent slots 1 and 2:

$$STG_{1,2}(\pi_x, \pi_y, \pi_z, \Pi_o, w_{\dot{L}}, \mu) = \sum_{\dot{l} \in \dot{L}_{-1,2}^{N(\mu)}(\Pi_o)} w_{\dot{L}}(\dot{l}) STO_{1,2}(\pi_x, \pi_y, \pi_z, \dot{l}, \mu) =$$

$$= \frac{1}{1} STO_{1,2}(\pi_x, \pi_y, \pi_z, (*, *, \pi_s, \pi_s), \mu)$$

[7]Note that every cell has an action that leads to a block or a boundary, therefore an agent performing this action stays at its current cell.

The following table shows us $STO_{1,2}$ for all the permutations of π_x, π_y, π_z.

AS 1		AS 2	AS 1		AS 2	AS 1		AS 2
π_x	$<$	π_y	π_x	$<$	π_z	π_y	$<$	π_x
π_y	$<$	π_z	π_z	$<$	π_y	π_x	$<$	π_z
π_x	$<$	π_z	π_x	$<$	π_y	π_y	$<$	π_z
AS 1		AS 2	AS 1		AS 2	AS 1		AS 2
π_y	$<$	π_z	π_z	$<$	π_x	π_z	$<$	π_y
π_z	$<$	π_x	π_x	$<$	π_y	π_y	$<$	π_x
π_y	$<$	π_x	π_z	$<$	π_y	π_z	$<$	π_x

It is possible to find a STO for the first permutation. In agent line-up $(\pi_x, \pi_y, \pi_s, \pi_s)$, π_x stays at the upper left corner and π_y chases it in that cell in the 3rd time step, obtaining the expected average rewards of -1 and 1 respectively. In agent line-up $(\pi_y, \pi_z, \pi_s, \pi_s)$, π_y reaches the 2nd row 2nd column cell and π_z chases it in that cell in the 3rd time step, obtaining the expected average rewards of -1 and 1 respectively. In agent line-up $(\pi_x, \pi_z, \pi_s, \pi_s)$, π_x stays at the upper left corner and π_z chases it in that cell in the 5th time step, obtaining the expected average rewards of -1 and 1 respectively. So:

$$STG_{1,2}(\pi_x, \pi_y, \pi_z, \Pi_o, w_{\dot{L}}, \mu) = \frac{1}{1}1 = 1$$

For agent slots 1 and 3, the following table shows us $STO_{1,3}$ for all the permutations of π_x, π_y, π_z.

AS 1		AS 3	AS 1		AS 3	AS 1		AS 3
π_x	$<$	π_y	π_x	$<$	π_z	π_y	$<$	π_x
π_y	$<$	π_z	π_z	$<$	π_y	π_x	$<$	π_z
π_x	$<$	π_z	π_x	$<$	π_y	π_y	$<$	π_z
AS 1		AS 3	AS 1		AS 3	AS 1		AS 3
π_y	$<$	π_z	π_z	$<$	π_x	π_z	$<$	π_y
π_z	$<$	π_x	π_x	$<$	π_y	π_y	$<$	π_x
π_y	$<$	π_x	π_z	$<$	π_y	π_z	$<$	π_x

It is also possible to find a STO for the first permutation. In agent line-up $(\pi_x, \pi_s, \pi_y, \pi_s)$, π_x stays at the upper left corner and π_y chases it in that cell in the 3rd time step, obtaining the expected average rewards of -1 and 1 respectively. In agent line-up $(\pi_y, \pi_s, \pi_z, \pi_s)$, π_y reaches the 2nd row 2nd column cell and π_z chases it in that cell in the 3rd time step, obtaining the expected average rewards of -1 and 1 respectively. In agent line-up $(\pi_x, \pi_s, \pi_z, \pi_s)$, π_x stays at the upper left corner and π_z chases it in that cell in the 5th time step, obtaining the expected average rewards of -1 and 1 respectively. So:

$$\begin{aligned} STG_{1,3}(\pi_x, \pi_y, \pi_z, \Pi_o, w_{\dot{L}}, \mu) &= \sum_{\dot{l} \in \dot{L}^{N(\mu)}_{-1,3}(\Pi_o)} w_{\dot{L}}(\dot{l}) STO_{1,3}(\pi_x, \pi_y, \pi_z, \dot{l}, \mu) = \\ &= \frac{1}{1} STO_{1,3}(\pi_x, \pi_y, \pi_z, (*, \pi_s, *, \pi_s), \mu) = \\ &= \frac{1}{1}1 = 1 \end{aligned}$$

For agent slots 1 and 4, the following table shows us $STO_{1,4}$ for all the permutations of π_x, π_y, π_z.

AS 1		AS 4	AS 1		AS 4	AS 1		AS 4
π_x	$<$	π_y	π_x	$<$	π_z	π_y	$<$	π_x
π_y	$<$	π_z	π_z	$<$	π_y	π_x	$<$	π_z
π_x	$<$	π_z	π_x	$<$	π_y	π_y	$<$	π_z
AS 1		AS 4	AS 1		AS 4	AS 1		AS 4
π_y	$<$	π_z	π_z	$<$	π_x	π_z	$<$	π_y
π_z	$<$	π_x	π_x	$<$	π_y	π_y	$<$	π_x
π_y	$<$	π_x	π_z	$<$	π_y	π_z	$<$	π_x

It is also possible to find a STO for the first permutation. In agent line-up $(\pi_x, \pi_s, \pi_s, \pi_y)$, π_x stays at the upper left corner and π_y chases it in that cell in the 6th time step, obtaining the expected average rewards of -1 and 1 respectively. In agent line-up $(\pi_y, \pi_s, \pi_s, \pi_z)$, π_y reaches the 2nd row 2nd column cell and π_z chases it in that cell in the 4th time step, obtaining the expected average rewards of -1 and 1 respectively. In agent line-up $(\pi_x, \pi_s, \pi_s, \pi_z)$, π_x stays at the upper left corner and π_z chases it in that cell in the 6th time step, obtaining the expected average rewards of -1 and 1 respectively. So:

$$
\begin{aligned}
STG_{1,4}(\pi_x, \pi_y, \pi_z, \Pi_o, w_{\dot{L}}, \mu) &= \sum_{\dot{l} \in \dot{L}_{-1,4}^{N(\mu)}(\Pi_o)} w_{\dot{L}}(\dot{l}) STO_{1,4}(\pi_x, \pi_y, \pi_z, \dot{l}, \mu) = \\
&= \frac{1}{1} STO_{1,4}(\pi_x, \pi_y, \pi_z, (*, \pi_s, \pi_s, *), \mu) = \\
&= \frac{1}{1} 1 = 1
\end{aligned}
$$

For agent slots 2 and 1, the following table shows us $STO_{2,1}$ for all the permutations of π_x, π_y, π_z.

AS 2		AS 1	AS 2		AS 1	AS 2		AS 1
π_x	$<$	π_y	π_x	$<$	π_z	π_y	$<$	π_x
π_y	$<$	π_z	π_z	$<$	π_y	π_x	$<$	π_z
π_x	$<$	π_z	π_x	$<$	π_y	π_y	$<$	π_z
AS 2		AS 1	AS 2		AS 1	AS 2		AS 1
π_y	$<$	π_z	π_z	$<$	π_x	π_z	$<$	π_y
π_z	$<$	π_x	π_x	$<$	π_y	π_y	$<$	π_x
π_y	$<$	π_x	π_z	$<$	π_y	π_z	$<$	π_x

It is also possible to find a STO for the first permutation. In agent line-up $(\pi_y, \pi_x, \pi_s, \pi_s)$, π_y reaches the 2nd row 2nd column cell and π_x never chases it in that cell, obtaining the expected average rewards of 1 and -1 respectively. In agent line-up $(\pi_z, \pi_y, \pi_s, \pi_s)$, π_z reaches the 2nd row 2nd column cell and π_y never chases it in that cell, obtaining the expected average rewards of 1 and -1 respectively. In agent line-up $(\pi_z, \pi_x, \pi_s, \pi_s)$, π_z reaches the 2nd row 2nd column cell and π_x never chases it in that cell, obtaining the expected average rewards of 1 and -1 respectively. So:

$$\begin{aligned}
STG_{2,1}(\pi_x, \pi_y, \pi_z, \Pi_o, w_{\dot{L}}, \mu) &= \sum_{\dot{l} \in \dot{L}_{-2,1}^{N(\mu)}(\Pi_o)} w_{\dot{L}}(\dot{l}) STO_{2,1}(\pi_x, \pi_y, \pi_z, \dot{l}, \mu) = \\
&= \frac{1}{1} STO_{2,1}(\pi_x, \pi_y, \pi_z, (*, *, \pi_s, \pi_s), \mu) = \\
&= \frac{1}{1} 1 = 1
\end{aligned}$$

For agent slots 2 and 3, it is not possible to find a STO, since both agents obtain the same expected average reward due they are in the same team. So:

$$\begin{aligned}
STG_{2,3}(\pi_x, \pi_y, \pi_z, \Pi_o, w_{\dot{L}}, \mu) &= \sum_{\dot{l} \in \dot{L}_{-2,3}^{N(\mu)}(\Pi_o)} w_{\dot{L}}(\dot{l}) STO_{2,3}(\pi_x, \pi_y, \pi_z, \dot{l}, \mu) = \\
&= \frac{1}{1} STO_{2,3}(\pi_x, \pi_y, \pi_z, (\pi_s, *, *, \pi_s), \mu) = \\
&= \frac{1}{1} 0 = 0
\end{aligned}$$

For agent slots 2 and 4, it is also not possible to find a STO, since again both agents obtain the same expected average reward due they are in the same team. So:

$$\begin{aligned}
STG_{2,4}(\pi_x, \pi_y, \pi_z, \Pi_o, w_{\dot{L}}, \mu) &= \sum_{\dot{l} \in \dot{L}_{-2,4}^{N(\mu)}(\Pi_o)} w_{\dot{L}}(\dot{l}) STO_{2,4}(\pi_x, \pi_y, \pi_z, \dot{l}, \mu) = \\
&= \frac{1}{1} STO_{2,4}(\pi_x, \pi_y, \pi_z, (\pi_s, *, \pi_s, *), \mu) = \\
&= \frac{1}{1} 0 = 0
\end{aligned}$$

For agent slots 3 and 1, the following table shows us $STO_{3,1}$ for all the permutations of π_x, π_y, π_z.

AS 3		AS 1	AS 3		AS 1	AS 3		AS 1
π_x	$<$	π_y	π_x	$<$	π_z	π_y	$<$	π_x
π_y	$<$	π_z	π_z	$<$	π_y	π_x	$<$	π_z
π_x	$<$	π_z	π_x	$<$	π_y	π_y	$<$	π_z
AS 3		AS 1	AS 3		AS 1	AS 3		AS 1
π_y	$<$	π_z	π_z	$<$	π_x	π_z	$<$	π_y
π_z	$<$	π_x	π_x	$<$	π_y	π_y	$<$	π_x
π_y	$<$	π_x	π_z	$<$	π_y	π_z	$<$	π_x

It is possible to find a STO for the first permutation. In agent line-up $(\pi_y, \pi_s, \pi_x, \pi_s)$, π_y reaches the 2nd row 2nd column cell and π_x never chases it in that cell, obtaining the expected average rewards of 1 and -1 respectively. In agent line-up $(\pi_z, \pi_s, \pi_y, \pi_s)$, π_z reaches the 2nd row 2nd column cell and π_y never chases it in that cell, obtaining the expected average rewards of 1 and -1 respectively. In agent line-up $(\pi_z, \pi_s, \pi_x, \pi_s)$, π_z reaches the 2nd row 2nd column cell and π_x never chases it in that cell, obtaining the expected average rewards of 1 and -1 respectively. So:

$$\begin{aligned}
STG_{3,1}(\pi_x,\pi_y,\pi_z,\Pi_o,w_{\dot{L}},\mu) &= \sum_{\dot{l}\in\dot{L}_{-3,1}^{N(\mu)}(\Pi_o)} w_{\dot{L}}(\dot{l})STO_{3,1}(\pi_x,\pi_y,\pi_z,\dot{l},\mu) = \\
&= \frac{1}{1}STO_{3,1}(\pi_x,\pi_y,\pi_z,(*,\pi_s,*,\pi_s),\mu) = \\
&= \frac{1}{1}1 = 1
\end{aligned}$$

For agent slots 3 and 2, it is not possible to find a STO, since again both agents obtain the same expected average reward due they are in the same team. So:

$$\begin{aligned}
STG_{3,2}(\pi_x,\pi_y,\pi_z,\Pi_o,w_{\dot{L}},\mu) &= \sum_{\dot{l}\in\dot{L}_{-3,2}^{N(\mu)}(\Pi_o)} w_{\dot{L}}(\dot{l})STO_{3,2}(\pi_x,\pi_y,\pi_z,\dot{l},\mu) = \\
&= \frac{1}{1}STO_{3,2}(\pi_x,\pi_y,\pi_z,(\pi_s,*,*,\pi_s),\mu) = \\
&= \frac{1}{1}0 = 0
\end{aligned}$$

For agent slots 3 and 4, it is also not possible to find a STO, since again both agents obtain the same expected average reward due they are in the same team. So:

$$\begin{aligned}
STG_{3,4}(\pi_x,\pi_y,\pi_z,\Pi_o,w_{\dot{L}},\mu) &= \sum_{\dot{l}\in\dot{L}_{-3,4}^{N(\mu)}(\Pi_o)} w_{\dot{L}}(\dot{l})STO_{3,4}(\pi_x,\pi_y,\pi_z,\dot{l},\mu) = \\
&= \frac{1}{1}STO_{3,4}(\pi_x,\pi_y,\pi_z,(\pi_s,\pi_s,*,*),\mu) = \\
&= \frac{1}{1}0 = 0
\end{aligned}$$

For agent slots 4 and 1, the following table shows us $STO_{4,1}$ for all the permutations of π_x,π_y,π_z.

AS 4		AS 1	AS 4		AS 1	AS 4		AS 1
π_x	$<$	π_y	π_x	$<$	π_z	π_y	$<$	π_x
π_y	$<$	π_z	π_z	$<$	π_y	π_x	$<$	π_z
π_x	$<$	π_z	π_x	$<$	π_y	π_y	$<$	π_z
AS 4		AS 1	AS 4		AS 1	AS 4		AS 1
π_y	$<$	π_z	π_z	$<$	π_x	π_z	$<$	π_y
π_z	$<$	π_x	π_x	$<$	π_y	π_y	$<$	π_x
π_y	$<$	π_x	π_z	$<$	π_y	π_z	$<$	π_x

It is possible to find a STO for the first permutation. In agent line-up $(\pi_y,\pi_s,\pi_s,\pi_x)$, π_y reaches the 2nd row 2nd column cell and π_x never chases it in that cell, obtaining the expected average rewards of 1 and -1 respectively. In agent line-up $(\pi_z,\pi_s,\pi_s,\pi_y)$, π_z reaches the 2nd row 2nd column cell and π_y never chases it in that cell, obtaining the expected average rewards of 1 and -1 respectively. In agent line-up $(\pi_z,\pi_s,\pi_s,\pi_x)$, π_z reaches the 2nd row 2nd column cell and π_x never chases it in that cell, obtaining the expected average rewards of 1 and -1 respectively. So:

$$\begin{aligned}STG_{4,1}(\pi_x,\pi_y,\pi_z,\Pi_o,w_{\dot{L}},\mu) &= \sum_{\dot{l}\in \dot{L}^{N(\mu)}_{-4,1}(\Pi_o)} w_{\dot{L}}(\dot{l})STO_{4,1}(\pi_x,\pi_y,\pi_z,\dot{l},\mu) = \\ &= \frac{1}{1}STO_{4,1}(\pi_x,\pi_y,\pi_z,(*,\pi_s,\pi_s,*),\mu) = \\ &= \frac{1}{1}1 = 1\end{aligned}$$

For agent slots 4 and 2, it is not possible to find a STO, since again both agents obtain the same expected average reward due they are in the same team. So:

$$\begin{aligned}STG_{4,2}(\pi_x,\pi_y,\pi_z,\Pi_o,w_{\dot{L}},\mu) &= \sum_{\dot{l}\in \dot{L}^{N(\mu)}_{-4,2}(\Pi_o)} w_{\dot{L}}(\dot{l})STO_{4,2}(\pi_x,\pi_y,\pi_z,\dot{l},\mu) = \\ &= \frac{1}{1}STO_{4,2}(\pi_x,\pi_y,\pi_z,(\pi_s,*,\pi_s,*),\mu) = \\ &= \frac{1}{1}0 = 0\end{aligned}$$

And for agent slots 4 and 3, it is also not possible to find a STO, since again both agents obtain the same expected average reward due they are in the same team. So:

$$\begin{aligned}STG_{4,3}(\pi_x,\pi_y,\pi_z,\Pi_o,w_{\dot{L}},\mu) &= \sum_{\dot{l}\in \dot{L}^{N(\mu)}_{-4,3}(\Pi_o)} w_{\dot{L}}(\dot{l})STO_{4,3}(\pi_x,\pi_y,\pi_z,\dot{l},\mu) = \\ &= \frac{1}{1}STO_{4,3}(\pi_x,\pi_y,\pi_z,(\pi_s,\pi_s,*,*),\mu) = \\ &= \frac{1}{1}0 = 0\end{aligned}$$

And finally, we calculate the STG value:

$$STG(\Pi_e,w_{\Pi_e},\Pi_o,w_{\dot{L}},\mu,w_S) = \eta_{\Pi^3}\sum_{\pi_1,\pi_2,\pi_3\in\Pi_e|\pi_1\neq\pi_2\neq\pi_3} w_{\Pi_e}(\pi_1)w_{\Pi_e}(\pi_2)w_{\Pi_e}(\pi_3)\eta_{S_1^2}\sum_{i=1}^{N(\mu)} w_S(i,\mu)\times$$

$$\begin{aligned}\times\left(\sum_{j=1}^{i-1} w_S(j,\mu)STG_{i,j}(\pi_1,\pi_2,\pi_3,\Pi_o,w_{\dot{L}},\mu)+\sum_{j=i+1}^{N(\mu)} w_S(j,\mu)STG_{i,j}(\pi_1,\pi_2,\pi_3,\Pi_o,w_{\dot{L}},\mu)\right) &= \\ = 6\frac{9}{2}\frac{1}{3}\frac{1}{3}\frac{1}{3}\frac{4}{3}\frac{1}{4}\frac{1}{4}\times & \\ \times\{STG_{1,2}(\pi_x,\pi_y,\pi_z,\Pi_o,w_{\dot{L}},\mu)+STG_{1,3}(\pi_x,\pi_y,\pi_z,\Pi_o,w_{\dot{L}},\mu)+ & \\ +STG_{1,4}(\pi_x,\pi_y,\pi_z,\Pi_o,w_{\dot{L}},\mu)+STG_{2,1}(\pi_x,\pi_y,\pi_z,\Pi_o,w_{\dot{L}},\mu)+ & \\ +STG_{2,3}(\pi_x,\pi_y,\pi_z,\Pi_o,w_{\dot{L}},\mu)+STG_{2,4}(\pi_x,\pi_y,\pi_z,\Pi_o,w_{\dot{L}},\mu)+ & \\ +STG_{3,1}(\pi_x,\pi_y,\pi_z,\Pi_o,w_{\dot{L}},\mu)+STG_{3,2}(\pi_x,\pi_y,\pi_z,\Pi_o,w_{\dot{L}},\mu)+ & \\ +STG_{3,4}(\pi_x,\pi_y,\pi_z,\Pi_o,w_{\dot{L}},\mu)+STG_{4,1}(\pi_x,\pi_y,\pi_z,\Pi_o,w_{\dot{L}},\mu)+ & \\ +STG_{4,2}(\pi_x,\pi_y,\pi_z,\Pi_o,w_{\dot{L}},\mu)+STG_{4,3}(\pi_x,\pi_y,\pi_z,\Pi_o,w_{\dot{L}},\mu)\} &= \\ = 6\frac{9}{2}\frac{1}{3}\frac{1}{3}\frac{1}{3}\frac{4}{3}\frac{1}{4}\frac{1}{4}\{6\times 0+6\times 1\} = \frac{1}{2} &\end{aligned}$$

Note that we avoided to calculate all the permutations of π_1, π_2, π_3 for $STG_{i,j}(\pi_1, \pi_2, \pi_3, \Pi_o, w_{\dot{L}}, \mu)$, since they provide the same result, by calculating only one permutation and multiplying the result by 6 (the number of permutations).

Since $\frac{1}{2}$ is the highest possible value that we can obtain for the strict total grading property, therefore predator-prey has $General_{max} = \frac{1}{2}$ for this property. □

Approximation 2. *$Left_{max}$ for the strict total grading (STG) property is equal to $\frac{1}{4}$ (as a lower approximation) for the predator-prey environment.*

Proof. To find $Left_{max}$ (equation 8.4), we need to find a Π_e that maximises the property value as much as possible while Π_o minimises it. Using $\Pi_e = \{\pi_{chase1}, \pi_{chase2}, \pi_{chase3}\}$ (a π_{chase} agent tries to be chased when interacting as the prey and tries to perfectly coordinate with the other predators to chase the prey when interacting as a predator) we find a *lower* approximation of this situation no matter which Π_o we use.

Following definition 30 we obtain the STG value for this $\langle \Pi_e, \Pi_o \rangle$ (where Π_o is instantiated with any permitted value). Since the multi-agent environment is not team symmetric, we need to calculate this property value for every pair of agent slots. Also, since $|\Pi_e| = 3$ we just need to calculate this property value for one trio of evaluated agents. Following definition 29 we calculate the STG value for the evaluated agents $\pi_{chase1}, \pi_{chase2}, \pi_{chase3} \in \Pi_e$ and each pair of agent slots. We start with agent slots 1 and 2:

$$STG_{1,2}(\pi_{chase1}, \pi_{chase2}, \pi_{chase3}, \Pi_o, w_{\dot{L}}, \mu) = \sum_{\dot{l} \in \dot{L}_{-1,2}^{N(\mu)}(\Pi_o)} w_{\dot{L}}(\dot{l}) STO_{1,2}(\pi_{chase1}, \pi_{chase2}, \pi_{chase3}, \dot{l}, \mu)$$

We do not know which Π_o we have, so we use a figurative agent line-up pattern $\dot{l} = (*, *, \pi_1, \pi_2)$ from $\dot{L}_{-1,2}^{N(\mu)}(\Pi_o)$:

$$STO_{1,2}(\pi_{chase1}, \pi_{chase2}, \pi_{chase3}, \dot{l}, \mu) = STO_{1,2}(\pi_{chase1}, \pi_{chase2}, \pi_{chase3}, (*, *, \pi_1, \pi_2), \mu)$$

The following table shows us $STO_{1,2}$ for all the permutations of $\pi_{chase1}, \pi_{chase2}, \pi_{chase3}$.

AS 1		AS 2	AS 1		AS 2	AS 1		AS 2
π_{chase1}	$<$	π_{chase2}	π_{chase1}	$<$	π_{chase3}	π_{chase2}	$<$	π_{chase1}
π_{chase2}	$<$	π_{chase3}	π_{chase3}	$<$	π_{chase2}	π_{chase1}	$<$	π_{chase3}
π_{chase1}	$<$	π_{chase3}	π_{chase1}	$<$	π_{chase2}	π_{chase2}	$<$	π_{chase3}
AS 1		AS 2	AS 1		AS 2	AS 1		AS 2
π_{chase2}	$<$	π_{chase3}	π_{chase3}	$<$	π_{chase1}	π_{chase3}	$<$	π_{chase2}
π_{chase3}	$<$	π_{chase1}	π_{chase1}	$<$	π_{chase2}	π_{chase2}	$<$	π_{chase1}
π_{chase2}	$<$	π_{chase1}	π_{chase3}	$<$	π_{chase2}	π_{chase3}	$<$	π_{chase1}

It is possible to find a STO for every permutation, since the prey is always chased[8], obtaining the agents in agent slots 1 (any π_{chase}) and 2 (any π_{chase}) the expected average rewards of -1 and 1 respectively. So:

$$STO_{1,2}(\pi_{chase1}, \pi_{chase2}, \pi_{chase3}, (*, *, \pi_1, \pi_2), \mu) = 1$$

[8]Note that only one predator trying to chase the prey is enough to chase a prey which wants to be chased.

We generalise $STG_{1,2}$ for any possible agent line-up pattern:

$$STG_{1,2}(\pi_{chase1}, \pi_{chase2}, \pi_{chase3}, \Pi_o, w_{\dot{L}}, \mu) = \sum_{\dot{l} \in \dot{L}^{N(\mu)}_{-1,2}(\Pi_o)} w_{\dot{L}}(\dot{l}) STO_{1,2}(\pi_{chase1}, \pi_{chase2}, \pi_{chase3}, \dot{l}, \mu) =$$
$$= \sum_{\dot{l} \in \dot{L}^{N(\mu)}_{-1,2}(\Pi_o)} w_{\dot{L}}(\dot{l}) 1 = 1$$

For agent slots 1 and 3, the following table shows us $STO_{1,3}$ for all the permutations of $\pi_{chase1}, \pi_{chase2}, \pi_{chase3}$.

AS 1		AS 3	AS 1		AS 3	AS 1		AS 3
π_{chase1}	$<$	π_{chase2}	π_{chase1}	$<$	π_{chase3}	π_{chase2}	$<$	π_{chase1}
π_{chase2}	$<$	π_{chase3}	π_{chase3}	$<$	π_{chase2}	π_{chase1}	$<$	π_{chase3}
π_{chase1}	$<$	π_{chase3}	π_{chase1}	$<$	π_{chase2}	π_{chase2}	$<$	π_{chase3}
AS 1		AS 3	AS 1		AS 3	AS 1		AS 3
π_{chase2}	$<$	π_{chase3}	π_{chase3}	$<$	π_{chase1}	π_{chase3}	$<$	π_{chase2}
π_{chase3}	$<$	π_{chase1}	π_{chase1}	$<$	π_{chase2}	π_{chase2}	$<$	π_{chase1}
π_{chase2}	$<$	π_{chase1}	π_{chase3}	$<$	π_{chase2}	π_{chase3}	$<$	π_{chase1}

It is also possible to find a STO, since again the agents in agent slots 1 (any π_{chase}) and 3 (any π_{chase}) obtain the expected average rewards of -1 and 1 respectively. So:

$$STG_{1,3}(\pi_{chase1}, \pi_{chase2}, \pi_{chase3}, \Pi_o, w_{\dot{L}}, \mu) = \sum_{\dot{l} \in \dot{L}^{N(\mu)}_{-1,3}(\Pi_o)} w_{\dot{L}}(\dot{l}) STO_{1,3}(\pi_{chase1}, \pi_{chase2}, \pi_{chase3}, \dot{l}, \mu) =$$
$$= \sum_{\dot{l} \in \dot{L}^{N(\mu)}_{-1,3}(\Pi_o)} w_{\dot{L}}(\dot{l}) 1 = 1$$

For agent slots 1 and 4, the following table shows us $STO_{1,4}$ for all the permutations of $\pi_{chase1}, \pi_{chase2}, \pi_{chase3}$.

