Hernández Orallo, J.; Martínez Plumed, F.; Schmid, U.; Siebers, M.; Dowe, DL. (2016). Computer models solving intelligence test problems: Progress and implications. Artificial Intelligence. 230:74-107. doi:10.1016/j.artint.2015.09.011.

Computer models solving intelligence test problems: progress and implications

José Hernández-Orallo[a], Fernando Martínez-Plumed[a], Ute Schmid[b], Michael Siebers[b], David L. Dowe[c]

[a] *Dept. of Computer Systems and Computation, Universitat Politècnica de València, 46022 Valencia, Spain*
[b] *Faculty of Information Systems and Applied Computer Science, University of Bamberg, 96045 Bamberg, Germany*
[c] *Computer Science and Software Engineering, Faculty of I.T., Monash University, Clayton, Vic. 3800, Australia*

Abstract

While some computational models of intelligence test problems were proposed throughout the second half of the XXth century, in the first years of the XXIst century we have seen an increasing number of computer systems being able to score well on particular intelligence test tasks. However, despite this increasing trend there has been no general account of all these works in terms of how they relate to each other and what their real achievements are. Also, there is poor understanding about what intelligence tests measure in machines, whether they are useful to evaluate AI systems, whether they are really challenging problems, and whether they are useful to understand (human) intelligence. In this paper, we provide some insight on these issues, in the form of nine specific questions, by giving a comprehensive account of about thirty computer models, from the 1960s to nowadays, and their relationships, focussing on the range of intelligence test tasks they address, the purpose of the models, how general or specialised these models are, the AI techniques they use in each case, their comparison with human performance, and their evaluation of item difficulty. As a conclusion, these tests and the computer models attempting them show that AI is still lacking general techniques to deal with a variety of problems at the same time. Nonetheless, a renewed attention on these problems and a more careful understanding of what intelligence tests offer for AI may help build new bridges between psychometrics, cognitive science, and AI; and may motivate new kinds of problem repositories.

Keywords: intelligence tests; cognitive models; artificial intelligence; intelligence evaluation.

1. Introduction

Artificial intelligence (AI) is typically defined as "the scientific understanding of the mechanisms underlying thought and intelligent behaviour and their embodiment in machines"[1]. Associated with the notion of natural and artificial intelligent agents is our (human-centred or anthropocentric) belief that intelligence underlies most human behaviour. The origin of AI research was based on the conviction that "intelligence is the computational part of the ability to achieve goals in the world" [1].

Indeed, AI research can claim some impressive milestones. For example, already in 1959 Arthur Samuel presented a self-learning program that could play checkers (or draughts) [2]. In 2002 the 1957 prophecy of Herbert Simon that within 10 years a computer would be world's chess champion (eventually) came true when Deep Blue won against the human chess champion Garry Kasparov [3]. In 2010 IBM's program Watson [4, 5] was the winner of the *Jeopardy!* TV quiz. However, one can ask whether the mechanisms underlying the behaviour of these programs is the same as or similar to the mechanism underlying human intelligent behaviour. In fact, this success in specialised tasks is a very illustrative demonstration of the

Email addresses: jorallo@dsic.upv.com (José Hernández-Orallo), fmartinez@dsic.upv.com (Fernando Martínez-Plumed), ute.schmid@uni-bamberg.de (Ute Schmid), michael.siebers@uni-bamberg.de (Michael Siebers), david.dowe@monash.edu (David L. Dowe)

[1] See homepage of the Association for the Advancement of Artificial Intelligence, http://www.aaai.org.

big switch approach in AI research. If we fix the switch to one particular problem, we can devise, after several years or decades of research, a system that performs better than humans. We can even embed many specialised programs into a system and devise an automated switch. For instance, if we have a good program for checkers, a good program for chess, etc., we can devise a meta-system that is able to recognise which kind of game has to be played and switch to the appropriate program. If AI evaluation is based on specific benchmarks that are known beforehand, non-intelligent systems will be able to thrive. Game playing is a good example where a reaction against this specialisation is beginning to flourish. Since 2005 the performance of game playing systems is evaluated in the game playing competition [6] on a wide variety of games—some invented ones not disclosed to the participants until the competition. Consequently, in the area of game playing, systems using a big switch approach are hardly successful in contrast to systems realising general game playing algorithms.

While successfully playing games can be seen as a special manifestation of intelligent behaviour, intelligence tests assess the underlying ability to act intelligently in many different domains [7]. In psychology research the classical approach to intelligence assessment is to apply psychometric tests measuring intelligence [7]. Some of these tests, the so-called IQ tests, are standardised in such a way that humans can be classified as below, about, or above average intelligence. Nonetheless, there are many other human intelligence tests and cognitive tests that also measure intelligence. In addition, other similar tests from other areas were not originally targeted to humans. In what follows, for simplicity, we will use the term *intelligence tests* for all of them. The intelligence test tasks address a variety of reasoning abilities, for example, solving number series problems, detecting regularities in spatial configurations, or understanding verbal analogies. Some types of problems are rather independent of the subject's educational and cultural background, others depend on background knowledge.

In early AI research, the intelligence test approach was considered as a useful approach for AI programs as well; Newell argued that one of the ways artificial intelligence could be achieved was "to construct a single program that would take a standard intelligence test, say the WAIS or the Stanford-Binet" [8]. And indeed, as early as 1963, Evans devised an AI program that could solve geometric analogy tasks from the WAIS (Wechsler Adult Intelligence Scale) test [9, 10] and, in the same year, Simon and Kotovsky [11] presented a program that could solve Thurstone letter series completion problems [12]. Both types of problems address the ability to identify regularities in patterns and generalise over them. This connection between inductive inference and intelligence tests was also identified early on, for instance by Blum and Blum [13]: "Intelligence tests occasionally require the extrapolation of an effective sequence".

After the initial interest of AI research in intelligence test problems, this branch of research sank into oblivion during the next decades. However, in the 1990s, cognitive science research recovered this line of research, and cognitive models were proposed to simulate the human cognitive processes that take place when solving inductive inference in intelligence test problems [14].

In AI, forty years after the work of Evans and Simon & Kotovsky, in 2003, computer programs solving intelligence tests became of interest again. On one hand, Sanghi and Dowe [15] wanted to make a conclusive point about how easy it was to make non-intelligent machines pass intelligence tests. On the other hand, Bringsjord and Schimanski aimed at resuscitating the role of psychometric tests—including not only intelligence tests but also tests about personality, artistic creativity, etc.—in AI [16]. They claimed that psychometric tests should not be dismissed but placed at a definitional, major role for what artificial intelligence is and proposed "psychometric artificial intelligence" (PAI) as a direction of research. While this approach moves towards an ability-oriented categorisation of problems instead of the classical task-oriented categorisation, it is not clear whether it is free from the big switch approach, especially if the kinds of tasks that appear in intelligence tests are known beforehand. In fact, Sanghi and Dowe's system used a big switch approach. Given these two opposite stances on the use of intelligence tests, it is hard to tell yet how influential they have been or will be.

But the fact is that the past ten (and especially five) years have been blooming with computational models aimed at solving intelligence test problems. The diversity of goals and approaches has also widened, including the use of intelligence tests for the analysis of what intelligence is, for the understanding of certain aspects of human cognition, for the evaluation of some AI techniques or systems, including robots, and, simply, to have more insights about what intelligence tests really represent. We will use the term *computer*

model for all these approaches, independently of their purpose, of the employed techniques, and of the range of problems they are able to address.

In the current state of research, there exist many systems addressing different intelligence tests. Currently, there is no general framework to characterise all intelligence tests. Therefore, it is unclear whether instances of problems are members of the same or different problem classes. With the exception of *big switch* approaches, like the one of Sanghi and Dowe [15], current systems are based on algorithmic approaches specifically designed to solve a special class of such problems and even only one specific test. It is an open question whether a general algorithmic approach for a diverse set of intelligence tests can be designed in principle. Comparing the current situation of computer models which solve intelligence tests with the history of game playing, it took about 55 years before the rise of general game playing. From the beginning of AI research and over the next decades, specialised programs for playing games, such as checkers [2], backgammon [17], chess [3] and GO [18, 19], were proposed. Around 1970, the first ideas about a system generally able to play games were discussed. For instance, Pitrat [20, 21] proposed an algorithmic framework for a class of board games and formulated some first ideas about a general language to describe the rules of the games. However, it was in 2005 when a suitable general game description language was introduced [22], which allows the characterisation of a large variety of games and the establishment of a repository based on this common representation. A common language was the prerequisite for a thorough and meaningful comparison of different approaches in the domain of game playing that is since 2005 realised in the annual general game playing competition [6, 23].

Extending the parallel from the history of general game playing to computer models solving intelligence tests, the composition of an inventory of problems and the proposal of some criteria to compare the different systems is a crucial step. We hope that our work can be a starting point towards a more systematic and general treatment of computer models solving intelligence tests and towards the development of (more) general intelligent systems passing arbitrary intelligence tests.

With regard to the classes of problems and systems considered in our inventory, we will not cover any informal conception about how any of these problems can be solved (e.g., from psychology) if this conception is not accompanied by a computer implementation and the application of the system to a set of intelligence test problems. Nonetheless, we will be as inclusive as possible in the purpose of these systems and the area or research question that motivated the study, be it psychology, artificial intelligence, cognitive science, or robotics. We analyse all the computer models taking intelligence tests (or as many as we could find, about thirty in total), starting with Evans's ANALOGY [9, 10] and going through to Spaun [24], a noteworthy 2.5-million-neuron artificial model brain that has received considerable interest [25, 26].

This analysis will also help us to have a better understanding of the relevance and connections of these approaches, and draw some conclusions about their usefulness. Overall, the main goal of the paper is to understand the meaning, utility, and impact of these computer models taking intelligence tests, and explore the progress and implications of this area of research.

The rest of the paper is organised as follows. In section 2 we discuss the diversity of approaches when assessing intelligence, in humans and artificial intelligence, and the features that intelligence tests represent in this context. In section 3 some of the most common types of intelligence test problems, and what they are supposed to measure in humans, are summarised, with a more detailed exposition of each problem in the appendix. In section 4 we will introduce, discuss, and explain the (nine) questions we want to investigate in this paper, and derive a set of criteria that will be used to analyse the computer models. In section 5, we will go chronologically through all the computer models. The results will be summarised in a table at the end of this section. A technical analysis of the systems and the relation with the kinds of problems solved is more deeply examined in section 6. From the table and the technical analysis, in section 7 we will address a series of questions, including a discussion about the insights and implications we can infer from this study. Finally, some possible directions and perspectives of integration close the paper.

2. Approaches to identifying intelligence

Intelligence is a trait that has been recognised in several degrees and qualities in humans, and to a lesser extent, other animals. While in this paper we will focus on computer models addressing intelligence test

problems, there are many other ways of recognising, identifying, and ultimately measuring intelligence of humans, non-human animals, and machines. We will briefly overview these approaches below. The following approaches (see Figure 1) explicitly or implicitly assume that intelligence can be recognised, identified, and measured by observation (i.e., by behaviour). This does not necessarily imply that these approaches take a stance in favour of Fodor's functionalism, Dennett's intentionalism, or other philosophical views about cognition, neither do they necessarily align with strong AI or weak AI (except perhaps section 2.2 below), but rather that these methods do not use brain scanning or other methods (e.g., inspecting a machine code) to identify internal mechanisms that could be related to intelligence or other cognitive abilities.

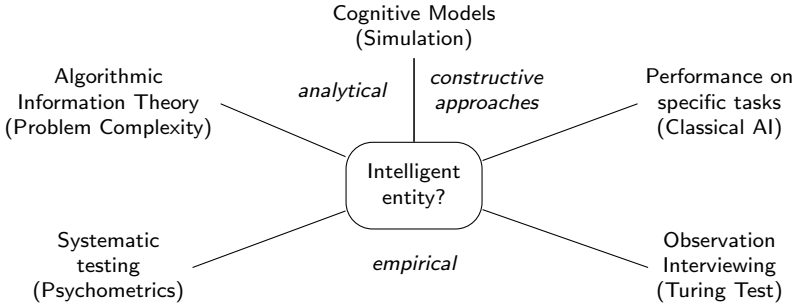


Figure 1: Different approaches to assess the intelligence of an entity.

2.1. Observation and interviewing

The oldest approach to identifying intelligence is through observation. By observing the behaviour of animals and humans [27], we have been able to determine (initially in an informal way but then more scientifically) that some animals have better abilities than others, and also that some humans are more intelligent than others. Observation is a very powerful tool, and ethology and psychology have been built upon observation of behaviour in natural or artificial environments.

A more efficient way of identifying intelligence and other cognitive abilities in humans is by conversation or, more technically, by interviewing. This is a common practice in psychology, educational contexts, and personnel selection. By carefully designing a set of questions, and by shrewdly adapting the questionnaire as the subject answers the previous questions, the assessment of cognitive abilities, personality traits, and even the identification of psychological problems can be done more efficiently than by mere observation.

A similar approach can be taken in artificial intelligence. Machines can be evaluated by observation and—for systems with a natural language interface, such as conversational bots—we can even conduct an interview. This approach to intelligence assessment is typically used by default, especially if there is agreement about the criteria to be met within the scientific community. For instance, state-of-the-art AI systems are said to be less intelligent than humans just by mere observation, or machine translators are said to be worse than (professional) human translators just by observation.

Turing's imitation game is, in fact, an interesting variant of this approach based on observation and interviewing. In his seminal paper [28], Alan Turing raised the question of whether machines can think by proposing a procedure to test whether a computer program is intelligent—the imitation game. Here, a human judge has to guess which of two players, one of them a human, one a computer, is human by some kind of Teletype interaction [29, 30, 31]. Some implementations exist, e.g., the Loebner prize competition (`http://www.loebner.net/Prizef/loebner-prize.html`).

The Turing Test provides a work-around to avoid the necessity to define the concept of thinking by only considering the relation between inputs (the questions) and outputs (the answers). The Turing Test was criticised, among others, by Searle [32], who introduced the thought experiment "The Chinese Room", which argued that a computer could pass the Turing Test by simple symbol manipulation without "thinking" in a way humans do. This critique is related to the common confusion between intelligence and consciousness. A much more fundamental criticism to the Turing Test (or a simple one-on-one interview) as a way to identify

intelligence is that non-intelligent systems can obtain relative success at the (imitation) game, such as ELIZA [33], which was accepted by many people as an expert counsellor despite the fact that the program was based only on the recognition of simple patterns in text. Despite the criticism, AI evaluation by interviewing or with variants of the Turing test in particular still has many advocates [34].

2.2. Performance on specific tasks

The mainstream approach in artificial intelligence to evaluate its systems more technically (especially when differences are subtle) is by the use of specific tasks. We have benchmarks and competitions for almost every general area in AI [35, 36, 6, 37, 38, 39], such as planning, learning, game playing, deductive reasoning, and also for many particular areas such as machine translation, driving vehicles, chess, robotics.

This has a relevant similarity to the way humans were evaluated before the advent of psychometrics. Some people were said to be intelligent because they read and wrote Latin, played chess well, or knew the names of all the Popes. However, this did not always indicate that these people could be able to perform well in other tasks. In fact, the term 'idiots savants' [40] was used for those subjects that have a good knowledge or skills about some particular things but are useless for many other problems. This represents, in a way, what is happening to AI nowadays. We have seen that many modern achievements such as the successful chess-playing Deep Blue [3], the theoretically optimal draughts player Chinook [41], the advent of autonomous cars [42], or Watson [4], the winner of the *Jeopardy!* TV quiz, show amazing capabilities, but are not generally considered very intelligent, as they are not able to do other things (without programming). These systems and the way knowledge is introduced or acquired are mainly designed (and programmed) to perform excellently in a very specialised area. As we mentioned in the introduction, this evaluation approach favours the *big switch* approach.

Another example of specific tasks are the CAPTCHAs (Completely Automated Public Turing test to tell Computers and Humans Apart) [43], where some specific tasks (e.g., pattern recognition problems) are used, as they are difficult for current AI technology. In fact, they are not used as *tests* of intelligence but just, as the acronym indicates, to tell humans and machines apart.

2.3. Systematic testing: psychometrics

One of the motivations behind the advent of psychometrics in the late XIXth century was the common confusion between 'idiots savants' and generally-able individuals [44]. Tests to measure intelligence in human children and adults consolidated in psychology in the first half of the last century. One common psychometric approach to the evaluation of intelligence is to determine the intelligence quotient (IQ), as a score obtained in a standardised test that quantifies the intelligence of a person [7]. However, there are many intelligence tests that are not used for IQ estimation.

Intelligence tests are typically based on a factor analytical model with sub-tests for different abilities. There is no complete consensus about the number of factors and, most especially, how they are related. However, there is a certain level of agreement on the existence of specific and general factors, and also on the distinction between knowledge-independent abilities and those that require the use of knowledge (and, of course, language). In this regard, one important notion is the distinction between fluid and crystallised intelligence [45]. Fluid intelligence refers to the capacity of reasoning and solving new problems, with a limited use of previously acquired knowledge. Crystallised intelligence, on the contrary, refers to the capacity of applying previous knowledge to new problems. It is important to clarify that crystallised intelligence is not the same as specialised knowledge. For instance, an 'idiot savant' is not generally able to use knowledge to generalise and relate concepts to new problems.

There are many different theories about how cognitive abilities (or factors) can be identified and arranged [46]. For instance, the Cattell-Horn-Carroll theory [47] and Thurstone's Primary Mental Abilities (PMA) theory [48, 12] identify a set of abilities (at one or more levels). The g-factor is a construct that is usually derived from a factorial analysis of the abilities, and is usually associated to the idea of 'general intelligence'.

Intelligence tests have been designed and work well for humans. What about using them for machines? As we will survey in this paper, this has been advocated (for many reasons) in the past. The most explicit claim of their use in AI was proposed by Bringsjord and Schimanski [16, 49], with the so-called 'Psychometric AI'

$k = 9$	:	a, d, g, j, ...	Answer: m
$k = 12$	:	a, a, z, c, y, e, x, ...	Answer: g
$k = 14$	:	c, a, b, d, b, c, c, e, c, d, ...	Answer: d

Figure 2: Examples of series of complexity 9, 12, and 14 used in the C-test [62]. The complexity is derived formally from the size of the shortest program that outputs the series

.

(PAI), as "the field devoted to building information-processing entities capable of at least solid performance on all established, validated tests of intelligence and mental ability, a class of tests that includes not just the rather restrictive IQ tests, but also tests of artistic and literary creativity, mechanical ability, and so on". Although, again, AI systems are evaluated with respect to their *performance*, the challenge of PAI is the broadness of scope that is demanded to be covered by a single system, as represented by intelligence tests. Note, however, that PAI proposes a roadmap for AI, but it does not claim that intelligence tests are the best way to evaluate AI systems. In fact, there is a strong debate (see, e.g., [50, 51] for two antagonistic positions) about whether a computer, if it can successfully solve a broad range of intelligence test problems, is intelligent in the way we use this term when we apply it to human intelligence.