AS 1		AS 4	AS 1		AS 4	AS 1		AS 4
π_{chase1}	$<$	π_{chase2}	π_{chase1}	$<$	π_{chase3}	π_{chase2}	$<$	π_{chase1}
π_{chase2}	$<$	π_{chase3}	π_{chase3}	$<$	π_{chase2}	π_{chase1}	$<$	π_{chase3}
π_{chase1}	$<$	π_{chase3}	π_{chase1}	$<$	π_{chase2}	π_{chase2}	$<$	π_{chase3}
AS 1		AS 4	AS 1		AS 4	AS 1		AS 4
π_{chase2}	$<$	π_{chase3}	π_{chase3}	$<$	π_{chase1}	π_{chase3}	$<$	π_{chase2}
π_{chase3}	$<$	π_{chase1}	π_{chase1}	$<$	π_{chase2}	π_{chase2}	$<$	π_{chase1}
π_{chase2}	$<$	π_{chase1}	π_{chase3}	$<$	π_{chase2}	π_{chase3}	$<$	π_{chase1}

It is also possible to find a STO, since again the agents in agent slots 1 (any π_{chase}) and 4 (any π_{chase}) obtain the expected average rewards of -1 and 1 respectively. So:

$$STG_{1,4}(\pi_{chase1}, \pi_{chase2}, \pi_{chase3}, \Pi_o, w_{\dot{L}}, \mu) = \sum_{\dot{l} \in \dot{L}^{N(\mu)}_{-1,4}(\Pi_o)} w_{\dot{L}}(\dot{l}) STO_{1,4}(\pi_{chase1}, \pi_{chase2}, \pi_{chase3}, \dot{l}, \mu) =$$
$$= \sum_{\dot{l} \in \dot{L}^{N(\mu)}_{-1,4}(\Pi_o)} w_{\dot{L}}(\dot{l}) 1 = 1$$

For agent slots 2 and 1, the following table shows us $STO_{2,1}$ for all the permutations of $\pi_{chase1}, \pi_{chase2}, \pi_{chase3}$.

AS 2		AS 1	AS 2		AS 1	AS 2		AS 1
π_{chase1}	$<$	π_{chase2}	π_{chase1}	$<$	π_{chase3}	π_{chase2}	$<$	π_{chase1}
π_{chase2}	$<$	π_{chase3}	π_{chase3}	$<$	π_{chase2}	π_{chase1}	$<$	π_{chase3}
π_{chase1}	$<$	π_{chase3}	π_{chase1}	$<$	π_{chase2}	π_{chase2}	$<$	π_{chase3}
AS 2		AS 1	AS 2		AS 1	AS 2		AS 1
π_{chase2}	$<$	π_{chase3}	π_{chase3}	$<$	π_{chase1}	π_{chase3}	$<$	π_{chase2}
π_{chase3}	$<$	π_{chase1}	π_{chase1}	$<$	π_{chase2}	π_{chase2}	$<$	π_{chase1}
π_{chase2}	$<$	π_{chase1}	π_{chase3}	$<$	π_{chase2}	π_{chase3}	$<$	π_{chase1}

It is not possible to find a STO for any permutation, since the agents in agent slots 2 (any π_{chase}) and 1 (any π_{chase}) obtain the expected average rewards of 1 and -1 respectively. So:

$$STG_{2,1}(\pi_{chase1}, \pi_{chase2}, \pi_{chase3}, \Pi_o, w_{\dot{L}}, \mu) = \sum_{\dot{l} \in \dot{L}^{N(\mu)}_{-2,1}(\Pi_o)} w_{\dot{L}}(\dot{l}) STO_{2,1}(\pi_{chase1}, \pi_{chase2}, \pi_{chase3}, \dot{l}, \mu) =$$
$$= \sum_{\dot{l} \in \dot{L}^{N(\mu)}_{-2,1}(\Pi_o)} w_{\dot{L}}(\dot{l}) 0 = 0$$

For agent slots 2 and 3, it is also not possible to find a STO, since both agents obtain the same expected average reward due they are in the same team. So:

$$STG_{2,3}(\pi_{chase1}, \pi_{chase2}, \pi_{chase3}, \Pi_o, w_{\dot{L}}, \mu) = \sum_{\dot{l} \in \dot{L}^{N(\mu)}_{-2,3}(\Pi_o)} w_{\dot{L}}(\dot{l}) STO_{2,3}(\pi_{chase1}, \pi_{chase2}, \pi_{chase3}, \dot{l}, \mu) =$$
$$= \sum_{\dot{l} \in \dot{L}^{N(\mu)}_{-2,3}(\Pi_o)} w_{\dot{L}}(\dot{l}) 0 = 0$$

For agent slots 2 and 4, it is also not possible to find a STO, since again both agents obtain the same expected average reward due they are in the same team. So:

$$STG_{2,4}(\pi_{chase1}, \pi_{chase2}, \pi_{chase3}, \Pi_o, w_{\dot{L}}, \mu) = \sum_{\dot{l} \in \dot{L}^{N(\mu)}_{-2,4}(\Pi_o)} w_{\dot{L}}(\dot{l}) STO_{2,4}(\pi_{chase1}, \pi_{chase2}, \pi_{chase3}, \dot{l}, \mu) =$$
$$= \sum_{\dot{l} \in \dot{L}^{N(\mu)}_{-2,4}(\Pi_o)} w_{\dot{L}}(\dot{l}) 0 = 0$$

For agent slots 3 and 1, the following table shows us $STO_{3,1}$ for all the permutations of $\pi_{chase1}, \pi_{chase2}, \pi_{chase3}$.

AS 3		AS 1	AS 3		AS 1	AS 3		AS 1
π_{chase1}	$<$	π_{chase2}	π_{chase1}	$<$	π_{chase3}	π_{chase2}	$<$	π_{chase1}
π_{chase2}	$<$	π_{chase3}	π_{chase3}	$<$	π_{chase2}	π_{chase1}	$<$	π_{chase3}
π_{chase1}	$<$	π_{chase3}	π_{chase1}	$<$	π_{chase2}	π_{chase2}	$<$	π_{chase3}
AS 3		AS 1	AS 3		AS 1	AS 3		AS 1
π_{chase2}	$<$	π_{chase3}	π_{chase3}	$<$	π_{chase1}	π_{chase3}	$<$	π_{chase2}
π_{chase3}	$<$	π_{chase1}	π_{chase1}	$<$	π_{chase2}	π_{chase2}	$<$	π_{chase1}
π_{chase2}	$<$	π_{chase1}	π_{chase3}	$<$	π_{chase2}	π_{chase3}	$<$	π_{chase1}

It is also not possible to find a STO for any permutation, since the agents in agent slots 3 (any π_{chase}) and 1 (any π_{chase}) obtain the expected average rewards of 1 and -1 respectively. So:

$$\begin{aligned} STG_{3,1}(\pi_{chase1}, \pi_{chase2}, \pi_{chase3}, \Pi_o, w_{\dot{L}}, \mu) &= \sum_{\dot{l} \in \dot{L}^{N(\mu)}_{-3,1}(\Pi_o)} w_{\dot{L}}(\dot{l}) STO_{3,1}(\pi_{chase1}, \pi_{chase2}, \pi_{chase3}, \dot{l}, \mu) = \\ &= \sum_{\dot{l} \in \dot{L}^{N(\mu)}_{-3,1}(\Pi_o)} w_{\dot{L}}(\dot{l}) 0 = 0 \end{aligned}$$

For agent slots 3 and 2, it is also not possible to find a STO, since both agents obtain the same expected average reward due they are in the same team. So:

$$\begin{aligned} STG_{3,2}(\pi_{chase1}, \pi_{chase2}, \pi_{chase3}, \Pi_o, w_{\dot{L}}, \mu) &= \sum_{\dot{l} \in \dot{L}^{N(\mu)}_{-3,2}(\Pi_o)} w_{\dot{L}}(\dot{l}) STO_{3,2}(\pi_{chase1}, \pi_{chase2}, \pi_{chase3}, \dot{l}, \mu) = \\ &= \sum_{\dot{l} \in \dot{L}^{N(\mu)}_{-3,2}(\Pi_o)} w_{\dot{L}}(\dot{l}) 0 = 0 \end{aligned}$$

For agent slots 3 and 4, it is also not possible to find a STO, since again both agents obtain the same expected average reward due they are in the same team. So:

$$\begin{aligned} STG_{3,4}(\pi_{chase1}, \pi_{chase2}, \pi_{chase3}, \Pi_o, w_{\dot{L}}, \mu) &= \sum_{\dot{l} \in \dot{L}^{N(\mu)}_{-3,4}(\Pi_o)} w_{\dot{L}}(\dot{l}) STO_{3,4}(\pi_{chase1}, \pi_{chase2}, \pi_{chase3}, \dot{l}, \mu) = \\ &= \sum_{\dot{l} \in \dot{L}^{N(\mu)}_{-3,4}(\Pi_o)} w_{\dot{L}}(\dot{l}) 0 = 0 \end{aligned}$$

For agent slots 4 and 1, the following table shows us $STO_{4,1}$ for all the permutations of $\pi_{chase1}, \pi_{chase2}, \pi_{chase3}$.

AS 4		AS 1	AS 4		AS 1	AS 4		AS 1
π_{chase1}	$<$	π_{chase2}	π_{chase1}	$<$	π_{chase3}	π_{chase2}	$<$	π_{chase1}
π_{chase2}	$<$	π_{chase3}	π_{chase3}	$<$	π_{chase2}	π_{chase1}	$<$	π_{chase3}
π_{chase1}	$<$	π_{chase3}	π_{chase1}	$<$	π_{chase2}	π_{chase2}	$<$	π_{chase3}
AS 4		AS 1	AS 4		AS 1	AS 4		AS 1
π_{chase2}	$<$	π_{chase3}	π_{chase3}	$<$	π_{chase1}	π_{chase3}	$<$	π_{chase2}
π_{chase3}	$<$	π_{chase1}	π_{chase1}	$<$	π_{chase2}	π_{chase2}	$<$	π_{chase1}
π_{chase2}	$<$	π_{chase1}	π_{chase3}	$<$	π_{chase2}	π_{chase3}	$<$	π_{chase1}

It is also not possible to find a STO for any permutation, since the agents in agent slots 4 (any π_{chase}) and 1 (any π_{chase}) obtain the expected average rewards of 1 and -1 respectively. So:

$$\begin{aligned} STG_{4,1}(\pi_{chase1}, \pi_{chase2}, \pi_{chase3}, \Pi_o, w_{\dot{L}}, \mu) &= \sum_{\dot{l} \in \dot{L}^{N(\mu)}_{-4,1}(\Pi_o)} w_{\dot{L}}(\dot{l}) STO_{4,1}(\pi_{chase1}, \pi_{chase2}, \pi_{chase3}, \dot{l}, \mu) = \\ &= \sum_{\dot{l} \in \dot{L}^{N(\mu)}_{-4,1}(\Pi_o)} w_{\dot{L}}(\dot{l}) 0 = 0 \end{aligned}$$

For agent slots 4 and 2, it is also not possible to find a STO, since both agents obtain the same expected average reward due they are in the same team. So:

$$STG_{4,2}(\pi_{chase1}, \pi_{chase2}, \pi_{chase3}, \Pi_o, w_{\dot{L}}, \mu) = \sum_{\dot{l} \in \dot{L}^{N(\mu)}_{-4,2}(\Pi_o)} w_{\dot{L}}(\dot{l}) STO_{4,2}(\pi_{chase1}, \pi_{chase2}, \pi_{chase3}, \dot{l}, \mu) = = \sum_{\dot{l} \in \dot{L}^{N(\mu)}_{-4,2}(\Pi_o)} w_{\dot{L}}(\dot{l}) 0 = 0$$

And for agent slots 4 and 3, it is also not possible to find a STO, since again both agents obtain the same expected average reward due they are in the same team. So:

$$STG_{4,3}(\pi_{chase1}, \pi_{chase2}, \pi_{chase3}, \Pi_o, w_{\dot{L}}, \mu) = \sum_{\dot{l} \in \dot{L}^{N(\mu)}_{-4,3}(\Pi_o)} w_{\dot{L}}(\dot{l}) STO_{4,3}(\pi_{chase1}, \pi_{chase2}, \pi_{chase3}, \dot{l}, \mu) = = \sum_{\dot{l} \in \dot{L}^{N(\mu)}_{-4,3}(\Pi_o)} w_{\dot{L}}(\dot{l}) 0 = 0$$

And finally, we calculate the STG value:

$$\begin{aligned}
STG(\Pi_e, w_{\Pi_e}, \Pi_o, w_{\dot{L}}, \mu, w_S) = \eta_{\Pi^3} \sum_{\pi_1,\pi_2,\pi_3 \in \Pi_e | \pi_1 \neq \pi_2 \neq \pi_3} w_{\Pi_e}(\pi_1) w_{\Pi_e}(\pi_2) w_{\Pi_e}(\pi_3) \eta_{S_1^2} \sum_{i=1}^{N(\mu)} w_S(i,\mu) \times \\
\times \left(\sum_{j=1}^{i-1} w_S(j,\mu) STG_{i,j}(\pi_1,\pi_2,\pi_3,\Pi_o,w_{\dot{L}},\mu) + \sum_{j=i+1}^{N(\mu)} w_S(j,\mu) STG_{i,j}(\pi_1,\pi_2,\pi_3,\Pi_o,w_{\dot{L}},\mu) \right) = \\
= 6 \frac{9}{2}\frac{1}{3}\frac{1}{3}\frac{1}{3}\frac{4}{3}\frac{1}{4}\frac{1}{4} \{ STG_{1,2}(\pi_{chase1}, \pi_{chase2}, \pi_{chase3}, \Pi_o, w_{\dot{L}}, \mu) + \\
+ STG_{1,3}(\pi_{chase1}, \pi_{chase2}, \pi_{chase3}, \Pi_o, w_{\dot{L}}, \mu) + \\
+ STG_{1,4}(\pi_{chase1}, \pi_{chase2}, \pi_{chase3}, \Pi_o, w_{\dot{L}}, \mu) + \\
+ STG_{2,1}(\pi_{chase1}, \pi_{chase2}, \pi_{chase3}, \Pi_o, w_{\dot{L}}, \mu) + \\
+ STG_{2,3}(\pi_{chase1}, \pi_{chase2}, \pi_{chase3}, \Pi_o, w_{\dot{L}}, \mu) + \\
+ STG_{2,4}(\pi_{chase1}, \pi_{chase2}, \pi_{chase3}, \Pi_o, w_{\dot{L}}, \mu) + \\
+ STG_{3,1}(\pi_{chase1}, \pi_{chase2}, \pi_{chase3}, \Pi_o, w_{\dot{L}}, \mu) + \\
+ STG_{3,2}(\pi_{chase1}, \pi_{chase2}, \pi_{chase3}, \Pi_o, w_{\dot{L}}, \mu) + \\
+ STG_{3,4}(\pi_{chase1}, \pi_{chase2}, \pi_{chase3}, \Pi_o, w_{\dot{L}}, \mu) + \\
+ STG_{4,1}(\pi_{chase1}, \pi_{chase2}, \pi_{chase3}, \Pi_o, w_{\dot{L}}, \mu) + \\
+ STG_{4,2}(\pi_{chase1}, \pi_{chase2}, \pi_{chase3}, \Pi_o, w_{\dot{L}}, \mu) + \\
+ STG_{4,3}(\pi_{chase1}, \pi_{chase2}, \pi_{chase3}, \Pi_o, w_{\dot{L}}, \mu) \} = \\
= 6 \frac{9}{2}\frac{1}{3}\frac{1}{3}\frac{1}{3}\frac{4}{3}\frac{1}{4}\frac{1}{4} \{ 9 \times 0 + 3 \times 1 \} = \frac{1}{4}
\end{aligned}$$

Note that we avoided to calculate all the permutations of π_1, π_2, π_3 for $STG_{i,j}(\pi_1, \pi_2, \pi_3, \Pi_o, w_{\dot{L}}, \mu)$, since they provide the same result, by calculating only one permutation and multiplying the result by 6 (the number of permutations).

So, for every Π_o we obtain the same result:

$$\forall \Pi_o : STG(\Pi_e, w_{\Pi_e}, \Pi_o, w_{\dot{L}}, \mu, w_S) = \frac{1}{4}$$

Therefore, predator-prey has $Left_{max} = \frac{1}{4}$ (as a *lower* approximation) for this property. □

Approximation 3. $Right_{min}$ for the strict total grading (STG) property is equal to $\frac{1}{4}$ (as a *higher* approximation) for the predator-prey environment.

Proof. To find $Right_{min}$ (equation 8.5), we need to find a Π_e that minimises the property value as much as possible while Π_o maximises it. Using $\Pi_e = \{\pi_{chase1}, \pi_{chase2}, \pi_{chase3}\}$ (a π_{chase} agent tries to be chased when interacting as the prey and tries to perfectly coordinate with the other predators to chase the prey when interacting as a predator) we find a *higher* approximation of this situation no matter which Π_o we use.

Following definition 30 we obtain the STG value for this $\langle \Pi_e, \Pi_o \rangle$ (where Π_o is instantiated with any permitted value). Since the multi-agent environment is not team symmetric, we need to calculate this property value for every pair of agent slots. Also, since $|\Pi_e| = 3$ we just need to calculate this property value for one trio of evaluated agents. Following definition 29 we calculate the STG value for the evaluated agents $\pi_{chase1}, \pi_{chase2}, \pi_{chase3} \in \Pi_e$ and each pair of agent slots. We start with agent slots 1 and 2:

$$STG_{1,2}(\pi_{chase1}, \pi_{chase2}, \pi_{chase3}, \Pi_o, w_{\dot{L}}, \mu) = \sum_{\dot{l} \in \dot{L}_{-1,2}^{N(\mu)}(\Pi_o)} w_{\dot{L}}(\dot{l}) STO_{1,2}(\pi_{chase1}, \pi_{chase2}, \pi_{chase3}, \dot{l}, \mu)$$

We do not know which Π_o we have, so we use a figurative agent line-up pattern $\dot{l} = (*, *, \pi_1, \pi_2)$ from $\dot{L}_{-1,2}^{N(\mu)}(\Pi_o)$:

$$STO_{1,2}(\pi_{chase1}, \pi_{chase2}, \pi_{chase3}, \dot{l}, \mu) = STO_{1,2}(\pi_{chase1}, \pi_{chase2}, \pi_{chase3}, (*, *, \pi_1, \pi_2), \mu)$$

The following table shows us $STO_{1,2}$ for all the permutations of $\pi_{chase1}, \pi_{chase2}, \pi_{chase3}$.

AS 1		AS 2	AS 1		AS 2	AS 1		AS 2
π_{chase1}	$<$	π_{chase2}	π_{chase1}	$<$	π_{chase3}	π_{chase2}	$<$	π_{chase1}
π_{chase2}	$<$	π_{chase3}	π_{chase3}	$<$	π_{chase2}	π_{chase1}	$<$	π_{chase3}
π_{chase1}	$<$	π_{chase3}	π_{chase1}	$<$	π_{chase2}	π_{chase2}	$<$	π_{chase3}
AS 1		AS 2	AS 1		AS 2	AS 1		AS 2
π_{chase2}	$<$	π_{chase3}	π_{chase3}	$<$	π_{chase1}	π_{chase3}	$<$	π_{chase2}
π_{chase3}	$<$	π_{chase1}	π_{chase1}	$<$	π_{chase2}	π_{chase2}	$<$	π_{chase1}
π_{chase2}	$<$	π_{chase1}	π_{chase3}	$<$	π_{chase2}	π_{chase3}	$<$	π_{chase1}

It is possible to find a STO for every permutation, since the prey is always chased[9], obtaining the agents in agent slots 1 (any π_{chase}) and 2 (any π_{chase}) the expected average rewards of -1 and 1 respectively. So:

$$STO_{1,2}(\pi_{chase1}, \pi_{chase2}, \pi_{chase3}, (*, *, \pi_1, \pi_2), \mu) = 1$$

[9]Note that only one predator trying to chase the prey is enough to chase a prey which wants to be chased.

We generalise $STG_{1,2}$ for any possible agent line-up pattern:

$$STG_{1,2}(\pi_{chase1}, \pi_{chase2}, \pi_{chase3}, \Pi_o, w_{\dot{L}}, \mu) = \sum_{\dot{l} \in \dot{L}^{N(\mu)}_{-1,2}(\Pi_o)} w_{\dot{L}}(\dot{l}) STO_{1,2}(\pi_{chase1}, \pi_{chase2}, \pi_{chase3}, \dot{l}, \mu) =$$
$$= \sum_{\dot{l} \in \dot{L}^{N(\mu)}_{-1,2}(\Pi_o)} w_{\dot{L}}(\dot{l}) 1 = 1$$

For agent slots 1 and 3, the following table shows us $STO_{1,3}$ for all the permutations of $\pi_{chase1}, \pi_{chase2}, \pi_{chase3}$.

AS 1		AS 3	AS 1		AS 3	AS 1		AS 3
π_{chase1}	$<$	π_{chase2}	π_{chase1}	$<$	π_{chase3}	π_{chase2}	$<$	π_{chase1}
π_{chase2}	$<$	π_{chase3}	π_{chase3}	$<$	π_{chase2}	π_{chase1}	$<$	π_{chase3}
π_{chase1}	$<$	π_{chase3}	π_{chase1}	$<$	π_{chase2}	π_{chase2}	$<$	π_{chase3}
AS 1		AS 3	AS 1		AS 3	AS 1		AS 3
π_{chase2}	$<$	π_{chase3}	π_{chase3}	$<$	π_{chase1}	π_{chase3}	$<$	π_{chase2}
π_{chase3}	$<$	π_{chase1}	π_{chase1}	$<$	π_{chase2}	π_{chase2}	$<$	π_{chase1}
π_{chase2}	$<$	π_{chase1}	π_{chase3}	$<$	π_{chase2}	π_{chase3}	$<$	π_{chase1}

It is also possible to find a STO, since again the agents in agent slots 1 (any π_{chase}) and 3 (any π_{chase}) obtain the expected average rewards of -1 and 1 respectively. So:

$$STG_{1,3}(\pi_{chase1}, \pi_{chase2}, \pi_{chase3}, \Pi_o, w_{\dot{L}}, \mu) = \sum_{\dot{l} \in \dot{L}^{N(\mu)}_{-1,3}(\Pi_o)} w_{\dot{L}}(\dot{l}) STO_{1,3}(\pi_{chase1}, \pi_{chase2}, \pi_{chase3}, \dot{l}, \mu) =$$
$$= \sum_{\dot{l} \in \dot{L}^{N(\mu)}_{-1,3}(\Pi_o)} w_{\dot{L}}(\dot{l}) 1 = 1$$

For agent slots 1 and 4, the following table shows us $STO_{1,4}$ for all the permutations of $\pi_{chase1}, \pi_{chase2}, \pi_{chase3}$.

AS 1		AS 4	AS 1		AS 4	AS 1		AS 4
π_{chase1}	$<$	π_{chase2}	π_{chase1}	$<$	π_{chase3}	π_{chase2}	$<$	π_{chase1}
π_{chase2}	$<$	π_{chase3}	π_{chase3}	$<$	π_{chase2}	π_{chase1}	$<$	π_{chase3}
π_{chase1}	$<$	π_{chase3}	π_{chase1}	$<$	π_{chase2}	π_{chase2}	$<$	π_{chase3}
AS 1		AS 4	AS 1		AS 4	AS 1		AS 4
π_{chase2}	$<$	π_{chase3}	π_{chase3}	$<$	π_{chase1}	π_{chase3}	$<$	π_{chase2}
π_{chase3}	$<$	π_{chase1}	π_{chase1}	$<$	π_{chase2}	π_{chase2}	$<$	π_{chase1}
π_{chase2}	$<$	π_{chase1}	π_{chase3}	$<$	π_{chase2}	π_{chase3}	$<$	π_{chase1}

It is also possible to find a STO, since again the agents in agent slots 1 (any π_{chase}) and 4 (any π_{chase}) obtain the expected average rewards of -1 and 1 respectively. So:

$$STG_{1,4}(\pi_{chase1}, \pi_{chase2}, \pi_{chase3}, \Pi_o, w_{\dot{L}}, \mu) = \sum_{\dot{l} \in \dot{L}^{N(\mu)}_{-1,4}(\Pi_o)} w_{\dot{L}}(\dot{l}) STO_{1,4}(\pi_{chase1}, \pi_{chase2}, \pi_{chase3}, \dot{l}, \mu) =$$
$$= \sum_{\dot{l} \in \dot{L}^{N(\mu)}_{-1,4}(\Pi_o)} w_{\dot{L}}(\dot{l}) 1 = 1$$

For agent slots 2 and 1, the following table shows us $STO_{2,1}$ for all the permutations of $\pi_{chase1}, \pi_{chase2}, \pi_{chase3}$.

AS 2		AS 1	AS 2		AS 1	AS 2		AS 1
π_{chase1}	$<$	π_{chase2}	π_{chase1}	$<$	π_{chase3}	π_{chase2}	$<$	π_{chase1}
π_{chase2}	$<$	π_{chase3}	π_{chase3}	$<$	π_{chase2}	π_{chase1}	$<$	π_{chase3}
π_{chase1}	$<$	π_{chase3}	π_{chase1}	$<$	π_{chase2}	π_{chase2}	$<$	π_{chase3}
AS 2		AS 1	AS 2		AS 1	AS 2		AS 1
π_{chase2}	$<$	π_{chase3}	π_{chase3}	$<$	π_{chase1}	π_{chase3}	$<$	π_{chase2}
π_{chase3}	$<$	π_{chase1}	π_{chase1}	$<$	π_{chase2}	π_{chase2}	$<$	π_{chase1}
π_{chase2}	$<$	π_{chase1}	π_{chase3}	$<$	π_{chase2}	π_{chase3}	$<$	π_{chase1}

It is not possible to find a STO for any permutation, since the agents in agent slots 2 (any π_{chase}) and 1 (any π_{chase}) obtain the expected average rewards of 1 and -1 respectively. So:

$$STG_{2,1}(\pi_{chase1}, \pi_{chase2}, \pi_{chase3}, \Pi_o, w_{\dot{L}}, \mu) = \sum_{\dot{l} \in \dot{L}^{N(\mu)}_{-2,1}(\Pi_o)} w_{\dot{L}}(\dot{l}) STO_{2,1}(\pi_{chase1}, \pi_{chase2}, \pi_{chase3}, \dot{l}, \mu) =$$

$$= \sum_{\dot{l} \in \dot{L}^{N(\mu)}_{-2,1}(\Pi_o)} w_{\dot{L}}(\dot{l}) 0 = 0$$

For agent slots 2 and 3, it is also not possible to find a STO, since both agents obtain the same expected average reward due they are in the same team. So:

$$STG_{2,3}(\pi_{chase1}, \pi_{chase2}, \pi_{chase3}, \Pi_o, w_{\dot{L}}, \mu) = \sum_{\dot{l} \in \dot{L}^{N(\mu)}_{-2,3}(\Pi_o)} w_{\dot{L}}(\dot{l}) STO_{2,3}(\pi_{chase1}, \pi_{chase2}, \pi_{chase3}, \dot{l}, \mu) =$$

$$= \sum_{\dot{l} \in \dot{L}^{N(\mu)}_{-2,3}(\Pi_o)} w_{\dot{L}}(\dot{l}) 0 = 0$$

For agent slots 2 and 4, it is also not possible to find a STO, since again both agents obtain the same expected average reward due they are in the same team. So:

$$STG_{2,4}(\pi_{chase1}, \pi_{chase2}, \pi_{chase3}, \Pi_o, w_{\dot{L}}, \mu) = \sum_{\dot{l} \in \dot{L}^{N(\mu)}_{-2,4}(\Pi_o)} w_{\dot{L}}(\dot{l}) STO_{2,4}(\pi_{chase1}, \pi_{chase2}, \pi_{chase3}, \dot{l}, \mu) =$$

$$= \sum_{\dot{l} \in \dot{L}^{N(\mu)}_{-2,4}(\Pi_o)} w_{\dot{L}}(\dot{l}) 0 = 0$$

For agent slots 3 and 1, the following table shows us $STO_{3,1}$ for all the permutations of $\pi_{chase1}, \pi_{chase2}, \pi_{chase3}$.

AS 3		AS 1	AS 3		AS 1	AS 3		AS 1
π_{chase1}	$<$	π_{chase2}	π_{chase1}	$<$	π_{chase3}	π_{chase2}	$<$	π_{chase1}
π_{chase2}	$<$	π_{chase3}	π_{chase3}	$<$	π_{chase2}	π_{chase1}	$<$	π_{chase3}
π_{chase1}	$<$	π_{chase3}	π_{chase1}	$<$	π_{chase2}	π_{chase2}	$<$	π_{chase3}
AS 3		AS 1	AS 3		AS 1	AS 3		AS 1
π_{chase2}	$<$	π_{chase3}	π_{chase3}	$<$	π_{chase1}	π_{chase3}	$<$	π_{chase2}
π_{chase3}	$<$	π_{chase1}	π_{chase1}	$<$	π_{chase2}	π_{chase2}	$<$	π_{chase1}
π_{chase2}	$<$	π_{chase1}	π_{chase3}	$<$	π_{chase2}	π_{chase3}	$<$	π_{chase1}

It is also not possible to find a STO for any permutation, since the agents in agent slots 3 (any π_{chase}) and 1 (any π_{chase}) obtain the expected average rewards of 1 and -1 respectively. So:

$$STG_{3,1}(\pi_{chase1}, \pi_{chase2}, \pi_{chase3}, \Pi_o, w_{\dot{L}}, \mu) = \sum_{\dot{l} \in \dot{L}_{-3,1}^{N(\mu)}(\Pi_o)} w_{\dot{L}}(\dot{l}) STO_{3,1}(\pi_{chase1}, \pi_{chase2}, \pi_{chase3}, \dot{l}, \mu) =$$
$$= \sum_{\dot{l} \in \dot{L}_{-3,1}^{N(\mu)}(\Pi_o)} w_{\dot{L}}(\dot{l}) 0 = 0$$

For agent slots 3 and 2, it is also not possible to find a STO, since both agents obtain the same expected average reward due they are in the same team. So:

$$STG_{3,2}(\pi_{chase1}, \pi_{chase2}, \pi_{chase3}, \Pi_o, w_{\dot{L}}, \mu) = \sum_{\dot{l} \in \dot{L}_{-3,2}^{N(\mu)}(\Pi_o)} w_{\dot{L}}(\dot{l}) STO_{3,2}(\pi_{chase1}, \pi_{chase2}, \pi_{chase3}, \dot{l}, \mu) =$$
$$= \sum_{\dot{l} \in \dot{L}_{-3,2}^{N(\mu)}(\Pi_o)} w_{\dot{L}}(\dot{l}) 0 = 0$$

For agent slots 3 and 4, it is also not possible to find a STO, since again both agents obtain the same expected average reward due they are in the same team. So:

$$STG_{3,4}(\pi_{chase1}, \pi_{chase2}, \pi_{chase3}, \Pi_o, w_{\dot{L}}, \mu) = \sum_{\dot{l} \in \dot{L}_{-3,4}^{N(\mu)}(\Pi_o)} w_{\dot{L}}(\dot{l}) STO_{3,4}(\pi_{chase1}, \pi_{chase2}, \pi_{chase3}, \dot{l}, \mu) =$$
$$= \sum_{\dot{l} \in \dot{L}_{-3,4}^{N(\mu)}(\Pi_o)} w_{\dot{L}}(\dot{l}) 0 = 0$$

For agent slots 4 and 1, the following table shows us $STO_{4,1}$ for all the permutations of $\pi_{chase1}, \pi_{chase2}, \pi_{chase3}$.

AS 4		AS 1	AS 4		AS 1	AS 4		AS 1
π_{chase1}	$<$	π_{chase2}	π_{chase1}	$<$	π_{chase3}	π_{chase2}	$<$	π_{chase1}
π_{chase2}	$<$	π_{chase3}	π_{chase3}	$<$	π_{chase2}	π_{chase1}	$<$	π_{chase3}
π_{chase1}	$<$	π_{chase3}	π_{chase1}	$<$	π_{chase2}	π_{chase2}	$<$	π_{chase3}
AS 4		AS 1	AS 4		AS 1	AS 4		AS 1
π_{chase2}	$<$	π_{chase3}	π_{chase3}	$<$	π_{chase1}	π_{chase3}	$<$	π_{chase2}
π_{chase3}	$<$	π_{chase1}	π_{chase1}	$<$	π_{chase2}	π_{chase2}	$<$	π_{chase1}
π_{chase2}	$<$	π_{chase1}	π_{chase3}	$<$	π_{chase2}	π_{chase3}	$<$	π_{chase1}

It is also not possible to find a STO for any permutation, since the agents in agent slots 4 (any π_{chase}) and 1 (any π_{chase}) obtain the expected average rewards of 1 and -1 respectively. So:

$$STG_{4,1}(\pi_{chase1}, \pi_{chase2}, \pi_{chase3}, \Pi_o, w_{\dot{L}}, \mu) = \sum_{\dot{l} \in \dot{L}_{-4,1}^{N(\mu)}(\Pi_o)} w_{\dot{L}}(\dot{l}) STO_{4,1}(\pi_{chase1}, \pi_{chase2}, \pi_{chase3}, \dot{l}, \mu) =$$
$$= \sum_{\dot{l} \in \dot{L}_{-4,1}^{N(\mu)}(\Pi_o)} w_{\dot{L}}(\dot{l}) 0 = 0$$

For agent slots 4 and 2, it is also not possible to find a STO, since both agents obtain the same expected average reward due they are in the same team. So:

$$STG_{4,2}(\pi_{chase1}, \pi_{chase2}, \pi_{chase3}, \Pi_o, w_{\dot{L}}, \mu) = \sum_{\dot{l} \in \dot{L}_{-4,2}^{N(\mu)}(\Pi_o)} w_{\dot{L}}(\dot{l}) STO_{4,2}(\pi_{chase1}, \pi_{chase2}, \pi_{chase3}, \dot{l}, \mu) =$$
$$= \sum_{\dot{l} \in \dot{L}_{-4,2}^{N(\mu)}(\Pi_o)} w_{\dot{L}}(\dot{l}) 0 = 0$$

And for agent slots 4 and 3, it is also not possible to find a STO, since again both agents obtain the same expected average reward due they are in the same team. So:

$$STG_{4,3}(\pi_{chase1}, \pi_{chase2}, \pi_{chase3}, \Pi_o, w_{\dot{L}}, \mu) = \sum_{\dot{l} \in \dot{L}_{-4,3}^{N(\mu)}(\Pi_o)} w_{\dot{L}}(\dot{l}) STO_{4,3}(\pi_{chase1}, \pi_{chase2}, \pi_{chase3}, \dot{l}, \mu) =$$
$$= \sum_{\dot{l} \in \dot{L}_{-4,3}^{N(\mu)}(\Pi_o)} w_{\dot{L}}(\dot{l}) 0 = 0$$

And finally, we calculate the STG value:

$$STG(\Pi_e, w_{\Pi_e}, \Pi_o, w_{\dot{L}}, \mu, w_S) = \eta_{\Pi^3} \sum_{\pi_1, \pi_2, \pi_3 \in \Pi_e | \pi_1 \neq \pi_2 \neq \pi_3} w_{\Pi_e}(\pi_1) w_{\Pi_e}(\pi_2) w_{\Pi_e}(\pi_3) \eta_{S_1^2} \sum_{i=1}^{N(\mu)} w_S(i, \mu) \times$$
$$\times \left(\sum_{j=1}^{i-1} w_S(j, \mu) STG_{i,j}(\pi_1, \pi_2, \pi_3, \Pi_o, w_{\dot{L}}, \mu) + \sum_{j=i+1}^{N(\mu)} w_S(j, \mu) STG_{i,j}(\pi_1, \pi_2, \pi_3, \Pi_o, w_{\dot{L}}, \mu) \right) =$$
$$= 6 \frac{9}{2} \frac{1}{3} \frac{1}{3} \frac{1}{3} \frac{4}{3} \frac{1}{4} \frac{1}{4} \{ STG_{1,2}(\pi_{chase1}, \pi_{chase2}, \pi_{chase3}, \Pi_o, w_{\dot{L}}, \mu) +$$
$$+ STG_{1,3}(\pi_{chase1}, \pi_{chase2}, \pi_{chase3}, \Pi_o, w_{\dot{L}}, \mu) +$$
$$+ STG_{1,4}(\pi_{chase1}, \pi_{chase2}, \pi_{chase3}, \Pi_o, w_{\dot{L}}, \mu) +$$
$$+ STG_{2,1}(\pi_{chase1}, \pi_{chase2}, \pi_{chase3}, \Pi_o, w_{\dot{L}}, \mu) +$$
$$+ STG_{2,3}(\pi_{chase1}, \pi_{chase2}, \pi_{chase3}, \Pi_o, w_{\dot{L}}, \mu) +$$
$$+ STG_{2,4}(\pi_{chase1}, \pi_{chase2}, \pi_{chase3}, \Pi_o, w_{\dot{L}}, \mu) +$$
$$+ STG_{3,1}(\pi_{chase1}, \pi_{chase2}, \pi_{chase3}, \Pi_o, w_{\dot{L}}, \mu) +$$
$$+ STG_{3,2}(\pi_{chase1}, \pi_{chase2}, \pi_{chase3}, \Pi_o, w_{\dot{L}}, \mu) +$$
$$+ STG_{3,4}(\pi_{chase1}, \pi_{chase2}, \pi_{chase3}, \Pi_o, w_{\dot{L}}, \mu) +$$
$$+ STG_{4,1}(\pi_{chase1}, \pi_{chase2}, \pi_{chase3}, \Pi_o, w_{\dot{L}}, \mu) +$$
$$+ STG_{4,2}(\pi_{chase1}, \pi_{chase2}, \pi_{chase3}, \Pi_o, w_{\dot{L}}, \mu) +$$
$$+ STG_{4,3}(\pi_{chase1}, \pi_{chase2}, \pi_{chase3}, \Pi_o, w_{\dot{L}}, \mu) \} =$$
$$= 6 \frac{9}{2} \frac{1}{3} \frac{1}{3} \frac{1}{3} \frac{4}{3} \frac{1}{4} \frac{1}{4} \{ 9 \times 0 + 3 \times 1 \} = \frac{1}{4}$$

Note that we avoided to calculate all the permutations of π_1, π_2, π_3 for $STG_{i,j}(\pi_1, \pi_2, \pi_3, \Pi_o, w_{\dot{L}}, \mu)$, since they provide the same result, by calculating only one permutation and multiplying the result by 6 (the number of permutations).

So, for every Π_o we obtain the same result:

$$\forall \Pi_o : STG(\Pi_e, w_{\Pi_e}, \Pi_o, w_{\dot{L}}, \mu, w_S) = \frac{1}{4}$$

Therefore, predator-prey has $Right_{min} = \frac{1}{4}$ (as a *higher* approximation) for this property. □

C.5 Partial Grading

Now we arrive to the partial grading (PG) property. As given in section 7.5.2, we want to know if there exists a partial ordering between the evaluated agents when interacting in the multi-agent environment. We use an average of rewards as the utility function to calculate an agent's result.

To simplify the notation, we use the next table to represent the PO: $R_i(\mu[\dot{l} \overset{i,j}{\leftarrow} \pi_1, \pi_2]) \leq R_j(\mu[\dot{l} \overset{i,j}{\leftarrow} \pi_1, \pi_2])$, $R_i(\mu[\dot{l} \overset{i,j}{\leftarrow} \pi_2, \pi_3]) \leq R_j(\mu[\dot{l} \overset{i,j}{\leftarrow} \pi_2, \pi_3])$ and $R_i(\mu[\dot{l} \overset{i,j}{\leftarrow} \pi_1, \pi_3]) \leq R_j(\mu[\dot{l} \overset{i,j}{\leftarrow} \pi_1, \pi_3])$.

AS i		AS j
π_1	$\leq$	π_2
π_2	$\leq$	π_3
π_1	$\leq$	π_3

where AS stands for agent slot.

Proposition 60. *General*$_{min}$ for the partial grading (PG) property is equal to $\frac{1}{2}$ for the predator-prey environment.

Proof. To find $General_{min}$ (equation 8.1), we need to find a pair $\langle \Pi_e, \Pi_o \rangle$ that minimises the property value as much as possible. We can have this situation by selecting $\Pi_e = \{\pi_x, \pi_y, \pi_z\}$ and $\Pi_o = \{\pi_s\}$ (a π_s agent always stays in the same cell[10] and π_x, π_y and π_z behave as shown in figures C.5, C.6 and C.7 respectively).

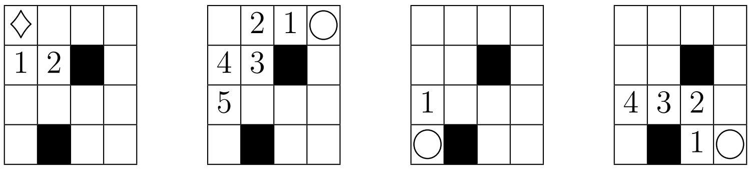


Figure C.5: Behaviour of π_x when interacting on each of the agent slots. Numbers represent the position of π_x on each time step.

Following definition 31 we obtain the PG value for this $\langle \Pi_e, \Pi_o \rangle$. Since the multi-agent environment is not team symmetric, we need to calculate this property value for every pair of agent slots. Also, since $|\Pi_e| = 3$ we just need to calculate this property value for one trio of evaluated agents. Following definition 29 (for PG) we calculate the PG value for the evaluated agents $\pi_x, \pi_y, \pi_z \in \Pi_e$ and each pair of agent slots. We start with agent slots 1 and 2:

[10]Note that every cell has an action that leads to a block or a boundary, therefore an agent performing this action stays at its current cell.

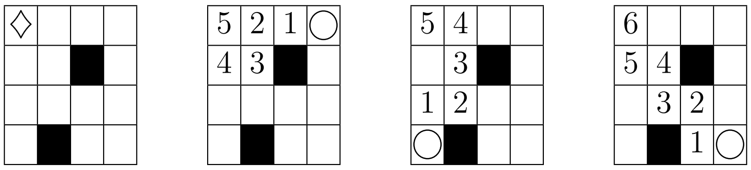


Figure C.6: Behaviour of π_y when interacting on each of the agent slots. Numbers represent the position of π_y on each time step.

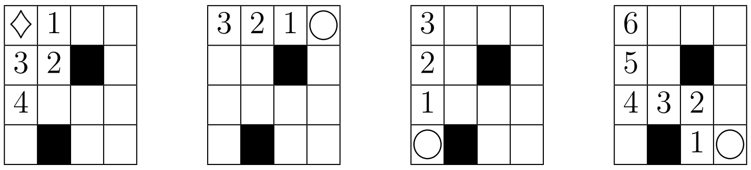


Figure C.7: Behaviour of π_z when interacting on each of the agent slots. Numbers represent the position of π_z on each time step.

$$PG_{1,2}(\pi_x, \pi_y, \pi_z, \Pi_o, w_{\dot{L}}, \mu) = \sum_{\dot{l} \in \dot{L}^{N(\mu)}_{-1,2}(\Pi_o)} w_{\dot{L}}(\dot{l}) PO_{1,2}(\pi_x, \pi_y, \pi_z, \dot{l}, \mu) =$$
$$= \frac{1}{1} PO_{1,2}(\pi_x, \pi_y, \pi_z, (*, *, \pi_s, \pi_s), \mu)$$

The following table shows us $PO_{1,2}$ for all the permutations of π_x, π_y, π_z.

AS 1		AS 2	AS 1		AS 2	AS 1		AS 2
π_x	$\leq$	π_y	π_x	$\leq$	π_z	π_y	$\leq$	π_x
π_y	$\leq$	π_z	π_z	$\leq$	π_y	π_x	$\leq$	π_z
π_x	$\leq$	π_z	π_x	$\leq$	π_y	π_y	$\leq$	π_z
AS 1		AS 2	AS 1		AS 2	AS 1		AS 2
π_y	$\leq$	π_z	π_z	$\leq$	π_x	π_z	$\leq$	π_y
π_z	$\leq$	π_x	π_x	$\leq$	π_y	π_y	$\leq$	π_x
π_y	$\leq$	π_x	π_z	$\leq$	π_y	π_z	$\leq$	π_x

It is not possible to find a PO, since for every permutation we have either the agent line-up $(\pi_x, \pi_z, \pi_s, \pi_s)$, $(\pi_y, \pi_x, \pi_s, \pi_s)$ or $(\pi_z, \pi_y, \pi_s, \pi_s)$. In the three cases the prey is never chased, obtaining the agents in agent slots 1 and 2 the expected average rewards of 1 and -1 respectively. So:

$$PG_{1,2}(\pi_x, \pi_y, \pi_z, \Pi_o, w_{\dot{L}}, \mu) = \frac{1}{1} 0 = 0$$

For agent slots 1 and 3, the following table shows us $PO_{1,3}$ for all the permutations of π_x, π_y, π_z.

AS 1		AS 3	AS 1		AS 3	AS 1		AS 3
π_x	$\leq$	π_y	π_x	$\leq$	π_z	π_y	$\leq$	π_x
π_y	$\leq$	π_z	π_z	$\leq$	π_y	π_x	$\leq$	π_z
π_x	$\leq$	π_z	π_x	$\leq$	π_y	π_y	$\leq$	π_z

AS 1		AS 3	AS 1		AS 3	AS 1		AS 3
π_y	$\leq$	π_z	π_z	$\leq$	π_x	π_z	$\leq$	π_y
π_z	$\leq$	π_x	π_x	$\leq$	π_y	π_y	$\leq$	π_x
π_y	$\leq$	π_x	π_z	$\leq$	π_y	π_z	$\leq$	π_x

It is also not possible to find a PO, since for every permutation we have either the agent line-up $(\pi_x, \pi_s, \pi_z, \pi_s)$, $(\pi_y, \pi_s, \pi_x, \pi_s)$ or $(\pi_z, \pi_s, \pi_y, \pi_s)$. Again, in the three cases the prey is never chased, obtaining the agents in agent slots 1 and 3 the expected average rewards of 1 and -1 respectively. So:

$$\begin{aligned} PG_{1,3}(\pi_x, \pi_y, \pi_z, \Pi_o, w_{\dot{L}}, \mu) &= \sum_{\dot{l} \in \dot{L}_{-1,3}^{N(\mu)}(\Pi_o)} w_{\dot{L}}(\dot{l}) PO_{1,3}(\pi_x, \pi_y, \pi_z, \dot{l}, \mu) = \\ &= \frac{1}{1} PO_{1,3}(\pi_x, \pi_y, \pi_z, (*, \pi_s, *, \pi_s), \mu) = \\ &= \frac{1}{1} 0 = 0 \end{aligned}$$

For agent slots 1 and 4, the following table shows us $PO_{1,4}$ for all the permutations of π_x, π_y, π_z.

AS 1		AS 4	AS 1		AS 4	AS 1		AS 4
π_x	$\leq$	π_y	π_x	$\leq$	π_z	π_y	$\leq$	π_x
π_y	$\leq$	π_z	π_z	$\leq$	π_y	π_x	$\leq$	π_z
π_x	$\leq$	π_z	π_x	$\leq$	π_y	π_y	$\leq$	π_z
AS 1		AS 4	AS 1		AS 4	AS 1		AS 4
π_y	$\leq$	π_z	π_z	$\leq$	π_x	π_z	$\leq$	π_y
π_z	$\leq$	π_x	π_x	$\leq$	π_y	π_y	$\leq$	π_x
π_y	$\leq$	π_x	π_z	$\leq$	π_y	π_z	$\leq$	π_x

It is also not possible to find a PO, since for every permutation we have either the agent line-up $(\pi_x, \pi_s, \pi_s, \pi_z)$, $(\pi_y, \pi_s, \pi_s, \pi_x)$ or $(\pi_z, \pi_s, \pi_s, \pi_y)$. Again, in the three cases the prey is never chased, obtaining the agents in agent slots 1 and 4 the expected average rewards of 1 and -1 respectively. So:

$$\begin{aligned} PG_{1,4}(\pi_x, \pi_y, \pi_z, \Pi_o, w_{\dot{L}}, \mu) &= \sum_{\dot{l} \in \dot{L}_{-1,4}^{N(\mu)}(\Pi_o)} w_{\dot{L}}(\dot{l}) PO_{1,4}(\pi_x, \pi_y, \pi_z, \dot{l}, \mu) = \\ &= \frac{1}{1} PO_{1,4}(\pi_x, \pi_y, \pi_z, (*, \pi_s, \pi_s, *), \mu) = \\ &= \frac{1}{1} 0 = 0 \end{aligned}$$

For agent slots 2 and 1, the following table shows us $PO_{2,1}$ for all the permutations of π_x, π_y, π_z.

AS 2		AS 1	AS 2		AS 1	AS 2		AS 1
π_x	$\leq$	π_y	π_x	$\leq$	π_z	π_y	$\leq$	π_x
π_y	$\leq$	π_z	π_z	$\leq$	π_y	π_x	$\leq$	π_z
π_x	$\leq$	π_z	π_x	$\leq$	π_y	π_y	$\leq$	π_z

AS 2		AS 1	AS 2		AS 1	AS 2		AS 1
π_y	$\leq$	π_z	π_z	$\leq$	π_x	π_z	$\leq$	π_y
π_z	$\leq$	π_x	π_x	$\leq$	π_y	π_y	$\leq$	π_x
π_y	$\leq$	π_x	π_z	$\leq$	π_y	π_z	$\leq$	π_x

It is also not possible to find a PO, since for every permutation we have either the agent line-up $(\pi_z, \pi_x, \pi_s, \pi_s)$, $(\pi_x, \pi_y, \pi_s, \pi_s)$ or $(\pi_y, \pi_z, \pi_s, \pi_s)$. In agent line-up $(\pi_z, \pi_x, \pi_s, \pi_s)$, π_z reaches the 3rd row 1st column cell and π_x chases it in that cell in the 5th time step, obtaining the expected average rewards of -1 and 1 respectively. In agent line-up $(\pi_x, \pi_y, \pi_s, \pi_s)$, π_x reaches the 2nd row 2nd column cell and π_y chases it in that cell in the 3rd time step, obtaining the expected average rewards of -1 and 1 respectively. In agent line-up $(\pi_y, \pi_z, \pi_s, \pi_s)$, π_y stays at the upper left corner and π_z chases it in that cell in the 3rd time step, obtaining the expected average rewards of -1 and 1 respectively. So:

$$\begin{aligned} PG_{2,1}(\pi_x, \pi_y, \pi_z, \Pi_o, w_{\dot{L}}, \mu) &= \sum_{\dot{l} \in \dot{L}_{-2,1}^{N(\mu)}(\Pi_o)} w_{\dot{L}}(\dot{l}) PO_{2,1}(\pi_x, \pi_y, \pi_z, \dot{l}, \mu) = \\ &= \frac{1}{1} PO_{2,1}(\pi_x, \pi_y, \pi_z, (*, *, \pi_s, \pi_s), \mu) = \\ &= \frac{1}{1} 0 = 0 \end{aligned}$$

For agent slots 2 and 3, it is possible to find a PO, since both agents obtain the same expected average reward due they are in the same team. So:

$$\begin{aligned} PG_{2,3}(\pi_x, \pi_y, \pi_z, \Pi_o, w_{\dot{L}}, \mu) &= \sum_{\dot{l} \in \dot{L}_{-2,3}^{N(\mu)}(\Pi_o)} w_{\dot{L}}(\dot{l}) PO_{2,3}(\pi_x, \pi_y, \pi_z, \dot{l}, \mu) = \\ &= \frac{1}{1} PO_{2,3}(\pi_x, \pi_y, \pi_z, (\pi_s, *, *, \pi_s), \mu) = \\ &= \frac{1}{1} 1 = 1 \end{aligned}$$

For agent slots 2 and 4, it is also possible to find a PO, since again both agents obtain the same expected average reward due they are in the same team. So:

$$\begin{aligned} PG_{2,4}(\pi_x, \pi_y, \pi_z, \Pi_o, w_{\dot{L}}, \mu) &= \sum_{\dot{l} \in \dot{L}_{-2,4}^{N(\mu)}(\Pi_o)} w_{\dot{L}}(\dot{l}) PO_{2,4}(\pi_x, \pi_y, \pi_z, \dot{l}, \mu) = \\ &= \frac{1}{1} PO_{2,4}(\pi_x, \pi_y, \pi_z, (\pi_s, *, \pi_s, *), \mu) = \\ &= \frac{1}{1} 1 = 1 \end{aligned}$$

For agent slots 3 and 1, the following table shows us $PO_{3,1}$ for all the permutations of π_x, π_y, π_z.

AS 3		AS 1	AS 3		AS 1	AS 3		AS 1
π_x	$\leq$	π_y	π_x	$\leq$	π_z	π_y	$\leq$	π_x
π_y	$\leq$	π_z	π_z	$\leq$	π_y	π_x	$\leq$	π_z
π_x	$\leq$	π_z	π_x	$\leq$	π_y	π_y	$\leq$	π_z

AS 3		AS 1	AS 3		AS 1	AS 3		AS 1
π_y	$\leq$	π_z	π_z	$\leq$	π_x	π_z	$\leq$	π_y
π_z	$\leq$	π_x	π_x	$\leq$	π_y	π_y	$\leq$	π_x
π_y	$\leq$	π_x	π_z	$\leq$	π_y	π_z	$\leq$	π_x

It is not possible to find a PO, since for every permutation we have either the agent line-up $(\pi_z, \pi_s, \pi_x, \pi_s)$, $(\pi_x, \pi_s, \pi_y, \pi_s)$ or $(\pi_y, \pi_s, \pi_z, \pi_s)$. In agent line-up $(\pi_z, \pi_s, \pi_x, \pi_s)$, π_z reaches the 3rd row 1st column cell and π_x chases it in that cell in the 4th time step, obtaining the expected average rewards of -1 and 1 respectively. In agent line-up $(\pi_x, \pi_s, \pi_y, \pi_s)$, π_x reaches the 2nd row 2nd column cell and π_y chases it in that cell in the 3rd time step, obtaining the expected average rewards of -1 and 1 respectively. In agent line-up $(\pi_y, \pi_s, \pi_z, \pi_s)$, π_y stays at the upper left corner and π_z chases it in that cell in the 3rd time step, obtaining the expected average rewards of -1 and 1 respectively. So:

$$\begin{aligned} PG_{3,1}(\pi_x, \pi_y, \pi_z, \Pi_o, w_{\dot{L}}, \mu) &= \sum_{\dot{l} \in \dot{L}^{N(\mu)}_{-3,1}(\Pi_o)} w_{\dot{L}}(\dot{l}) PO_{3,1}(\pi_x, \pi_y, \pi_z, \dot{l}, \mu) = \\ &= \frac{1}{1} PO_{3,1}(\pi_x, \pi_y, \pi_z, (*, \pi_s, *, \pi_s), \mu) = \\ &= \frac{1}{1} 0 = 0 \end{aligned}$$

For agent slots 3 and 2, it is possible to find a PO, since both agents obtain the same expected average reward due they are in the same team. So:

$$\begin{aligned} PG_{3,2}(\pi_x, \pi_y, \pi_z, \Pi_o, w_{\dot{L}}, \mu) &= \sum_{\dot{l} \in \dot{L}^{N(\mu)}_{-3,2}(\Pi_o)} w_{\dot{L}}(\dot{l}) PO_{3,2}(\pi_x, \pi_y, \pi_z, \dot{l}, \mu) = \\ &= \frac{1}{1} PO_{3,2}(\pi_x, \pi_y, \pi_z, (\pi_s, *, *, \pi_s), \mu) = \\ &= \frac{1}{1} 1 = 1 \end{aligned}$$

For agent slots 3 and 4, it is also possible to find a PO, since again both agents obtain the same expected average reward due they are in the same team. So:

$$\begin{aligned} PG_{3,4}(\pi_x, \pi_y, \pi_z, \Pi_o, w_{\dot{L}}, \mu) &= \sum_{\dot{l} \in \dot{L}^{N(\mu)}_{-3,4}(\Pi_o)} w_{\dot{L}}(\dot{l}) PO_{3,4}(\pi_x, \pi_y, \pi_z, \dot{l}, \mu) = \\ &= \frac{1}{1} PO_{3,4}(\pi_x, \pi_y, \pi_z, (\pi_s, \pi_s, *, *), \mu) = \\ &= \frac{1}{1} 1 = 1 \end{aligned}$$

For agent slots 4 and 1, the following table shows us $PO_{4,1}$ for all the permutations of π_x, π_y, π_z.