One inherent characteristic of intelligence tests is that they are composed of items with variable item difficulty. Item difficulty is determined by the percentage of subjects that are able to solve this item, using functional models as in Item Response Theory [52, 53]. However, this cannot explain why one item is more difficult than another. For such an explanation, the investigated system has to be modelled as an information processing system as proposed by Newell and Simon [54]. Taking this perspective, item difficulty can be either explained completely independent of humans by means of algorithmic information theory or based on the assumed complexity of the cognitive processes and representations necessary to solve a test item. These two perspectives are introduced in the following subsections.

2.4. Algorithmic information theory approach

Algorithmic information theory and associated notions such as Solomonoff universal probability [55], Kolmogorov complexity [56], and Wallace's Minimum Message Length (MML) [57, 58] provide a further perspective to understanding and identifying intelligence. Using these concepts, intelligence is seen as a special kind of information processing tool that can be defined and evaluated using mathematical constructs. In addition, this perspective would, in principle, be able to determine the theoretical difficulty of any task, and the intelligence scale derived from a set of tasks, not relative to a (human) population. This approach is based on the assumption that intelligence is a universal concept.

This universal perspective has led in the past fifteen years or more to new intelligence definitions and tests, such as [59, 60, 61, 62], where tasks can be formally derived and their difficulty can be explicitly quantified. An example of one of these tests formally derived from computational principles is shown in Figure 2, which resembles some exercises found in intelligence tests but with a theoretical assessment of difficulty (based on the size of the shortest program that generates the problem instance) instead of an experimental one based on how difficult humans find them.

Other more general approaches followed, extending the above from static evaluation items to dynamic evolution items, where agents are evaluated in an environment with actions, observations, and rewards [63]. One crucial notion that has followed this trend is the idea of *universal* test [64, 65, 66], i.e., tests should be theoretically conceived to be administered to machines and humans (and other animals) alike.

The information-theoretic approach is not isolated from some of the approaches seen so far in sections 2.1, 2.2, and 2.3, and some hybridisations and integrated approaches have been proposed [67, 68, 69]. Actually, the development of well-grounded tests for humans, animals, robots, agents, animats, hybrids, swarms, etc., has been presented as an emerging new (but integrating) discipline, dubbed 'universal psychometrics' [70], which may inherit and integrate many important concepts from psychometrics, cognition, and computation.

2.5. Simulating mental operations: cognitive systems and cognitive models

While modern AI research typically focusses on designing efficient algorithms for complex problems, early AI research was inspired by the goal of simulating or creating intelligent behaviour. In the 1980s this goal was mainly adopted by cognitive science. Currently, we find the original goal of AI distributed over three, partially overlapping, communities: Cognitive Systems [71, 72, 73, 74], where research focusses on specific,

implemented simulation models based on standard AI methods, Artificial General Intelligence [75, 76], where researchers aim at autonomous, artificial systems that show human-like performance over a broad range of domains, and Cognitive Modelling, where researchers build simulation models based on cognitive theories and empirically compare similarity between models and human performance in specific tasks [77, 78, 79]. A domain extensively researched in cognitive science is analogy making with numerous approaches from cognitive systems as well as from cognitive modelling [80, 81]. Modelling mental operations also includes brain models and simulations of neurological processes [82], but these involve more than just behaviour (i.e., brain neurophysiology and imaging). All address the design of computer algorithms that *simulate* cognitive processes, so the emphasis is put on the similarities in processing effort and the responses the models and computer programs make. When a range of processes are integrated into a system, we usually talk about a cognitive architecture, such as SOAR [83, 84] (State, Operator And Result), which is based on a production system. Some of the computer models that we will see in section 5 are inspired by this architecture.

There is some variation in how the similarity between human and program is evaluated. On the one hand, algorithms are designed to capture an empirically-known phenomenon of cognition, for example, that humans transfer as many related knowledge elements as possible from one problem to another when reasoning by analogy [85, 86]; on the other hand, models are compared with specific empirical data such as solution times or errors [87]. In other words, these cognitive approaches are not meant to build systems that are able to perform well (or perfectly) in some tasks (or even in intelligence test problems), but rather that perform in the same way humans do, solving problems that humans usually solve but also failing at problems where humans fail.

Overall, in this section we have seen that intelligence tests are not the mainstream approach for evaluating intelligence in AI. However, the types of tasks that are found in intelligence tests are related to all the approaches seen in this section, either because they were introduced as a systematic alternative to the identification of abilities by interviewing and observation, as a reaction to the early confusion between intelligence and knowledge, or because of the connections that have been established between the difficulty and character of some intelligence test tasks and some information-theoretic principles. We shortly introduce a sample of typical intelligence test problems in the next section. A detailed discussion of the computer models that attempt to solve them will be presented in sections 5 and 6.

3. Intelligence test tasks and similar problems

In this section, we will briefly review some of the tasks that appear in intelligence tests and similar tests. We refer to some of these tests as 'similar' to intelligence test because those tests do not originate from psychometrics but from AI or cognitive science, and have never been used to evaluate humans systematically. Nonetheless, they are similar (intentionally or not) to human intelligence tests, and we think it is worth taking them into consideration. Basically, the criteria for inclusion in this paper are (1) that the tests have been developed as part of human intelligence tests or other tests of mental abilities, or have been introduced in the context of cognitive systems addressing aspects of human intelligence, and (2) that the tests have been attempted by at least one computer model, as we will see in section 5.

Well known intelligence tests are Raven's Progressive Matrices (RPM), the German Intelligenz-Struktur-Test 2000 (IST-2000), and the Wechsler Intelligence Scales for Adults, School and Preschool Children (WAIS, WISC, WPPSI). There are computer models dealing with Raven's Standard Progressive Matrices (denoted by SPM), number series (Numb-S) which are present in different standard intelligence tests, such as IST-2000, verbal common-sense reasoning problems (WPPSI), and block design problems (WAIS-B). Furthermore, different tests for mental abilities were investigated with computer models: Thurstone letter series completion task (Lett-S) based on Thurstone's Theory of Primary Mental Abilities (PMA), geometric analogies (ACE-A) in American Council of Education (ACE) tests, Odd-one-out problems (OOO, used nowadays in many test batteries, such as `http://www.cambridgebrainsciences.com/`), Bennett mechanical comprehension tests (BMCT), word analogies (SAT-A) from the Scholastic Assessment Test (SAT), and Montessori's object matching (Mont-O). Problems introduced in the context of cognitive systems are Bongard's analogy problems (Bong-A), string analogies (Str-A), and Hofstadter's anagrams (Jumbles). More information on these tests is given in the appendix. These tests represent the selection of problems covered by the computer models

discussed in this paper. We intentionally leave out related work where specific problems included in school or college tests are addressed [88, 89, 90, 91], because the tests are not intended to evaluate general intelligence.

The selection of which tests have been attempted by computer models and which tests have not is most informative. For instance, it seems pointless to apply a memory test to a computer model, even if this kind of test is usually included in many human intelligence test batteries. Also, some verbal tests, especially those about story completion, have only been attempted by a computer model very recently because they are really challenging. Another important factor that explains why some tasks have not been used in artificial intelligence or cognitive science is due to the restricted availability of many psychometric tests. The reason behind this is mostly to prevent the task instances from becoming public, as this would allow humans to prepare and *specialise* for the tests. A third factor influencing the selection of tests is the degree and complexity of required previous knowledge. Usually tasks that are very abstract and do not require previous knowledge (aiming at measuring fluid intelligence) are preferred in the context of computer models. However, there are also computer models addressing more knowledge-intensive tests (aiming at measuring crystallised intelligence). Finally, some tests issue new editions regularly, such as the WAIS (e.g., WAIS-IV was released in 2008), and the definition of items and tasks is hence more volatile.

Because of this volatility and their extensional definition by a set of instances, the tasks that we consider are not easy to separate or characterise by a single criterion, as there is a high degree of overlap in the processes and abilities that each of the following tasks involve. It is more convenient to analyse the abilities that each test covers. Table 1 shows a list of mental abilities on the rows and an identifier for each test task on the columns. We follow the list of primary mental abilities as described in [46], which is mostly based on Thurstone's Mental Abilities (PMA) theory [48, 12]. Even if there is no consensus on a list of abilities or factors, this list includes "the most important factors, in order of the proportion of individual differences explained" [46]. We exclude Associate Memory and Perceptual Speed, as memory and speed are not very interesting from the point of view of machines. We include Deductive Reasoning as it was originally present in Thurstone's PMA theory.

Ability	Lett-S	Numb-S	ACE-A	SPM	WAIS-B	OOO	Bong-A	Str-A	Mont-O	Jumbles	WPPSI	BMCT	SAT-A
Verbal Comprehension											×	×	×
Spatial Orientation			×	×	×	×	×		×			×	
Inductive Reasoning	×	×	×	×	×	×	×	×	×	×			×
Number Facility		×											
Word Fluency										×	×		×
Deductive Reasoning												×	

Table 1: Correspondence between the tasks given in the appendix (using the task identifiers as they appear in each subsection heading) and the mental abilities they measure according to [46].

From the point of view of artificial intelligence, the list of mental abilities in Table 1 can be mapped to the areas in artificial intelligence. For instance, Table 2 shows a list of AI subdisciplines[2]. Our understanding of "knowledge representation" is not just mere representation, which obviously affects all problems, but the storage and retrieval of *previous* knowledge from knowledge bases (linguistic, common-sense, rules, constructs, etc.) and its application for new problems.

One of the first things that we observe in these two tables is that inductive reasoning is predominant in Table 1 while deductive reasoning is predominant in Table 2.

4. Main issues for investigating computer models solving intelligence test problems

In the introduction we mentioned that the main goal of the paper was to understand the meaning, usefulness, and impact of several computer models taking intelligence test problems like those seen in the

[2]These are taken from the list of topics of this journal, excluding "AI and Philosophy", "Cognitive aspects of AI", "Heuristic search", "Intelligent interfaces", and "Intelligent robotics" for being metalevel, hybrid, or instrumental.

AI area	Verbal	Spatial	Inductive	Number	Word	Deductive
Automated reasoning and (deductive) reasoning						×
Commonsense reasoning	×	×				×
Constraint processing				×		×
Computer vision (and perception)		×				
Knowledge representation	×				×	×
Machine Learning			×			
Multiagent systems						
Natural Language Processing	×		×		×	
Planning and theories of action						×
Reasoning under uncertainty or imprecision						×

Table 2: Approximate correspondence between the mental abilities as shown in Table 1 and main areas in AI. The area of multiagent systems does not find a match in the list of abilities, as social abilities are not in the list. Constraint processing may only be related to the Number Facility ability when dealing with numeric constraints.

previous section. In order to get a comprehensive insight of many different approaches spanning over five decades, we will focus on a series of questions that are crucial in this understanding. But before discussing the questions, we will also identify some criteria that will be used to shed light on these questions.

The criteria that we will use are shown in Table 3. The 'years' criterion gives us a chronological perspective that can show trends about the interest in computer models solving intelligence test problems. The 'range' specifies the kinds of tests addressed by the models. This criterion is fundamental to understand whether the computer models are specialised or general. The 'intention' criterion specifies the purpose of the study: to better understand human cognition, to understand general principles of intelligence, to propose a metric for AI evaluation (i.e., psychometric AI), or to make a philosophical/epistemological question (human cognition, AI principles, AI evaluation, philosophical). This provides key information to truly understand the model achievements. The 'techniques' criterion shows which techniques are used by each model, which is relevant to see the role and progress of AI techniques or other ad-hoc techniques. The 'representation' criterion indicates whether the data was *transformed* to other representations or the *original* representation was used. 'Performance' specifies what the kind of comparison with humans was made while 'Difficulty' specifies the kind of difficulty assessment derived from the model. For these two last criteria, we specify whether it is at the test level or item by item (values can be *no*, *global*, or *itemwise*).

These criteria are identified to characterise all the computer models we will analyse in sections 5 and 6 in a most informative way, covering the facts about what, when, why, and how, so that the questions we formulate in this section can be answered in section 7.

Once we have enumerated the criteria, we will focus on the key questions we want to investigate in this paper. Table 3 shows the correspondence between the criteria and the (forthcoming) questions.

	Questions								
Criterion	Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Q9
Years	×						×		
Range		×	×	×		×	×		×
Intention	×		×	×				×	
Techniques					×		×		
Representation		×			×	×	×		×
Performance		×	×			×		×	
Difficulty		×		×		×		×	×

Table 3: Criteria to describe the computer models and the questions for which each criterion contributes.

The first question (Q1) is about the relevance of this area of research: **Are computer models solving intelligence tests becoming more wide-spread and more relevant for AI?** This is not a technical question but rather a question about whether this is becoming a trending research area. We will investigate

this question by looking at two criteria from Table 3: 'years' and 'intention', in order to see whether the number and frequency of publications in this area is increasing, and also the goals these models want to solve with a special focus on their use as evaluation tools for AI. We will also identify possible reasons why it is or is not relevant, and also if this area is expected to become more necessary in the future, as more diverse and intelligent systems are being developed so that benchmarks and challenges will be required. This leads us to the question of how challenging these problems are.

So, the second question (Q2) is: **How challenging are intelligence test problems for AI and what are the challenges?** Clearly, we need to understand where the difficulties of these problems lie and why some problems are more difficult than others. We refer here to full problems (or tasks) and not to problem instances (or items). To answer this question we will need to see first the 'range' of problems, as some types of problems might be easy but others could be very challenging. Another important criterion for the analysis of this question will be the 'representation' of the problem, since a change in the interface can make some particular problems less challenging. Finally, it is of utmost importance to see the approaches that have tried to understand the 'difficulty' of instances and also whether the way of assessing this difficulty is problem-independent and related to the difficulty humans find on the same problems (human 'performance').

Apart from possibly being useful as benchmarks, these models are usually proposed as tools to improve our understanding of intelligence. In particular, the third question (Q3) is: **Are computer models for solving intelligence test problems useful for understanding *human* cognition?** We want to know whether these tasks are good for cognitive modelling and to ascertain what constructs and cognitive mechanisms are needed to solve these problems. It is also crucial to see why and how humans solve (some of) them. In order to answer this question, we will require several criteria, mostly 'intention', 'range', and 'performance'. Basically, this question is about the implications for cognitive science.

While human cognition is a key reference, we might want to have insights from a more universal perspective about intelligence. Namely, our fourth question (Q4) is: **Are these models useful for understanding intelligence *in general*?** This question raises a more general, more theoretical (or even philosophical) question than the previous one, as some of the basic principles of intelligence could be unveiled (or not) by these kinds of studies. We would get more insight about whether these computer programs tell us something about the 'essence' of intelligence. The criteria that will be used are: 'intention', 'range', and 'difficulty'. We want to know if this can have implications for (universal) psychometrics.

While questions Q2, Q3, and Q4 are related to a description of the problems, we want to characterise the way these problems are (or can be) solved. Our fifth question (Q5) is: **What is the correspondence between the areas in AI and the abilities covered in intelligence tests?** We are interested in the techniques and algorithms they use, in order to determine which areas in AI are most useful for these tests. In particular, we want to know whether they are those subdisciplines in AI that deal with more general problems, such as machine learning. Also, we need to identify whether there are ad-hoc mechanisms to delegate different types of problems to different specialised techniques, either as a big switch or by mapping the representation. The categories 'techniques' and 'representation' will be useful for this.

The sixth question is more methodological and deals with the degree of connection among all the existing approaches so far. Our sixth question (Q6) will be: **Are the systems comparable and what are the requirements for a general and meaningful comparison?** With this question we want to determine whether the systems are comparable as they are, perform—if possible—a comparison of at least some of them for one kind of problem, and more importantly, to analyse what would be required to have a more general and meaningful comparison in the future. The criteria that will be used here are: 'range', 'representation', 'performance', and 'difficulty'.

A seventh question is perhaps more controversial: (Q7) **Is there progress in techniques within the field and has this progress impacted on AI techniques in general?** We are interested in whether the systems revise and extend earlier models and use previous experiences, or whether they ignore previous achievements and start from scratch. We want to see whether there is an incremental progress in results and the range of problems, and to see whether this results in a better understanding of the problems or the improvement of the AI techniques that are used. Also, we want to know whether the same problem can be solved by different approaches. Do the techniques adapt to new problems or are in fact the researchers adapting the techniques to new problems? Finally, we are interested in whether any new technique or

adaptation to these problems has been transferred to other areas of AI research. The criteria that will be used here are: 'years', 'range', 'representation', and 'techniques'.

An eighth question deals with the potential of the interdisciplinarity of this area of research. The eighth question (Q8) is: **Can these models help bridge research in psychometrics, cognitive science, and AI?** Can this change the way cognitive science and AI are evaluated? Can this make psychometrics more interested in evaluating AI systems? We want to see how concepts and ideas can be adapted from one area to the others. The criteria that will be used here are: 'intention', 'performance', and 'difficulty'.

The last question is about how to make these tests more available, flawless, and useful for AI evaluation: (Q9) **How should these tests be arranged and changed in order to serve as evaluation tools for AI?** Is a repository of the existing intelligence tests enough? Should we create new types of tests? Should the tests be public and fixed, or generated on demand? For this we will use the following criteria: 'range', 'difficulty', and 'representation'.

The nine questions above are used as a guideline to inspect about 30 approaches to intelligence test problems in the following two sections.

5. Account of computer models solving intelligence test problems

In this section we shortly describe all the computer models that have addressed intelligence tests and related tests following a chronological order. The systems we will review address tests introduced in section 3 (Table 1). The purpose of this section is to focus on the main features and achievements for all of these models, trying to determine the values of each of them for the criteria introduced in Table 3. The values for the criteria will be summarised in Table 4. A further discussion about the technical details of and the connections between these models will be given in section 6. Afterwards, the results of both analyses will be used to answer the questions raised in section 4.

5.1. Early systems: 1961–1991

The relation between artificial intelligence and psychometrics started more than fifty years ago with Evans [9, 10] and his program ANALOGY, which was "capable of solving a wide class of the so-called 'geometric-analogy' problems ("A is to B as C is to ?") frequently encountered on intelligence tests" [10] such as the ACE Tests. Evans's ANALOGY analyses the geometric figures in terms of intersections, decompositions, similarities, transformations (rotation, scaling), etc., using pattern matching.

At least for this seminal paper, it is interesting to know the reasons why Evans thought intelligence test problems were appropriate for the construction of heuristic problem-solving programs. He thought that the choice of geometric-analogy problems was suitable because: (i) "problems of this type require elaborate processing of complex line drawings", (ii) "the form of problems, [...] more speculative, [...] presents an interesting paradigm of 'reasoning by analogy' ", and (iii) "problems of this type are widely regarded as requiring a considerable degree of intelligence for their solution and in fact are used as a touchstone of intelligence in various intelligence tests" [10]. The intention was then to better understand some principles of analogy and its presentation, and the problems were converted from visual to symbolic. Evans did not say that these tests could be taken as a sufficient or a necessary condition for intelligence, just that "this suggests a non-trivial aspect of any attempt to mechanize their solution". In fact, when he compared his results to humans', no mention is made of this being interpreted as ANALOGY showing intelligence.

Evans's program ANALOGY was highly acclaimed and cited in the following years, but not because of its results being comparable to humans (as this was probably considered irrelevant or anecdotal). Some people (e.g., Solomonoff [92]) suggested that ANALOGY could be extended to cover a much wider range of problems, but the approach was never continued or extended.

A different, but almost simultaneous, approach to Evans's was taken by Simon and Kotovsky [11]. They chose "Thurstone letter series completion" tasks. The goal of the research was to understand how humans solved these kinds of problems and their difficulty, through the use of a computer model. In order to make a system that would capture the patterns, Simon and Kotovsky formulated a simple procedure for pattern descriptions using IPL-V, Newell's information processing language V [93]. Their work included a number

of different pattern generator variants conceived to solve the series with different levels of performance (variants A to D, becoming progressively more challenging). One of the programs (variant D) was able to score better than 10 of 12 human subjects on 15 problems of the Thurstone Letter Series Completion [11, Table 3]. Similarly to Evans, this work did not claim that these results could lead to seeing these programs becoming closer to human intelligence.

Despite the relevance of these works, it took almost two decades to see more models. With the major goal of understanding analogy, Hofstadter developed a series of computational models, Jumbo [94], and others, in the Copycat project [74]. Hofstadter considered analogy as key to recognition and categorisation, namely, the core of "high-level perception" and creative thought. The project and tasks were inspired by the Bongard problems. However, because of their visual difficulty, the problems were simplified and remade in a microdomain with a symbolic representation whose basic elements are letters and strings of letters: jumbles (anagrams from a given set of letters) and letter sequences analogies. The cognitive architecture and ideas used therein were very specific and generally unrelated to other techniques in AI. The architecture follows a biological metaphor in its development by using different constituent elements (long-term memory, short-term memory, and subcognitive processing mechanisms) which interplay in parallel in order to generate obvious or (in cases) creative solutions. The results were not fully compared with those of humans.

Carpenter et al. [14] addressed Raven's Progressive Matrices (RPM) [95]. Yet again, the goal was to better understand human intelligence and the nature of the tests. The general outline of how the model performs is divided into three main categories: (a) *perceptual analysis*, where the model encodes the information about the figures of the first pair of entries of each row, determines the correspondences between their attributes and the figures in the remaining entry are compared to obtain a pattern of pairwise similarities and differences; (b) *conceptual analysis*, induction and generalisation of the rules that account for the variation among the figures and attributes in each of the first two rows. This process is incremental and the rules are induced one by one; (c) *response generation*, the rules for the top two rows are generalised and applied to the third row to generate the solution. Carpenter et al. analysed the rules needed to solve the RPMs and identified five relations: "constant in a row", "distribution of three values", "quantitative pairwise progression", "figure addition", and "distribution of two values". Strictly, this is not a big switch approach, but shows that the system would not work for a similar problem with different relations. As an indirect result and based on the previous relations, they produced "a pair (FAIRAVEN and BETTERAVEN) of computer simulation models that performed like the median or best college students in the sample", which were based on the analysis of eye fixations, verbal protocols, error patterns, and the identification of the previous rules.

As the reasons to choose RPM, Carpenter et al. say: "the correlation between Raven test scores and measures of intellectual achievement suggests that the underlying processes may be general, rather than specific to this one test" [14]. Again, it is not argued or mentioned that these models are intelligent. Carpenter et al.'s approach does not handle re-representation or encoding of the problem. Nonetheless, for the first time, one of these cognitive models is generalised as a theory of intelligence on its own, but without any explicit reference that this could be used for machines or for artificial intelligence.

Simon et al. [96] presented the cognitive computer model SC-Soar to solve series completion tasks based on the ideas of Simon and Kotovsky [11]. Their system was an adaptation of the cognitive architecture SOAR. In particular, SC-Soar solved letter series completion tasks by casting them as comprehension tasks so as to find relations that characterise the series. The system was able to solve ten out of fifteen series in the original set [11]. Although they did not make any comparison with humans, they argued that the number of decision cycles (measure of duration) that the system took to obtain a right solution provided a measure of complexity of these series that could be closely related to the difficulty of these series for the subjects. The authors also claimed that the five series SC-Soar cannot solve were the most difficult.

5.2. Later systems: 2003–2009

A decade later, Bringsjord and Schimanski refreshed the interest of psychometric tests in artificial intelligence [16]. In fact, they started by wondering "what went wrong" with Evans's ANALOGY program, as it should have been "the first system in a long-standing, comprehensive research program", but, after forty years, they found themselves "trying to start that very program" [16]. One of their explanations was based on the distinction between a narrow view of intelligence tests (Spearman's general intelligence g [97], as

illustrated by Raven's Progressive Matrices [95]) in front of a broader view of intelligence tests (Thurstone's view of a range of problems [48], as in WAIS [98]). As an example of how to deal with WAIS (at least partially), they introduced PERI, whose name stands for "Psychometric Experimental Robotic Intelligence". PERI is based on cognitive robotics and makes no claim of being a cognitive model. It uses brute force for problem solving and space search. While it is specialised to one kind of problem (Block Design), Bringsjord and Schimanski argued that PERI was not only able to solve the particular Block Design problems in the WAIS, but any Block Design puzzle given to it.

Simultaneously with Bringsjord and Schimanski, Sanghi and Dowe [16, 15] developed a computer program (without any supporting cognitive model) which was used to make a clear point about the relevance of machines passing intelligence tests. A third year undergraduate student (Sanghi) produced an obviously non-intelligent small program written in Perl which was able to pass some specific intelligence tests featuring number, letter, and word series. The authors claimed that some other kinds of more difficult intelligence tests (involving matrices, pictures, and questions) could be passed in the same way (using predefined patterns), but they were not implemented in the program. The experiment was conclusive, as nobody would assign intelligence—not even the smallest degree—to this program. This system is a clear example of the big switch approach. The code was full of *if-then-else* and *case* instructions trying to identify what kind of problem they were facing and delegate it to the appropriate module. As a result, the analysis of this work has to be done in terms of the specialisation of the techniques that were used, their performance and the range of problems, hence raising important doubts about the use of intelligence tests for evaluating AI systems.

The two previous (opposed) approaches [16, 15] for solving intelligence tests led to an increasing number of works in the area (although some of them were possibly unaware of these opposed views, as they do not cite Bringsjord and Schimanski's paper or Sanghi and Dowe's paper). For instance, Tomai et al. [99] revisits Evans's problems but using a more abstract and general approach, based on general-purpose simulation models: sKEA [100] and the *Structure-Mapping Engine* (SME) [85]. Tomai et al.'s goal was to show that through the generality of their system, they were able to solve a set of classic visual analogy problems. The results on the twenty problems used by Evans were similar to those of Evans.

A different attempt to address a psychometric test with the aim of understanding humans' cognitive mechanisms was Phaeaco [101], which focussed on Bongard problems [102]. Phaeaco uses a cognitive architecture with an image processing module, pattern matching techniques, long-term memory, and learning mechanisms with the aim to tell how similar such figures are and trying to emulate how humans solve these problems (hardwired mechanisms and holistic and analytic views). Phaeaco is able to solve some Bongard problems. Overall, its performance was shown to be slightly worse than humans.

Similar to Tomai et al., Lovett et al. addressed three visual problem-solving tasks: geometric analogies [103, 104], Raven's Progressive Matrices [105, 77], and odd-one-out intelligence tests [106, 107]. With a major goal of modeling human cognition, their models provide novel insights about which cognitive operations are easier or harder for human visual problem-solving. Lovett et al. demonstrated that qualitative spatial representations extracted by using CogSketch [108] (their sketch understanding system) and visual comparisons made by the structure mapping engine (SME) can be used to solve geometric analogy problems [99, 103, 104]. The authors combine these two models with two complementary theories of how people perform it: (1) *visual inference*, where the answer of a problem is inferred applying the differences found between the images "A" and "B" over the image "C" in order to obtain the solution "D"; and (b), *second-order comparison*, where the differences found between the images "A" and "B" are then compared to the differences between "C" and each possible answer "i", selecting the most similar one.

In order to address RPMs [105, 77], Lovett et al. used SME (without using any processes specifically designed for the task) in a two-stage mapping process, also using CogSketch and a series of strategies based on structure-mapping techniques which are used to map the different elements in the matrix in order to obtain patterns of structural relationships (similarities and dissimilarities). Lovett et al. claimed that their model overcame the limitations of Carpenter et al.'s model [14], using visual representations and task-general processes. Regarding the results, the first system scored like an average American adult, while an improved version scored "above [...] most adults" [77].

Finally, in [106, 107], Lovett et al. addressed odd-one-out problems in the same way as the previous models: using SME with the analogical generalisation module known as SEQL. The results were slightly

better than those of American adults [107, Table 3]. While these comparisons could be indicative of the achievements of these techniques, and one of the most general approaches to intelligence tests to date, we note that the goal of these works was to construct a model to better understand item difficulty.

It should be noted that, although those models of Lovett et al. were first developed before 2010 [103, 105, 106], we find improved or more advanced versions of them from 2010 onwards [104, 77, 107], so it is somewhat arbitrary to include them in this subsection or in the following one.

5.3. Recent explosion: 2010–today

A surge of new systems has taken place since 2010. For instance, a different approach has been started by Sinapov and Sotytchev [109], where an intelligence test is taken by a robot in a rich sensorimotor scenario. The robot's aim was to solve personalised odd-one-out tasks. The tests were performed on six natural object categories (pop cans, plastic cups, metal objects, empty bottles, soft objects, and objects with contents) where a group of four objects (three of the same category and one outside the category) is presented. Cognitively, the model uses similarities between objects to select the most dissimilar object. The results were not compared to humans, but the robot obtained results better than chance for all six object categories.

Another attempt to address Raven's Progressive Matrices was undertaken by McGreggor, Kunda, and Goel [110, 111, 112, 113]. Unlike previous models focussing exclusively on modal propositional accounts, they proposed two complementary approaches that use purely iconic visual representations of test inputs (with different level of resolution): the "fractal" and an "affine" method. Both methods are used to judge the similarity between images and induce all possible transformations (selecting the best one) so as to predict an answer that will be compared to each given answer choice. The affine and fractal models were tested on all problems from the Raven's Standard and Advanced Progressive Matrices tests. Even with this original visual representation, the performance was shown to be similar to the human norm. McGreggor and Goel also applied the fractal approach to the odd-one-out problem in [114, 115], where the system automatically adjusts the level of the resolution of the images and increases (also automatically) the complexity of the relationships (from simpler to higher-order) to resolve ambiguity. Their approach was evaluated against 2976 randomly selected odd-one-out problems from a large corpus with a span of difficulty from the very easiest (level one) up to the most difficult (level 20). From them, 1647 problems were solved and, although the performance was not compared with humans, we think that its results must be below human performance. The main goal of these works was to evaluate whether visual analogy problems (RPMs and odd-one-out problems) could be solved using visual representations.

A special journal issue on psychometric artificial intelligence [49] triggered more computer models. For instance, Klenk et al. [116] challenged a new kind of test, Bennett's Mechanical Comprehension Tests [117], with important content about physics and contextual information. Klenk et al. also used sKEA and SME (as [99] and [105]), integrated in (and extending) the Companion Cognitive Architecture [118], in order to address these problems. Unlike the previous approaches, they did not make any comparison with humans.

Turney [119] introduced the system PairClass as an improvement of other systems by the same author. PairClass is a system for analogy perception that recognises lexical proportional analogies (A:B::C:D, meaning "A is to B as C is to D" but with words) by using word frequency-based features vectors (searching in a corpus) and some supervised machine learning algorithms to classify word pairs. PairClass is able to solve word analogies found in the SAT college entrance exam, TOEFL (test of English as a foreign language) synonyms, ESL (English as a second language) synonyms, some synonym and antonym problems previously used in computational linguistics, similar/associated/both word pairs, and noun-modifier relations. Although only the SAT college entrance exam resembles an intelligence test (with specific knowledge), we think that the other tests are also valid as human-level tests and show that PairClass can address many kinds of linguistic problems provided it has access to a corpus.

Ruiz [120] addresses the odd-one-out problems. The aim of this work was to show that a simple two-step algorithm can help to understand the relationships between symbols and the dissociation/association process (based on symbols and sets) in this kind of intelligence test problem. The results were comparable to those of humans. This has to be interpreted carefully, as instead of re-representation, the system performed a hand translation (RASCM). Also, the ad-hoc use of a Hamming distance (to find most frequent symbols) suggests that the system would need to be changed dramatically for other similar problems.

Id	Model	Years	Range	Intention	Techniques	Performance	Difficulty
EvansGEO	Evans's ANALOGY [9, 10]	1961–65	geometric analogy*	AI principles	own rules	global	no
Sim+LET1	Simon & Kotovsky [11]	1963	letter series	human cog.	list-processing	itemwise	itemwise
HofJUMBO	Hofstadter's JUMBO [94, 74]	1983	word jumbles	AI principles	own cog. arch.	no	no
HofCOPYC	Hofstadter's COPYCAT [94, 74]	1984	let. seq. analogy	AI principles	own cog. arch.	no	no
Carp+RPM	Carpenter et al. [14]	1990	Raven's PMs*	human cog.	production-system	itemwise	itemwise
Sim+LET2	Simon et al. [96]	1991	letter series	human cog.	own rules	global	itemwise
Bri+PERI	Bringsjord & Schimanski's PERI [16]	2003	psychomotor WAIS blocks*	AI evaluation	robot + own rules + NLP	global	no
San+PERL	Sanghi & Dowe [15]	2003	several	philosophical	own rules	global	no
Toma+GEO	Tomai et al. [99]	2005	geometric analogy	AI principles	sketches + SME	global	no
FouPHAEA	Foundalis's PHAEACO [101]	2006	Bongard probs.	human cog.	own rules	itemwise	no
Lov+RPM	Lovett et al. [105, 77]	2007–10	Raven's PMs	human cog.	sketches + SME + SEQL	itemwise	itemwise
Lov+OOO	Lovett et al. [106, 107]	2008–10	odd-one-out	human cog.	sketches + SME + SEQL	itemwise	itemwise
Lov+GEO	Lovett et al. [103, 104]	2009–12	geometric analogy	human cog.	sketches + SME	itemwise	itemwise
Sin+ROOO	Sinapov et al. [109]	2010	psychomotor odd-one-out	AI principles	robot + similarities	no	itemwise
McG+RPMf	McGreggor et al. [110, 111, 112]	2010–12	Raven's PMs	AI principles	similarities + fractal	global	no
McG+OOO	McGreggor & Goel [114]	2011	odd-one-out	AI principles	similarities	global	no
Kle+MECH	Klenk et al. [116]	2011	Bennett's Mech.	AI evaluation	sketches + SME	no	no
TurPAIRC	Turney's PairClass [119]	2011	word analogy/synonyms	AI principles	corpus + SVM + RBF	no	no
RuizOOO	Ruiz [120]	2011	odd-one-out*	human cog.	similarities + clusters	itemwise	itemwise
Rag+NUMS	Ragni & Klein [121]	2011	number series	AI principles	ANN	global	no
Rag+RPM	Ragni & Neubert [122, 123]	2012	Raven's PMs*	human cog.	ACT-R	global	itemwise
Bay+WORD	Bayoudh et al. [124]	2012	word analogy	AI principles	corpus + complexity	no	no
Pra+RPMp	Prade et al. (pixels) [125, 126, 127]	2012	Raven's PMs	AI principles	similarities	global	no
Pra+RPMf	Prade et al. (features) [125, 126, 127]	2012	Raven's PMs*	AI principles	similarities	global	no
Eli+SPAU	Eliasmith et al.'s SPAUN [24]	2012	series and analogies*	human cog.	ANN	itemwise	no
Sie+NUMS	Siebers & Schmid [128]	2012	number series	AI principles	own rules	no	no
Sch+ROBO	Schenck et al. [129]	2012–13	psychomotor, several	AI principles	robot + similarities	no	itemwise
McG+RPMa	McGreggor et al. (affine) [112, 113]	2012–13	Raven's PMs	AI principles	similarities	global	no
Pra+OOO	Prade et al. [130, 127]	2013	odd-one-out*	AI principles	similarities	no	no
Str+ASOL	Strannegård et al.'s ASolver[131]	2013	number series	AI principles	ind. prog. (compression)	global	no
Str+SSOL	Strannegård et al.'s SeqSolver [132]	2013	number series	AI principles	ind. prog. (compression)	global	no
Str+RPM	Strannegård et al. RPMs [133]	2013	Raven's PMs*	AI principles	own rules	itemwise	no
Ohl+CNET	Ohlsson et al.'s ConceptNet4 [134]	2013	verbal WPPSI	AI evaluation	NLP + knowledge base	global	no
Hof+NUMS	Kitzelmann & Schmid's IGOR [135]	2014	number series	AI principles	ind. prog.	itemwise	itemwise

Table 4: Taxonomy of computer models solving intelligence tests. All systems are automated. The categories (which were introduced in section 4, see Table 3 and accompanying text) are the development or publication 'Years', the 'Range' of problems (an asterisk shows that the 'representation' has been transformed), the authors' 'Intention' with the model, the 'Techniques' that were used, the comparison with human 'Performance', and the analysis of item 'Difficulty'.

Ragni and Klein [121] worked on number series completion, which is very common in many intelligence tests and also occasionally in artificial intelligence [136]. They took over 50,000 number series from the Online Encyclopedia of Integer Sequences (OEIS) and applied a general method using artificial neural networks (ANNs) and dynamic learning. Although the overall results were comparable to those of humans, the error distribution was very different, probably because the way in which the problems were solved is very different to the way humans solve them.