AS 4		AS 1	AS 4		AS 1	AS 4		AS 1
π_x	$\leq$	π_y	π_x	$\leq$	π_z	π_y	$\leq$	π_x
π_y	$\leq$	π_z	π_z	$\leq$	π_y	π_x	$\leq$	π_z
π_x	$\leq$	π_z	π_x	$\leq$	π_y	π_y	$\leq$	π_z

AS 4		AS 1	AS 4		AS 1	AS 4		AS 1
π_y	$\leq$	π_z	π_z	$\leq$	π_x	π_z	$\leq$	π_y
π_z	$\leq$	π_x	π_x	$\leq$	π_y	π_y	$\leq$	π_x
π_y	$\leq$	π_x	π_z	$\leq$	π_y	π_z	$\leq$	π_x

It is not possible to find a PO, since for every permutation we have either the agent line-up $(\pi_z, \pi_s, \pi_s, \pi_x)$, $(\pi_x, \pi_s, \pi_s, \pi_y)$ or $(\pi_y, \pi_s, \pi_s, \pi_z)$. In agent line-up $(\pi_z, \pi_s, \pi_s, \pi_x)$, π_z reaches the 3rd row 1st column cell and π_x chases it in that cell in the 4th time step, obtaining the expected average rewards of -1 and 1 respectively. In agent line-up $(\pi_x, \pi_s, \pi_s, \pi_y)$, π_x reaches the 2nd row 2nd column cell and π_y chases it in that cell in the 4th time step, obtaining the expected average rewards of -1 and 1 respectively. In agent line-up $(\pi_y, \pi_s, \pi_s, \pi_z)$, π_y stays at the upper left corner and π_z chases it in that cell in the 6th time step, obtaining the expected average rewards of -1 and 1 respectively. So:

$$\begin{aligned} PG_{4,1}(\pi_x, \pi_y, \pi_z, \Pi_o, w_{\dot{L}}, \mu) &= \sum_{\dot{l} \in \dot{L}^{N(\mu)}_{-4,1}(\Pi_o)} w_{\dot{L}}(\dot{l}) PO_{4,1}(\pi_x, \pi_y, \pi_z, \dot{l}, \mu) = \\ &= \frac{1}{1} PO_{4,1}(\pi_x, \pi_y, \pi_z, (*, \pi_s, \pi_s, *), \mu) = \\ &= \frac{1}{1} 0 = 0 \end{aligned}$$

For agent slots 4 and 2, it is possible to find a PO, since both agents obtain the same expected average reward due they are in the same team. So:

$$\begin{aligned} PG_{4,2}(\pi_x, \pi_y, \pi_z, \Pi_o, w_{\dot{L}}, \mu) &= \sum_{\dot{l} \in \dot{L}^{N(\mu)}_{-4,2}(\Pi_o)} w_{\dot{L}}(\dot{l}) PO_{4,2}(\pi_x, \pi_y, \pi_z, \dot{l}, \mu) = \\ &= \frac{1}{1} PO_{4,2}(\pi_x, \pi_y, \pi_z, (\pi_s, *, \pi_s, *), \mu) = \\ &= \frac{1}{1} 1 = 1 \end{aligned}$$

For agent slots 4 and 3, it is also possible to find a PO, since again both agents obtain the same expected average reward due they are in the same team. So:

$$\begin{aligned} PG_{4,3}(\pi_x, \pi_y, \pi_z, \Pi_o, w_{\dot{L}}, \mu) &= \sum_{\dot{l} \in \dot{L}^{N(\mu)}_{-4,3}(\Pi_o)} w_{\dot{L}}(\dot{l}) PO_{4,3}(\pi_x, \pi_y, \pi_z, \dot{l}, \mu) = \\ &= \frac{1}{1} PO_{4,3}(\pi_x, \pi_y, \pi_z, (\pi_s, \pi_s, *, *), \mu) = \\ &= \frac{1}{1} 1 = 1 \end{aligned}$$

And finally, we calculate the PG value:

$$PG(\Pi_e, w_{\Pi_e}, \Pi_o, w_{\dot{L}}, \mu, w_S) = \eta_{\Pi^3} \sum_{\pi_1,\pi_2,\pi_3 \in \Pi_e | \pi_1 \neq \pi_2 \neq \pi_3} w_{\Pi_e}(\pi_1) w_{\Pi_e}(\pi_2) w_{\Pi_e}(\pi_3) \eta_{S_1^2} \sum_{i=1}^{N(\mu)} w_S(i,\mu) \times$$

$$\times \left(\sum_{j=1}^{i-1} w_S(j,\mu) PG_{i,j}(\pi_1,\pi_2,\pi_3,\Pi_o,w_{\dot{L}},\mu) + \sum_{j=i+1}^{N(\mu)} w_S(j,\mu) PG_{i,j}(\pi_1,\pi_2,\pi_3,\Pi_o,w_{\dot{L}},\mu) \right) =$$

$$= 6 \frac{9}{2}\frac{1}{3}\frac{1}{3}\frac{1}{3}\frac{4}{3}\frac{1}{4}\frac{1}{4} \times$$
$$\times \{PG_{1,2}(\pi_x,\pi_y,\pi_z,\Pi_o,w_{\dot{L}},\mu) + PG_{1,3}(\pi_x,\pi_y,\pi_z,\Pi_o,w_{\dot{L}},\mu) +$$
$$+ PG_{1,4}(\pi_x,\pi_y,\pi_z,\Pi_o,w_{\dot{L}},\mu) + PG_{2,1}(\pi_x,\pi_y,\pi_z,\Pi_o,w_{\dot{L}},\mu) +$$
$$+ PG_{2,3}(\pi_x,\pi_y,\pi_z,\Pi_o,w_{\dot{L}},\mu) + PG_{2,4}(\pi_x,\pi_y,\pi_z,\Pi_o,w_{\dot{L}},\mu) +$$
$$+ PG_{3,1}(\pi_x,\pi_y,\pi_z,\Pi_o,w_{\dot{L}},\mu) + PG_{3,2}(\pi_x,\pi_y,\pi_z,\Pi_o,w_{\dot{L}},\mu) +$$
$$+ PG_{3,4}(\pi_x,\pi_y,\pi_z,\Pi_o,w_{\dot{L}},\mu) + PG_{4,1}(\pi_x,\pi_y,\pi_z,\Pi_o,w_{\dot{L}},\mu) +$$
$$+ PG_{4,3}(\pi_x,\pi_y,\pi_z,\Pi_o,w_{\dot{L}},\mu) + PG_{4,2}(\pi_x,\pi_y,\pi_z,\Pi_o,w_{\dot{L}},\mu)\} =$$
$$= 6 \frac{9}{2}\frac{1}{3}\frac{1}{3}\frac{1}{3}\frac{4}{3}\frac{1}{4}\frac{1}{4} \{6 \times 0 + 6 \times 1\} = \frac{1}{2}$$

Note that we avoided to calculate all the permutations of π_1, π_2, π_3 for $PG_{i,j}(\pi_1,\pi_2,\pi_3,\Pi_o,w_{\dot{L}},\mu)$, since they provide the same result, by calculating only one permutation and multiplying the result by 6 (the number of permutations).

Since $\frac{1}{2}$ is the lowest possible value that we can obtain for the partial grading property, therefore predator-prey has $General_{min} = \frac{1}{2}$ for this property. □

Proposition 61. *$General_{max}$ for the partial grading (PG) property is equal to 1 for the predator-prey environment.*

Proof. To find $General_{max}$ (equation 8.2), we need to find a pair $\langle \Pi_e, \Pi_o \rangle$ that maximises the property value as much as possible. We can have this situation by selecting $\Pi_e = \{\pi_{s1}, \pi_{s2}, \pi_{s3}\}$ and $\Pi_o = \{\pi_x\}$ (a π_s agent always stays in the same cell[11], and a π_x agent acts stochastically with a probability of $1/\sqrt{2}$ to do not reach the upper left corner and a probability of $1 - 1/\sqrt{2}$ to reach this corner).

Following definition 31 we obtain the PG value for this $\langle \Pi_e, \Pi_o \rangle$. Since the multi-agent environment is not team symmetric, we need to calculate this property value for every pair of agent slots. Also, since $|\Pi_e| = 3$ we just need to calculate this property value for one trio of evaluated agents. Following definition 29 (for PG) we calculate the PG value for the evaluated agents $\pi_{s1}, \pi_{s2}, \pi_{s3} \in \Pi_e$ and each pair of agent slots. We start with agent slots 1 and 2:

$$PG_{1,2}(\pi_{s1},\pi_{s2},\pi_{s3},\Pi_o,w_{\dot{L}},\mu) = \sum_{\dot{l} \in \dot{L}_{-1,2}^{N(\mu)}(\Pi_o)} w_{\dot{L}}(\dot{l}) PO_{1,2}(\pi_{s1},\pi_{s2},\pi_{s3},\dot{l},\mu) =$$
$$= \frac{1}{1} PO_{1,2}(\pi_{s1},\pi_{s2},\pi_{s3},(*,*,\pi_x,\pi_x),\mu)$$

The following table shows us $PO_{1,2}$ for all the permutations of $\pi_{s1}, \pi_{s2}, \pi_{s3}$.

[11]Note that every cell has an action that leads to a block or a boundary, therefore an agent performing this action stays at its current cell.

AS 1		AS 2	AS 1		AS 2	AS 1		AS 2
π_{s1}	$\leq$	π_{s2}	π_{s1}	$\leq$	π_{s3}	π_{s2}	$\leq$	π_{s1}
π_{s2}	$\leq$	π_{s3}	π_{s3}	$\leq$	π_{s2}	π_{s1}	$\leq$	π_{s3}
π_{s1}	$\leq$	π_{s3}	π_{s1}	$\leq$	π_{s2}	π_{s2}	$\leq$	π_{s3}
AS 1		AS 2	AS 1		AS 2	AS 1		AS 2
π_{s2}	$\leq$	π_{s3}	π_{s3}	$\leq$	π_{s1}	π_{s3}	$\leq$	π_{s2}
π_{s3}	$\leq$	π_{s1}	π_{s1}	$\leq$	π_{s2}	π_{s2}	$\leq$	π_{s1}
π_{s2}	$\leq$	π_{s1}	π_{s3}	$\leq$	π_{s2}	π_{s3}	$\leq$	π_{s1}

It is possible to find a PO for every permutation, since the agents in agent slots 3 (π_x) and 4 (π_x) have a probability of $(1/\sqrt{2}) \times (1/\sqrt{2}) = 1/2$ to do not chase the prey (any π_s) and the same probability $1 - 1/2 = 1/2$ to chase the prey (any π_s), making the agents in agent slots 1 (any π_s) and 2 (any π_s) to obtain the same expected average reward (0). So:

$$PG_{1,2}(\pi_{s1}, \pi_{s2}, \pi_{s3}, \Pi_o, w_{\dot{L}}, \mu) = \frac{1}{1}1 = 1$$

For agent slots 1 and 3, the following table shows us $PO_{1,3}$ for all the permutations of $\pi_{s1}, \pi_{s2}, \pi_{s3}$.

AS 1		AS 3	AS 1		AS 3	AS 1		AS 3
π_{s1}	$\leq$	π_{s2}	π_{s1}	$\leq$	π_{s3}	π_{s2}	$\leq$	π_{s1}
π_{s2}	$\leq$	π_{s3}	π_{s3}	$\leq$	π_{s2}	π_{s1}	$\leq$	π_{s3}
π_{s1}	$\leq$	π_{s3}	π_{s1}	$\leq$	π_{s2}	π_{s2}	$\leq$	π_{s3}
AS 1		AS 3	AS 1		AS 3	AS 1		AS 3
π_{s2}	$\leq$	π_{s3}	π_{s3}	$\leq$	π_{s1}	π_{s3}	$\leq$	π_{s2}
π_{s3}	$\leq$	π_{s1}	π_{s1}	$\leq$	π_{s2}	π_{s2}	$\leq$	π_{s1}
π_{s2}	$\leq$	π_{s1}	π_{s3}	$\leq$	π_{s2}	π_{s3}	$\leq$	π_{s1}

It is also possible to find a PO for every permutation, making again the agents in agent slots 1 (any π_s) and 3 (any π_s) to obtain the same expected average reward (0). So:

$$\begin{aligned}
PG_{1,3}(\pi_{s1}, \pi_{s2}, \pi_{s3}, \Pi_o, w_{\dot{L}}, \mu) = & \sum_{\dot{l} \in \dot{L}^{N(\mu)}_{-1,3}(\Pi_o)} w_{\dot{L}}(\dot{l}) PO_{1,3}(\pi_{s1}, \pi_{s2}, \pi_{s3}, \dot{l}, \mu) = \\
= & \frac{1}{1} PO_{1,3}(\pi_{s1}, \pi_{s2}, \pi_{s3}, (*, \pi_x, *, \pi_x), \mu) = \\
= & \frac{1}{1}1 = 1
\end{aligned}$$

For agent slots 1 and 4, the following table shows us $PO_{1,4}$ for all the permutations of $\pi_{s1}, \pi_{s2}, \pi_{s3}$.

AS 1		AS 4	AS 1		AS 4	AS 1		AS 4
π_{s1}	$\leq$	π_{s2}	π_{s1}	$\leq$	π_{s3}	π_{s2}	$\leq$	π_{s1}
π_{s2}	$\leq$	π_{s3}	π_{s3}	$\leq$	π_{s2}	π_{s1}	$\leq$	π_{s3}
π_{s1}	$\leq$	π_{s3}	π_{s1}	$\leq$	π_{s2}	π_{s2}	$\leq$	π_{s3}
AS 1		AS 4	AS 1		AS 4	AS 1		AS 4
π_{s2}	$\leq$	π_{s3}	π_{s3}	$\leq$	π_{s1}	π_{s3}	$\leq$	π_{s2}
π_{s3}	$\leq$	π_{s1}	π_{s1}	$\leq$	π_{s2}	π_{s2}	$\leq$	π_{s1}
π_{s2}	$\leq$	π_{s1}	π_{s3}	$\leq$	π_{s2}	π_{s3}	$\leq$	π_{s1}

It is also possible to find a PO for every permutation, making again the agents in agent slots 1 (any π_s) and 4 (any π_s) to obtain the same expected average reward (0). So:

$$\begin{aligned} PG_{1,4}(\pi_{s1}, \pi_{s2}, \pi_{s3}, \Pi_o, w_{\dot{L}}, \mu) &= \sum_{\dot{l}\in \dot{L}_{-1,4}^{N(\mu)}(\Pi_o)} w_{\dot{L}}(\dot{l}) PO_{1,4}(\pi_{s1}, \pi_{s2}, \pi_{s3}, \dot{l}, \mu) = \\ &= \frac{1}{1} PO_{1,4}(\pi_{s1}, \pi_{s2}, \pi_{s3}, (*, \pi_x, \pi_x, *), \mu) = \\ &= \frac{1}{1} 1 = 1 \end{aligned}$$

For agent slots 2 and 1, the following table shows us $PO_{2,1}$ for all the permutations of $\pi_{s1}, \pi_{s2}, \pi_{s3}$.

AS 2		AS 1	AS 2		AS 1	AS 2		AS 1
π_{s1}	$\leq$	π_{s2}	π_{s1}	$\leq$	π_{s3}	π_{s2}	$\leq$	π_{s1}
π_{s2}	$\leq$	π_{s3}	π_{s3}	$\leq$	π_{s2}	π_{s1}	$\leq$	π_{s3}
π_{s1}	$\leq$	π_{s3}	π_{s1}	$\leq$	π_{s2}	π_{s2}	$\leq$	π_{s3}
AS 2		AS 1	AS 2		AS 1	AS 2		AS 1
π_{s2}	$\leq$	π_{s3}	π_{s3}	$\leq$	π_{s1}	π_{s3}	$\leq$	π_{s2}
π_{s3}	$\leq$	π_{s1}	π_{s1}	$\leq$	π_{s2}	π_{s2}	$\leq$	π_{s1}
π_{s2}	$\leq$	π_{s1}	π_{s3}	$\leq$	π_{s2}	π_{s3}	$\leq$	π_{s1}

It is also possible to find a PO for any permutation, making again the agents in agent slots 2 (any π_s) and 1 (any π_s) to obtain the same expected average reward (0). So:

$$\begin{aligned} PG_{2,1}(\pi_{s1}, \pi_{s2}, \pi_{s3}, \Pi_o, w_{\dot{L}}, \mu) &= \sum_{\dot{l}\in \dot{L}_{-2,1}^{N(\mu)}(\Pi_o)} w_{\dot{L}}(\dot{l}) PO_{2,1}(\pi_{s1}, \pi_{s2}, \pi_{s3}, \dot{l}, \mu) = \\ &= \frac{1}{1} PO_{2,1}(\pi_{s1}, \pi_{s2}, \pi_{s3}, (*, *, \pi_x, \pi_x), \mu) = \\ &= \frac{1}{1} 1 = 1 \end{aligned}$$

For agent slots 2 and 3, it is also possible to find a PO, since both agents obtain the same expected average reward due they are in the same team. So:

$$\begin{aligned} PG_{2,3}(\pi_{s1}, \pi_{s2}, \pi_{s3}, \Pi_o, w_{\dot{L}}, \mu) &= \sum_{\dot{l}\in \dot{L}_{-2,3}^{N(\mu)}(\Pi_o)} w_{\dot{L}}(\dot{l}) PO_{2,3}(\pi_{s1}, \pi_{s2}, \pi_{s3}, \dot{l}, \mu) = \\ &= \frac{1}{1} PO_{2,3}(\pi_{s1}, \pi_{s2}, \pi_{s3}, (\pi_x, *, *, \pi_x), \mu) = \\ &= \frac{1}{1} 1 = 1 \end{aligned}$$

For agent slots 2 and 4, it is also possible to find a PO, since again both agents obtain the same expected average reward due they are in the same team. So:

$$PG_{2,4}(\pi_{s1},\pi_{s2},\pi_{s3},\Pi_o,w_{\dot{L}},\mu) = \sum_{\dot{l}\in \dot{L}^{N(\mu)}_{-2,4}(\Pi_o)} w_{\dot{L}}(\dot{l})PO_{2,4}(\pi_{s1},\pi_{s2},\pi_{s3},\dot{l},\mu) =$$
$$= \frac{1}{1}PO_{2,4}(\pi_{s1},\pi_{s2},\pi_{s3},(\pi_x,*,\pi_x,*),\mu) =$$
$$= \frac{1}{1}1 = 1$$

For agent slots 3 and 1, the following table shows us $PO_{3,1}$ for all the permutations of $\pi_{s1},\pi_{s2},\pi_{s3}$.

AS 3		AS 1	AS 3		AS 1	AS 3		AS 1
π_{s1}	$\leq$	π_{s2}	π_{s1}	$\leq$	π_{s3}	π_{s2}	$\leq$	π_{s1}
π_{s2}	$\leq$	π_{s3}	π_{s3}	$\leq$	π_{s2}	π_{s1}	$\leq$	π_{s3}
π_{s1}	$\leq$	π_{s3}	π_{s1}	$\leq$	π_{s2}	π_{s2}	$\leq$	π_{s3}
AS 3		AS 1	AS 3		AS 1	AS 3		AS 1
π_{s2}	$\leq$	π_{s3}	π_{s3}	$\leq$	π_{s1}	π_{s3}	$\leq$	π_{s2}
π_{s3}	$\leq$	π_{s1}	π_{s1}	$\leq$	π_{s2}	π_{s2}	$\leq$	π_{s1}
π_{s2}	$\leq$	π_{s1}	π_{s3}	$\leq$	π_{s2}	π_{s3}	$\leq$	π_{s1}

It is also possible to find a PO for any permutation, making the agents in agent slots 3 (any π_s) and 1 (any π_s) to obtain the same expected average reward (0). So:

$$PG_{3,1}(\pi_{s1},\pi_{s2},\pi_{s3},\Pi_o,w_{\dot{L}},\mu) = \sum_{\dot{l}\in \dot{L}^{N(\mu)}_{-3,1}(\Pi_o)} w_{\dot{L}}(\dot{l})PO_{3,1}(\pi_{s1},\pi_{s2},\pi_{s3},\dot{l},\mu) =$$
$$= \frac{1}{1}PO_{3,1}(\pi_{s1},\pi_{s2},\pi_{s3},(*,\pi_x,*,\pi_x),\mu) =$$
$$= \frac{1}{1}1 = 1$$

For agent slots 3 and 2, it is also possible to find a PO, since both agents obtain the same expected average reward due they are in the same team. So:

$$PG_{3,2}(\pi_{s1},\pi_{s2},\pi_{s3},\Pi_o,w_{\dot{L}},\mu) = \sum_{\dot{l}\in \dot{L}^{N(\mu)}_{-3,2}(\Pi_o)} w_{\dot{L}}(\dot{l})PO_{3,2}(\pi_{s1},\pi_{s2},\pi_{s3},\dot{l},\mu) =$$
$$= \frac{1}{1}PO_{3,2}(\pi_{s1},\pi_{s2},\pi_{s3},(\pi_x,*,*,\pi_x),\mu) =$$
$$= \frac{1}{1}1 = 1$$

For agent slots 3 and 4, it is also possible to find a PO, since again both agents obtain the same expected average reward due they are in the same team. So:

$$PG_{3,4}(\pi_{s1},\pi_{s2},\pi_{s3},\Pi_o,w_{\dot{L}},\mu) = \sum_{\dot{l}\in \dot{L}^{N(\mu)}_{-3,4}(\Pi_o)} w_{\dot{L}}(\dot{l})PO_{3,4}(\pi_{s1},\pi_{s2},\pi_{s3},\dot{l},\mu) =$$
$$= \frac{1}{1}PO_{3,4}(\pi_{s1},\pi_{s2},\pi_{s3},(\pi_x,\pi_x,*,*),\mu) =$$
$$= \frac{1}{1}1 = 1$$

For agent slots 4 and 1, the following table shows us $PO_{4,1}$ for all the permutations of $\pi_{s1}, \pi_{s2}, \pi_{s3}$.

AS 4		AS 1	AS 4		AS 1	AS 4		AS 1
π_{s1}	$\leq$	π_{s2}	π_{s1}	$\leq$	π_{s3}	π_{s2}	$\leq$	π_{s1}
π_{s2}	$\leq$	π_{s3}	π_{s3}	$\leq$	π_{s2}	π_{s1}	$\leq$	π_{s3}
π_{s1}	$\leq$	π_{s3}	π_{s1}	$\leq$	π_{s2}	π_{s2}	$\leq$	π_{s3}
AS 4		AS 1	AS 4		AS 1	AS 4		AS 1
π_{s2}	$\leq$	π_{s3}	π_{s3}	$\leq$	π_{s1}	π_{s3}	$\leq$	π_{s2}
π_{s3}	$\leq$	π_{s1}	π_{s1}	$\leq$	π_{s2}	π_{s2}	$\leq$	π_{s1}
π_{s2}	$\leq$	π_{s1}	π_{s3}	$\leq$	π_{s2}	π_{s3}	$\leq$	π_{s1}

It is also possible to find a PO for any permutation, making the agents in agent slots 4 (any π_s) and 1 (any π_s) to obtain the same expected average reward (0). So:

$$\begin{aligned} PG_{4,1}(\pi_{s1}, \pi_{s2}, \pi_{s3}, \Pi_o, w_{\dot{L}}, \mu) &= \sum_{\dot{l} \in \dot{L}_{-4,1}^{N(\mu)}(\Pi_o)} w_{\dot{L}}(\dot{l}) PO_{4,1}(\pi_{s1}, \pi_{s2}, \pi_{s3}, \dot{l}, \mu) = \\ &= \frac{1}{1} PO_{4,1}(\pi_{s1}, \pi_{s2}, \pi_{s3}, (*, \pi_x, \pi_x, *), \mu) = \\ &= \frac{1}{1} 1 = 1 \end{aligned}$$

For agent slots 4 and 2, it is also possible to find a PO, since both agents obtain the same expected average reward due they are in the same team. So:

$$\begin{aligned} PG_{4,2}(\pi_{s1}, \pi_{s2}, \pi_{s3}, \Pi_o, w_{\dot{L}}, \mu) &= \sum_{\dot{l} \in \dot{L}_{-4,2}^{N(\mu)}(\Pi_o)} w_{\dot{L}}(\dot{l}) PO_{4,2}(\pi_{s1}, \pi_{s2}, \pi_{s3}, \dot{l}, \mu) = \\ &= \frac{1}{1} PO_{4,2}(\pi_{s1}, \pi_{s2}, \pi_{s3}, (\pi_x, *, \pi_x, *), \mu) = \\ &= \frac{1}{1} 1 = 1 \end{aligned}$$

And for agent slots 4 and 3, it is also possible to find a PO, since again both agents obtain the same expected average reward due they are in the same team. So:

$$\begin{aligned} PG_{4,3}(\pi_{s1}, \pi_{s2}, \pi_{s3}, \Pi_o, w_{\dot{L}}, \mu) &= \sum_{\dot{l} \in \dot{L}_{-4,3}^{N(\mu)}(\Pi_o)} w_{\dot{L}}(\dot{l}) PO_{4,3}(\pi_{s1}, \pi_{s2}, \pi_{s3}, \dot{l}, \mu) = \\ &= \frac{1}{1} PO_{4,3}(\pi_{s1}, \pi_{s2}, \pi_{s3}, (\pi_x, \pi_x, *, *), \mu) = \\ &= \frac{1}{1} 1 = 1 \end{aligned}$$

And finally, we calculate the PG value:

$$PG(\Pi_e, w_{\Pi_e}, \Pi_o, w_{\dot{L}}, \mu, w_S) = \eta_{\Pi^3} \sum_{\pi_1,\pi_2,\pi_3 \in \Pi_e | \pi_1 \neq \pi_2 \neq \pi_3} w_{\Pi_e}(\pi_1) w_{\Pi_e}(\pi_2) w_{\Pi_e}(\pi_3) \eta_{S_1^2} \sum_{i=1}^{N(\mu)} w_S(i,\mu) \times$$

$$\times \left(\sum_{j=1}^{i-1} w_S(j,\mu) PG_{i,j}(\pi_1,\pi_2,\pi_3,\Pi_o,w_{\dot{L}},\mu) + \sum_{j=i+1}^{N(\mu)} w_S(j,\mu) PG_{i,j}(\pi_1,\pi_2,\pi_3,\Pi_o,w_{\dot{L}},\mu) \right) =$$

$$= 6 \frac{9}{2} \frac{1}{3} \frac{1}{3} \frac{1}{3} \frac{4}{3} \frac{1}{4} \frac{1}{4} \times$$
$$\times \{PG_{1,2}(\pi_{s1},\pi_{s2},\pi_{s3},\Pi_o,w_{\dot{L}},\mu) + PG_{1,3}(\pi_{s1},\pi_{s2},\pi_{s3},\Pi_o,w_{\dot{L}},\mu) +$$
$$+ PG_{1,4}(\pi_{s1},\pi_{s2},\pi_{s3},\Pi_o,w_{\dot{L}},\mu) + PG_{2,1}(\pi_{s1},\pi_{s2},\pi_{s3},\Pi_o,w_{\dot{L}},\mu) +$$
$$+ PG_{2,3}(\pi_{s1},\pi_{s2},\pi_{s3},\Pi_o,w_{\dot{L}},\mu) + PG_{2,4}(\pi_{s1},\pi_{s2},\pi_{s3},\Pi_o,w_{\dot{L}},\mu) +$$
$$+ PG_{3,1}(\pi_{s1},\pi_{s2},\pi_{s3},\Pi_o,w_{\dot{L}},\mu) + PG_{3,2}(\pi_{s1},\pi_{s2},\pi_{s3},\Pi_o,w_{\dot{L}},\mu) +$$
$$+ PG_{3,4}(\pi_{s1},\pi_{s2},\pi_{s3},\Pi_o,w_{\dot{L}},\mu) + PG_{4,1}(\pi_{s1},\pi_{s2},\pi_{s3},\Pi_o,w_{\dot{L}},\mu) +$$
$$+ PG_{4,2}(\pi_{s1},\pi_{s2},\pi_{s3},\Pi_o,w_{\dot{L}},\mu) + PG_{4,3}(\pi_{s1},\pi_{s2},\pi_{s3},\Pi_o,w_{\dot{L}},\mu)\} =$$
$$= 6 \frac{9}{2} \frac{1}{3} \frac{1}{3} \frac{1}{3} \frac{4}{3} \frac{1}{4} \frac{1}{4} \{12 \times 1\} = 1$$

Note that we avoided to calculate all the permutations of π_1, π_2, π_3 for $PG_{i,j}(\pi_1,\pi_2,\pi_3,\Pi_o,w_{\dot{L}},\mu)$, since they provide the same result, by calculating only one permutation and multiplying the result by 6 (the number of permutations).