Ragni and Neubert [122, 123] presented another system for Raven's Progressive Matrices, implemented in the cognitive architecture ACT-R. The system requires the identification of relational rules, as Carpenter did. In particular, Ragni and Neubert use five: constant in a row, distribution of three values, quantitative pairwise progression, figure addition, and distribution of two values. Unlike the previous work [121], the motivation of this work is to solve the problems in a cognitive way, i.e., explaining why the system fails (ambiguousness, objects neglected, incorrect rules applied, ...) and why some problems are more difficult than others (number and type of rules applied, number of calls to the declarative memory, objects stored, ...). Considering the results obtained, the authors claim that the visual complexity involved in solving this kind of problems is smaller than the functional difficulty (rules to be applied). The results were compared to humans in terms of correlations for the accuracy (subjects error rates) and to Carpenter's BETTERAVEN (where a human comparative study was made), with similar (or slightly better) score, so we can estimate the results to be around human average.

Closely related to the approach of Turney we find the work of Bayoudh et al. [124], where a feature vector similar to Turney's is used to represent the concepts but with completely different semantics. The main idea is that each word in an analogical proportion problem carries an "information content" that can be formally defined via its Kolmogorov complexity K. To evaluate their approach, they used a set of 147 pairs of words coming from the SAT college entrance exam that are to be completed with another pair of words to be selected among 5 options, performing slightly better than a pure random choice.

Prade, Richard, and Correa [125, 126, 127] developed a logical representation of the notion of "analogical proportion" (i.e., statements of the form "A is to B as C is to D"). This logical view tries to depict how similar (*homogeneous proportions*) or dissimilar (*heterogeneous proportions*) the items in an analogical proportion are by representing them using binary or multiple-valued features. Following this perception of similarity, the authors developed a system for addressing Raven's Progressive Matrices based on solving analogical proportion equations using an extended scheme where proportion $(a, b) : f(a, b) :: (c, d) : f(c, d)$ holds for lines and proportion $(a, b) : g(a, b) :: (c, d) : g(c, d)$ for columns. Their approach may be applicable at different levels of representation: feature-based approach (propositional representation of the matrices) and pixel-based approach (bitmaps). For both approaches a simple algorithm finds both horizontal and vertical patterns (analogical proportions) between the Boolean values of the features or pixels of the pictures. The authors mainly compared their work with those which use *Structure-Mapping Engine* (SME) approaches [105] claiming the simplicity and generality of their approach. The results for the pixel-based approach were worse than those for the feature-based approach (because their algorithm is unable to provide the information needed to get a proper solution of several tests), the latter being about human average.

Following the previous perception of dissimilarity between items in a formal setting of logical proportions, Prade and Richard [130, 127] addressed odd-one-out problems. By using the same Boolean representation and equation-solving principle, the idea is to find the item which has most dissimilarities with regard to the rest of items for each feature: when a *homogeneous proportion* equations holds between four Boolean values, there is no intruder among them, however, if heterogeneous proportions equations hold for an item but not for the rest, an intruder is found. This approach was not evaluated.

With quite a different perspective, Eliasmith et al. [24] have recently produced a 2.5-million-neuron artificial model brain (called Spaun, short for Semantic Pointer Architecture Unified Network). Since Spaun is highlighted by its functionality, attention has been drawn to its having a similar ability on certain aptitude test questions to what might be found in some humans [25, 26]. Spaun is evaluated against eight types of problems (A0–A7). A7 is "fluid reasoning", which is arguably said to be "isomorphic to the induction problems from the Raven's Progressive Matrices (RPM) test for fluid intelligence. [...] This task requires completing patterns of the form: 1 2 3; 5 6 7; 3 4 ?" [24]. Using a "match-adjusted success rate" (since Spaun must generate the correct answer, not choosing from a set of answers), and comparing to humans,

the authors conclude that "Spaun is similarly good at these tasks" [24].

Siebers and Schmid [128] address the number series problem with a cognitive model, differently to other previous approaches for the same problem introduced before [121, 136]. They use their own (public) series collection formed by 25,000 randomly created number series (using the basic numerical operators), for which the accuracy was 93.2%. No comparisons with humans were made.

Schenck et al., in a series of papers [137, 138] and in Schenck's Master thesis [129], address a series of tasks found in intelligence tests with an upper-torso humanoid robot performing a set of stereotyped exploratory behaviours and recording sensorimotor feedback, as in Sinapov's paper [109]. Three different types of problems were tried. First, some Montessori tasks [137] [129, chapter 4] were included in the experiments with sound cylinders, weight cylinders, pressure cylinders, and sound boxes. Second, order completion problems [138] [129, chapter 5] were arranged and three tasks were prepared: ordering by weight, ordering by compliance (black and white categories) and height. They posed 150 order completion tasks to the robot (3 sets of objects, 50 tasks per set). Third, matrix completion problems were arranged [129, chapter 6]. They were similar to simple RPMs, where size, colour, and content were used. "The robot was presented with a set of objects arranged in a grid, with the lower-right object missing, and with a set of candidate objects". They posed "500 randomly generated matrix reasoning tasks to the robot".

Strannegård et al. [131] address the number series problem by developing ASolver, a Haskell-coded anthropomorphic cognitive system based on the idea of limited working memory. The main strategy is as follows: (a) construct a term-based language that describes number sequences ($1, 2, 3, 4, \ldots$ is described as $f(n-1)+1$); (b) define a term rewriting system to compute mathematical terms; (c) define bounded cognitive resources in terms of computations and size; (d) look for the smallest term that computes the input sequence. This model was not intended as a cognitive model but only for the purpose of problem solving. Nevertheless, it has some relations to human cognition. The same problem was undertaken by Strannegård et al. [132] with SeqSolver, which, following the same pattern discovery as in [131], is based on Kolmogorov complexity for limiting the set of computations. ASolver was tested on the 11 number sequence problems of the IQ test PJP [139], obtaining scores above IQ 130–140. On the other hand, SeqSolver was tested on number series from IST [140] and obtained scores of at least IQ 130.

Closely related and simultaneously, Strannegård et al. [133] address Raven's progressive matrices with an anthropomorphic cognitive model in the sense that it uses certain problem solving strategies that were reported by high-achieving human solvers. Some principles are different to those of [131] but the system is also based on a repertoire of patterns that are combined to reach the solution. To solve the matrices, the system relies on pattern matching and the use of the repertoire of patterns similar to those of Carpenter et al. [14] (identity, one of each, numeric progression, translation, AND, OR, XOR). The system program was tested on the sets C, D, and E of Raven's Standard Progressive Matrices test and produced correct solutions for 28 of the 36 problems considered. The results are said to be roughly comparable to an IQ of 100.

The approach presented by Ohlsson et al. [134] is one of the few approaches to verbal intelligence tests. In particular, the authors proposed the use of the Wechsler Preschool and Primary Scale of Intelligence (WPPSI-III, Third Edition) to evaluate common-sense AI systems. Their method is based on the capabilities available in the commonsense knowledgebase and natural-language-processing toolkit *ConceptNet* [141], an open-source project run by the MIT Common Sense Computing Initiative. The system obtained results which were comparable to results obtained by an average 4-year-old (which does not mean that the system has the verbal abilities of a 4-year-old).

Hofmann, Kitzelmann, and Schmid [135] demonstrate that the inductive programming [142, 143] system IGOR2 can be applied to number series problems. IGOR is a system for learning (recursive) functional programs from input/output examples, which has been applied to many problem solving domains [144]. The authors present an empirical evaluation with 100 number series systematically varied along three dimensions: size of starting number (small/large), type of operator ($+, \times$), and structural complexity. The latter involves variations with respect to reference (last number, second last number, third last number), linear vs. tree recursion (as for factorial vs. Fibonacci), and usage of one or more operators. These dimensions were identified from a psychological study of the difficulty of number series problems by Holzmann et al. [145]. Although IGOR2 is an AI system, the authors propose that IGOR2 can also be viewed as a plausible cognitive approach to learning complex rules.

6. Technical analysis

In this section we present a more detailed study of the techniques that are used by the systems introduced in the previous section. We will group the systems by the techniques they used. Furthermore, we will show the evolution of the techniques and their progress and whether some kinds of techniques are more appropriate for some types of problems. This analysis will be crucial to answer questions Q5 and Q7 (from sec. 4) in the following section. Our analysis is arranged in two parts: (1) how the representation of the input problems is addressed, and (2) which techniques are used to compute the solutions.

6.1. Input representation techniques

Regarding the input problem's representation we find two big different approaches: (a) symbolic representations (hand-coded or not) and (b) visual representations (pixel-based or analogical visual representations). Former systems like Evans's ANALOGY [9, 10] or those developed by Carpenter et al. [14] relied on hand-coded symbolic descriptions of geometric analogies and RPMs. In particular, Evans's system made a complex preprocessing of the input problems decomposing each hand-made figure description (coordinates) into very precise internal symmetry descriptions (LISP-based representations): both a specified set of geometrical analogy relations (inside, above, ...) and a set of "Euclidean similarity" calculations (rotations, uniform scale changes, and certain types of reflections), which were determined for each and every pair of objects by using a substantial repertoire of "analytic geometry" routines and a topological pattern matching process. By contrast, more recent works, such as those of Tomai et al. [99] and Lovett et al. [103, 104, 106, 107], are based on the process of forming high-level conceptual representations from raw data (without hand-coding). These approaches exploit qualitative visual structure computing abilities from sketching programs such as sKEA [100] or its successor *CogSketch* [108]. Both are general-purpose architectures for conceptual sketch understanding (visual and spatial properties) that allow the generation of representations from raw input and provide a convenient platform for cognitive experiments that use visual stimuli. Specifically, in *CogSketch* the user has to label the input objects (*glyphs*) and *CogSketch* uses them to compute spatial relations (e.g., edges, groups of objects, relative position, topology, overlapping) which can lead to inferences about the spatial relationships between the content of those glyphs.

Notwithstanding, many modern computational systems keep assuming that hand-coded representations are available as inputs. Systems such as those of Ragni and Neubert [122, 123] and Strannegård et al. [133] continue addressing Raven's progressive matrices by generating hand-coded propositional attribute-value vectors with, respectively, a fixed set of attributes (shape, size, number of sides, width, height, colour, rotation, position, and quantity); and *vector graphics* where each matrix is encoded as vectors of attribute-value pairs using different abstraction levels (features, elements, groups, and cells) by using the XML-based specification language from Microsoft, *XAML*.

Prade, Richard, and Correa's approach [125, 126, 127] is a special case, applicable to different levels of representation (resolution). One is a feature-based approach, where the RPM pictures are represented as vectors of Boolean features (coded manually), in some way like Ragni and Neubert's, but the features can be different for each problem and can denote, for instance, the presence of a *Big Square*, a *Small Square*, a *Circle*, or a *Cross* in a picture. A pixel-based representation is also used in a number of cases.

Clearly, a big technical advance in intelligence tests' input has been the ability of using low level image representations (bitmaps). Here we find those systems for solving RPMs of Prade, Richard, and Correa's (pixel-based approach) aforementioned [125, 126, 127], the system of Foundalis [101] for solving Bongard problems, or the (robotic or not) approaches with video input systems [16, 109, 137, 138, 129, 24]. Also, the fractal and pixel-based computational models from McGreggor, Kunda, and Goel [110, 111, 112, 113, 114, 115] work directly on visual inputs from both RPMs and odd-one-out problems without any need to extract propositional representations from them. All these techniques, computationally infeasible in previous decades (due to the hardware and software requirements), have shown that they perform surprisingly well in those kind of tasks and, thus, are valid alternatives to the verbal representations.

Finally, note that those systems addressing word, letter, or number-based tasks are able to use raw "strings" or number/letter sequences directly [11, 96, 15, 121, 128, 131, 132], meanwhile others need specific types for input problems: Hofmann, Kitzelmann, and Schmid's system [135] needs algebraic data-types for

the target function together with a small set of examples; Ragni and Klein's system [121] and Siebers and Schmid's system [128] need to transform the sequences into different train and test sets.

6.2. Computational techniques

A crucial difference that makes systems distinctive is not only the kind of problems they solve, but also the great deal of techniques that have been used for solving the same or different intelligence tests. This correspondence between tasks and kinds of problems is illustrated in Figure 3 and will be developed by the technical details below, attempting to clarify the evolution over the years.

One of the most used techniques during the early years was what we call **rule-matching** techniques. This includes, among others, the use of the authors' own code or the use of production-systems [146] with hand-coded production rules. In this category we include those systems such as Evans's ANALOGY, which used a substantial amount of pattern-matching techniques between pairs of figures and heuristic problem-solving mechanisms in order to generate a set of rules transforming figure A into figure B (specifying how the objects of figure A are removed, added to, or altered to generate figure B). These sets of rules are generalised and a similarity matching is carried out between figure C and each of the five answer figures given in the analogy problem to find (if possible) a correct answer. Another example is [14], where Carpenter et al. used their own production system CAPS [147] (for Concurrent, Activation-Based Production System), a collection of procedural hand-coded *if-then-else* rules (including the five rules discovered needed to solve the RPMs) specifying what symbolic manipulation should be made when a given information condition arises in working memory. Unlike conventional production systems, (a) CAPS allows for parallel execution of all the productions whose conditions are satisfied, and (b) instead of having knowledge elements either present or absent from working memory, they can have varying degrees of activation. As commented in the previous section, Carpenter et al. produced a pair (FAIRAVEN and BETTERAVEN) of computer simulation models. The BETTERAVEN model differs from FAIRAVEN in two major ways: BETTERAVEN has the ability to induce more abstract relations than FAIRAVEN, and it has the ability to manage a larger set of goals in working memory and hence can solve more complex problems.

Currently, several computational systems still use their own *if-then-else* patterns to pass some specific intelligence tests. This is the case with Sanghi and Dowe's program written in Perl (960 lines of code) [15], which, having access to a word-list of approximately 25,000 words and using some predefined *if-then-else* patterns, was able to pass some specific intelligence tests featuring number, letter, and word series. Furthermore, some other systems use hand-coded variations of those patterns found by Carpenter. Strannegård et al. [133] also relied on a repertoire of patterns similar to the one of Carpenter et al. [14], which can operate with one or more abstraction levels—attributes (features), elements, groups, and cells—and can be processed row or column-wise in order to obtain a prediction value for the solution cell.

On the other hand, Ragni and Neubert [122, 123] also relied on the use of some predefined hand-coded rules or patterns for solving RPMs. However, in their case, these rules were implemented using the well-known **cognitive architecture** ACT-R [148], a cognitive architecture for simulating and understanding human cognition implemented as a production system, where each symbolic processes is controlled by subsymbolic equations (responsible for most learning processes) which estimate (and then decide) the relative cost and benefit associated with their execution. Their system also requires the identification of relational rules, as Carpenter did.

Former systems such as those of Hofstadter also relied on cognitive architectures. In this case, Hofstadter developed its own cognitive architecture in the Copycat project [74]. Particularly, *Jumbo* [94] and *Copycat* [74] were developed for solving different letter-based intelligence tests and, furthermore, both are composed of three constituent elements following a biological metaphor. Jumbo's architecture, used to make plausible anagrams from a given set of letters, is composed by: the *chunkabet*, storing a database of pondered chunks (small sequences of letters); the *cytoplasm*, modelling the working memory and containing partial associations of letters; and the *coderack*, formed by *codelets* (fragments of code) that run in parallel (based on the current state of the *chunkabet* and *cytoplasm*) and can perform modifications over the structures in the *chunkabet* and *cytoplasm*. On the other hand, Copycat's architecture, which is similar to Jumbo's, was used for solving letter sequence analogies. Its main components are: the *slipnet*, the *workspace*, and the *coderack*. The *slipnet* models the long-term memory in humans by a semantic network composed of nodes

that represent permanent concepts about the letter-string world (*sameness*, *leftmost*, *opposite*, ...) and their weighted relations (strength of associations between concepts). In total there are more than 60 such concepts. Both the relevance of the concepts (related to their activation) and the distance of the links between them change dynamically depending on CopyCat's perspective on the given problem. On the other hand, the *workspace* is the site of subcognitive processing activity that is in charge of modelling the short-term memory where partial structures can be formed (single letters, descriptions, groups, bonds, and bridges). These are temporary combinations built up entirely from *Slipnet* concepts, e.g., *"jjfg"* may be described as the *sameness* group *"jj"* and the *successor group*, consisting of the letters *"f"* and *"g"*. The third main component of the architecture is the *coderack*, which is equal to Jumbo's but based on the current state of the *slipnet* and *workspace*. *Codelets* in the *coderack* examine in parallel the letters of an analogy problem trying to build a coherent set of structures around them and possibly chunking them together into groups based on a common relationship, representing a particular interpretation of the problem. Since analogy problems can have many interpretations (giving rise to a vast space of potential configurations in the *workspace*), Copycat runs according to the *parallel terraced scan*—introduced by Jumbo—that executes several probabilistically selected possible processes in parallel.

Simon et al. [96] used the cognitive computer model SC-Soar, an adaptation of the cognitive architecture SOAR [83] for solving letter series completion. Based on a production system, the SOAR architecture formulates the task in a *problem space* in which different operators are selectively applied to bring the system gradually closer to its goal state. In each decision cycle SOAR brings different pieces of knowledge from its long term recognition memory to the working memory and decides which action to be taken. In particular, SC-Soar used letter series completion tasks and solved them by casting them as comprehension tasks: comprehension operators are applied in order to find relationships between items reaching different states (which encode the knowledge about the relations that characterise the series) for a goal state.

Foundalis's Phaeaco [101] is another cognitive architecture for visual pattern recognition and abstraction. Phaeaco, which creates abstract representations of geometric figures received as input (pixels), uses three mechanisms for solving Bongard's problems. The first mechanism is the *hardwired recall*, related to those problems that can be solved by means of mechanisms hardwired in the human brain. Phaeaco employs a mechanism ("zero variance"), very different from the one humans use, as a first attempt to solve a Bongard problem. If this first stage fails, Phaeaco enters its *holistic view*, related to the initial strategy of conducting a panoramic overview of the problem for a brief period of time in an attempt to see apparent differences. Finally, if the previous stages fail, the systems enters the *the analytic view*, where individual images are selected and reexamined in an effort to come up with fresh ideas due to Phaeaco's nondeterministic nature.

Notwithstanding the previous techniques, the best-known approach to analogy-making is the use of **simulation models** and, in particular, the use of *Structure-mapping Engine* (SME) [85]. SME is a program for analysing analogical processing which compares the similarities and differences between objects. SME has been a significant advance facilitating a lightweight high-level visual matching solving analogy problems. The SME is based on Gentner's structure-mapping theory [149] and operates over symbolic representations (entities, attributes, and relations) and thus, it is used together with sketch understanding systems (such as sKea or CogSketch) using the qualitative spatial representations returned by the latter. SME takes as input two propositional descriptions, a *base* and a *target*, and produces a set of mappings between the pictures aligning their common relational structure. Each mapping consist of: 1) correspondences linking items in the base and target (commonalities in the representations and corresponding objects in two pictures); 2) a structural evaluation score which provides an indication of match quality (giving thus an overall similarity between the pictures); and 3) *candidate inferences* which identify particular differences between pictures. Furthermore, SME accepts input constraints to bias the mappings.

This combination of sketch understanding systems and a structure-mapping engine has been used for solving a wide range of intelligence tests: Tomai et al. [99] revisited Evans's geometric analogy problems. Lovett et al. addressed geometric analogies [103, 104], Raven's Progressive Matrices [105, 77] and odd-one-out problems [106, 107]. For solving both RPMs and odd-one-out problems, Lovett's models also perform a generalisation of the patterns founded using SEQL [150], a model of analogical generalisation (describing what is common between two structural representations of objects) through a process of *progressive abstraction* [151] built upon SME. The generalisation can then be compared to new objects: each given solution in a

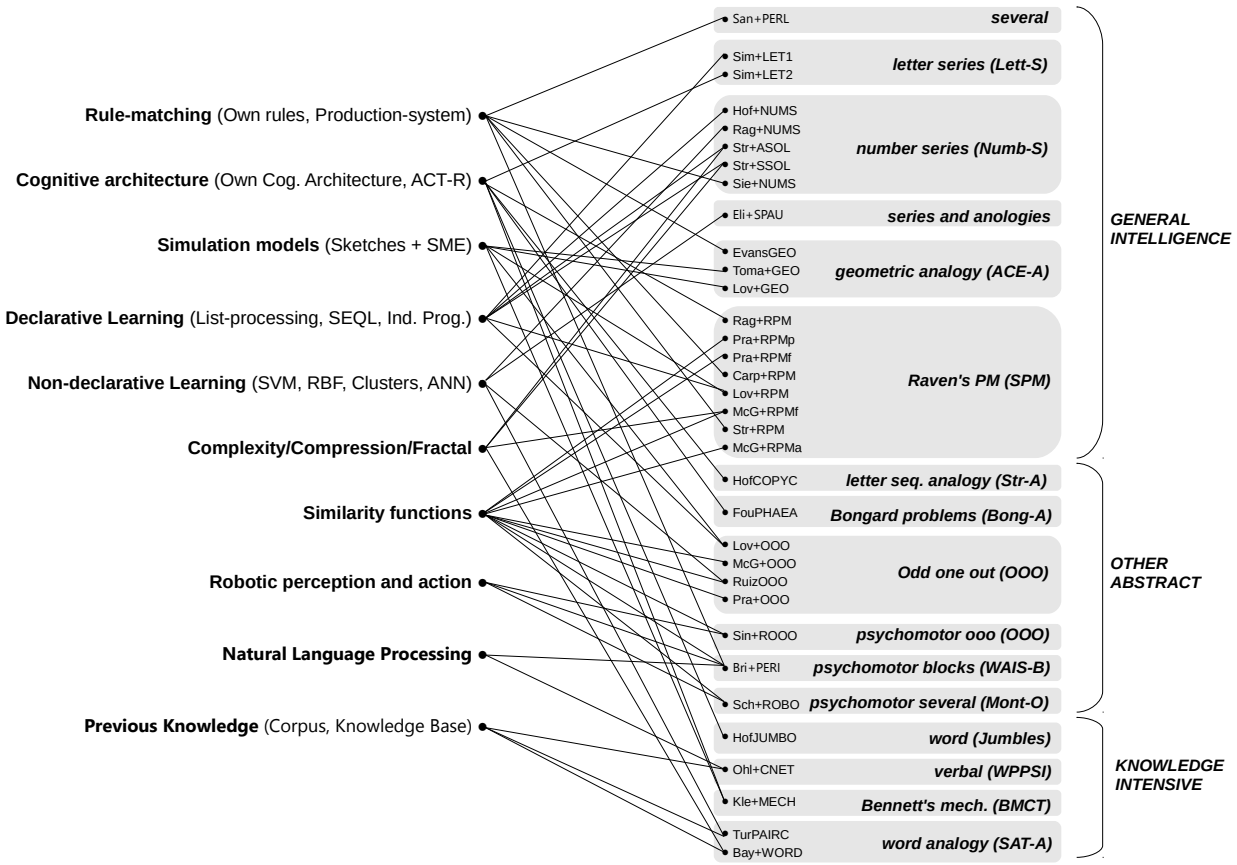


Figure 3: Correspondence between kinds of techniques and problems. Compare with tables 1 and 2. The id of the computer models can be found on the first column of Table 4.

RPM problem is inserted into the matrix, returning the one with the closest matching structural relationship, or, each individual image in an odd-one-out problem can be compared to the generalisation thus returning the noticeably least similar as the solution. Furthermore, since SME automatically figures out what kinds of things can be matched [152], this combination of qualitative representations and structure-mapping engine has also been used (all together with the *Companion Cognitive Architecture* to perform qualitative reasoning) to solve Bennett's Mechanical Comprehension Tests [116], demonstrating its generality.

Techniques related to **declarative learning** such as list processing mechanisms or inductive programming [142, 143] have also been widely used for solving letter and number series intelligence tests. Simon and Kotovsky [11] proposed a list processing procedure to induce pattern descriptions from segments using IPL-V, Newell's information processing language V [93]. The main task of this procedure is to seek for initial conditions, periodicity and relations in the given sequence, and to arrange them in the corresponding pattern. Simon and Kotovsky also developed an extremely simple IPL-V based procedure capable of generating sequences from pattern descriptions by executing elementary list processes called for by the descriptions.

Likewise, Siebers and Schmid [128], Strannegård et al. [131, 132] and Hofmann, Kitzelmann and Schmid [135] demonstrate that the number series problem can be addressed with the use of inductive programming and bounded cognitive resources (use of heuristics emulating certain limitations of human cognition). Siebers and Schmid [128] use basic numerical operators (addition, subtraction, division, multiplication, and exponentiation) to construct the patterns that lead to extrapolate the sequences. Following a human-like strategy, the hypothesis formation is guided by an analytical strategy.

Strannegård et al. [131] developed two Haskell-coded anthropomorphic cognitive systems based also on

the idea of limited working memory: ASolver and SeqSolver. Both systems have a repertoire of predefined patterns (syntactic expressions defining sequences), mathematical operators ($+$, $-$, $*$, $\div$) and simplifications (odd, even, exception, last) used to describe number sequences; and a limited set of computations with bounded cognitive resources (SeqSolver based on Kolmogorov complexity for limiting the set of computations),, that are used to decode patterns (rejecting solution candidates which are too computationally demanding), thus turning them into number sequences.

Hofmann, Kitzelmann, and Schmid [135] used their inductive programming system IGOR2 (a system for learning (recursive) functional programs from input/output examples). IGOR2 can be applied to number series problems using different example presentation: a list of the initial elements of the series as input and the next element as output, a position as input and an enumeration of the series up to this position as output, or a position as input and the value at this position as output.

In addition, some systems use general machine learning techniques, what we have called **non-declarative learning**. In this group of techniques we find the large-scale model of the functioning brain developed by Eliasmith et al. [24]. It is remarkable that the system learns to recognise the problems (with digit recognition) and also writes the answer with a robotic arm, which gives the system an even more impressive look. In Spaun, the tasks are learned, which is clearly different to a hardwired big switch approach. Nonetheless, the re-representation that is made here is different to the one needed in many intelligence test tasks with visual presentation, such as RPM, since the components of each item are digits, unlike the original RPMs.

Ragni and Klein [121], for their part, worked on number series completion applying a general method using artificial neural networks (ANNs) and dynamic learning where the learning rate, the number of input nodes, the number of hidden nodes, and the number of training iterations are manually varied in order to allow for a comparison of the different ANNs. Like other previous techniques, most models based on non-declarative techniques need to rely on other mechanisms or techniques shown on Figure 3. Turney's PairClass [119] uses a standard supervised machine learning algorithm: a sequential minimal optimisation (SMO) support vector machine (SVM) with a radial basis function (RBF) kernel from the machine learning library WEKA to classify word pairs according to their semantic relations. PairClass represents the semantic relations between two words using a high-dimensional feature vector, in which the elements are based on frequencies of patterns in a corpus—consisting of a large corpus of plain text (280GB), gathered by a web crawler and retrieved by the search engine Wumpus [153]—obtained using templates.

Ruiz [120] addresses odd-one-out problems, combining a clustering technique with a similarity function. Ruiz uses the so-called "Ruiz-Absolute Scale of Complexity Management" (RASCM) in order to code or discretise the objects as character strings. For instance, a problem composed of two circles and a square is represented as (A;A;B). In order to represent objects with several features, other letters are used for each object, such as (AC;AD;AE;BF;AG), so representing four circles and a square with different sizes. Once this transformation is made, the problems are addressed by a two-step clustering algorithm that computes the average Hamming distance of the sets in the first step, and, for those misclassified items, it makes a second recoding so that the most frequent symbol within that set becomes A, the second most frequent symbol becomes B, and so on. For example, the three following sets: (AAAD, BBBE, CCDE) become respectively: (AAAB, AAAB, AABC), which clearly unveil AABC as the least similar.

Among the approaches based on **complexity, compression or fractal** techniques, Bayoudh et al. [124] uses the relationship between Kolmogorov complexity and the universal distribution, and searching on a structured text corpus (i.e., the Web), are able to address word analogies. In more detail, the authors use the Kolmogorov complexity as a measure of the quantity of information needed to go from the word w_1 to the word w_2, namely, to define agreement and disagreement between concepts. The estimation for a given pair of words w_1, w_2 is calculated as the log inverse of the words' frequencies within a given corpus.

McGreggor, Kunda, and Goel [110, 111, 112, 113, 114, 115] proposed two different image resolution-based methods (the "fractal" and a "affine" method) for solving RPMs and odd-one-out intelligence test problems. Both methods compare images under a variety of transformations (image rotations and mirrors, scaling, addition, and subtraction), and must judge the similarity between images (based upon features which arise from them) to determine similarity for each possible answer. The main difference is the interpretation of what constitutes a feature in each model: (a) in the fractal method, the representation of an image is a set of fractal codes which compactly describe the geometric alteration and colourisation of fragments

(sets of points) of the source image that will transform into the target image, and the features are derived from these representations (which will be used to determine the similarity for each possible answer across all the relationships present in the problem); (b) in the affine method, a feature is defined to be a single greyscale pixel with each pixel associated with a single intensity value. Both approaches induce all possible transformations for the matrix, both row-wise and column-wise and select the best one according to some measure of fitness. The model applies this transformation to the incomplete row/column to predict an answer which will be compared to each given answer choice according to a similarity measure.

The above combines a fractal approach with similarity. **Similarity functions** have also been in a wide range of computer models. Prade, Richard, and Correa [125, 126, 127], following a logical view of similarity, developed a system for addressing Raven's Progressive Matrices based on solving analogical proportion equations. As seen before (section 6.1), their approach may be applicable at different levels of representation. Their approach starts with the truth table of the analogical proportion (i.e., 0000 or 0101 satisfy an analogical proportion, but 1000 or 1011 do not satisfy it), where each Boolean digit represents the absence or not of a property. The equation-solving principle is used to build the missing item D (rather than selecting it from a set of potential candidates) in an incomplete proportion involving A, B, C. Furthermore, following the logical view of dissimilarity and by using the same Boolean representation and equation-solving principle, Prade and Richard [130, 127] developed an approach to address the odd-one-out problem. The idea is to find the item which has most dissimilarities with respect to the rest of items for each feature.

Unimaginable in the early years, we find **robotic** approaches developed to solve several intelligence test problems (such as WAIS Block problems [16], Montessori tasks and RPMs [137, 129], or odd-one-out problems [109]) making use of an upper-torso humanoid robot with auditory, proprioceptive, and visual sensory abilities. Apart from the robotic abilities, other rule-matching or similarity-based techniques are needed. As an example of how to deal with WAIS (at least partially), Bringsjord and Schimanski's PERI [16] is capable of logic/reasoning, vision, physical manipulation through a five-degree-of-freedom vertically articulated robotic arm, speech, and hearing. At the core of PERI we find a complex Lisp program and an associated "Scorbot Advanced Control Language Library". Meanwhile, Sinapov and Sotytchev [109] present a framework that allows the robot to interact with a set of fifty household objects (cups, bottles, and toys) with the aim of solving personalised odd-one-out tasks. The robot explores each object to detect many of its physical properties (auditory and proprioceptive sensory feedback) to infer the pairwise similarity matrix for the objects that will be used to select the most dissimilar object in the tests. Finally, Schenck et al. [137, 138, 129] address Montessori tasks and RPMs with an upper-torso humanoid robot, as in Sinapov's paper [109]. The robot grounded its representation for the different objects in each task in terms of the auditory and proprioceptive outcomes that they produced in response to a series of exploratory behaviours. Combining information from each sensorimotor context and using similarity measures, the robot was able to estimate the perceptual distance (similarity score) on a given set of objects.

On the other hand, some systems use **natural language processing** methods based on keywords and statistics in order to try to understand basic facts and queries. Systems like the open-source crowdsourced knowledge base *ConceptNet* [141] support many practical textual-reasoning tasks. Ohlsson's system [134] uses ConceptNet tools to map the input words to concepts in the system, Python algorithms to wire the different parts of the system, and the *AnalogySpace*, a concise version of the large common-sense knowledge base of ConceptNet, which is queried to obtain relations (between concepts). For a question such as, "Where can you find a penguin?", their system will query its knowledge base for the words "find" and "penguin".

Finally, Turney's PairClass [119], Ohlsson's CNET [134] and Bayoudh's system seen above use **previous knowledge** more intensively, in integration with other techniques. Looking at Figure 3, the first seven kinds of techniques are general and can be applied to any type of problem. The bottom three, though, are clearly more restricted to some types of problems. While some categories can be mapped to AI areas seen in Table 2, and ultimately to Table 1, it is still difficult to tell for a particular problem which kinds of AI techniques may be more suitable. This resembles what have been observed in the area of game playing. Some games are well solved with machine learning (reinforcement learning, neural networks, etc.), others are solved with planning techniques, others with constraint processing, or a combination of techniques. However, from the experience of general game playing, it is clear that the ad hoc "rule-matching" approach does not work if the system is aimed at a range of problems [23].

7. Discussion

After giving a historical overview of the systems in section 5 and a discussion of the techniques of the systems in section 6 we will now discuss the key questions introduced in section 4. The criteria used for answering the questions were summarised in Table 3 and the systems were characterised based on these criteria in Table 4. In this table, we see a row for each model (or a group of models) and the value for each criterion. An overview of the table indicates that most approaches are very recent. If we look at the 'range' column, we see that the computer models usually address one problem and the most common one has been RPMs. An asterisk is shown in the column 'representation' to indicate that the representation of the problem has been transformed. This is very common for RPMs. The next column shows that the intention behind the models is usually 'AI principles' or 'human cognition', whereas 'evaluation' is not very common. There is a wide variety in the techniques used, from more ad-hoc to more general AI techniques. Finally, there is also a huge diversity in whether performance and difficulty are assessed. Overall, this variability in values for all criteria will help us to address the questions we formulated in section 4.

7.1. Are computer models solving intelligence tests becoming more wide-spread and more relevant for AI?

To answer the first question (Q1) we have to look at the column 'years' in Table 4, where we see a burgeoning increase in the past four years. Is it an indication of relevance? According to the venues, we see that they go from mainstream AI to cognitive science, or even psychology, and some of them are in leading conferences and journals in these areas or even in interdisciplinary general outlets. According to the 'intention' criterion, it seems that most approaches aim at unveiling general (artificial) intelligence principles in ways that are not necessarily connected to the way humans solve these tests. This suggests that this is attracting more interest in artificial intelligence and cognitive science than in psychology.

What about the use of these tests for AI evaluation? Are they becoming more common? It has been recently argued—from human intelligence researchers—that intelligence tests are the right tool to evaluate AI systems. In February 2011, Douglas K. Detterman, the editor-in-chief of *Intelligence* wrote an editorial after seeing how successful IBM's program Watson [4] (the recent winner of the *Jeopardy!* TV quiz show at the time) had been (at least as a way to acknowledge the progress of artificial intelligence for lay people). As Watson was clearly unable to do other tasks and clearly non-intelligent, Detterman claimed that AI systems should be better measured by classical intelligence tests. The challenge was set in this way [50]: "I, the editorial board of *Intelligence*, and members of the International Society for Intelligence Research will develop a unique battery of intelligence tests that would be administered to that computer and would result in an actual IQ score". It is important to note that Detterman's challenge had two levels: the first level allowed the challenge designers to look at the (kinds of) tests beforehand, and construct their system according to this information, while the second (true) level allowed the committee to use any possible test not previously disclosed to the designer or the system. This is a very appropriate distinction in the context of this paper, since in the previous sections we have seen systems that could come close to the first level, but there is no attempt that could come close to the second level.

Nonetheless, we do not see that artificial intelligence has changed its evaluation protocols following this increase of models taking intelligence tests (the only exceptions are the works by Sinapov [15], Sotytchev [109], and Schenck et al. [137, 138, 129] for robots and model brains such as Spaun [24]). But overall, Detterman's challenge has had almost no impact in AI. We see that these models are becoming more common and widespread as testbeds for experimentation, but not so as regular tools for AI evaluation.

7.2. How challenging are intelligence test problems for AI and what are the challenges?

We need to be clear that focussing on the overall results of a computer model and comparing them with the results of humans (column 'performance' on Table 4) is not very informative about how challenging the problem is (question Q2). First, humans are general-purpose systems and it is not fair to compare them with some systems that are only able to solve one problem—even if the problem comes from an intelligence test. Second, many of these intelligence test problems have been developed for humans, and hence it can be unfair to evaluate AI systems limitations with anthropocentric measures. Third, some of the works perform an interesting analysis in terms of difficulty (column 'difficulty' on Table 4). The purpose is to determine

what instances are more difficult, but this is not very related to how challenging the problem is. In fact, focussing on the most difficult problems may even make the system more specialised to the intelligence test task at hand. Some of the previous works have studied whether difficulty is related to the size of the working memory, the size of the pattern, or the number of elements that need to be combined or retrieved from background knowledge [11, 14, 131, 132]. These notions of difficulty are much more general and can work independently of the problem and the representation.