Since 1 is the highest possible value for the partial grading property, therefore predator-prey has $General_{max} = 1$ for this property. □

Approximation 4. *$Left_{max}$ for the partial grading (PG) property is equal to $\frac{3}{4}$ (as a lower approximation) for the predator-prey environment.*

Proof. To find $Left_{max}$ (equation 8.4), we need to find a Π_e that maximises the property value as much as possible while Π_o minimises it. Using $\Pi_e = \{\pi_{chase1}, \pi_{chase2}, \pi_{chase3}\}$ (a π_{chase} agent tries to be chased when interacting as the prey and tries to perfectly coordinate with the other predators to chase the prey when interacting as a predator) we find a *lower* approximation of this situation no matter which Π_o we use.

Following definition 31 we obtain the PG value for this $\langle \Pi_e, \Pi_o \rangle$ (where Π_o is instantiated with any permitted value). Since the multi-agent environment is not team symmetric, we need to calculate this property value for every pair of agent slots. Also, since $|\Pi_e| = 3$ we just need to calculate this property value for one trio of evaluated agents. Following definition 29 (for PG) we calculate the PG value for the evaluated agents $\pi_{chase1}, \pi_{chase2}, \pi_{chase3} \in \Pi_e$ and each pair of agent slots. We start with agent slots 1 and 2:

$$PG_{1,2}(\pi_{chase1},\pi_{chase2},\pi_{chase3},\Pi_o,w_{\dot{L}},\mu) = \sum_{\dot{l} \in \dot{L}_{-1,2}^{N(\mu)}(\Pi_o)} w_{\dot{L}}(\dot{l}) PO_{1,2}(\pi_{chase1},\pi_{chase2},\pi_{chase3},\dot{l},\mu)$$

We do not know which Π_o we have, so we use a figurative agent line-up pattern $\dot{l} = (*,*,\pi_1,\pi_2)$ from $\dot{L}_{-1,2}^{N(\mu)}(\Pi_o)$:

$$PO_{1,2}(\pi_{chase1},\pi_{chase2},\pi_{chase3},\dot{l},\mu) = PO_{1,2}(\pi_{chase1},\pi_{chase2},\pi_{chase3},(*,*,\pi_1,\pi_2),\mu)$$

The following table shows us $PO_{1,2}$ for all the permutations of $\pi_{chase1}, \pi_{chase2}, \pi_{chase3}$.

AS 1		AS 2	AS 1		AS 2	AS 1		AS 2
π_{chase1}	$\leq$	π_{chase2}	π_{chase1}	$\leq$	π_{chase3}	π_{chase2}	$\leq$	π_{chase1}
π_{chase2}	$\leq$	π_{chase3}	π_{chase3}	$\leq$	π_{chase2}	π_{chase1}	$\leq$	π_{chase3}
π_{chase1}	$\leq$	π_{chase3}	π_{chase1}	$\leq$	π_{chase2}	π_{chase2}	$\leq$	π_{chase3}
AS 1		AS 2	AS 1		AS 2	AS 1		AS 2
π_{chase2}	$\leq$	π_{chase3}	π_{chase3}	$\leq$	π_{chase1}	π_{chase3}	$\leq$	π_{chase2}
π_{chase3}	$\leq$	π_{chase1}	π_{chase1}	$\leq$	π_{chase2}	π_{chase2}	$\leq$	π_{chase1}
π_{chase2}	$\leq$	π_{chase1}	π_{chase3}	$\leq$	π_{chase2}	π_{chase3}	$\leq$	π_{chase1}

It is possible to find a PO for every permutation, since the prey is always chased[12], obtaining the agents in agent slots 1 (any π_{chase}) and 2 (any π_{chase}) the expected average rewards of -1 and 1 respectively. So:

$$PO_{1,2}(\pi_{chase1}, \pi_{chase2}, \pi_{chase3}, (*, *, \pi_1, \pi_2), \mu) = 1$$

We generalise $PG_{1,2}$ for any possible agent line-up pattern:

$$\begin{aligned} PG_{1,2}(\pi_{chase1}, \pi_{chase2}, \pi_{chase3}, \Pi_o, w_{\dot{L}}, \mu) &= \sum_{\dot{l} \in \dot{L}^{N(\mu)}_{-1,2}(\Pi_o)} w_{\dot{L}}(\dot{l}) PO_{1,2}(\pi_{chase1}, \pi_{chase2}, \pi_{chase3}, \dot{l}, \mu) = \\ &= \sum_{\dot{l} \in \dot{L}^{N(\mu)}_{-1,2}(\Pi_o)} w_{\dot{L}}(\dot{l}) 1 = 1 \end{aligned}$$

For agent slots 1 and 3, the following table shows us $PO_{1,3}$ for all the permutations of $\pi_{chase1}, \pi_{chase2}, \pi_{chase3}$.

AS 1		AS 3	AS 1		AS 3	AS 1		AS 3
π_{chase1}	$\leq$	π_{chase2}	π_{chase1}	$\leq$	π_{chase3}	π_{chase2}	$\leq$	π_{chase1}
π_{chase2}	$\leq$	π_{chase3}	π_{chase3}	$\leq$	π_{chase2}	π_{chase1}	$\leq$	π_{chase3}
π_{chase1}	$\leq$	π_{chase3}	π_{chase1}	$\leq$	π_{chase2}	π_{chase2}	$\leq$	π_{chase3}
AS 1		AS 3	AS 1		AS 3	AS 1		AS 3
π_{chase2}	$\leq$	π_{chase3}	π_{chase3}	$\leq$	π_{chase1}	π_{chase3}	$\leq$	π_{chase2}
π_{chase3}	$\leq$	π_{chase1}	π_{chase1}	$\leq$	π_{chase2}	π_{chase2}	$\leq$	π_{chase1}
π_{chase2}	$\leq$	π_{chase1}	π_{chase3}	$\leq$	π_{chase2}	π_{chase3}	$\leq$	π_{chase1}

It is also possible to find a PO, since again the agents in agent slots 1 (any π_{chase}) and 3 (any π_{chase}) obtain the expected average rewards of -1 and 1 respectively. So:

$$\begin{aligned} PG_{1,3}(\pi_{chase1}, \pi_{chase2}, \pi_{chase3}, \Pi_o, w_{\dot{L}}, \mu) &= \sum_{\dot{l} \in \dot{L}^{N(\mu)}_{-1,3}(\Pi_o)} w_{\dot{L}}(\dot{l}) PO_{1,3}(\pi_{chase1}, \pi_{chase2}, \pi_{chase3}, \dot{l}, \mu) = \\ &= \sum_{\dot{l} \in \dot{L}^{N(\mu)}_{-1,3}(\Pi_o)} w_{\dot{L}}(\dot{l}) 1 = 1 \end{aligned}$$

For agent slots 1 and 4, the following table shows us $PO_{1,4}$ for all the permutations of $\pi_{chase1}, \pi_{chase2}, \pi_{chase3}$.

[12]Note that only one predator trying to chase the prey is enough to chase a prey which wants to be chased.

AS 1		AS 4	AS 1		AS 4	AS 1		AS 4
π_{chase1}	$\leq$	π_{chase2}	π_{chase1}	$\leq$	π_{chase3}	π_{chase2}	$\leq$	π_{chase1}
π_{chase2}	$\leq$	π_{chase3}	π_{chase3}	$\leq$	π_{chase2}	π_{chase1}	$\leq$	π_{chase3}
π_{chase1}	$\leq$	π_{chase3}	π_{chase1}	$\leq$	π_{chase2}	π_{chase2}	$\leq$	π_{chase3}
AS 1		AS 4	AS 1		AS 4	AS 1		AS 4
π_{chase2}	$\leq$	π_{chase3}	π_{chase3}	$\leq$	π_{chase1}	π_{chase3}	$\leq$	π_{chase2}
π_{chase3}	$\leq$	π_{chase1}	π_{chase1}	$\leq$	π_{chase2}	π_{chase2}	$\leq$	π_{chase1}
π_{chase2}	$\leq$	π_{chase1}	π_{chase3}	$\leq$	π_{chase2}	π_{chase3}	$\leq$	π_{chase1}

It is also possible to find a PO, since again the agents in agent slots 1 (any π_{chase}) and 4 (any π_{chase}) obtain the expected average rewards of -1 and 1 respectively. So:

$$PG_{1,4}(\pi_{chase1}, \pi_{chase2}, \pi_{chase3}, \Pi_o, w_{\dot{L}}, \mu) = \sum_{\dot{l} \in \dot{L}_{-1,4}^{N(\mu)}(\Pi_o)} w_{\dot{L}}(\dot{l}) PO_{1,4}(\pi_{chase1}, \pi_{chase2}, \pi_{chase3}, \dot{l}, \mu) =$$

$$= \sum_{\dot{l} \in \dot{L}_{-1,4}^{N(\mu)}(\Pi_o)} w_{\dot{L}}(\dot{l}) 1 = 1$$

For agent slots 2 and 1, the following table shows us $PO_{2,1}$ for all the permutations of $\pi_{chase1}, \pi_{chase2}, \pi_{chase3}$.

AS 2		AS 1	AS 2		AS 1	AS 2		AS 1
π_{chase1}	$\leq$	π_{chase2}	π_{chase1}	$\leq$	π_{chase3}	π_{chase2}	$\leq$	π_{chase1}
π_{chase2}	$\leq$	π_{chase3}	π_{chase3}	$\leq$	π_{chase2}	π_{chase1}	$\leq$	π_{chase3}
π_{chase1}	$\leq$	π_{chase3}	π_{chase1}	$\leq$	π_{chase2}	π_{chase2}	$\leq$	π_{chase3}
AS 2		AS 1	AS 2		AS 1	AS 2		AS 1
π_{chase2}	$\leq$	π_{chase3}	π_{chase3}	$\leq$	π_{chase1}	π_{chase3}	$\leq$	π_{chase2}
π_{chase3}	$\leq$	π_{chase1}	π_{chase1}	$\leq$	π_{chase2}	π_{chase2}	$\leq$	π_{chase1}
π_{chase2}	$\leq$	π_{chase1}	π_{chase3}	$\leq$	π_{chase2}	π_{chase3}	$\leq$	π_{chase1}

It is not possible to find a PO for any permutation, since the agents in agent slots 2 (any π_{chase}) and 1 (any π_{chase}) obtain the expected average rewards of 1 and -1 respectively. So:

$$PG_{2,1}(\pi_{chase1}, \pi_{chase2}, \pi_{chase3}, \Pi_o, w_{\dot{L}}, \mu) = \sum_{\dot{l} \in \dot{L}_{-2,1}^{N(\mu)}(\Pi_o)} w_{\dot{L}}(\dot{l}) PO_{2,1}(\pi_{chase1}, \pi_{chase2}, \pi_{chase3}, \dot{l}, \mu) =$$

$$= \sum_{\dot{l} \in \dot{L}_{-2,1}^{N(\mu)}(\Pi_o)} w_{\dot{L}}(\dot{l}) 0 = 0$$

For agent slots 2 and 3, it is possible to find a PO, since both agents obtain the same expected average reward due they are in the same team. So:

$$PG_{2,3}(\pi_{chase1}, \pi_{chase2}, \pi_{chase3}, \Pi_o, w_{\dot{L}}, \mu) = \sum_{\dot{l} \in \dot{L}_{-2,3}^{N(\mu)}(\Pi_o)} w_{\dot{L}}(\dot{l}) PO_{2,3}(\pi_{chase1}, \pi_{chase2}, \pi_{chase3}, \dot{l}, \mu) =$$

$$= \sum_{\dot{l} \in \dot{L}_{-2,3}^{N(\mu)}(\Pi_o)} w_{\dot{L}}(\dot{l}) 1 = 1$$

For agent slots 2 and 4, it is also possible to find a PO, since again both agents obtain the same expected average reward due they are in the same team. So:

$$\begin{aligned} PG_{2,4}(\pi_{chase1}, \pi_{chase2}, \pi_{chase3}, \Pi_o, w_{\dot{L}}, \mu) &= \sum_{\dot{l} \in \dot{L}^{N(\mu)}_{-2,4}(\Pi_o)} w_{\dot{L}}(\dot{l}) PO_{2,4}(\pi_{chase1}, \pi_{chase2}, \pi_{chase3}, \dot{l}, \mu) = \\ &= \sum_{\dot{l} \in \dot{L}^{N(\mu)}_{-2,4}(\Pi_o)} w_{\dot{L}}(\dot{l}) 1 = 1 \end{aligned}$$

For agent slots 3 and 1, the following table shows us $PO_{3,1}$ for all the permutations of $\pi_{chase1}, \pi_{chase2}, \pi_{chase3}$.

AS 3		AS 1	AS 3		AS 1	AS 3		AS 1
π_{chase1}	$\leq$	π_{chase2}	π_{chase1}	$\leq$	π_{chase3}	π_{chase2}	$\leq$	π_{chase1}
π_{chase2}	$\leq$	π_{chase3}	π_{chase3}	$\leq$	π_{chase2}	π_{chase1}	$\leq$	π_{chase3}
π_{chase1}	$\leq$	π_{chase3}	π_{chase1}	$\leq$	π_{chase2}	π_{chase2}	$\leq$	π_{chase3}
AS 3		AS 1	AS 3		AS 1	AS 3		AS 1
π_{chase2}	$\leq$	π_{chase3}	π_{chase3}	$\leq$	π_{chase1}	π_{chase3}	$\leq$	π_{chase2}
π_{chase3}	$\leq$	π_{chase1}	π_{chase1}	$\leq$	π_{chase2}	π_{chase2}	$\leq$	π_{chase1}
π_{chase2}	$\leq$	π_{chase1}	π_{chase3}	$\leq$	π_{chase2}	π_{chase3}	$\leq$	π_{chase1}

It is not possible to find a PO for any permutation, since the agents in agent slots 3 (any π_{chase}) and 1 (any π_{chase}) obtain the expected average rewards of 1 and -1 respectively. So:

$$\begin{aligned} PG_{3,1}(\pi_{chase1}, \pi_{chase2}, \pi_{chase3}, \Pi_o, w_{\dot{L}}, \mu) &= \sum_{\dot{l} \in \dot{L}^{N(\mu)}_{-3,1}(\Pi_o)} w_{\dot{L}}(\dot{l}) PO_{3,1}(\pi_{chase1}, \pi_{chase2}, \pi_{chase3}, \dot{l}, \mu) = \\ &= \sum_{\dot{l} \in \dot{L}^{N(\mu)}_{-3,1}(\Pi_o)} w_{\dot{L}}(\dot{l}) 0 = 0 \end{aligned}$$

For agent slots 3 and 2, it is possible to find a PO, since both agents obtain the same expected average reward due they are in the same team. So:

$$\begin{aligned} PG_{3,2}(\pi_{chase1}, \pi_{chase2}, \pi_{chase3}, \Pi_o, w_{\dot{L}}, \mu) &= \sum_{\dot{l} \in \dot{L}^{N(\mu)}_{-3,2}(\Pi_o)} w_{\dot{L}}(\dot{l}) PO_{3,2}(\pi_{chase1}, \pi_{chase2}, \pi_{chase3}, \dot{l}, \mu) = \\ &= \sum_{\dot{l} \in \dot{L}^{N(\mu)}_{-3,2}(\Pi_o)} w_{\dot{L}}(\dot{l}) 1 = 1 \end{aligned}$$

For agent slots 3 and 4, it is also possible to find a PO, since again both agents obtain the same expected average reward due they are in the same team. So:

$$\begin{aligned} PG_{3,4}(\pi_{chase1}, \pi_{chase2}, \pi_{chase3}, \Pi_o, w_{\dot{L}}, \mu) &= \sum_{\dot{l} \in \dot{L}^{N(\mu)}_{-3,4}(\Pi_o)} w_{\dot{L}}(\dot{l}) PO_{3,4}(\pi_{chase1}, \pi_{chase2}, \pi_{chase3}, \dot{l}, \mu) = \\ &= \sum_{\dot{l} \in \dot{L}^{N(\mu)}_{-3,4}(\Pi_o)} w_{\dot{L}}(\dot{l}) 1 = 1 \end{aligned}$$

For agent slots 4 and 1, the following table shows us $PO_{4,1}$ for all the permutations of $\pi_{chase1}, \pi_{chase2}, \pi_{chase3}$.

AS 4		AS 1	AS 4		AS 1	AS 4		AS 1
π_{chase1}	$\leq$	π_{chase2}	π_{chase1}	$\leq$	π_{chase3}	π_{chase2}	$\leq$	π_{chase1}
π_{chase2}	$\leq$	π_{chase3}	π_{chase3}	$\leq$	π_{chase2}	π_{chase1}	$\leq$	π_{chase3}
π_{chase1}	$\leq$	π_{chase3}	π_{chase1}	$\leq$	π_{chase2}	π_{chase2}	$\leq$	π_{chase3}
AS 4		AS 1	AS 4		AS 1	AS 4		AS 1
π_{chase2}	$\leq$	π_{chase3}	π_{chase3}	$\leq$	π_{chase1}	π_{chase3}	$\leq$	π_{chase2}
π_{chase3}	$\leq$	π_{chase1}	π_{chase1}	$\leq$	π_{chase2}	π_{chase2}	$\leq$	π_{chase1}
π_{chase2}	$\leq$	π_{chase1}	π_{chase3}	$\leq$	π_{chase2}	π_{chase3}	$\leq$	π_{chase1}

It is not possible to find a PO for any permutation, since the agents in agent slots 4 (any π_{chase}) and 1 (any π_{chase}) obtain the expected average rewards of 1 and -1 respectively. So:

$$
\begin{aligned}
PG_{4,1}(\pi_{chase1}, \pi_{chase2}, \pi_{chase3}, \Pi_o, w_{\dot{L}}, \mu) = & \sum_{\dot{l} \in \dot{L}^{N(\mu)}_{-4,1}(\Pi_o)} w_{\dot{L}}(\dot{l}) PO_{4,1}(\pi_{chase1}, \pi_{chase2}, \pi_{chase3}, \dot{l}, \mu) = \\
= & \sum_{\dot{l} \in \dot{L}^{N(\mu)}_{-4,1}(\Pi_o)} w_{\dot{L}}(\dot{l}) 0 = 0
\end{aligned}
$$

For agent slots 4 and 2, it is possible to find a PO, since both agents obtain the same expected average reward due they are in the same team. So:

$$
\begin{aligned}
PG_{4,2}(\pi_{chase1}, \pi_{chase2}, \pi_{chase3}, \Pi_o, w_{\dot{L}}, \mu) = & \sum_{\dot{l} \in \dot{L}^{N(\mu)}_{-4,2}(\Pi_o)} w_{\dot{L}}(\dot{l}) PO_{4,2}(\pi_{chase1}, \pi_{chase2}, \pi_{chase3}, \dot{l}, \mu) = \\
= & \sum_{\dot{l} \in \dot{L}^{N(\mu)}_{-4,2}(\Pi_o)} w_{\dot{L}}(\dot{l}) 1 = 1
\end{aligned}
$$

And for agent slots 4 and 3, it is also possible to find a PO, since again both agents obtain the same expected average reward due they are in the same team. So:

$$
\begin{aligned}
PG_{4,3}(\pi_{chase1}, \pi_{chase2}, \pi_{chase3}, \Pi_o, w_{\dot{L}}, \mu) = & \sum_{\dot{l} \in \dot{L}^{N(\mu)}_{-4,3}(\Pi_o)} w_{\dot{L}}(\dot{l}) PO_{4,3}(\pi_{chase1}, \pi_{chase2}, \pi_{chase3}, \dot{l}, \mu) = \\
= & \sum_{\dot{l} \in \dot{L}^{N(\mu)}_{-4,3}(\Pi_o)} w_{\dot{L}}(\dot{l}) 1 = 1
\end{aligned}
$$

And finally, we calculate the PG value:

$$PG(\Pi_e, w_{\Pi_e}, \Pi_o, w_{\dot{L}}, \mu, w_S) = \eta_{\Pi^3} \sum_{\pi_1,\pi_2,\pi_3 \in \Pi_e | \pi_1 \neq \pi_2 \neq \pi_3} w_{\Pi_e}(\pi_1) w_{\Pi_e}(\pi_2) w_{\Pi_e}(\pi_3) \eta_{S_1^2} \sum_{i=1}^{N(\mu)} w_S(i,\mu) \times$$
$$\times \left(\sum_{j=1}^{i-1} w_S(j,\mu) PG_{i,j}(\pi_1,\pi_2,\pi_3,\Pi_o,w_{\dot{L}},\mu) + \sum_{j=i+1}^{N(\mu)} w_S(j,\mu) PG_{i,j}(\pi_1,\pi_2,\pi_3,\Pi_o,w_{\dot{L}},\mu) \right) =$$
$$= 6\frac{9}{2}\frac{1}{3}\frac{1}{3}\frac{1}{3}\frac{4}{3}\frac{1}{4}\frac{1}{4}\{PG_{1,2}(\pi_{chase1},\pi_{chase2},\pi_{chase3},\Pi_o,w_{\dot{L}},\mu)+$$
$$+ PG_{1,3}(\pi_{chase1},\pi_{chase2},\pi_{chase3},\Pi_o,w_{\dot{L}},\mu)+$$
$$+ PG_{1,4}(\pi_{chase1},\pi_{chase2},\pi_{chase3},\Pi_o,w_{\dot{L}},\mu)+$$
$$+ PG_{2,1}(\pi_{chase1},\pi_{chase2},\pi_{chase3},\Pi_o,w_{\dot{L}},\mu)+$$
$$+ PG_{2,3}(\pi_{chase1},\pi_{chase2},\pi_{chase3},\Pi_o,w_{\dot{L}},\mu)+$$
$$+ PG_{2,4}(\pi_{chase1},\pi_{chase2},\pi_{chase3},\Pi_o,w_{\dot{L}},\mu)+$$
$$+ PG_{3,1}(\pi_{chase1},\pi_{chase2},\pi_{chase3},\Pi_o,w_{\dot{L}},\mu)+$$
$$+ PG_{3,2}(\pi_{chase1},\pi_{chase2},\pi_{chase3},\Pi_o,w_{\dot{L}},\mu)+$$
$$+ PG_{3,4}(\pi_{chase1},\pi_{chase2},\pi_{chase3},\Pi_o,w_{\dot{L}},\mu)+$$
$$+ PG_{4,1}(\pi_{chase1},\pi_{chase2},\pi_{chase3},\Pi_o,w_{\dot{L}},\mu)+$$
$$+ PG_{4,2}(\pi_{chase1},\pi_{chase2},\pi_{chase3},\Pi_o,w_{\dot{L}},\mu)+$$
$$+ PG_{4,3}(\pi_{chase1},\pi_{chase2},\pi_{chase3},\Pi_o,w_{\dot{L}},\mu)\} =$$
$$= 6\frac{9}{2}\frac{1}{3}\frac{1}{3}\frac{1}{3}\frac{4}{3}\frac{1}{4}\frac{1}{4}\{3 \times 0 + 9 \times 1\} = \frac{3}{4}$$

Note that we avoided to calculate all the permutations of π_1, π_2, π_3 for $PG_{i,j}(\pi_1,\pi_2,\pi_3,\Pi_o,w_{\dot{L}},\mu)$, since they provide the same result, by calculating only one permutation and multiplying the result by 6 (the number of permutations).

So, for every Π_o we obtain the same result:

$$\forall \Pi_o : PG(\Pi_e, w_{\Pi_e}, \Pi_o, w_{\dot{L}}, \mu, w_S) = \frac{3}{4}$$

Therefore, predator-prey has $Left_{max} = \frac{3}{4}$ (as a *lower* approximation) for this property. □

Approximation 5. *$Right_{min}$ for the partial grading (PG) property is equal to $\frac{3}{4}$ (as a higher approximation) for the predator-prey environment.*

Proof. To find $Right_{min}$ (equation 8.5), we need to find a Π_e that minimises the property value as much as possible while Π_o maximises it. Using $\Pi_e = \{\pi_{chase1}, \pi_{chase2}, \pi_{chase3}\}$ (a π_{chase} agent tries to be chased when interacting as the prey and tries to perfectly coordinate with the other predators to chase the prey when interacting as a predator) we find a *higher* approximation of this situation no matter which Π_o we use.

Following definition 31 we obtain the PG value for this $\langle \Pi_e, \Pi_o \rangle$ (where Π_o is instantiated with any permitted value). Since the multi-agent environment is not team symmetric, we need to calculate this property value for every pair of agent slots. Also, since $|\Pi_e| = 3$ we just need to calculate this property value for one trio of evaluated agents. Following definition 29 (for PG) we calculate the PG value for the evaluated agents $\pi_{chase1}, \pi_{chase2}, \pi_{chase3} \in \Pi_e$ and each pair of agent slots. We start with agent slots 1 and 2:

$$PG_{1,2}(\pi_{chase1}, \pi_{chase2}, \pi_{chase3}, \Pi_o, w_{\dot{L}}, \mu) = \sum_{\dot{l} \in \dot{L}^{N(\mu)}_{-1,2}(\Pi_o)} w_{\dot{L}}(\dot{l}) PO_{1,2}(\pi_{chase1}, \pi_{chase2}, \pi_{chase3}, \dot{l}, \mu)$$

We do not know which Π_o we have, so we use a figurative agent line-up pattern $\dot{l} = (*, *, \pi_1, \pi_2)$ from $\dot{L}^{N(\mu)}_{-1,2}(\Pi_o)$:

$$PO_{1,2}(\pi_{chase1}, \pi_{chase2}, \pi_{chase3}, \dot{l}, \mu) = PO_{1,2}(\pi_{chase1}, \pi_{chase2}, \pi_{chase3}, (*, *, \pi_1, \pi_2), \mu)$$

The following table shows us $PO_{1,2}$ for all the permutations of $\pi_{chase1}, \pi_{chase2}, \pi_{chase3}$.