The key issue is to consider a greater diversity of problems (this claim is made by Psychometric AI, and Detterman's second level). As we can see on Table 4, very few approaches address more than one kind of test. Actually, the more specific a test is the easier it is to develop specific solutions. Likewise, some problems that were very 'challenging' (e.g., chess) for machines are now excelled at by specialised programs. In addition, 'representation' is also crucial, as some problems involving complex pattern recognition (those involving images) need some processing. In fact, the same problem (e.g., RPM) is very challenging if it is presented visually—as it is presented for humans—but it is not so if translated to an appropriate symbolic representation. Nevertheless, we can still specialise a system to do the re-representation for just one task. This point is clearly shown by current CAPTCHAs [43], where humans that pass CAPTCHAs more quickly or effectively—usually without thinking—are not necessarily the most intelligent ones. Diversity and representation are hence crucial to distinguish between specific and general AI systems. The challenging goal should be to develop a single general system that is able to solve *all* of them *as they are*.

All in all, the answer to Q2 is that these intelligence test problems were challenging in isolation for a time, but many of them are now solvable by specific approaches. As a consequence, the challenge (and real usefulness) of these kinds of exercises relies on the use of a diverse battery of tests, keeping the original representation, in order to obtain a meaningful interpretation.

7.3. Are computer models for solving intelligence test problems useful for understanding human *cognition?*

To answer the third question (Q3) we have to look at the similarities between the AI models and human cognition. Some of the works featuring computer models have tried to mimic how humans solve these problems. Nonetheless, many anthropomorphic models have succeeded and failed on the same problem items that humans do, but some other non-anthropomorphic approaches have also shown some degree of coincidence with humans, even if the mechanism to solve the items was completely different. Also, very good results have been obtained with techniques that are clearly different from those that humans use. Interestingly, when a problem is challenging for humans but not so for machines, this is very explanatory. For instance, machines are much better at arithmetic than humans. Nonetheless, we can see this from a different view. As we are using human tests, why are these kinds of problems (and not others) useful to measure human abilities? Why are Raven progressive matrices different from chess or multiplication? The use of intelligence tests for machines gives very insightful information about what intelligence tests measure and what they do not and, ultimately, about what characterises intelligence in humans.

Most interestingly, using computer models for intelligence tests can elucidate the relation between different tasks, as the factorial analysis in psychometrics reflects how abilities (or tasks) are correlated for humans. However, psychometrics says nothing about how abilities are correlated in principle, in a computational way. For instance, psychometrics may have found that the g factor is correlated to many other abilities. But does this happen for machines? Can we implement models with good results on tests measuring g and very poor results on the other tests? This seems to be possible, as some of the results shown in section 5 suggest. In fact, this even sheds more doubts about the validity of comprehensive tests such as WAIS for machines, because they are inspired or derived from the knowledge about factorial analysis of human abilities in the past century. Nonetheless this can also be seen as an opportunity of computer models, AI, and other approaches based on information theory, to help improving intelligence tests.

Overall, the answer to this third question is that some of these models have been useful to provide insights and valuable information about how human cognition works. This is especially the case when there is a coincidence of results between a model and humans even if the model was not conceived to follow exactly what humans do. Nonetheless, a systematic disagreement in results or ability correlations may also be very informative. In fact, these studies can be very useful to better understand what intelligence tests really measure and to better understand the correlations between the abilities found in humans.

7.4. Are these models useful for understanding intelligence in general*?*

The fourth question (Q4) is more general. Most of the models shown in the column 'intention' of Table 4 were motivated by a better understanding of what intelligence is and how to recreate it. Actually, the criteria 'range' and 'specialisation' are most meaningful here. We see that some intelligence test problems integrate too many processes and may take many others for granted. By seeing what tasks are meaningful for humans and not for machines and vice versa, we can construct a less anthropomorphic view of intelligence tests. This is related to the early distinction between knowledge and intelligence that guided the early days of psychometrics. As seen in Table 4, most problems can be solved without a knowledge base, especially those featuring inductive inference, such as letter and number series, analogy problems, and matrices. A notable exception are tests based on vocabulary, where the knowledge of a language is needed. If we focus on fluid intelligence tests, we see that most problems are cases of inductive inference where the patterns can be inferred from very few examples. This problem was more mainstream in the early days of artificial intelligence and machine learning, but is no more a priority area in machine learning, which has shifted its focus on the analysis of big data. Some areas in artificial intelligence, however, such as inductive programming [142, 143] and programming by example are still interested in the problem of learning complex patterns from limited evidence. Natural language acquisition is an example of how learning can take place from limited (and mostly partial) evidence. However, we must note here that most intelligence test tasks involving natural language evaluate the ability of applying language (by solving language analogies, completing a story, or finding synonyms and antonyms) rather than evaluating how new words and structures can be acquired. This may suggest that general intelligence tests may lack some tasks that are not usually included for standard human evaluation, and should be complemented with more incremental and developmental tasks, as the cumulative acquisition of knowledge may be a key element in what we understand by intelligence, such as developing and learning a language from scratch.

Similarly, the view of difficulty requires different foundations. For instance, some of the proposals for evaluating machines seen in section 2.4 are based on a non-anthropocentric measurement of intelligence and other cognitive abilities via (algorithmic) information theory. Again, the evaluation of tasks in isolation may be misleading, as difficulty depends on previous constructs being available. As a computational formalisation of all this can be difficult, only the integration of all these approaches from psychometrics, artificial intelligence, and other areas may lead to a full understanding of what intelligence is.

Consequently, the answer of this fourth question is that, in our opinion, some of these models are shedding light about the kind of problems a general AI system should solve, but all this shows that other more universal and formal approaches for the definition and analysis of tasks will be required.

7.5. What is the correspondence between the areas in AI and the abilities covered in intelligence tests?

The fifth question (Q5) deals with the techniques that have been used in these models. As was discussed in section 6, many models use specific techniques, either by developing new techniques from scratch or by performing a very particular adaptation of existing techniques. In fact, only a few common AI techniques (see column 'techniques' in Table 4) are used, mostly from machine learning, pattern recognition, automated reasoning, and natural language processing (apart from some other more miscellaneous techniques such as fractals and ad-hoc rules). It is interesting that a few cases have developed new techniques.

If we look at Table 1 and especially Table 2, we see a correspondence between AI subdisciplines and cognitive abilities. However, if we look at the models and the techniques through Table 4 and Figure 4, it seems that there is no clear correspondence between the kind of problem and the techniques from AI that have been used to solve it. There is again more prevalence of techniques related to inductive reasoning, but some specialised techniques have also been used in some problems to circumvent this need. Some approaches (many using their 'own rules') are actually variants of a big switch.

We also see that some perception techniques are used for the representation mapping when the problems are originally presented in a visual way. These techniques are of course unrelated to whether the task is inductive or deductive, but are just necessary to process the visual tasks. As a consequence, there is some correspondence between techniques and abilities but the alignment is far from perfect, due to the specialisation, as many systems are still more task-oriented than ability-oriented.

Model	Raven's SPM					Raven's APM	
	A (12)	B (12)	C (12)	D (12)	E (12)	Set I (12)	Set II (36)
Carp+RPM (FAIRAVEN)	-	-	-	-	-	7[a]	16[b]
Carp+RPM (BETTERAVEN)	-	-	-	-	-	7[a]	25[b]
Lov+RPM			44[c]			-	-
Rag+RPM	-	-	12	12	11	-	31
McG+RPMf	11	7	5	7	2	12	26
McG+RPMa[d]	12	11	8	1	3	7	14
Pra+RPMp	-	-	-	-	-	-	16
Pra+RPMf	-	-	-	-	-	-	16 + 16[e]
Str+RPM	-	-	-	-	-	7	16

Table 5: Comparison of systems addressing RPMs using the results given in the corresponding papers. Many of these systems did not have high accuracy as a goal or used different transformations of the instances. Notes: [a]Out of 7 attempted. [b]Out of 27 attempted. [c]From sets B to E. [d]Results shown for the *standard* configuration. [e]The pixel-based approach was able to solve 16 problems and the feature-based approach solved 16 additional problems.

7.6. Are the systems comparable and what are the requirements for a general and meaningful comparison?

For this sixth question (Q6) we need to look first at the dependencies between the models in Table 4. There are two types of comparisons that can be made with these systems. First, we can perform a *bona fide* comparison using the data from the associated papers. For instance, from all the systems in Table 4, there is only sufficient data in the papers to compare the *results* for those addressing RPMs. This is shown in Table 5. The insight from this table is narrow, for many different reasons. First, many of the systems were not designed to achieve good performance but to study the problems or the human performance on them. Second, they do not use the same subsets (the Set II of APM is the one for which we can compare the largest number of systems: just 5). This is sometimes caused by confidentiality agreements and copyrights. Third, many systems could have been overfitted for the subset of items they used, and the results may not generalise to other items. Fourth, we do not have information in most cases about the results for particular items, so a system may be successful on the most difficult ones, or vice versa.

A more meaningful analysis could be performed if we were able to analyse more systems and use different item sets. In order to determine whether this comparison could be possible, we performed a survey (a series of questionnaires sent by email to the systems' authors) in order to determine the availability of the systems and the input format the systems use. We complemented this information with the set of problems these systems can handle. The result is given in Figure 4, where we show that systems can only be compared for isolated clusters (given by the problems they can handle). An additional difficulty has been the public availability of some models, thus making their comparison impossible. Furthermore, an extra challenge lies in the input representation. Note, for instance, that solid links between systems solving RPMs (and also for those solving odd-one-out problems) indicates that, although almost all of them use propositional input representations, the dimension, number, type, syntax, and semantics of the features used vary for each model. This makes it extremely difficult to ensure equivalence between the input problems to be solved in each model. Therefore, we see that a comparison of several systems for a range of problems is not possible. Making this explicit is in fact one of the crucial observations of this paper, and again another insightful parallel with general game playing (as it was in 2005, where game playing systems could not be compared). It seems that apart from the RPM cluster in Figure 4 (for which we have already discussed the comparison in Table 5 there are a few models that could be compared for number series, only three of them (Rag+NUMS, San+PERL, and Sie+NUMS) share a similar representation (raw numbers with minor variations). However, there is no point in making a comparison between these three systems since their aims were completely different: the goal of Sanghi and Dowe [15] was not to excel in any specific task, the aim of Ragni and Klein [121] was to successfully solve large sets of number series from OEIS, and the goal of Siebers and Schmid [128] was to get cognitive plausibility for series actually found in intelligence tests (including failure on those difficult series for which humans fail) instead of excelling on performance.

As a result, the comparison would become much more meaningful with systems that are able to handle several kinds of problems. An effort must be made towards a standardised format for problem description and the availability of a large variety of instances—so that system cannot prepare for a particular set.This leads to the construction of a repository, as we will discuss below in question Q9.

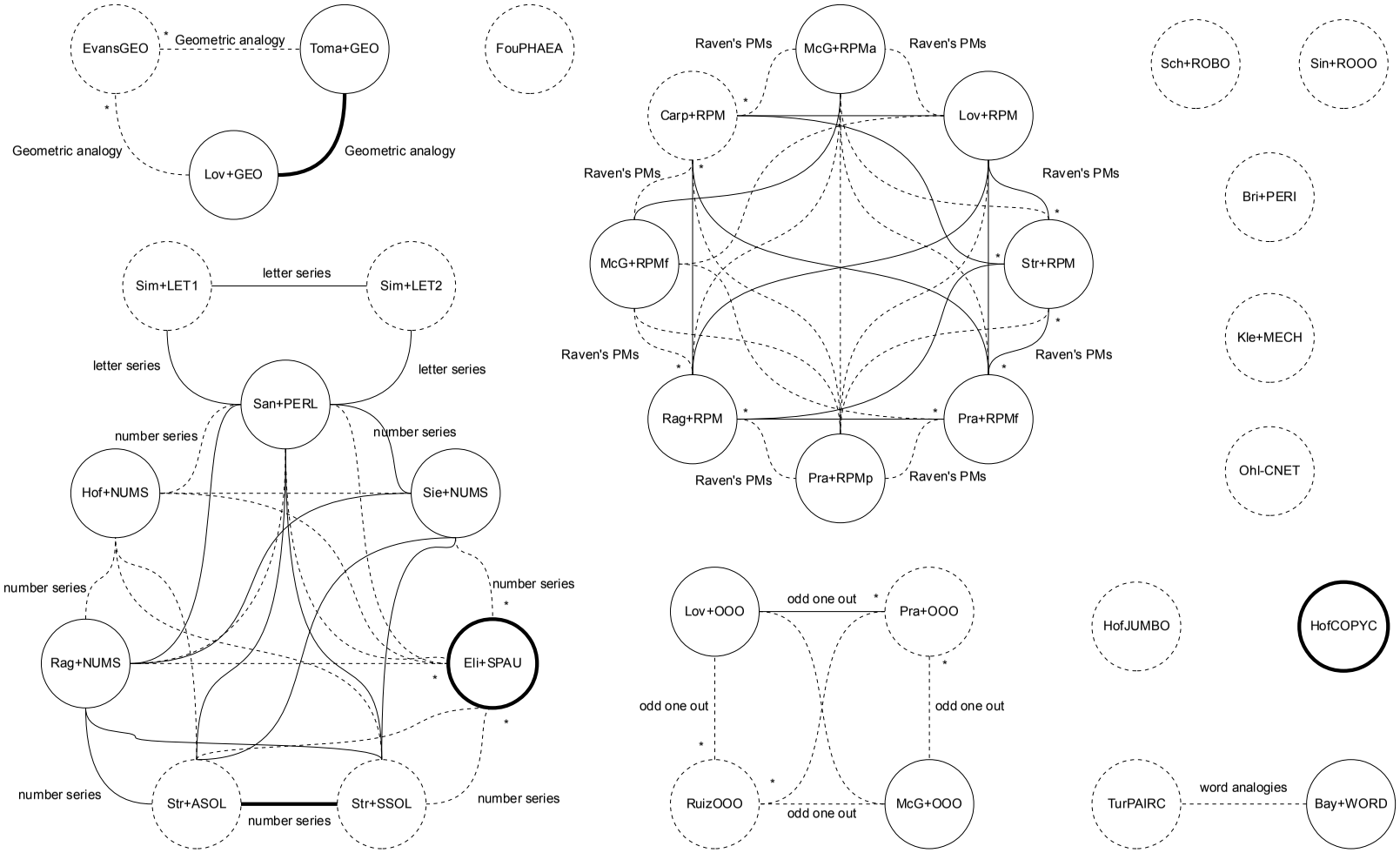


Figure 4: Complete account of all systems and whether they can be compared and to which extent. The identifier inside the circles corresponds to the first column in Table 4. Dashed circles indicate that the code is not available. Solid circles show availability on demand (thin line) or publicly (thick line). Dashed edges mean that both systems are comparable in principle but they use different representations. Solid edges correspond to similar representation (thick solid means that the format is compatible). An asterisk is shown when there is a (handcrafted) preprocessing transformation on the representation. We see that problems determine the clusters, as most systems are specialised for just one problem. Note that thick solid links correspond to approaches developed (and evolved) by (almost) the same authors.

7.7. Is there progress in techniques within the field and has it impact on AI techniques in general?

The seventh question (Q7) is more controversial. In the case of intelligence test tasks, we have seen in sections 5 and 6, that improvement from the early systems exists. However, if we look at the columns 'years' and 'techniques' in Table 4, we see that only a few systems have reused the same techniques, and in many cases it is because there are common authors. Also, even for those cases where the techniques are similar, the 'range' of tasks is different. In fact, some works we have seen in section 5 do not even cite all the relevant literature for the same task, which clearly limits the transfer between systems. The lack of correspondence that we observed between Tables 1 and 2 and the techniques that we see in Table 4 and Figure 3 must be an indication that something is wrong here. Of course there may be many interpretations for all this. First, there is a possibility that AI techniques are not appropriate (or too specific) to solve these problems. Second, there is the possibility that many of these works were not able to adapt AI techniques more smoothly. But there is also the possibility that AI has not considered these problems interesting enough, or simply, as mentioned above, that the systems have not been designed in many cases to excel in the results, but to better understand the nature of the problems and how humans solve them. Actually, this was the motivation behind one of the systems, Sanghi and Dowe's, [15], which showed that if the mere goal is to score well in these tests, one can do a very simple switch approach (with no AI techniques whatsoever).

Also, if we compare Tables 1 and 2 we see a mismatchbetween deduction and induction. This may suggest that inductive reasoning is still the area where more progress is required. In fact, for most approaches the system does not learn to solve the problems but it is programmed to solve the problems. In other words, *the task is hard-coded into the program* and it can be easier to become 'superhuman' in many specific tasks, as happens with chess, draughts, some kinds of planning, and many other tasks. But humans are not programmed to do intelligence tests. Only for a few cases (such as Spaun, but with many limitations and just some kinds of tasks), the system is *trained* to solve intelligence test tasks. We think that this is one of the lessons learnt in AI. Many techniques that have been developed in the past decades have to be adapted for a specific problem. But the ultimate goal of AI is to make systems that learn to solve new tasks they have never seen before. Not even machine learning systems are ready to do this.

It seems that the main issue is how to make more general systems that can take a range of problems, not even anticipated during design. Since we do not assume that the systems will be able to handle instructions given in natural language (not even in a specific description language, as in the general game playing competition), we expect that they identify what to do from some examples. An important thing here is that we do not expect systems to learn from hundreds of examples, as this is not the way humans handle a new test. Tasks in intelligence tests are sufficiently abstract such that the identification of what has to be done can be inferred from a few illustrative examples with solutions, usually at the beginning of the test.Furthermore, humans are able to understand what to do, without further instructions. The creation of systems capable of doing this is more related to the areas of learning by demonstration and inductive programming [142, 143], as mentioned above, rather than other areas of artificial intelligence and machine learning.

As a result, the answer to this seventh question is that there has been limited feedback between the systems, and the progress is caused by the painstaking integration of more methods from AI and some incremental development for some particular techniques. However, the techniques that originated from these systems have had very limited impact on mainstream AI.

7.8. Can these models help bridge research in psychometrics, cognitive science, and AI?

To answer this eighth question (Q8) we need to identify possible links between psychometrics, cognitive science, and AI. There seems to be broad interest in computer models solving intelligence tests in all three disciplines if we look at the column 'intention' in Table 4.