AS 1		AS 2	AS 1		AS 2	AS 1		AS 2
π_{chase1}	$\leq$	π_{chase2}	π_{chase1}	$\leq$	π_{chase3}	π_{chase2}	$\leq$	π_{chase1}
π_{chase2}	$\leq$	π_{chase3}	π_{chase3}	$\leq$	π_{chase2}	π_{chase1}	$\leq$	π_{chase3}
π_{chase1}	$\leq$	π_{chase3}	π_{chase1}	$\leq$	π_{chase2}	π_{chase2}	$\leq$	π_{chase3}
AS 1		AS 2	AS 1		AS 2	AS 1		AS 2
π_{chase2}	$\leq$	π_{chase3}	π_{chase3}	$\leq$	π_{chase1}	π_{chase3}	$\leq$	π_{chase2}
π_{chase3}	$\leq$	π_{chase1}	π_{chase1}	$\leq$	π_{chase2}	π_{chase2}	$\leq$	π_{chase1}
π_{chase2}	$\leq$	π_{chase1}	π_{chase3}	$\leq$	π_{chase2}	π_{chase3}	$\leq$	π_{chase1}

It is possible to find a PO for every permutation, since the prey is always chased[13], obtaining the agents in agent slots 1 (any π_{chase}) and 2 (any π_{chase}) the expected average rewards of -1 and 1 respectively. So:

$$PO_{1,2}(\pi_{chase1}, \pi_{chase2}, \pi_{chase3}, (*, *, \pi_1, \pi_2), \mu) = 1$$

We generalise $PG_{1,2}$ for any possible agent line-up pattern:

$$PG_{1,2}(\pi_{chase1}, \pi_{chase2}, \pi_{chase3}, \Pi_o, w_{\dot{L}}, \mu) = \sum_{\dot{l} \in \dot{L}^{N(\mu)}_{-1,2}(\Pi_o)} w_{\dot{L}}(\dot{l}) PO_{1,2}(\pi_{chase1}, \pi_{chase2}, \pi_{chase3}, \dot{l}, \mu) =$$
$$= \sum_{\dot{l} \in \dot{L}^{N(\mu)}_{-1,2}(\Pi_o)} w_{\dot{L}}(\dot{l}) 1 = 1$$

For agent slots 1 and 3, the following table shows us $PO_{1,3}$ for all the permutations of $\pi_{chase1}, \pi_{chase2}, \pi_{chase3}$.

AS 1		AS 3	AS 1		AS 3	AS 1		AS 3
π_{chase1}	$\leq$	π_{chase2}	π_{chase1}	$\leq$	π_{chase3}	π_{chase2}	$\leq$	π_{chase1}
π_{chase2}	$\leq$	π_{chase3}	π_{chase3}	$\leq$	π_{chase2}	π_{chase1}	$\leq$	π_{chase3}
π_{chase1}	$\leq$	π_{chase3}	π_{chase1}	$\leq$	π_{chase2}	π_{chase2}	$\leq$	π_{chase3}
AS 1		AS 3	AS 1		AS 3	AS 1		AS 3
π_{chase2}	$\leq$	π_{chase3}	π_{chase3}	$\leq$	π_{chase1}	π_{chase3}	$\leq$	π_{chase2}
π_{chase3}	$\leq$	π_{chase1}	π_{chase1}	$\leq$	π_{chase2}	π_{chase2}	$\leq$	π_{chase1}
π_{chase2}	$\leq$	π_{chase1}	π_{chase3}	$\leq$	π_{chase2}	π_{chase3}	$\leq$	π_{chase1}

[13]Note that only one predator trying to chase the prey is enough to chase a prey which wants to be chased.

It is also possible to find a PO, since again the agents in agent slots 1 (any π_{chase}) and 3 (any π_{chase}) obtain the expected average rewards of -1 and 1 respectively. So:

$$PG_{1,3}(\pi_{chase1}, \pi_{chase2}, \pi_{chase3}, \Pi_o, w_{\dot{L}}, \mu) = \sum_{\dot{l} \in \dot{L}_{-1,3}^{N(\mu)}(\Pi_o)} w_{\dot{L}}(\dot{l}) PO_{1,3}(\pi_{chase1}, \pi_{chase2}, \pi_{chase3}, \dot{l}, \mu) =$$
$$= \sum_{\dot{l} \in \dot{L}_{-1,3}^{N(\mu)}(\Pi_o)} w_{\dot{L}}(\dot{l}) 1 = 1$$

For agent slots 1 and 4, the following table shows us $PO_{1,4}$ for all the permutations of $\pi_{chase1}, \pi_{chase2}, \pi_{chase3}$.

AS 1		AS 4	AS 1		AS 4	AS 1		AS 4
π_{chase1}	$\leq$	π_{chase2}	π_{chase1}	$\leq$	π_{chase3}	π_{chase2}	$\leq$	π_{chase1}
π_{chase2}	$\leq$	π_{chase3}	π_{chase3}	$\leq$	π_{chase2}	π_{chase1}	$\leq$	π_{chase3}
π_{chase1}	$\leq$	π_{chase3}	π_{chase1}	$\leq$	π_{chase2}	π_{chase2}	$\leq$	π_{chase3}
AS 1		AS 4	AS 1		AS 4	AS 1		AS 4
π_{chase2}	$\leq$	π_{chase3}	π_{chase3}	$\leq$	π_{chase1}	π_{chase3}	$\leq$	π_{chase2}
π_{chase3}	$\leq$	π_{chase1}	π_{chase1}	$\leq$	π_{chase2}	π_{chase2}	$\leq$	π_{chase1}
π_{chase2}	$\leq$	π_{chase1}	π_{chase3}	$\leq$	π_{chase2}	π_{chase3}	$\leq$	π_{chase1}

It is also possible to find a PO, since again the agents in agent slots 1 (any π_{chase}) and 4 (any π_{chase}) obtain the expected average rewards of -1 and 1 respectively. So:

$$PG_{1,4}(\pi_{chase1}, \pi_{chase2}, \pi_{chase3}, \Pi_o, w_{\dot{L}}, \mu) = \sum_{\dot{l} \in \dot{L}_{-1,4}^{N(\mu)}(\Pi_o)} w_{\dot{L}}(\dot{l}) PO_{1,4}(\pi_{chase1}, \pi_{chase2}, \pi_{chase3}, \dot{l}, \mu) =$$
$$= \sum_{\dot{l} \in \dot{L}_{-1,4}^{N(\mu)}(\Pi_o)} w_{\dot{L}}(\dot{l}) 1 = 1$$

For agent slots 2 and 1, the following table shows us $PO_{2,1}$ for all the permutations of $\pi_{chase1}, \pi_{chase2}, \pi_{chase3}$.

AS 2		AS 1	AS 2		AS 1	AS 2		AS 1
π_{chase1}	$\leq$	π_{chase2}	π_{chase1}	$\leq$	π_{chase3}	π_{chase2}	$\leq$	π_{chase1}
π_{chase2}	$\leq$	π_{chase3}	π_{chase3}	$\leq$	π_{chase2}	π_{chase1}	$\leq$	π_{chase3}
π_{chase1}	$\leq$	π_{chase3}	π_{chase1}	$\leq$	π_{chase2}	π_{chase2}	$\leq$	π_{chase3}
AS 2		AS 1	AS 2		AS 1	AS 2		AS 1
π_{chase2}	$\leq$	π_{chase3}	π_{chase3}	$\leq$	π_{chase1}	π_{chase3}	$\leq$	π_{chase2}
π_{chase3}	$\leq$	π_{chase1}	π_{chase1}	$\leq$	π_{chase2}	π_{chase2}	$\leq$	π_{chase1}
π_{chase2}	$\leq$	π_{chase1}	π_{chase3}	$\leq$	π_{chase2}	π_{chase3}	$\leq$	π_{chase1}

It is not possible to find a PO for any permutation, since the agents in agent slots 2 (any π_{chase}) and 1 (any π_{chase}) obtain the expected average rewards of 1 and -1 respectively. So:

$$PG_{2,1}(\pi_{chase1}, \pi_{chase2}, \pi_{chase3}, \Pi_o, w_{\dot{L}}, \mu) = \sum_{\dot{l} \in \dot{L}_{-2,1}^{N(\mu)}(\Pi_o)} w_{\dot{L}}(\dot{l}) PO_{2,1}(\pi_{chase1}, \pi_{chase2}, \pi_{chase3}, \dot{l}, \mu) =$$
$$= \sum_{\dot{l} \in \dot{L}_{-2,1}^{N(\mu)}(\Pi_o)} w_{\dot{L}}(\dot{l}) 0 = 0$$

For agent slots 2 and 3, it is possible to find a PO, since both agents obtain the same expected average reward due they are in the same team. So:

$$PG_{2,3}(\pi_{chase1}, \pi_{chase2}, \pi_{chase3}, \Pi_o, w_{\dot{L}}, \mu) = \sum_{\dot{l} \in \dot{L}^{N(\mu)}_{-2,3}(\Pi_o)} w_{\dot{L}}(\dot{l}) PO_{2,3}(\pi_{chase1}, \pi_{chase2}, \pi_{chase3}, \dot{l}, \mu) =$$
$$= \sum_{\dot{l} \in \dot{L}^{N(\mu)}_{-2,3}(\Pi_o)} w_{\dot{L}}(\dot{l}) 1 = 1$$

For agent slots 2 and 4, it is also possible to find a PO, since again both agents obtain the same expected average reward due they are in the same team. So:

$$PG_{2,4}(\pi_{chase1}, \pi_{chase2}, \pi_{chase3}, \Pi_o, w_{\dot{L}}, \mu) = \sum_{\dot{l} \in \dot{L}^{N(\mu)}_{-2,4}(\Pi_o)} w_{\dot{L}}(\dot{l}) PO_{2,4}(\pi_{chase1}, \pi_{chase2}, \pi_{chase3}, \dot{l}, \mu) =$$
$$= \sum_{\dot{l} \in \dot{L}^{N(\mu)}_{-2,4}(\Pi_o)} w_{\dot{L}}(\dot{l}) 1 = 1$$

For agent slots 3 and 1, the following table shows us $PO_{3,1}$ for all the permutations of $\pi_{chase1}, \pi_{chase2}, \pi_{chase3}$.

AS 3		AS 1	AS 3		AS 1	AS 3		AS 1
π_{chase1}	$\leq$	π_{chase2}	π_{chase1}	$\leq$	π_{chase3}	π_{chase2}	$\leq$	π_{chase1}
π_{chase2}	$\leq$	π_{chase3}	π_{chase3}	$\leq$	π_{chase2}	π_{chase1}	$\leq$	π_{chase3}
π_{chase1}	$\leq$	π_{chase3}	π_{chase1}	$\leq$	π_{chase2}	π_{chase2}	$\leq$	π_{chase3}
AS 3		AS 1	AS 3		AS 1	AS 3		AS 1
π_{chase2}	$\leq$	π_{chase3}	π_{chase3}	$\leq$	π_{chase1}	π_{chase3}	$\leq$	π_{chase2}
π_{chase3}	$\leq$	π_{chase1}	π_{chase1}	$\leq$	π_{chase2}	π_{chase2}	$\leq$	π_{chase1}
π_{chase2}	$\leq$	π_{chase1}	π_{chase3}	$\leq$	π_{chase2}	π_{chase3}	$\leq$	π_{chase1}

It is not possible to find a PO for any permutation, since the agents in agent slots 3 (any π_{chase}) and 1 (any π_{chase}) obtain the expected average rewards of 1 and -1 respectively. So:

$$PG_{3,1}(\pi_{chase1}, \pi_{chase2}, \pi_{chase3}, \Pi_o, w_{\dot{L}}, \mu) = \sum_{\dot{l} \in \dot{L}^{N(\mu)}_{-3,1}(\Pi_o)} w_{\dot{L}}(\dot{l}) PO_{3,1}(\pi_{chase1}, \pi_{chase2}, \pi_{chase3}, \dot{l}, \mu) =$$
$$= \sum_{\dot{l} \in \dot{L}^{N(\mu)}_{-3,1}(\Pi_o)} w_{\dot{L}}(\dot{l}) 0 = 0$$

For agent slots 3 and 2, it is possible to find a PO, since both agents obtain the same expected average reward due they are in the same team. So:

$$PG_{3,2}(\pi_{chase1}, \pi_{chase2}, \pi_{chase3}, \Pi_o, w_{\dot{L}}, \mu) = \sum_{\dot{l} \in \dot{L}^{N(\mu)}_{-3,2}(\Pi_o)} w_{\dot{L}}(\dot{l}) PO_{3,2}(\pi_{chase1}, \pi_{chase2}, \pi_{chase3}, \dot{l}, \mu) =$$
$$= \sum_{\dot{l} \in \dot{L}^{N(\mu)}_{-3,2}(\Pi_o)} w_{\dot{L}}(\dot{l}) 1 = 1$$

For agent slots 3 and 4, it is also possible to find a PO, since again both agents obtain the same expected average reward due they are in the same team. So:

$$PG_{3,4}(\pi_{chase1}, \pi_{chase2}, \pi_{chase3}, \Pi_o, w_{\dot{L}}, \mu) = \sum_{\dot{l} \in \dot{L}^{N(\mu)}_{-3,4}(\Pi_o)} w_{\dot{L}}(\dot{l}) PO_{3,4}(\pi_{chase1}, \pi_{chase2}, \pi_{chase3}, \dot{l}, \mu) =$$
$$= \sum_{\dot{l} \in \dot{L}^{N(\mu)}_{-3,4}(\Pi_o)} w_{\dot{L}}(\dot{l}) 1 = 1$$

For agent slots 4 and 1, the following table shows us $PO_{4,1}$ for all the permutations of $\pi_{chase1}, \pi_{chase2}, \pi_{chase3}$.

AS 4		AS 1	AS 4		AS 1	AS 4		AS 1
π_{chase1}	$\leq$	π_{chase2}	π_{chase1}	$\leq$	π_{chase3}	π_{chase2}	$\leq$	π_{chase1}
π_{chase2}	$\leq$	π_{chase3}	π_{chase3}	$\leq$	π_{chase2}	π_{chase1}	$\leq$	π_{chase3}
π_{chase1}	$\leq$	π_{chase3}	π_{chase1}	$\leq$	π_{chase2}	π_{chase2}	$\leq$	π_{chase3}
AS 4		AS 1	AS 4		AS 1	AS 4		AS 1
π_{chase2}	$\leq$	π_{chase3}	π_{chase3}	$\leq$	π_{chase1}	π_{chase3}	$\leq$	π_{chase2}
π_{chase3}	$\leq$	π_{chase1}	π_{chase1}	$\leq$	π_{chase2}	π_{chase2}	$\leq$	π_{chase1}
π_{chase2}	$\leq$	π_{chase1}	π_{chase3}	$\leq$	π_{chase2}	π_{chase3}	$\leq$	π_{chase1}

It is not possible to find a PO for any permutation, since the agents in agent slots 4 (any π_{chase}) and 1 (any π_{chase}) obtain the expected average rewards of 1 and -1 respectively. So:

$$PG_{4,1}(\pi_{chase1}, \pi_{chase2}, \pi_{chase3}, \Pi_o, w_{\dot{L}}, \mu) = \sum_{\dot{l} \in \dot{L}^{N(\mu)}_{-4,1}(\Pi_o)} w_{\dot{L}}(\dot{l}) PO_{4,1}(\pi_{chase1}, \pi_{chase2}, \pi_{chase3}, \dot{l}, \mu) =$$
$$= \sum_{\dot{l} \in \dot{L}^{N(\mu)}_{-4,1}(\Pi_o)} w_{\dot{L}}(\dot{l}) 0 = 0$$

For agent slots 4 and 2, it is possible to find a PO, since both agents obtain the same expected average reward due they are in the same team. So:

$$PG_{4,2}(\pi_{chase1}, \pi_{chase2}, \pi_{chase3}, \Pi_o, w_{\dot{L}}, \mu) = \sum_{\dot{l} \in \dot{L}^{N(\mu)}_{-4,2}(\Pi_o)} w_{\dot{L}}(\dot{l}) PO_{4,2}(\pi_{chase1}, \pi_{chase2}, \pi_{chase3}, \dot{l}, \mu) =$$
$$= \sum_{\dot{l} \in \dot{L}^{N(\mu)}_{-4,2}(\Pi_o)} w_{\dot{L}}(\dot{l}) 1 = 1$$

And for agent slots 4 and 3, it is also possible to find a PO, since again both agents obtain the same expected average reward due they are in the same team. So:

$$PG_{4,3}(\pi_{chase1}, \pi_{chase2}, \pi_{chase3}, \Pi_o, w_{\dot{L}}, \mu) = \sum_{\dot{l} \in \dot{L}^{N(\mu)}_{-4,3}(\Pi_o)} w_{\dot{L}}(\dot{l}) PO_{4,3}(\pi_{chase1}, \pi_{chase2}, \pi_{chase3}, \dot{l}, \mu) =$$
$$= \sum_{\dot{l} \in \dot{L}^{N(\mu)}_{-4,3}(\Pi_o)} w_{\dot{L}}(\dot{l}) 1 = 1$$

And finally, we calculate the PG value:

$$PG(\Pi_e, w_{\Pi_e}, \Pi_o, w_{\dot{L}}, \mu, w_S) = \eta_{\Pi^3} \sum_{\pi_1,\pi_2,\pi_3 \in \Pi_e | \pi_1 \neq \pi_2 \neq \pi_3} w_{\Pi_e}(\pi_1) w_{\Pi_e}(\pi_2) w_{\Pi_e}(\pi_3) \eta_{S_1^2} \sum_{i=1}^{N(\mu)} w_S(i,\mu) \times$$

$$\times \left(\sum_{j=1}^{i-1} w_S(j,\mu) PG_{i,j}(\pi_1,\pi_2,\pi_3,\Pi_o,w_{\dot{L}},\mu) + \sum_{j=i+1}^{N(\mu)} w_S(j,\mu) PG_{i,j}(\pi_1,\pi_2,\pi_3,\Pi_o,w_{\dot{L}},\mu) \right) =$$

$$\begin{aligned}
&= 6\frac{9}{2}\frac{1}{3}\frac{1}{3}\frac{1}{3}\frac{4}{3}\frac{1}{4}\frac{1}{4}\{PG_{1,2}(\pi_{chase1},\pi_{chase2},\pi_{chase3},\Pi_o,w_{\dot{L}},\mu)+\\
&+ PG_{1,3}(\pi_{chase1},\pi_{chase2},\pi_{chase3},\Pi_o,w_{\dot{L}},\mu)+\\
&+ PG_{1,4}(\pi_{chase1},\pi_{chase2},\pi_{chase3},\Pi_o,w_{\dot{L}},\mu)+\\
&+ PG_{2,1}(\pi_{chase1},\pi_{chase2},\pi_{chase3},\Pi_o,w_{\dot{L}},\mu)+\\
&+ PG_{2,3}(\pi_{chase1},\pi_{chase2},\pi_{chase3},\Pi_o,w_{\dot{L}},\mu)+\\
&+ PG_{2,4}(\pi_{chase1},\pi_{chase2},\pi_{chase3},\Pi_o,w_{\dot{L}},\mu)+\\
&+ PG_{3,1}(\pi_{chase1},\pi_{chase2},\pi_{chase3},\Pi_o,w_{\dot{L}},\mu)+\\
&+ PG_{3,2}(\pi_{chase1},\pi_{chase2},\pi_{chase3},\Pi_o,w_{\dot{L}},\mu)+\\
&+ PG_{3,4}(\pi_{chase1},\pi_{chase2},\pi_{chase3},\Pi_o,w_{\dot{L}},\mu)+\\
&+ PG_{4,1}(\pi_{chase1},\pi_{chase2},\pi_{chase3},\Pi_o,w_{\dot{L}},\mu)+\\
&+ PG_{4,2}(\pi_{chase1},\pi_{chase2},\pi_{chase3},\Pi_o,w_{\dot{L}},\mu)+\\
&+ PG_{4,3}(\pi_{chase1},\pi_{chase2},\pi_{chase3},\Pi_o,w_{\dot{L}},\mu)\} =\\
&= 6\frac{9}{2}\frac{1}{3}\frac{1}{3}\frac{1}{3}\frac{4}{3}\frac{1}{4}\frac{1}{4}\{3\times 0 + 9\times 1\} = \frac{3}{4}
\end{aligned}$$

Note that we avoided to calculate all the permutations of π_1, π_2, π_3 for $PG_{i,j}(\pi_1,\pi_2,\pi_3,\Pi_o,w_{\dot{L}},\mu)$, since they provide the same result, by calculating only one permutation and multiplying the result by 6 (the number of permutations).

So, for every Π_o we obtain the same result:

$$\forall \Pi_o : PG(\Pi_e, w_{\Pi_e}, \Pi_o, w_{\dot{L}}, \mu, w_S) = \frac{3}{4}$$

Therefore, predator-prey has $Right_{min} = \frac{3}{4}$ (as a *higher* approximation) for this property. □

C.6 Slot Result Dependency

Next we see the slot result dependency (SRD) property. As given in section 7.3.2, we want to know how much competitive or cooperative the multi-agent environment is. We use an average of rewards as the utility function to calculate an agent's result.

Proposition 62. *General* range for the slot result dependency (SRD) property is equal to $[0, 0]$ for the predator-prey environment.

Proof. Following definition 21 we obtain the SRD value for $\langle \Pi_e, \Pi_o \rangle$ (where Π_e and Π_o are instantiated with any permitted values). Since the multi-agent environment is not team symmetric, we need to calculate this property value for every pair of agent slots. Also, we do not know which Π_e we have, so we use a figurative evaluated agent π_1 from Π_e. Following definition 20 we calculate the SRD value for this figurative evaluated agent $\pi_1 \in \Pi_e$ and each pair of agent slots. We start with agent slots 1 and 2:

$$SRD_{1,2}(\pi_1, \Pi_o, w_{\dot{L}}, \mu) = corr_{\dot{l} \in \dot{L}_{-1}^{N(\mu)}(\Pi_o)}[w_{\dot{L}}(\dot{l})](R_1(\mu[\dot{l} \overset{1}{\leftarrow} \pi_1]), R_2(\mu[\dot{l} \overset{1}{\leftarrow} \pi_1]))$$

We do not know which Π_o we have, so we use a figurative agent line-up pattern $\dot{l} = (*, \pi_2, \pi_3, \pi_4)$ from $\dot{L}_{-1}^{N(\mu)}(\Pi_o)$:

$$corr(R_1(\mu[\dot{l} \overset{1}{\leftarrow} \pi_1]), R_2(\mu[\dot{l} \overset{1}{\leftarrow} \pi_1])) = corr(R_1(\mu[\pi_1, \pi_2, \pi_3, \pi_4]), R_2(\mu[\pi_1, \pi_2, \pi_3, \pi_4]))$$

When the agent in agent slot 1 (any π_1) obtains a reward (r) the agent in agent slot 2 (any π_2) obtains the opposite reward ($-r$), and this relation is propagated to expected average rewards as well. Since we use a correlation function between expected average rewards, and the agents in agent slots 1 and 2 always obtain opposite expected average rewards, then the correlation function always obtains the same value[14] of -1. So, for any π_1 we obtain the same result:

$$corr(R_1(\mu[\pi_1, \pi_2, \pi_3, \pi_4]), R_2(\mu[\pi_1, \pi_2, \pi_3, \pi_4])) = -1$$

We generalise $SRD_{1,2}$ for any possible agent line-up pattern:

$$\begin{aligned} SRD_{1,2}(\pi_1, \Pi_o, w_{\dot{L}}, \mu) &= corr_{\dot{l} \in \dot{L}_{-1}^{N(\mu)}(\Pi_o)}[w_{\dot{L}}(\dot{l})](R_1(\mu[\dot{l} \overset{1}{\leftarrow} \pi_1]), R_2(\mu[\dot{l} \overset{1}{\leftarrow} \pi_1])) = \\ &= corr_{\dot{l} \in \dot{L}_{-1}^{N(\mu)}(\Pi_o)}[w_{\dot{L}}(\dot{l})]\,(-1) = -1 \end{aligned}$$

For agent slots 1 and 3, the correlation function also always obtains the same value of -1. So:

$$\begin{aligned} SRD_{1,3}(\pi_1, \Pi_o, w_{\dot{L}}, \mu) &= corr_{\dot{l} \in \dot{L}_{-1}^{N(\mu)}(\Pi_o)}[w_{\dot{L}}(\dot{l})](R_1(\mu[\dot{l} \overset{1}{\leftarrow} \pi_1]), R_3(\mu[\dot{l} \overset{1}{\leftarrow} \pi_1])) = \\ &= corr_{\dot{l} \in \dot{L}_{-1}^{N(\mu)}(\Pi_o)}[w_{\dot{L}}(\dot{l})]\,(-1) = -1 \end{aligned}$$

For agent slots 1 and 4, the correlation function also always obtains the same value of -1. So:

$$\begin{aligned} SRD_{1,4}(\pi_1, \Pi_o, w_{\dot{L}}, \mu) &= corr_{\dot{l} \in \dot{L}_{-1}^{N(\mu)}(\Pi_o)}[w_{\dot{L}}(\dot{l})](R_1(\mu[\dot{l} \overset{1}{\leftarrow} \pi_1]), R_4(\mu[\dot{l} \overset{1}{\leftarrow} \pi_1])) = \\ &= corr_{\dot{l} \in \dot{L}_{-1}^{N(\mu)}(\Pi_o)}[w_{\dot{L}}(\dot{l})]\,(-1) = -1 \end{aligned}$$

For agent slots 2 and 1, the correlation function also always obtains the same value of -1. So:

$$\begin{aligned} SRD_{2,1}(\pi_1, \Pi_o, w_{\dot{L}}, \mu) &= corr_{\dot{l} \in \dot{L}_{-2}^{N(\mu)}(\Pi_o)}[w_{\dot{L}}(\dot{l})](R_2(\mu[\dot{l} \overset{2}{\leftarrow} \pi_1]), R_1(\mu[\dot{l} \overset{2}{\leftarrow} \pi_1])) = \\ &= corr_{\dot{l} \in \dot{L}_{-2}^{N(\mu)}(\Pi_o)}[w_{\dot{L}}(\dot{l})]\,(-1) = -1 \end{aligned}$$

For agent slots 2 and 3, the correlation function obtains the value of 1, since both agents obtain the same expected average reward due they are in the same team. So:

[14] Provided there is at least one game which is not a tie.

$$SRD_{2,3}(\pi_1, \Pi_o, w_{\dot{L}}, \mu) = corr_{\dot{l} \in \dot{L}_{-2}^{N(\mu)}(\Pi_o)}[w_{\dot{L}}(\dot{l})](R_2(\mu[\dot{l} \overset{2}{\leftarrow} \pi_1]), R_3(\mu[\dot{l} \overset{2}{\leftarrow} \pi_1])) =$$
$$= corr_{\dot{l} \in \dot{L}_{-2}^{N(\mu)}(\Pi_o)}[w_{\dot{L}}(\dot{l})]1 = 1$$

For agent slots 2 and 4, the correlation function also obtains the value of 1, since again both agents obtain the same expected average reward due they are in the same team. So:

$$SRD_{2,4}(\pi_1, \Pi_o, w_{\dot{L}}, \mu) = corr_{\dot{l} \in \dot{L}_{-2}^{N(\mu)}(\Pi_o)}[w_{\dot{L}}(\dot{l})](R_2(\mu[\dot{l} \overset{2}{\leftarrow} \pi_1]), R_4(\mu[\dot{l} \overset{2}{\leftarrow} \pi_1])) =$$
$$= corr_{\dot{l} \in \dot{L}_{-2}^{N(\mu)}(\Pi_o)}[w_{\dot{L}}(\dot{l})]1 = 1$$

For agent slots 3 and 1, the correlation function always obtains the same value of −1. So:

$$SRD_{3,1}(\pi_1, \Pi_o, w_{\dot{L}}, \mu) = corr_{\dot{l} \in \dot{L}_{-3}^{N(\mu)}(\Pi_o)}[w_{\dot{L}}(\dot{l})](R_3(\mu[\dot{l} \overset{3}{\leftarrow} \pi_1]), R_1(\mu[\dot{l} \overset{3}{\leftarrow} \pi_1])) =$$
$$= corr_{\dot{l} \in \dot{L}_{-3}^{N(\mu)}(\Pi_o)}[w_{\dot{L}}(\dot{l})]\,(-1) = -1$$

For agent slots 3 and 2, the correlation function obtains the value of 1, since both agents obtain the same expected average reward due they are in the same team. So:

$$SRD_{3,2}(\pi_1, \Pi_o, w_{\dot{L}}, \mu) = corr_{\dot{l} \in \dot{L}_{-3}^{N(\mu)}(\Pi_o)}[w_{\dot{L}}(\dot{l})](R_3(\mu[\dot{l} \overset{3}{\leftarrow} \pi_1]), R_2(\mu[\dot{l} \overset{3}{\leftarrow} \pi_1])) =$$
$$= corr_{\dot{l} \in \dot{L}_{-3}^{N(\mu)}(\Pi_o)}[w_{\dot{L}}(\dot{l})]1 = 1$$

For agent slots 3 and 4, the correlation function also obtains the value of 1, since again both agents obtain the same expected average reward due they are in the same team. So:

$$SRD_{3,4}(\pi_1, \Pi_o, w_{\dot{L}}, \mu) = corr_{\dot{l} \in \dot{L}_{-3}^{N(\mu)}(\Pi_o)}[w_{\dot{L}}(\dot{l})](R_3(\mu[\dot{l} \overset{3}{\leftarrow} \pi_1]), R_4(\mu[\dot{l} \overset{3}{\leftarrow} \pi_1])) =$$
$$= corr_{\dot{l} \in \dot{L}_{-3}^{N(\mu)}(\Pi_o)}[w_{\dot{L}}(\dot{l})]1 = 1$$

For agent slots 4 and 1, the correlation function always obtains the same value of −1. So:

$$SRD_{4,1}(\pi_1, \Pi_o, w_{\dot{L}}, \mu) = corr_{\dot{l} \in \dot{L}_{-4}^{N(\mu)}(\Pi_o)}[w_{\dot{L}}(\dot{l})](R_4(\mu[\dot{l} \overset{4}{\leftarrow} \pi_1]), R_1(\mu[\dot{l} \overset{4}{\leftarrow} \pi_1])) =$$
$$= corr_{\dot{l} \in \dot{L}_{-4}^{N(\mu)}(\Pi_o)}[w_{\dot{L}}(\dot{l})]\,(-1) = -1$$

For agent slots 4 and 2, the correlation function obtains the value of 1, since both agents obtain the same expected average reward due they are in the same team. So:

$$SRD_{4,2}(\pi_1, \Pi_o, w_{\dot{L}}, \mu) = corr_{\dot{l} \in \dot{L}_{-4}^{N(\mu)}(\Pi_o)}[w_{\dot{L}}(\dot{l})](R_4(\mu[\dot{l} \overset{4}{\leftarrow} \pi_1]), R_2(\mu[\dot{l} \overset{4}{\leftarrow} \pi_1])) =$$
$$= corr_{\dot{l} \in \dot{L}_{-4}^{N(\mu)}(\Pi_o)}[w_{\dot{L}}(\dot{l})]1 = 1$$

And for agent slots 4 and 3, the correlation function also obtains the value of 1, since again both agents obtain the same expected average reward due they are in the same team. So:

$$SRD_{4,3}(\pi_1, \Pi_o, w_{\dot{L}}, \mu) = corr_{\dot{l} \in \dot{L}_{-4}^{N(\mu)}(\Pi_o)}[w_{\dot{L}}(\dot{l})](R_4(\mu[\dot{l} \overset{4}{\leftarrow} \pi_1]), R_3(\mu[\dot{l} \overset{4}{\leftarrow} \pi_1])) =$$
$$= corr_{\dot{l} \in \dot{L}_{-4}^{N(\mu)}(\Pi_o)}[w_{\dot{L}}(\dot{l})]1 = 1$$

And finally, we calculate the SRD value generalising for any possible evaluated agent:

$$SRD(\Pi_e, w_{\Pi_e}, \Pi_o, w_{\dot{L}}, \mu, w_S) = \sum_{\pi \in \Pi_e} w_{\Pi_e}(\pi) \eta_{S_1^2} \sum_{i=1}^{N(\mu)} w_S(i,\mu) \times$$
$$\times \left(\sum_{j=1}^{i-1} w_S(j,\mu) SRD_{i,j}(\pi, \Pi_o, w_{\dot{L}}, \mu) + \sum_{j=i+1}^{N(\mu)} w_S(j,\mu) SRD_{i,j}(\pi, \Pi_o, w_{\dot{L}}, \mu) \right) =$$
$$= \sum_{\pi \in \Pi_e} w_{\Pi_e}(\pi) \frac{4}{3}\frac{1}{4}\frac{1}{4} \times$$
$$\times \{ SRD_{1,2}(\pi, \Pi_o, w_{\dot{L}}, \mu) + SRD_{1,3}(\pi, \Pi_o, w_{\dot{L}}, \mu) +$$
$$+ SRD_{1,4}(\pi, \Pi_o, w_{\dot{L}}, \mu) + SRD_{2,1}(\pi, \Pi_o, w_{\dot{L}}, \mu) +$$
$$+ SRD_{2,3}(\pi, \Pi_o, w_{\dot{L}}, \mu) + SRD_{2,4}(\pi, \Pi_o, w_{\dot{L}}, \mu) +$$
$$+ SRD_{3,1}(\pi, \Pi_o, w_{\dot{L}}, \mu) + SRD_{3,2}(\pi, \Pi_o, w_{\dot{L}}, \mu) +$$
$$+ SRD_{3,4}(\pi, \Pi_o, w_{\dot{L}}, \mu) + SRD_{4,1}(\pi, \Pi_o, w_{\dot{L}}, \mu) +$$
$$+ SRD_{4,2}(\pi, \Pi_o, w_{\dot{L}}, \mu) + SRD_{4,3}(\pi, \Pi_o, w_{\dot{L}}, \mu) \} =$$
$$= \sum_{\pi \in \Pi_e} w_{\Pi_e}(\pi) \frac{4}{3}\frac{1}{4}\frac{1}{4} \{ 6 \times (-1) + 6 \times 1 \} = 0$$

So, for every pair $\langle \Pi_e, \Pi_o \rangle$ we obtain the same result:

$$\forall \Pi_e, \Pi_o : SRD(\Pi_e, w_{\Pi_e}, \Pi_o, w_{\dot{L}}, \mu, w_S) = 0$$

Therefore, predator-prey has $General = [0, 0]$ for this property. □

C.7 Competitive Anticipation

Then, we follow with the competitive anticipation (AComp) property. As given in section 7.4.1, we want to know how much benefit the evaluated agents obtain when they anticipate competing agents. We use an average of rewards as the utility function to calculate an agent's result.

Proposition 63. *$General_{min}$ for the competitive anticipation (AComp) property is equal to -1 for the predator-prey environment.*

Proof. To find $General_{min}$ (equation 8.1), we need to find a pair $\langle \Pi_e, \Pi_o \rangle$ that minimises the property value as much as possible. We can have this situation by selecting $\Pi_e = \{\pi_{win/win'}\}$ and $\Pi_o = \{\pi_{win}\}$ (a π_{win} agent tries not to be chased when interacting as the prey and tries to perfectly coordinate with the other predators to chase the prey when interacting as a predator, and a $\pi_{win/win'}$ agent tries not to be chased when interacting as the prey and tries to perfectly coordinate with the other predators to chase the prey when interacting as a predator but from the fifth time step avoids chasing the prey).

Following definition 23 we obtain the AComp value for this $\langle \Pi_e, \Pi_o \rangle$. Since the multi-agent environment is not team symmetric, we need to calculate this property value for every pair of agent slots in different teams. Also, since $|\Pi_e| = 1$ we just need to calculate this property value for one evaluated agent. Following definition 22 we calculate the Ant value for the evaluated agent $\pi_{win/win'} \in \Pi_e$ and each pair of agent slots in different teams. We start with agent slots 1 and 2:

$$\begin{aligned}
Ant_{1,2}(\pi_{win/win'}, \Pi_o, w_{\dot{L}}, \mu) = & \sum_{\dot{l} \in \dot{L}_{-1,2}^{N(\mu)}(\Pi_o)} \sum_{\pi \in \Pi_o} w_{\dot{L}}(\dot{l} \overset{2}{\leftarrow} \pi) \times \\
& \times \frac{1}{2} \Big(R_1(\mu[\dot{l} \overset{1,2}{\leftarrow} \pi_{win/win'}, \pi]) - R_1(\mu[\dot{l} \overset{1,2}{\leftarrow} \pi_{win/win'}, \pi_r]) \Big) = \\
& = \frac{1}{1}\frac{1}{2} \big(R_1(\mu[\pi_{win/win'}, \pi_{win}, \pi_{win}, \pi_{win}]) - \\
& - R_1(\mu[\pi_{win/win'}, \pi_r, \pi_{win}, \pi_{win}]) \big)
\end{aligned}$$

In agent line-up $(\pi_{win/win'}, \pi_{win}, \pi_{win}, \pi_{win})$, the predators perfectly coordinate to always chase the prey (as seen in lemma 3), obtaining the agent in agent slot 1 $(\pi_{win/win'})$ the expected average reward of -1, while in agent line-up $(\pi_{win/win'}, \pi_r, \pi_{win}, \pi_{win})$, the agent in agent slot 1 $(\pi_{win/win'})$ is almost always able to avoid the predators, almost obtaining the expected average reward of 1. So:

$$Ant_{1,2}(\pi_{win/win'}, \Pi_o, w_{\dot{L}}, \mu) = \frac{1}{1}\frac{1}{2} [(-1) - 1] = -1$$

For agent slots 1 and 3, in agent line-up $(\pi_{win/win'}, \pi_{win}, \pi_{win}, \pi_{win})$, the agent in agent slot 1 $(\pi_{win/win'})$ obtains the expected average reward of -1, while in agent line-up $(\pi_{win/win'}, \pi_{win}, \pi_r, \pi_{win})$, the agent in agent slot 1 $(\pi_{win/win'})$ almost obtains the expected average reward of 1. So:

$$\begin{aligned}
Ant_{1,3}(\pi_{win/win'}, \Pi_o, w_{\dot{L}}, \mu) = & \sum_{\dot{l} \in \dot{L}_{-1,3}^{N(\mu)}(\Pi_o)} \sum_{\pi \in \Pi_o} w_{\dot{L}}(\dot{l} \overset{3}{\leftarrow} \pi) \times \\
& \times \frac{1}{2} \Big(R_1(\mu[\dot{l} \overset{1,3}{\leftarrow} \pi_{win/win'}, \pi]) - R_1(\mu[\dot{l} \overset{1,3}{\leftarrow} \pi_{win/win'}, \pi_r]) \Big) = \\
& = \frac{1}{1}\frac{1}{2} \big(R_1(\mu[\pi_{win/win'}, \pi_{win}, \pi_{win}, \pi_{win}]) - \\
& - R_1(\mu[\pi_{win/win'}, \pi_{win}, \pi_r, \pi_{win}]) \big) = \\
& = \frac{1}{1}\frac{1}{2} [(-1) - 1] = -1
\end{aligned}$$

For agent slots 1 and 4, in agent line-up $(\pi_{win/win'}, \pi_{win}, \pi_{win}, \pi_{win})$, the agent in agent slot 1 $(\pi_{win/win'})$ obtains the expected average reward of -1, while in agent line-up $(\pi_{win/win'}, \pi_{win}, \pi_{win}, \pi_r)$, the agent in agent slot 1 $(\pi_{win/win'})$ almost obtains the expected average reward of 1. So:

$$\begin{aligned}
Ant_{1,4}(\pi_{win/win'}, \Pi_o, w_{\dot{L}}, \mu) = & \sum_{\dot{l} \in \dot{L}_{-1,4}^{N(\mu)}(\Pi_o)} \sum_{\pi \in \Pi_o} w_{\dot{L}}(\dot{l} \overset{4}{\leftarrow} \pi) \times \\
& \times \frac{1}{2} \Big(R_1(\mu[\dot{l} \overset{1,4}{\leftarrow} \pi_{win/win'}, \pi]) - R_1(\mu[\dot{l} \overset{1,4}{\leftarrow} \pi_{win/win'}, \pi_r]) \Big) = \\
& = \frac{1}{1}\frac{1}{2} \big(R_1(\mu[\pi_{win/win'}, \pi_{win}, \pi_{win}, \pi_{win}]) - \\
& - R_1(\mu[\pi_{win/win'}, \pi_{win}, \pi_{win}, \pi_r]) \big) = \\
& = \frac{1}{1}\frac{1}{2} [(-1) - 1] = -1
\end{aligned}$$

For agent slots 2 and 1, in agent line-up $(\pi_{win}, \pi_{win/win'}, \pi_{win}, \pi_{win})$, they prey is not chased due to the miss-coordination of $\pi_{win/win'}$ in the last time steps, obtaining the agent in agent slot 2 $(\pi_{win/win'})$ the expected average reward of -1, while in agent line-up $(\pi_r, \pi_{win/win'}, \pi_{win}, \pi_{win})$, the prey is almost always chased by the predators, almost obtaining the agent in agent slot 2 $(\pi_{win/win'})$ the expected average reward of 1. So:

$$\begin{aligned}
Ant_{2,1}(\pi_{win/win'}, \Pi_o, w_{\dot{L}}, \mu) = & \sum_{\dot{l} \in \dot{L}_{-2,1}^{N(\mu)}(\Pi_o)} \sum_{\pi \in \Pi_o} w_{\dot{L}}(\dot{l} \overset{1}{\leftarrow} \pi) \times \\
& \times \frac{1}{2} \Big(R_2(\mu[\dot{l} \overset{2,1}{\leftarrow} \pi_{win/win'}, \pi]) - R_2(\mu[\dot{l} \overset{2,1}{\leftarrow} \pi_{win/win'}, \pi_r]) \Big) = \\
& = \frac{1}{1}\frac{1}{2} \big(R_2(\mu[\pi_{win}, \pi_{win/win'}, \pi_{win}, \pi_{win}]) - \\
& - R_2(\mu[\pi_r, \pi_{win/win'}, \pi_{win}, \pi_{win}]) \big) = \\
& = \frac{1}{1}\frac{1}{2} [(-1) - 1] = -1
\end{aligned}$$

For agent slots 3 and 1, in agent line-up $(\pi_{win}, \pi_{win}, \pi_{win/win'}, \pi_{win})$, the agent in agent slot 3 $(\pi_{win/win'})$ obtains the expected average reward of -1, while in agent line-up $(\pi_r, \pi_{win}, \pi_{win/win'}, \pi_{win})$, the agent in agent slot 3 $(\pi_{win/win'})$ almost obtains the expected average reward of 1. So:

$$\begin{aligned}
Ant_{3,1}(\pi_{win/win'}, \Pi_o, w_{\dot{L}}, \mu) = & \sum_{\dot{l} \in \dot{L}_{-3,1}^{N(\mu)}(\Pi_o)} \sum_{\pi \in \Pi_o} w_{\dot{L}}(\dot{l} \overset{1}{\leftarrow} \pi) \times \\
& \times \frac{1}{2} \Big(R_3(\mu[\dot{l} \overset{3,1}{\leftarrow} \pi_{win/win'}, \pi]) - R_3(\mu[\dot{l} \overset{3,1}{\leftarrow} \pi_{win/win'}, \pi_r]) \Big) = \\
& = \frac{1}{1}\frac{1}{2} \big(R_3(\mu[\pi_{win}, \pi_{win}, \pi_{win/win'}, \pi_{win}]) - \\
& - R_3(\mu[\pi_r, \pi_{win}, \pi_{win/win'}, \pi_{win}]) \big) = \\
& = \frac{1}{1}\frac{1}{2} [(-1) - 1] = -1
\end{aligned}$$

And for agent slots 4 and 1, in agent line-up $(\pi_{win}, \pi_{win}, \pi_{win}, \pi_{win/win'})$, the agent in agent slot 4 $(\pi_{win/win'})$ obtains the expected average reward of -1, while in agent line-up

$(\pi_r, \pi_{win}, \pi_{win}, \pi_{win/win'})$, the agent in agent slot 4 $(\pi_{win/win'})$ almost obtains the expected average reward of 1. So:

$$\begin{aligned}
Ant_{4,1}(\pi_{win/win'}, \Pi_o, w_{\dot{L}}, \mu) = & \sum_{\dot{l} \in \dot{L}^{N(\mu)}_{-4,1}(\Pi_o)} \sum_{\pi \in \Pi_o} w_{\dot{L}}(\dot{l} \overset{1}{\leftarrow} \pi) \times \\
& \times \frac{1}{2} \Big(R_4(\mu[\dot{l} \overset{4,1}{\leftarrow} \pi_{win/win'}, \pi]) - R_4(\mu[\dot{l} \overset{4,1}{\leftarrow} \pi_{win/win'}, \pi_r]) \Big) = \\
& = \frac{1}{1}\frac{1}{2} \big(R_4(\mu[\pi_{win}, \pi_{win}, \pi_{win}, \pi_{win/win'}]) - \\
& - R_4(\mu[\pi_r, \pi_{win}, \pi_{win}, \pi_{win/win'}]) \big) = \\
& = \frac{1}{1}\frac{1}{2} [(-1) - 1] = -1
\end{aligned}$$

And finally, we calculate the AComp value:

$$\begin{aligned}
AComp(\Pi_e, w_{\Pi_e}, \Pi_o, w_{\dot{L}}, \mu, w_S) = & \sum_{\pi \in \Pi_e} w_{\Pi_e}(\pi) \eta_{S_2^2} \sum_{t_1, t_2 \in \tau | t_1 \neq t_2} \sum_{i \in t_1} w_S(i, \mu) \sum_{j \in t_2} w_S(j, \mu) \times \\
& \times Ant_{i,j}(\pi, \Pi_o, w_{\dot{L}}, \mu) = \\
& = \frac{1}{1}\frac{8}{3}\frac{1}{4}\frac{1}{4} \times \\
& \times \{ Ant_{1,2}(\pi_{win/win'}, \Pi_o, w_{\dot{L}}, \mu) + Ant_{1,3}(\pi_{win/win'}, \Pi_o, w_{\dot{L}}, \mu) + \\
& + Ant_{1,4}(\pi_{win/win'}, \Pi_o, w_{\dot{L}}, \mu) + Ant_{2,1}(\pi_{win/win'}, \Pi_o, w_{\dot{L}}, \mu) + \\
& + Ant_{3,1}(\pi_{win/win'}, \Pi_o, w_{\dot{L}}, \mu) + Ant_{4,1}(\pi_{win/win'}, \Pi_o, w_{\dot{L}}, \mu) \} = \\
& = \frac{1}{1}\frac{8}{3}\frac{1}{4}\frac{1}{4} \{ 6 \times (-1) \} = -1
\end{aligned}$$

Since -1 is the lowest possible value for the competitive anticipation property, therefore predator-prey has $General_{min} = -1$ for this property. □

Proposition 64. *$General_{max}$ for the competitive anticipation (AComp) property is equal to 1 for the predator-prey environment.*

Proof. To find $General_{max}$ (equation 8.2), we need to find a pair $\langle \Pi_e, \Pi_o \rangle$ that maximises the property value as much as possible. We can have this situation by selecting $\Pi_e = \{\pi_{33'/br}\}$ and $\Pi_o = \{\pi_{br}\}$ (a π_{br} agent always goes to the bottom right corner and a $\pi_{33'/br}$ agent directly goes to the 3rd row 3rd column cell, but if it notices that not all the predators are directly going to the bottom right corner, then it goes to the bottom right corner when interacting as the prey and always goes to the bottom right corner when interacting as a predator).

Following definition 23 we obtain the AComp value for this $\langle \Pi_e, \Pi_o \rangle$. Since the multi-agent environment is not team symmetric, we need to calculate this property value for every pair of agent slots in different teams. Also, since $|\Pi_e| = 1$ we just need to calculate this property value for one evaluated agent. Following definition 22 we calculate the Ant value for the evaluated agent $\pi_{33'/br} \in \Pi_e$ and each pair of agent slots in different teams. We start with agent slots 1 and 2:

$$\begin{aligned}
Ant_{1,2}(\pi_{33'/br}, \Pi_o, w_{\dot{L}}, \mu) &= \sum_{\dot{l} \in \dot{L}^{N(\mu)}_{-1,2}(\Pi_o)} \sum_{\pi \in \Pi_o} w_{\dot{L}}(\dot{l} \overset{2}{\leftarrow} \pi) \times \\
&\times \frac{1}{2} \Big(R_1(\mu[\dot{l} \overset{1,2}{\leftarrow} \pi_{33'/br}, \pi]) - R_1(\mu[\dot{l} \overset{1,2}{\leftarrow} \pi_{33'/br}, \pi_r]) \Big) = \\
&= \frac{1}{1}\frac{1}{2} \left(R_1(\mu[\pi_{33'/br}, \pi_{br}, \pi_{br}, \pi_{br}]) - R_1(\mu[\pi_{33'/br}, \pi_r, \pi_{br}, \pi_{br}]) \right)
\end{aligned}$$

In agent line-up $(\pi_{33'/br}, \pi_{br}, \pi_{br}, \pi_{br})$, the agent in agent slot 1 $(\pi_{33'/br})$ obtains the expected average reward of 1, while in agent line-up $(\pi_{33'/br}, \pi_r, \pi_{br}, \pi_{br})$, the agent in agent slot 1 $(\pi_{33'/br})$ is almost always chased in the bottom right corner due it almost always notices the random movement of π_r, almost obtaining the expected average reward of -1. So:

$$Ant_{1,2}(\pi_{33'/br}, \Pi_o, w_{\dot{L}}, \mu) = \frac{1}{1}\frac{1}{2}\,[1 - (-1)] = 1$$

For agent slots 1 and 3, in agent line-up $(\pi_{33'/br}, \pi_{br}, \pi_{br}, \pi_{br})$, the agent in agent slot 1 $(\pi_{33'/br})$ obtains the expected average reward of 1, while in agent line-up $(\pi_{33'/br}, \pi_{br}, \pi_r, \pi_{br})$, the agent in agent slot 1 $(\pi_{33'/br})$ almost obtains the expected average reward of -1. So:

$$\begin{aligned}
Ant_{1,3}(\pi_{33'/br}, \Pi_o, w_{\dot{L}}, \mu) &= \sum_{\dot{l} \in \dot{L}^{N(\mu)}_{-1,3}(\Pi_o)} \sum_{\pi \in \Pi_o} w_{\dot{L}}(\dot{l} \overset{3}{\leftarrow} \pi) \times \\
&\times \frac{1}{2} \Big(R_1(\mu[\dot{l} \overset{1,3}{\leftarrow} \pi_{33'/br}, \pi]) - R_1(\mu[\dot{l} \overset{1,3}{\leftarrow} \pi_{33'/br}, \pi_r]) \Big) = \\
&= \frac{1}{1}\frac{1}{2} \left(R_1(\mu[\pi_{33'/br}, \pi_{br}, \pi_{br}, \pi_{br}]) - R_1(\mu[\pi_{33'/br}, \pi_{br}, \pi_r, \pi_{br}]) \right) = \\
&= \frac{1}{1}\frac{1}{2}\,[1 - (-1)] = 1
\end{aligned}$$

For agent slots 1 and 4, in agent line-up $(\pi_{33'/br}, \pi_{br}, \pi_{br}, \pi_{br})$, the agent in agent slot 1 $(\pi_{33'/br})$ obtains the expected average reward of 1, while in agent line-up $(\pi_{33'/br}, \pi_{br}, \pi_{br}, \pi_r)$, the agent in agent slot 1 $(\pi_{33'/br})$ almost obtains the expected average reward of -1. So:

$$\begin{aligned}
Ant_{1,4}(\pi_{33'/br}, \Pi_o, w_{\dot{L}}, \mu) &= \sum_{\dot{l} \in \dot{L}^{N(\mu)}_{-1,4}(\Pi_o)} \sum_{\pi \in \Pi_o} w_{\dot{L}}(\dot{l} \overset{4}{\leftarrow} \pi) \times \\
&\times \frac{1}{2} \Big(R_1(\mu[\dot{l} \overset{1,4}{\leftarrow} \pi_{33'/br}, \pi]) - R_1(\mu[\dot{l} \overset{1,4}{\leftarrow} \pi_{33'/br}, \pi_r]) \Big) = \\
&= \frac{1}{1}\frac{1}{2} \left(R_1(\mu[\pi_{33'/br}, \pi_{br}, \pi_{br}, \pi_{br}]) - R_1(\mu[\pi_{33'/br}, \pi_{br}, \pi_{br}, \pi_r]) \right) = \\
&= \frac{1}{1}\frac{1}{2}\,[1 - (-1)] = 1
\end{aligned}$$

For agent slots 2 and 1, in agent line-up $(\pi_{br}, \pi_{33'/br}, \pi_{br}, \pi_{br})$, the agent in agent slot 2 $(\pi_{33'/br})$ obtains the expected average reward of 1, while in agent line-up $(\pi_r, \pi_{33'/br}, \pi_{br}, \pi_{br})$, the agent in agent slot 2 $(\pi_{33'/br})$ almost obtains the expected average reward of -1. So:

$$\begin{aligned}
Ant_{2,1}(\pi_{33'/br}, \Pi_o, w_{\dot{L}}, \mu) = &\sum_{\dot{l} \in \dot{L}_{-2,1}^{N(\mu)}(\Pi_o)} \sum_{\pi \in \Pi_o} w_{\dot{L}}(\dot{l} \overset{1}{\leftarrow} \pi) \times \\
&\times \frac{1}{2}\left(R_2(\mu[\dot{l} \overset{2,1}{\leftarrow} \pi_{33'/br}, \pi]) - R_2(\mu[\dot{l} \overset{2,1}{\leftarrow} \pi_{33'/br}, \pi_r])\right) = \\
&= \frac{1}{1}\frac{1}{2}\left(R_2(\mu[\pi_{br}, \pi_{33'/br}, \pi_{br}, \pi_{br}]) - R_2(\mu[\pi_r, \pi_{33'/br}, \pi_{br}, \pi_{br}])\right) = \\
&= \frac{1}{1}\frac{1}{2}[1 - (-1)] = 1
\end{aligned}$$