A first common topic is the assessment of problem difficulty. Psychometrics and cognitive science have a view of difficulty in terms of *instances*, while artificial intelligence is more concerned about *problem* class complexity. Psychometrics usually derives difficulty empirically from human populations, cognitive science does this using theoretical models of item difficulty. We have seen some approaches in section 5 where instance difficulty is derived from the space and time resources that are needed. This approach is

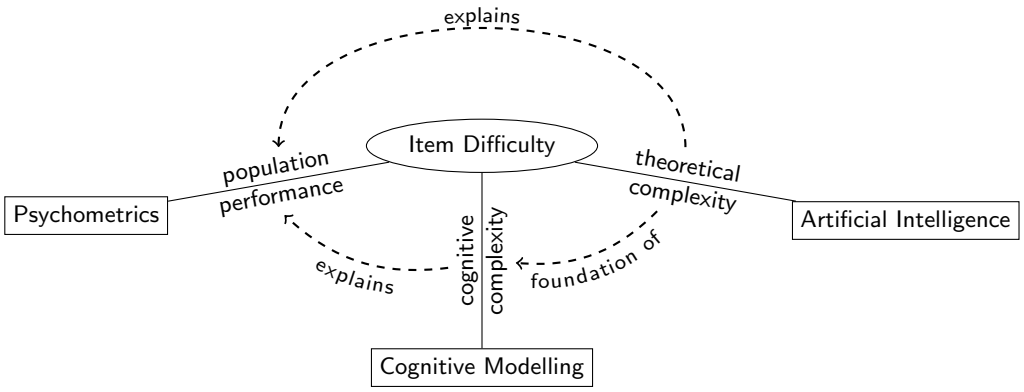


Figure 5: Bridge research between psychometrics, cognitive science, and AI. The explanation from AI to psychometrics (without a cognitive model) usually implies that the explanation is more universal and depends on the problem and not on how it is solved in particular (as humans do and cognitive models try to simulate).

consistent with psychometrics, cognitive science, and artificial intelligence, and has a solid foundation in algorithmic information theory (AIT). Combining the different perspectives on item difficulty provided by these research areas might give rise to a more general understanding of what makes a problem difficult for an intelligent agent—human or artificial—and may be a very productive bridge for these three disciplines (see Figure 5). Computational models of analytical complexity using AIT or the more traditional computational complexity theory could be brought in relation to the empirically derived measures of item difficulty from psychometrics and thereby provide an explanation why some items are solved only by a small number of subjects. Cognitive models, which simulate the reasoning process solving intelligence test problems, can also contribute to explain why some problems are harder to solve for most humans—in terms of the algorithmic effort of encoding the problem, detecting regularities, and generalising over them. An analysis of cognitive models based on AIT can not only provide a theoretical foundation of these models but also provide a way of generating the problems (as done in [62]). In consequence, future developments in psychometrics can profit from models that can generate new problem instances and from the formal assessment of their difficulty and the abilities they are really measuring. Item response curves could be, in principle, be derived from these models, as already suggested in [70].

Developmental robotics is an area that can also contribute to bridge the three disciplines. In fact, we have seen that some approaches presented in section 5 originate from this area, and that they require a broad range of psychometric tests, not only intelligence tests.

7.9. How should these tests be arranged and changed in order to serve as evaluation tools for AI?

The ninth and last question (Q9) focusses on the use of these tests as evaluation tools for AI. It seems clear that the use of one single test that is known a priori is mostly useless for evaluation, as we can specialise a system to solve that problem. As has been mentioned above, Detterman claimed that AI needed a "battery of intelligence tests" [50]. The systems we have seen in section 5 are, in general, only able to deal with one kind of test. Nonetheless, it is not difficult to imagine a system integrating the approaches in Table 4, in order to attempt the first level of the challenge with a big switch approach. In other words, if the battery becomes publicly available (like PEBL, `http://pebl.sourceforge.net/battery.html`), systems will specialise to it. This happens to some extent with other benchmarks and batteries in computer science, and is reduced as the battery becomes larger . This is not easy for (standardised psychometric testssolve this issue by making the tests copyrighted and confidential).

This general-vs-specific distinction may seem sufficient to clarify many issues about the use of intelligence tests to evaluate AI systems. This is also in agreement with the Psychometric Artificial Intelligence roadmap for artificial intelligence. However, not everyone seems to agree. Dowe and Hernández-Orallo [51] argued, in a response to Detterman, that the second level is not appropriate either for evaluating the progress of

artificial intelligence, as the test batteries must be changed as AI advances. They also argued that intelligence tests may be too anthropocentric to properly evaluate artificial intelligence.

In order to be useful evaluation tools for AI, several things must be considered. Instead of a collection of problems, a better approach may be a collection of instance generators, by integrating some of the existing ones we saw in section 5 and developing new ones. The collection must be large, in order to avoid the big switch approach applied to a small repertoire (or ranges as in Table 4). Moreover, brand-new problems could be generated by the combination of existing ones or by the development of more abstract problem generators (instead of instance generators). Different presentations and difficulty levels should be explored. The categories and overlaps between problems could be assessed via theoretical models, instead of using factor analysis as in psychometrics. In other words, a theoretical alternative to the classification of mental abilities—as presented in Tables 1 and 2—should be endeavoured.

About how the tests are selected and conceived, there seem to be convincing reasons to use tests that are *universal* [64, 66], i.e., they should be administered to machine and human (and other animals) alike. Note that if tests are generated anew, we do not have the standard human results for them. That means that these tests would have to be applied to humans (and animals, if possible) in order to get empirical evidence about whether algorithmically generated problems correspond to problem difficulties experienced by humans. In other words, even if we derive the difficulty of items from theoretical models and apply it to construct adaptive tests using item response theory, we would need to compare them with the difficulty found for humans. As has happened with many problem items in the past of for whole sections of test batteries, a model of problems or a generator can be discarded if it is not in agreement with the results found in humans. This does not mean that human results should be the ultimate reference, but at least any set of items we claim that measures intelligence should be minimally consistent to measuring intelligence in humans, and should be human-normed if the results are to be compared to other tests. Note that some of the models we have seen in this paper start from some results from humans and then try to fit a difficulty model to them. The ultimate goal is to do it the other way round, to find good theories in artificial intelligence (based on computation or information theory) such that the difficulty that we estimate corresponds to the difficulty that is experimentally found by human subjects (e.g., this was the case in [62]).

Generators should ensure that the same instances are generated for all participants, as is done in the general game playing competition. However, if the evaluations are performed during a longer period of time, we must be sure that the comparisons are fair. For instance, generators that create instances with a difficulty value are well suited for adaptive testing. One feature of adaptive testing is that the probability that two subjects are evaluated with the same instances is very low, requiring that the sampling procedures are well designed and well tested before proceeding with the systematic evaluation.

An additional aspect is whether the systems are signalled in some way when the task changes. In other words, can a system receive RPM instances and letter series problems in any order, without being notified that the task has changed? This does not only mean that the system needs to learn the task but has to *identify* the task. For instance, Spaun, which is trained for each task, gets confused if tasks are mixed.

As the above seems too ambitious as a short-term goal, the construction of a repository using some human intelligence tests (such as PEBL), despite its limitations, could be a seed for a more proper universal benchmark in the future. This would be beneficial (at least at this moment and in the near future), as several computer models could work with the same task instances, using the same presentation, as well a previously fixed difficulty assessment. The repository could also record some previous results and the kind of abilities they usually represent in humans (as seen in section 2). This repository could also integrate some other measurement tools from AI that try to be general and domain-independent [154].

Of course, the development of an evaluation testbed for AI (or universally) would require an effort and integration of interests and communities for which we cannot anticipate whether and when this will take place. Nonetheless, we propose a roadmap in Figure 6. At a first stage, a repository would contain problems represented in some generic format and system authors would have to write their own interfaces to transform the test problems in a format required by their specific system. A similar strategy was used by the AI planning community before the problem domain description language PDDL was established [155]. A second stage would shift from problem sets to problem generators—using hybridisation between the problems, with better understanding of their difficulty and their relations. A final third stage would

contain a series of ability-oriented tests, based on first principles from information theory and computation.

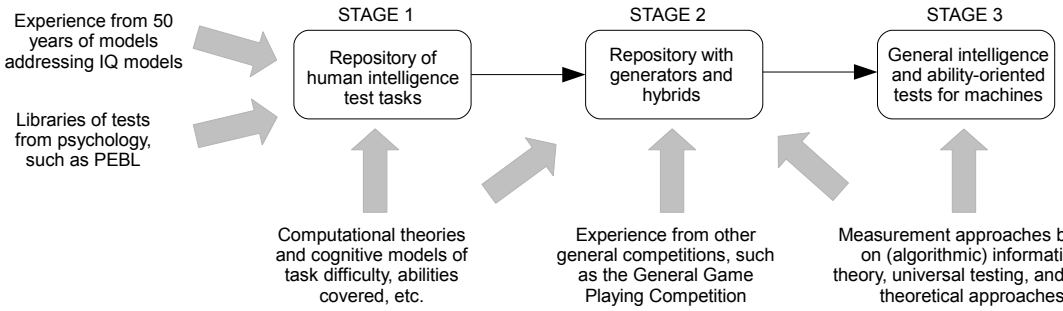


Figure 6: Suggested roadmap about a repository of intelligence test tasks at a first stage, elaborated into a repository mostly based on generators for a second stage and a third stage where tests are generated from computational principles.

The roadmap in Figure 6 is a possible suggestion under the assumption that an incremental development of testbeds reusing some of the experience from human intelligence tests is useful. Shortcuts could be found to reach the third stage without the use of human intelligence tests at all—though this pathway is not precisely easy [62, 63, 64]—or through a generalisation of task-oriented evaluation, as the proposed *Turing championship* [156] and other proposals arising from the increasing interest in AI evaluation (`http://www.math.unipd.it/~frossi/BeyondTuring2015/`).

One way or the other, there seems to be an agreement that there will be an increasing number of machines in the near future which show a range of cognitive abilities, and that we will require evaluation mechanisms for them. These mechanisms will have to give scores for several cognitive abilities so that we can compare them with humans and other animals. As a realisation of how much work needs to be done here, the development of well-grounded tests for humans, animals, robots, agents, animats, hybrids, swarms, etc., has been proposed in an emerging new (but integrating) discipline, dubbed 'universal psychometrics' [70], which may inherit and integrate many important concepts from psychometrics, cognition, and AI.

8. Conclusions

This paper was motivated by an observed explosion of the number of papers featuring computer models addressing intelligence test problems. Among the around 30 papers we have analysed, half of them have appeared since 2011. We wanted to investigate whether this increase was casual or was motivated by an increasing need of these tests and the computer models solving them. When we began our investigation we soon realised that computer models addressing intelligence tests have different purposes and applications: to advance AI by the use of challenging problems (this is the Psychometric AI approach), to use them for the evaluation of AI systems, to better understand intelligence tests and what they measure (including item difficulty), and, finally, to better understand what (human) intelligence is.

Note that if any or all of the above reasons were spurious, its negation would still be most interesting. Namely, if intelligence tests were not challenging, were not useful for the evaluation of AI systems, did not measure the purported abilities beyond humans, or were useless to understand what human intelligence is, then this would represent important insights for psychometrics, cognitive science, and artificial intelligence. In fact, the authors of this paper may have different stances on some of these issues, which partially explains why all of us agree that this area of research is worth being pursued and empowered in the near future. Another thing that motivated this paper is that there seems to be a limited connection between these works, as many of them seem to ignore results and ideas already present in previous approaches. This includes the (mis)understanding of what a computer model passing an intelligence test really means. We hope that this work could facilitate and encourage any future computer model taking intelligence test problems to link with and build upon previous research.

This is, in fact, a lesson learned for the field of AI. We have seen that even for supposedly general tasks that are designed for evaluation, many approaches have the (understandable) tendency to specialise to the

task and hard-wire parts (or most) of the solution. This is yet another indication of a discipline like AI that is being extremely successful in task-specific applications—from playing chess to driving a car—but yet of limited success in general-purpose systems. In a nutshell, we could say that AI has become a big switch discipline. This problem is being recognised in AI itself, as several benchmarks and competitions are now aiming at more general classes of problems (e.g., the general game playing competition [6]) and we hope that this paper contributes to a more widespread realisation of this when constructing new benchmarks, as relevant things are happening in AI evaluation [157], such as the proposal of a *Turing championship* [156].

As mentioned in the previous section, a very ambitious goal would be to create a repository or generator of all these problems. We know that many intelligence tests are not publicly available, and many of the approaches we have surveyed here have used alternative formulations because of this. It would be very useful for AI to arrange these problems, record the results of computer models and humans over them and organise competitions. This first stage of the incremental roadmap suggested in Figure 6 should have some conditions. The benchmark should be broad (including a wide range of tests), standard (using some kind of general protocol for inputs and outputs), characterised (accompanied with a catalogue of information about their difficulty, the abilities they cover, etc.), available (on a web or problem library) and renewed (new items are generated or disclosed so that systems cannot rote-learn them). The General Game Playing competition can be used as a good example that follows all these conditions.

Finally, a continued and enlarged research program on computer models taking intelligence tests would help with the delimiting and, when possible, integration of the approaches discussed in section 2 and illustrated in Figure 5. One of the key issues about these tests is to ascertain to what degree they are anthropocentric and what parts in them (or which of them) could remain as universal, hence applicable to AI systems and humans alike.

Acknowledgements: We thank the editor and reviewers for their thorough and insightful comments.

Appendix A.

In this appendix we extend the problems briefly introduced and classified in section 3. For the sake of exposition, we will arrange the test problems into three categories: general intelligence problems, other abstract pattern problems and knowledge-intensive problems. The two first categories do not require previous knowledge (or it is restricted to some basic arithmetic or geometrical operations). The last category requires the application of previous knowledge (and language).

Appendix A.1. General intelligence problems

In this first category we include several fluid intelligence problems that highly correlate with the g-factor. Typically, solving these problems involves inductive inference, abstraction, search and combination.

Thurstone letter series completion (Lett-S): The letter series completion problems were introduced as part of Thurstone's PMA theory [12]. The goal is to identify the following letter in a series (see Figure A.7) from five letter choices. To solve a letter series, an abstract pattern has to be identified which captures the regularity of the sequence. Correctness in the context of induction problems typically presupposes that there is one continuation which is most plausible with respect to the given regularity. Typically, problems are carefully constructed in such a way that there is a unique solution. Simon and Kotovsky [11] made the first attempt to solve them and understand their complexity using a computer model.

abababab__
aaabbbcccdd__
cadaeafa__
wxaxybyzczadab__
mnlnknjn__

Figure A.7: Examples of Thurstone letter series completion problems [11].

Number series (Numb-S): Similar to the Thurstone letter series completion problems we find the continuation of number series. This kind of problem is also included in some intelligence tests (such as the German IST-2000 [158]). Number series problems can be constructed to be much more complex than letter series due to the greater number of combinations and operations, as, e.g., the Fibonacci sequence $1, 1, 2, 3, 5, 8$. In fact, the problem gets closer to a crystallised intelligence task (instead of a fluid intelligence task) when sequences are generated by the use of a large library of mathematical functions, so requiring a correct memory search and retrieval by the subject. For instance, some problems can be just obtained from the Online Encyclopedia of Integer Sequences (`http://oeis.org`)[159]. There are many computer models exploring these problems [121, 24, 128, 131, 132].

Geometric analogy in American Council of Education tests (ACE-A): The ACE tests [160] were designed to predict or measure the scholastic aptitude (or general intelligence) of the participants. The examination consists of six tests arranged into two groups: *Quantitative Tests*, which includes arithmetical reasoning, number series, and figure analogies tasks; and *Linguistic Tests*, which includes same-opposite, completion, and verbal analogies tasks. In particular, we are interested in the the so-called "geometric-analogy" or "figure analogies" problems, as these have been addressed by some computer models [9, 10, 99, 103], as seen in section 5. The task to perform can be described as: "figure A is to figure B as figure C is to _" where "A", "B", and "C" are given and "D" should be derived (Figure A.8). Note that in order to solve these tasks, the process must involve some spatial reasoning and similarity recognition to identify the figures and occasionally perform spatial operations on them.

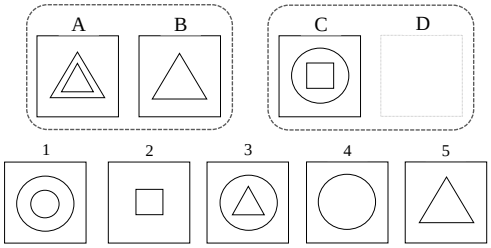


Figure A.8: ACE problem figure (redrawn following [10], originally from the 1942 edition of the Psychological Test for College Freshmen of the American Council on Education, ACE).

Raven's Progressive Matrices (RPM): Originally from Raven [95], Raven's Progressive Matrices consist of a pattern or a set of items where a missing part or item has to be guessed. The most typical case is a 3×3 grid where a figure is placed at each of the nine positions except the bottom-right cell, which is empty (see Figure A.9)[3]. Eight possible choices ('distractors') to fill in the gap are displayed at the bottom. There is a logical relation between the figures, which can be seen either horizontally (rows) or vertically (columns). There are three different sets of Raven Progressive Matrices (RPM) for participants of different abilities [95]: the original Standard Progressive Matrices (SPM), the simpler Coloured Progressive Matrices (CPM), and the Advanced Progressive Matrices (APM). We will just focus on SPM, which consist of 5 sets with 12 items in each set—60 items in total. Sets A and B do not use a 3×3 grid structure, so computer models usually attempt sets C, D, and E, as they share the same structure (although the difficulty varies). SPMs are, by far, the task that has been attempted by most computer models (partially or completely) [14, 105, 77, 110, 111, 122, 112, 113, 125, 126, 24, 133]. As in the previous case, these problems involve processes that require spatial reasoning abilities, similarity recognition, and analogies, but also sequences.

Appendix A.2. Other abstract pattern problems

In this category, we include some problems that do not require the application of a significant amount of knowledge but cannot be considered general, as they measure very specific abilities.

[3]Please note that items from standardised IQ tests are not allowed to be published due to copyright issues. Therefore, we can only present some examples or problems which are similar but not identical to the ones used in IQ tests.

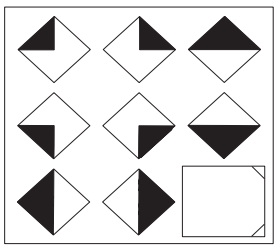


Figure A.9: Raven's Progressive Matrices example [own example].

Block design in WAIS (WAIS-B): These are geometrical problems where blocks with only white, only red, and both white and red sides have to be arranged according to a presented pattern (see Figure A.10). This task aims at measuring spatial perception, visual abstract processing, and problem solving abilities. The difficulty lies in figuring out how to rotate and combine the blocks such that they correspond to a *given* pattern. Block design in WAIS has been used in some computer models we survey in this paper [16].