For agent slots 3 and 1, in agent line-up $(\pi_{br}, \pi_{br}, \pi_{33'/br}, \pi_{br})$, the agent in agent slot 3 $(\pi_{33'/br})$ obtains the expected average reward of 1, while in agent line-up $(\pi_r, \pi_{br}, \pi_{33'/br}, \pi_{br})$, the agent in agent slot 3 $(\pi_{33'/br})$ almost obtains the expected average reward of -1. So:

$$\begin{aligned}
Ant_{3,1}(\pi_{33'/br}, \Pi_o, w_{\dot{L}}, \mu) = &\sum_{\dot{l} \in \dot{L}_{-3,1}^{N(\mu)}(\Pi_o)} \sum_{\pi \in \Pi_o} w_{\dot{L}}(\dot{l} \overset{1}{\leftarrow} \pi) \times \\
&\times \frac{1}{2}\left(R_3(\mu[\dot{l} \overset{3,1}{\leftarrow} \pi_{33'/br}, \pi]) - R_3(\mu[\dot{l} \overset{3,1}{\leftarrow} \pi_{33'/br}, \pi_r])\right) = \\
&= \frac{1}{1}\frac{1}{2}\left(R_3(\mu[\pi_{br}, \pi_{br}, \pi_{33'/br}, \pi_{br}]) - R_3(\mu[\pi_r, \pi_{br}, \pi_{33'/br}, \pi_{br}])\right) = \\
&= \frac{1}{1}\frac{1}{2}[1 - (-1)] = 1
\end{aligned}$$

And for agent slots 4 and 1, in agent line-up $(\pi_{br}, \pi_{br}, \pi_{br}, \pi_{33'/br})$, the agent in agent slot 4 $(\pi_{33'/br})$ obtains the expected average reward of 1, while in agent line-up $(\pi_r, \pi_{br}, \pi_{br}, \pi_{33'/br})$, the agent in agent slot 4 $(\pi_{33'/br})$ almost obtains the expected average reward of -1. So:

$$\begin{aligned}
Ant_{4,1}(\pi_{33'/br}, \Pi_o, w_{\dot{L}}, \mu) = &\sum_{\dot{l} \in \dot{L}_{-4,1}^{N(\mu)}(\Pi_o)} \sum_{\pi \in \Pi_o} w_{\dot{L}}(\dot{l} \overset{1}{\leftarrow} \pi) \times \\
&\times \frac{1}{2}\left(R_4(\mu[\dot{l} \overset{4,1}{\leftarrow} \pi_{33'/br}, \pi]) - R_4(\mu[\dot{l} \overset{4,1}{\leftarrow} \pi_{33'/br}, \pi_r])\right) = \\
&= \frac{1}{1}\frac{1}{2}\left(R_4(\mu[\pi_{br}, \pi_{br}, \pi_{br}, \pi_{33'/br}]) - R_4(\mu[\pi_r, \pi_{br}, \pi_{br}, \pi_{33'/br}])\right) = \\
&= \frac{1}{1}\frac{1}{2}[1 - (-1)] = 1
\end{aligned}$$

And finally, we calculate the AComp value:

$$\begin{aligned}
AComp(\Pi_e, w_{\Pi_e}, \Pi_o, w_{\dot{L}}, \mu, w_S) = &\sum_{\pi \in \Pi_e} w_{\Pi_e}(\pi) \eta_{S_2^2} \sum_{t_1, t_2 \in \tau | t_1 \neq t_2} \sum_{i \in t_1} w_S(i, \mu) \sum_{j \in t_2} w_S(j, \mu) \times \\
&\times Ant_{i,j}(\pi, \Pi_o, w_{\dot{L}}, \mu) = \\
&= \frac{1}{1}\frac{8}{3}\frac{1}{4}\frac{1}{4}\{Ant_{1,2}(\pi_{33'/br}, \Pi_o, w_{\dot{L}}, \mu) + Ant_{1,3}(\pi_{33'/br}, \Pi_o, w_{\dot{L}}, \mu) + \\
&+ Ant_{1,4}(\pi_{33'/br}, \Pi_o, w_{\dot{L}}, \mu) + Ant_{2,1}(\pi_{33'/br}, \Pi_o, w_{\dot{L}}, \mu) + \\
&+ Ant_{3,1}(\pi_{33'/br}, \Pi_o, w_{\dot{L}}, \mu) + Ant_{4,1}(\pi_{33'/br}, \Pi_o, w_{\dot{L}}, \mu)\} = \\
&= \frac{1}{1}\frac{8}{3}\frac{1}{4}\frac{1}{4}\{6 \times 1\} = 1
\end{aligned}$$

Since 1 is the highest possible value for the competitive anticipation property, therefore predator-prey has $General_{max} = 1$ for this property. $\square$

C.8 Cooperative Anticipation

Finally, we follow with the cooperative anticipation (ACoop) property. As given in section 7.4.2, we want to know how much benefit the evaluated agents obtain when they anticipate cooperating agents. We use an average of rewards as the utility function to calculate an agent's result.

Proposition 65. *$General_{min}$ for the cooperative anticipation (ACoop) property is equal to -1 for the predator-prey environment.*

Proof. To find $General_{min}$ (equation 8.1), we need to find a pair $\langle \Pi_e, \Pi_o \rangle$ that minimises the property value as much as possible. We can have this situation by selecting $\Pi_e = \{\pi_{br'}\}$ and $\Pi_o = \{\pi_{33/br}\}$ (a $\pi_{br'}$ agent directly goes to the bottom right corner, but if it notices that not all the predators are directly going to this corner, then it goes to the 3rd row 3rd column cell, and a $\pi_{33/br}$ agent always goes to the 3rd row 3rd column cell when interacting as the prey and directly goes to the bottom right corner when interacting as a predator).

Following definition 24 we obtain the ACoop value for this $\langle \Pi_e, \Pi_o \rangle$. Since the multi-agent environment is not team symmetric and it only has two teams, we need to calculate this property value for every pair of agent slots in the same team. Also, since $|\Pi_e| = 1$ we just need to calculate this property value for one evaluated agent. Following definition 22 we calculate the Ant value for the evaluated agent $\pi_{br'} \in \Pi_e$ and each pair of agent slots in the same team. We start with agent slots 2 and 3:

$$\begin{aligned} Ant_{2,3}(\pi_{br'}, \Pi_o, w_{\dot{L}}, \mu) = & \sum_{\dot{l} \in \dot{L}_{-2,3}^{N(\mu)}(\Pi_o)} \sum_{\pi \in \Pi_o} w_{\dot{L}}(\dot{l} \overset{3}{\leftarrow} \pi) \times \\ & \times \frac{1}{2} \Big(R_2(\mu[\dot{l} \overset{2,3}{\leftarrow} \pi_{br'}, \pi]) - R_2(\mu[\dot{l} \overset{2,3}{\leftarrow} \pi_{br'}, \pi_r]) \Big) = \\ = & \frac{1}{1}\frac{1}{2} \left(R_2(\mu[\pi_{33/br}, \pi_{br'}, \pi_{33/br}, \pi_{33/br}]) - R_2(\mu[\pi_{33/br}, \pi_{br'}, \pi_r, \pi_{33/br}]) \right) \end{aligned}$$

In agent line-up $(\pi_{33/br}, \pi_{br'}, \pi_{33/br}, \pi_{33/br})$, the predators directly go to the bottom right cell and the prey goes to the 3rd row 3rd column cell, obtaining the agent in agent slot 2 ($\pi_{br'}$) the expected average reward of -1, while in agent line-up $(\pi_{33/br}, \pi_{br'}, \pi_r, \pi_{33/br})$, the agent in agent slot 2 ($\pi_{br'}$) almost always chases the prey in the 3rd row 3rd column cell due it almost always notices the random movement of π_r, almost obtaining the expected average reward of 1. So:

$$Ant_{2,3}(\pi_{br'}, \Pi_o, w_{\dot{L}}, \mu) = \frac{1}{1}\frac{1}{2}[(-1) - 1] = -1$$

For agent slots 2 and 4, in agent line-up $(\pi_{33/br}, \pi_{br'}, \pi_{33/br}, \pi_{33/br})$, the agent in agent slot 2 ($\pi_{br'}$) obtains the expected average reward of -1, while in agent line-up $(\pi_{33/br}, \pi_{br'}, \pi_{33/br}, \pi_r)$, the agent in agent slot 2 ($\pi_{br'}$) almost obtains the expected average reward of 1. So:

$$\begin{aligned}
Ant_{2,4}(\pi_{br'}, \Pi_o, w_{\dot{L}}, \mu) = & \sum_{\dot{l} \in \dot{L}^{N(\mu)}_{-2,4}(\Pi_o)} \sum_{\pi \in \Pi_o} w_{\dot{L}}(\dot{l} \overset{4}{\leftarrow} \pi) \times \\
& \times \frac{1}{2} \Big(R_2(\mu[\dot{l} \overset{2,4}{\leftarrow} \pi_{br'}, \pi]) - R_2(\mu[\dot{l} \overset{2,4}{\leftarrow} \pi_{br'}, \pi_r]) \Big) = \\
& = \frac{1}{1}\frac{1}{2} \big(R_2(\mu[\pi_{33/br}, \pi_{br'}, \pi_{33/br}, \pi_{33/br}]) - R_2(\mu[\pi_{33/br}, \pi_{br'}, \pi_{33/br}, \pi_r]) \big) = \\
& = \frac{1}{1}\frac{1}{2} [(-1) - 1] = -1
\end{aligned}$$

For agent slots 3 and 2, in agent line-up $(\pi_{33/br}, \pi_{33/br}, \pi_{br'}, \pi_{33/br})$, the agent in agent slot 3 $(\pi_{br'})$ obtains the expected average reward of -1, while in agent line-up $(\pi_{33/br}, \pi_r, \pi_{br'}, \pi_{33/br})$, the agent in agent slot 3 $(\pi_{br'})$ almost obtains the expected average reward of 1. So:

$$\begin{aligned}
Ant_{3,2}(\pi_{br'}, \Pi_o, w_{\dot{L}}, \mu) = & \sum_{\dot{l} \in \dot{L}^{N(\mu)}_{-3,2}(\Pi_o)} \sum_{\pi \in \Pi_o} w_{\dot{L}}(\dot{l} \overset{2}{\leftarrow} \pi) \times \\
& \times \frac{1}{2} \Big(R_3(\mu[\dot{l} \overset{3,2}{\leftarrow} \pi_{br'}, \pi]) - R_3(\mu[\dot{l} \overset{3,2}{\leftarrow} \pi_{br'}, \pi_r]) \Big) = \\
& = \frac{1}{1}\frac{1}{2} \big(R_3(\mu[\pi_{33/br}, \pi_{33/br}, \pi_{br'}, \pi_{33/br}]) - R_3(\mu[\pi_{33/br}, \pi_r, \pi_{br'}, \pi_{33/br}]) \big) = \\
& = \frac{1}{1}\frac{1}{2} [(-1) - 1] = -1
\end{aligned}$$

For agent slots 3 and 4, in agent line-up $(\pi_{33/br}, \pi_{33/br}, \pi_{br'}, \pi_{33/br})$, the agent in agent slot 3 $(\pi_{br'})$ obtains the expected average reward of -1, while in agent line-up $(\pi_{33/br}, \pi_{33/br}, \pi_{br'}, \pi_r)$, the agent in agent slot 3 $(\pi_{br'})$ almost obtains the expected average reward of 1. So:

$$\begin{aligned}
Ant_{3,4}(\pi_{br'}, \Pi_o, w_{\dot{L}}, \mu) = & \sum_{\dot{l} \in \dot{L}^{N(\mu)}_{-3,4}(\Pi_o)} \sum_{\pi \in \Pi_o} w_{\dot{L}}(\dot{l} \overset{4}{\leftarrow} \pi) \times \\
& \times \frac{1}{2} \Big(R_3(\mu[\dot{l} \overset{3,4}{\leftarrow} \pi_{br'}, \pi]) - R_3(\mu[\dot{l} \overset{3,4}{\leftarrow} \pi_{br'}, \pi_r]) \Big) = \\
& = \frac{1}{1}\frac{1}{2} \big(R_3(\mu[\pi_{33/br}, \pi_{33/br}, \pi_{br'}, \pi_{33/br}]) - R_3(\mu[\pi_{33/br}, \pi_{33/br}, \pi_{br'}, \pi_r]) \big) = \\
& = \frac{1}{1}\frac{1}{2} [(-1) - 1] = -1
\end{aligned}$$

For agent slots 4 and 2, in agent line-up $(\pi_{33/br}, \pi_{33/br}, \pi_{33/br}, \pi_{br'})$, the agent in agent slot 4 $(\pi_{br'})$ obtains the expected average reward of -1, while in agent line-up $(\pi_{33/br}, \pi_r, \pi_{33/br}, \pi_{br'})$, the agent in agent slot 4 $(\pi_{br'})$ almost obtains the expected average reward of 1. So:

$$\begin{aligned}
Ant_{4,2}(\pi_{br'}, \Pi_o, w_{\dot{L}}, \mu) = & \sum_{\dot{l} \in \dot{L}_{-4,2}^{N(\mu)}(\Pi_o)} \sum_{\pi \in \Pi_o} w_{\dot{L}}(\dot{l} \overset{2}{\leftarrow} \pi) \times \\
& \times \frac{1}{2} \Big(R_4(\mu[\dot{l} \overset{4,2}{\leftarrow} \pi_{br'}, \pi]) - R_4(\mu[\dot{l} \overset{4,2}{\leftarrow} \pi_{br'}, \pi_r]) \Big) = \\
& = \frac{1}{1}\frac{1}{2} \big(R_4(\mu[\pi_{33/br}, \pi_{33/br}, \pi_{33/br}, \pi_{br'}]) - R_4(\mu[\pi_{33/br}, \pi_r, \pi_{33/br}, \pi_{br'}]) \big) = \\
& = \frac{1}{1}\frac{1}{2} [(-1) - 1] = -1
\end{aligned}$$

And for agent slots 4 and 3, in agent line-up $(\pi_{33/br}, \pi_{33/br}, \pi_{33/br}, \pi_{br'})$, the agent in agent slot 4 $(\pi_{br'})$ obtains the expected average reward of -1, while in agent line-up $(\pi_{33/br}, \pi_{33/br}, \pi_r, \pi_{br'})$, the agent in agent slot 4 $(\pi_{br'})$ almost obtains the expected average reward of 1. So:

$$\begin{aligned}
Ant_{4,3}(\pi_{br'}, \Pi_o, w_{\dot{L}}, \mu) = & \sum_{\dot{l} \in \dot{L}_{-4,3}^{N(\mu)}(\Pi_o)} \sum_{\pi \in \Pi_o} w_{\dot{L}}(\dot{l} \overset{3}{\leftarrow} \pi) \times \\
& \times \frac{1}{2} \Big(R_4(\mu[\dot{l} \overset{4,3}{\leftarrow} \pi_{br'}, \pi]) - R_4(\mu[\dot{l} \overset{4,3}{\leftarrow} \pi_{br'}, \pi_r]) \Big) = \\
& = \frac{1}{1}\frac{1}{2} \big(R_4(\mu[\pi_{33/br}, \pi_{33/br}, \pi_{33/br}, \pi_{br'}]) - R_4(\mu[\pi_{33/br}, \pi_{33/br}, \pi_r, \pi_{br'}]) \big) = \\
& = \frac{1}{1}\frac{1}{2} [(-1) - 1] = -1
\end{aligned}$$

And finally, we calculate the ACoop value:

$$\begin{aligned}
ACoop(\Pi_e, w_{\Pi_e}, \Pi_o, w_{\dot{L}}, \mu, w_S) = & \sum_{\pi \in \Pi_e} w_{\Pi_e}(\pi) \eta_{S_3^2} \sum_{t \in \tau} \sum_{i,j \in t | i \neq j} w_S(i, \mu) w_S(j, \mu) \times \\
& \times Ant_{i,j}(\pi, \Pi_o, w_{\dot{L}}, \mu) = \\
& = \frac{1}{1}\frac{8}{3}\frac{1}{4}\frac{1}{4} \{ Ant_{2,3}(\pi_{br'}, \Pi_o, w_{\dot{L}}, \mu) + Ant_{2,4}(\pi_{br'}, \Pi_o, w_{\dot{L}}, \mu) + \\
& + Ant_{3,2}(\pi_{br'}, \Pi_o, w_{\dot{L}}, \mu) + Ant_{3,4}(\pi_{br'}, \Pi_o, w_{\dot{L}}, \mu) + \\
& + Ant_{4,2}(\pi_{br'}, \Pi_o, w_{\dot{L}}, \mu) + Ant_{4,3}(\pi_{br'}, \Pi_o, w_{\dot{L}}, \mu) \} = \\
& = \frac{1}{1}\frac{8}{3}\frac{1}{4}\frac{1}{4} \{ 6 \times (-1) \} = -1
\end{aligned}$$

Since -1 is the lowest possible value for the cooperative anticipation property, therefore predator-prey has $General_{min} = -1$ for this property. □

Proposition 66. *$General_{max}$ for the cooperative anticipation (ACoop) property is equal to 1 for the predator-prey environment.*

Proof. To find $General_{max}$ (equation 8.2), we need to find a pair $\langle \Pi_e, \Pi_o \rangle$ that maximises the property value as much as possible. We can have this situation by selecting $\Pi_e = \{\pi_{win}\}$ and $\Pi_o = \{\pi_{win}\}$ (a π_{win} agent tries not to be chased when interacting as the prey and tries to perfectly coordinate with the other predators to chase the prey when interacting as a predator).

Following definition 24 we obtain the ACoop value for this $\langle \Pi_e, \Pi_o \rangle$. Since the multi-agent environment is not team symmetric and it only has two teams, we need to calculate this property value for every pair of agent slots in the same team. Also, since $|\Pi_e| = 1$ we just need to calculate this property value for one evaluated agent. Following definition 22 we calculate the Ant value for the evaluated agent $\pi_{win} \in \Pi_e$ and each pair of agent slots in the same team. We start with agent slots 2 and 3:

$$\begin{aligned} Ant_{2,3}(\pi_{win}, \Pi_o, w_{\dot{L}}, \mu) &= \sum_{\dot{l} \in \dot{L}_{-2,3}^{N(\mu)}(\Pi_o)} \sum_{\pi \in \Pi_o} w_{\dot{L}}(\dot{l} \overset{3}{\leftarrow} \pi) \times \\ &\times \frac{1}{2} \Big(R_2(\mu[\dot{l} \overset{2,3}{\leftarrow} \pi_{win}, \pi]) - R_2(\mu[\dot{l} \overset{2,3}{\leftarrow} \pi_{win}, \pi_r]) \Big) = \\ &= \frac{1}{1}\frac{1}{2} \left(R_2(\mu[\pi_{win}, \pi_{win}, \pi_{win}, \pi_{win}]) - R_2(\mu[\pi_{win}, \pi_{win}, \pi_r, \pi_{win}]) \right) \end{aligned}$$

In agent line-up $(\pi_{win}, \pi_{win}, \pi_{win}, \pi_{win})$, the predators perfectly coordinate to always chase the prey (as seen in lemma 3), obtaining the agent in agent slot 2 (π_{win}) the expected average reward of 1, while in agent line-up $(\pi_{win}, \pi_{win}, \pi_r, \pi_{win})$, the agent in agent slot 2 (π_{win}) almost never chases the prey due the random agent does not coordinate with the other predators, almost obtaining the expected average reward of -1. So:

$$Ant_{2,3}(\pi_{win}, \Pi_o, w_{\dot{L}}, \mu) = \frac{1}{1}\frac{1}{2} [1 - (-1)] = 1$$

For agent slots 2 and 4, in agent line-up $(\pi_{win}, \pi_{win}, \pi_{win}, \pi_{win})$, the agent in agent slot 2 (π_{win}) obtains the expected average reward of 1, while in agent line-up $(\pi_{win}, \pi_{win}, \pi_{win}, \pi_r)$, the agent in agent slot 2 (π_{win}) almost obtains the expected average reward of -1. So:

$$\begin{aligned} Ant_{2,4}(\pi_{win}, \Pi_o, w_{\dot{L}}, \mu) &= \sum_{\dot{l} \in \dot{L}_{-2,4}^{N(\mu)}(\Pi_o)} \sum_{\pi \in \Pi_o} w_{\dot{L}}(\dot{l} \overset{4}{\leftarrow} \pi) \times \\ &\times \frac{1}{2} \Big(R_2(\mu[\dot{l} \overset{2,4}{\leftarrow} \pi_{win}, \pi]) - R_2(\mu[\dot{l} \overset{2,4}{\leftarrow} \pi_{win}, \pi_r]) \Big) = \\ &= \frac{1}{1}\frac{1}{2} \left(R_2(\mu[\pi_{win}, \pi_{win}, \pi_{win}, \pi_{win}]) - R_2(\mu[\pi_{win}, \pi_{win}, \pi_{win}, \pi_r]) \right) = \\ &= \frac{1}{1}\frac{1}{2} [1 - (-1)] = 1 \end{aligned}$$

For agent slots 3 and 2, in agent line-up $(\pi_{win}, \pi_{win}, \pi_{win}, \pi_{win})$, the agent in agent slot 3 (π_{win}) obtains the expected average reward of 1, while in agent line-up $(\pi_{win}, \pi_r, \pi_{win}, \pi_{win})$, the agent in agent slot 3 (π_{win}) almost obtains the expected average reward of -1. So:

$$\begin{aligned} Ant_{3,2}(\pi_{win}, \Pi_o, w_{\dot{L}}, \mu) &= \sum_{\dot{l} \in \dot{L}_{-3,2}^{N(\mu)}(\Pi_o)} \sum_{\pi \in \Pi_o} w_{\dot{L}}(\dot{l} \overset{2}{\leftarrow} \pi) \times \\ &\times \frac{1}{2} \Big(R_3(\mu[\dot{l} \overset{3,2}{\leftarrow} \pi_{win}, \pi]) - R_3(\mu[\dot{l} \overset{3,2}{\leftarrow} \pi_{win}, \pi_r]) \Big) = \\ &= \frac{1}{1}\frac{1}{2} \left(R_3(\mu[\pi_{win}, \pi_{win}, \pi_{win}, \pi_{win}]) - R_3(\mu[\pi_{win}, \pi_r, \pi_{win}, \pi_{win}]) \right) = \\ &= \frac{1}{1}\frac{1}{2} [1 - (-1)] = 1 \end{aligned}$$

For agent slots 3 and 4, in agent line-up $(\pi_{win}, \pi_{win}, \pi_{win}, \pi_{win})$, the agent in agent slot 3 (π_{win}) obtains the expected average reward of 1, while in agent line-up $(\pi_{win}, \pi_{win}, \pi_{win}, \pi_r)$, the agent in agent slot 3 (π_{win}) almost obtains the expected average reward of -1. So:

$$\begin{aligned} Ant_{3,4}(\pi_{win}, \Pi_o, w_{\dot{L}}, \mu) = & \sum_{\dot{l} \in \dot{L}_{-3,4}^{N(\mu)}(\Pi_o)} \sum_{\pi \in \Pi_o} w_{\dot{L}}(\dot{l} \overset{4}{\leftarrow} \pi) \times \\ & \times \frac{1}{2} \left(R_3(\mu[\dot{l} \overset{3,4}{\leftarrow} \pi_{win}, \pi]) - R_3(\mu[\dot{l} \overset{3,4}{\leftarrow} \pi_{win}, \pi_r]) \right) = \\ & = \frac{1}{1}\frac{1}{2} \left(R_3(\mu[\pi_{win}, \pi_{win}, \pi_{win}, \pi_{win}]) - R_3(\mu[\pi_{win}, \pi_{win}, \pi_{win}, \pi_r]) \right) = \\ & = \frac{1}{1}\frac{1}{2} [1 - (-1)] = 1 \end{aligned}$$

For agent slots 4 and 2, in agent line-up $(\pi_{win}, \pi_{win}, \pi_{win}, \pi_{win})$, the agent in agent slot 4 (π_{win}) obtains the expected average reward of 1, while in agent line-up $(\pi_{win}, \pi_r, \pi_{win}, \pi_{win})$, the agent in agent slot 4 (π_{win}) almost obtains the expected average reward of -1. So:

$$\begin{aligned} Ant_{4,2}(\pi_{win}, \Pi_o, w_{\dot{L}}, \mu) = & \sum_{\dot{l} \in \dot{L}_{-4,2}^{N(\mu)}(\Pi_o)} \sum_{\pi \in \Pi_o} w_{\dot{L}}(\dot{l} \overset{2}{\leftarrow} \pi) \times \\ & \times \frac{1}{2} \left(R_4(\mu[\dot{l} \overset{4,2}{\leftarrow} \pi_{win}, \pi]) - R_4(\mu[\dot{l} \overset{4,2}{\leftarrow} \pi_{win}, \pi_r]) \right) = \\ & = \frac{1}{1}\frac{1}{2} \left(R_4(\mu[\pi_{win}, \pi_{win}, \pi_{win}, \pi_{win}]) - R_4(\mu[\pi_{win}, \pi_r, \pi_{win}, \pi_{win}]) \right) = \\ & = \frac{1}{1}\frac{1}{2} [1 - (-1)] = 1 \end{aligned}$$

And for agent slots 4 and 3, in agent line-up $(\pi_{win}, \pi_{win}, \pi_{win}, \pi_{win})$, the agent in agent slot 4 (π_{win}) obtains the expected average reward of 1, while in agent line-up $(\pi_{win}, \pi_{win}, \pi_r, \pi_{win})$, the agent in agent slot 4 (π_{win}) almost obtains the expected average reward of -1. So:

$$\begin{aligned} Ant_{4,3}(\pi_{win}, \Pi_o, w_{\dot{L}}, \mu) = & \sum_{\dot{l} \in \dot{L}_{-4,3}^{N(\mu)}(\Pi_o)} \sum_{\pi \in \Pi_o} w_{\dot{L}}(\dot{l} \overset{3}{\leftarrow} \pi) \times \\ & \times \frac{1}{2} \left(R_4(\mu[\dot{l} \overset{4,3}{\leftarrow} \pi_{win}, \pi]) - R_4(\mu[\dot{l} \overset{4,3}{\leftarrow} \pi_{win}, \pi_r]) \right) = \\ & = \frac{1}{1}\frac{1}{2} \left(R_4(\mu[\pi_{win}, \pi_{win}, \pi_{win}, \pi_{win}]) - R_4(\mu[\pi_{win}, \pi_{win}, \pi_r, \pi_{win}]) \right) = \\ & = \frac{1}{1}\frac{1}{2} [1 - (-1)] = 1 \end{aligned}$$

And finally, we calculate the ACoop value:

$$
\begin{aligned}
ACoop(\Pi_e, w_{\Pi_e}, \Pi_o, w_{\dot{L}}, \mu, w_S) &= \sum_{\pi \in \Pi_e} w_{\Pi_e}(\pi) \eta_{S_3^2} \sum_{t \in \tau} \sum_{i,j \in t | i \neq j} w_S(i, \mu) w_S(j, \mu) \times \\
&\times Ant_{i,j}(\pi, \Pi_o, w_{\dot{L}}, \mu) = \\
&= \frac{1}{1}\frac{8}{3}\frac{1}{4}\frac{1}{4} \{ Ant_{2,3}(\pi_{win}, \Pi_o, w_{\dot{L}}, \mu) + Ant_{2,4}(\pi_{win}, \Pi_o, w_{\dot{L}}, \mu) + \\
&+ Ant_{3,2}(\pi_{win}, \Pi_o, w_{\dot{L}}, \mu) + Ant_{3,4}(\pi_{win}, \Pi_o, w_{\dot{L}}, \mu) + \\
&+ Ant_{4,2}(\pi_{win}, \Pi_o, w_{\dot{L}}, \mu) + Ant_{4,3}(\pi_{win}, \Pi_o, w_{\dot{L}}, \mu) \} = \\
&= \frac{1}{1}\frac{8}{3}\frac{1}{4}\frac{1}{4} \{ 6 \times 1 \} = 1
\end{aligned}
$$

Since 1 is the highest possible value for the cooperative anticipation property, therefore predator-prey has $General_{max} = 1$ for this property. □