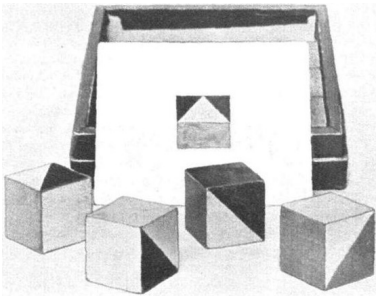


Figure A.10: A block problem such as those found in Wechsler Adult Intelligent Scale, WAIS [98]. The picture shows one problem from the Kohs Block Design test developed by Samuel Calmin Kohs and subsequently adapted into WAIS by David Wechsler. (Taken from Wikimedia Commons: `http://commons.wikimedia.org/wiki/File:Kohs_Block_Design_Test_-_Figure_1.jpg`)

Odd-one-out problems (OOO): The odd-one-out problems are focussed on geometry and spatial understanding, where the goal is to spot the most dissimilar object from the rest (see Figure A.11). They were first introduced in [161, 162] as a non-verbal test in the study of comparative animal learning (Zentall et al. [161] assessed pigeon's intelligence). The oddity task has also been used for cross-cultural testing to probe the conceptual primitives of geometry in the Mundurukú, an isolated Amazonian indigenous group, for which Deheane et al. [163] designed a visual oddity task. Typically, the presented items can vary along several dimensions (e.g., shape, size, quantity). The problems can be automatically generated by using a complex set of algorithms (`http://www.cambridgebrainsciences.com/`).

To our knowledge, the computer models that have addressed this kind of problem are [106, 109, 114, 120].

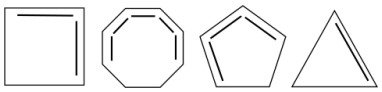


Figure A.11: An example of an odd-one-out problem [own example].

Bongard's analogy problems (Bong-A): Bongard's analogy problems are a set of puzzles about visual categorisation that were originally introduced for *visual pattern recognition*. They were invented by the Russian computer scientist M. M. Bongard [102], who provided a notable subset of a hundred problems in the appendix of the original publication. Each problem is formed by six boxes on the left side and another six on the right side containing relatively simple diagrams (see Figure A.12). The goal is to guess the pattern, or

rule, that appears in the left set which is lacking in all the diagrams of the set on the right. A few years later, Hofstadter [164] was more interested in the problems themselves as abstract analogy problems, and referred to them as "Bongard Problems", which is the term that has prevailed thereafter in the literature. Many computational architectures have been devised to solve Bongard problems from cognitive science [164, 101] and AI [165] becoming a common dataset in areas such as inductive logic programming [166, 167, 168, 169] and other areas of machine learning and pattern recognition. To our knowledge, only one system, Phaeaco [101], has attempted the resolution of an important number of these problems.

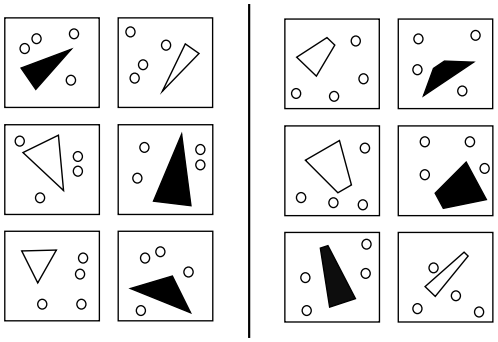


Figure A.12: An example of a Bongard problem [own example].

String analogies (Str-A): Originally referred to as "word analogies" (we use the term 'string analogies', as they are not really words of a language), Hofstadter [94] introduced short alphabetic sequence puzzles (see Figure A.13) as input problems. He focussed on Bongard's problems, and simplified them in order to avoid the difficulties arising when constructing representations for visual problems. The processes involved in this problem are similar to the letter series problem but with the additional use of analogical transfer.

abc ⇒ abd	ijk ⇒ ?
aabc ⇒ aabd	ijkk ⇒ ?
aac ⇒ abd	kji ⇒ ?
abc ⇒ abd	mrrjjj ⇒ ?
abc ⇒ abd	xyz ⇒ ?

Figure A.13: String analogy problems (originally referred to as "word analogies") analysed by the Copycat project [170].

Montessori's object matching (Mont-O): The Montessori method [171] is an educational method developed by Maria Montessori, an influential Italian physician and educator, in 1897 (but popularised in the 1910s). This method encouraged children to focus their attention on one particular 'quality', working at his/her own optimum level, so it makes emphasis on independence and freedom within limits. Several tasks require the children to compare objects, moving, rotating, or touching them (see Figure A.14). One typical task inside the Montessori method is *Object-to-object matching*, where the child is asked to find the matches from one set to another, e.g., matching coloured tiles, 3-dimensional shapes, or pieces of textured cloth. This problem requires some cognitive abilities, including visual encoding, object grouping, similarity assessment, category recognition, and object ordering. Recently, a sensorimotor robot has been able to solve this kind of object pairing task [137].

Appendix A.3. Knowledge-intensive problems

Finally, we group another set of tasks that depend on the crystallised use of knowledge and experience.

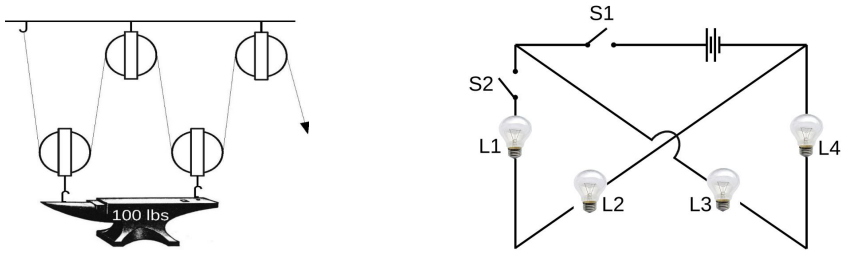
Hofstadter's anagrams (Jumbles): Frequently found in newspaper puzzle sections, jumbles are games where five or six letters are given with the aim of unscrambling them into an English word. In order to

Figure A.14: Materials used in some exercises and tests of the Montessori method. ["Montessori Materials", Diamond Montessori, retrieved from `https://www.flickr.com/photos/rossmenot/351505158/`, CC BY 2.0]

solve them, humans try to find common letter patterns in order to generate blocks. For instance, the jumble "yurfip" refers to "purify". Clearly, these tasks require the use of a large background knowledge (vocabulary) but also the ability of making combinations. Jumbles were used by Hofstadter as a domain of his prototype system Jumbo [94], prelude to the Copycat project [74].

Verbal common-sense reasoning problems in WPPSI (WPPSI): Wechsler developed several variants of WAIS that were aimed for individuals aged under 16: the Wechsler Intelligence Scale for Children (WISC) and the Wechsler Preschool and Primary Scale of Intelligence (WPPSI). The verbal part of the WPPSI (WPPSI-III, third Edition) includes questions of the type: "What color is the sky?", "What are pancakes made out of?", "Finish what I say. Pen and pencil are both ...", "You can see through it", "It is square and you can open it", etc. This task has been used to evaluate the common-sense reasoning system ConceptNet 4 [134]. The processes involved are language understanding and common-sense reasoning.

Bennett mechanical comprehension tests (BMCT): The Bennett Mechanical Comprehension Test (BMCT) [117] is an aptitude test for the evaluation of human common sense relying on everyday reasoning. Namely, it measures the ability to perceive, understand, and reason about the physical world (with an important content about physics and contextual information) and is commonly used to evaluate applicants for technical positions (typically, electrical and mechanical positions). The BMCT is composed of 68 diagrams depicting physical situations involving many different mechanisms, and is accompanied by multiple-choice questions about their qualitative properties. To solve this test, some abilities are required: spatial reasoning, conceptual knowledge spanning a broad range of domains (dynamics, acoustics, statics, electricity, and heat), and experience with a wide variety of everyday situations (boats, trains, bicycles, cranes, hoists, ...). In Figure A.15 we can see two examples that represent the two general types of problems that appear on the BMCT [116]. One computer model [116] has dealt with this type of problem so far.



1. Approximately how much force is needed to lift the anvil?

a) 100 lbs b) 50 lbs c) 25 lbs d) 10 lbs e) 200 lbs

2. If the switch S1 is closed and switch S2 remains open, which lights will turn on?

a) L1&L2 b) All c) L3&L4 d) L1 &L4 e) None

Figure A.15: Figurative recreation of Bennett mechanical comprehension test problems [own example].

Word analogies in Scholastic Assessment Test (SAT-A): The Scholastic Assessment Test (SAT reasoning test) is a standardised test (or entrance exam) developed by the non-profit organisation "College Board", which is used in the USA as an admission prerequisite. The test is intended to assess literacy and writing skills that are needed for university. SAT consists of three major sections: *Critical Reading*, including sentence completions and questions about short passages; *Mathematics*, including multiple choice and grid-in about topics such as algebra and geometry (calculator use is allowed); and *Writing*, including multiple choice questions and a brief essay. It includes some tasks that are very similar to the problems we may find in intelligence tests. For instance, the SAT analogy questions (word analogies) include exercises of the form *teacher : chalk :: soldier : ?* or *sun : planet :: earth : ?*. The procedure to solve word analogy problems corresponds to the procedure to solve geometrical analogies or letter string analogies. However, to detect the relation between the first pair of words and to create the solution, conceptual knowledge and verbal knowledge is necessary. This problem has been used by [119] (374 multiple-choice questions).

References

[1] J. McCarthy, What is artificial intelligence, URL: http://www-formal.stanford.edu/jmc/whatisai.html.
[2] A. L. Samuel, Some studies in machine learning using the game of checkers, IBM Journal of Research and Development 3 (3) (1959) 210–229.
[3] M. Campbell, A. J. Hoane, F. Hsu, Deep Blue, Artificial Intelligence 134 (1-2) (2002) 57–83.
[4] D. Ferrucci, E. Brown, J. Chu-Carroll, J. Fan, D. Gondek, A. A. Kalyanpur, A. Lally, J. Murdock, E. Nyberg, J. Prager, et al., Building Watson: An overview of the DeepQA project, AI Magazine 31 (3) (2010) 59–79.
[5] D. Ferrucci, A. Levas, S. Bagchi, D. Gondek, E. T. Mueller, Watson: Beyond jeopardy!, Artificial Intelligence 199 (2013) 93–105.
[6] M. Genesereth, N. Love, B. Pell, General game playing: Overview of the aaai competition, AI Magazine 26 (2) (2005) 62–72.
[7] R. J. Sternberg (ed.), Handbook of intelligence, Cambridge University Press, 2000.
[8] A. Newell, You can't play 20 questions with nature and win: Projective comments on the papers of this symposium, in: Visual Information Processing, ed. W. Chase, New York: Academic Press, 1973, pp. 283–308.
[9] T. Evans, A heuristic program of solving geometric analogy problems, Ph.D. thesis, Mass. Inst. Tech., Cambridge, Mass., U.S.A., also available from AF Cambridge Research Lab, Hanscom AFB, Bedford, Mass., U.S.A.: Data Sciences Lab, Phys and Math Sci Res Paper 64, Project 4641 (1963).
[10] T. Evans, A heuristic program to solve geometric-analogy problems, in: Proc. SJCC, Vol. 25, 1965, pp. 327–339, vol. 25.
[11] H. A. Simon, K. Kotovsky, Human acquisition of concepts for sequential patterns., Psychological Review 70 (6) (1963) 534.
[12] L. Thurstone, T. Thurstone, Factorial studies of intelligence, Psychometrika monograph suplements, The University of Chicago press, 1941.
[13] L. Blum, M. Blum, Toward a mathematical theory of inductive inference, Information and Control 28 (2) (1975) 125–155.
[14] P. A. Carpenter, M. A. Just, P. Shell, What one intelligence test measures: A theoretical account of the processing in the Raven Progressive Matrices test, Psychological review 97 (1990) 404–431.
[15] P. Sanghi, D. L. Dowe, A computer program capable of passing IQ tests, in: 4th Intl. Conf. on Cognitive Science (ICCS'03), Sydney, 2003, pp. 570–575.
[16] S. Bringsjord, B. Schimanski, What is artificial intelligence? Psychometric AI as an answer, in: International Joint Conference on Artificial Intelligence, 2003, pp. 887–893.
[17] H. J. Berliner, Backgammon computer program beats world champion, Artificial Intelligence 14 (2) (1980) 205–220.
[18] D. B. Benson, Life in the game of go, Information Sciences 10 (2) (1976) 1729.
[19] B. Bouzy, T. Cazenave, Computer go: An AI oriented survey, Artificial Intelligence 132 (1) (2001) 39–103.
[20] J. Pitrat, Realization of a general game-playing program, in: A. J. H. Morrell (Ed.), Information Processing 68, North-holland Publishing Co., 1968, pp. 1570–1574.
[21] J. Pitrat, A general game-playing program, in: N. V. Findler, B. Meltzer (Eds.), Artificial Intelligence and Heuristic Programming, Edinburgh University Press, 1971, pp. 125–155.
[22] N. Love, T. Hinrichs, M. Genesereth, General game playing: Game description language specification, Tech. Rep. LG-2006-01, Stanford Logic Group, Computer Science Department, Stanford University (2006).
[23] M. Genesereth, Y. Bjrnsson, The international general game playing competition, AI Magazine 34 (2013) 107–111.
[24] C. Eliasmith, T. C. Stewart, X. Choo, T. Bekolay, T. DeWolf, C. Tang, D. Rasmussen, A large-scale model of the functioning brain, Science 338 (6111) (2012) 1202–1205.
[25] E. Yong, A large-scale model of the functioning brain, Nature 29.
[26] C. K. Machens, Building the human brain, Science 338 (6111) (2012) 1156–1157.
[27] I. Eibl-Eibesfeldt, Human ethology, Transaction Publishers, 2007.
[28] A. M. Turing, Computing machinery and intelligence, Mind 59 (1950) 433–460.
[29] S. Harnad, The Turing Test is not a trick: Turing indistinguishability is a scientific criterion, ACM SIGART Bulletin 3 (4) (1992) 9–10.

[30] S. Harnad, The annotation game: On Turing (1950) on computing, machinery, and intelligence, The Turing Test Sourcebook: Philosophical and Methodological Issues in the Quest for the Thinking Computer.
[31] G. Oppy, D. L. Dowe, The Turing Test, in: E. N. Zalta (Ed.), Stanford Encyclopedia of Philosophy, 2011, pp. Stanford University, http://plato.stanford.edu/entries/turing–test/.
[32] J. Searle, Minds, brains, and programs, The Behavioral and Brain Sciences 3 (1980) 417–457.
[33] J. Weizenbaum, ELIZA – a computer program for the study of natural language communication between man and machine, Communications of the ACM 9 (1) (1966) 36–45.
[34] D. Proudfoot, Anthropomorphism and AI: Turing's much misunderstood imitation game, Artificial Intelligence 175 (5) (2011) 950–957.
[35] D. M. McDermott, The 1998 AI planning systems competition, AI magazine 21 (2) (2000) 35.
[36] G. Sutcliffe, C. Suttner, Evaluating general purpose automated theorem proving systems, Artificial Intelligence 131 (1) (2001) 39–54.
[37] B. Starkie, M. van Zaanen, D. Estival, The Tenjinno machine translation competition, in: Grammatical Inference: Algorithms and Applications, Springer, 2006, pp. 214–226.
[38] R. Madhavan, E. Tunstel, E. Messina, Performance Evaluation and Benchmarking of Intelligent Systems, Springer, September, 2009.
[39] J. Anderson, J. Baltes, C. T. Cheng, Robotics competitions as benchmarks for AI research, The Knowledge Engineering Review 26 (01) (2011) 11–17.
[40] M. Miller, The savant syndrome: intellectual impairment and exceptional skill, Psychol Bull. 125(1) (1999) 31–46.
[41] J. Schaeffer, N. Burch, Y. Bjornsson, A. Kishimoto, M. Muller, R. Lake, P. Lu, S. Sutphen, Checkers is solved, Science 317 (5844) (2007) 1518.
[42] M. Campbell, M. Egerstedt, J. P. How, R. M. Murray, Autonomous driving in urban environments: approaches, lessons and challenges, Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences 368 (1928) (2010) 4649–4672.
[43] L. von Ahn, M. Blum, J. Langford, Telling humans and computers apart automatically, Communications of the ACM 47 (2) (2004) 56–60.
[44] A. T. Cianciolo, R. J. Sternberg, Intelligence: A brief history, John Wiley & Sons, 2008.
[45] R. B. Cattell, Theory of fluid and crystallized intelligence: A critical experiment, Journal of educational psychology 54 (1–22) (1963) 1.
[46] K. W. Schaie, Primary Mental Abilities, John Wiley & Sons, Inc., 2010.
[47] T. Z. Keith, M. R. Reynolds, Cattell–Horn–Carroll abilities and cognitive tests: What we've learned from 20 years of research, Psychology in the Schools 47 (7) (2010) 635–650.
[48] L. L. Thurstone, Primary mental abilities, Chicago, Ill. : The University of Chicago press, 1938.
[49] S. Bringsjord, Psychometric artificial intelligence, Journal of Experimental & Theoretical Artificial Intelligence 23 (3) (2011) 271–277.
[50] D. K. Detterman, A challenge to Watson, Intelligence 39 (2-3) (2011) 77–78.
[51] D. L. Dowe, J. Hernández-Orallo, IQ tests are not for machines, yet, Intelligence 40 (2) (2012) 77–81.
[52] F. M. Lord, Applications of item response theory to practical testing problems, Mahwah, NJ: Erlbaum, 1980.
[53] S. E. Embretson, S. P. Reise, Item response theory for psychologists, L. Erlbaum, 2000.
[54] A. Newell, H. Simon, GPS, A program that simulates human thought, in: E. Feigenbaum, J. Feldman (Eds.), Computers and Thought, McGraw-Hill, New York, 1963, pp. 279–293.
[55] R. J. Solomonoff, A formal theory of inductive inference. Part I, Information and control 7 (1) (1964) 1–22.
[56] M. Li, P. Vitányi, An introduction to Kolmogorov complexity and its applications (3rd ed.), Springer-Verlag, 2008.
[57] C. S. Wallace, D. M. Boulton, An information measure for classification, Computer Journal 11 (2) (1968) 185–194.
[58] C. S. Wallace, D. L. Dowe, Minimum message length and Kolmogorov complexity, Computer Journal 42 (4) (1999) 270–283, special issue on Kolmogorov complexity.
[59] D. L. Dowe, A. R. Hajek, A computational extension to the Turing Test, proc. of the 4th conference of the australasian cognitive science society, university of newcastle, nsw, australia (1997).
[60] J. Hernández-Orallo, N. Minaya-Collado, A formal definition of intelligence based on an intensional variant of Kolmogorov complexity, in: Proc. Intl Symposium of Engineering of Intelligent Systems (EIS'98), ICSC Press, 1998, pp. 146–163.
[61] D. L. Dowe, A. R. Hajek, A non-behavioural, computational extension to the Turing Test, in: Intl. Conf. on Computational Intelligence & multimedia applications (ICCIMA'98), Gippsland, Australia, 1998, pp. 101–106.
[62] J. Hernández-Orallo, Beyond the Turing Test, J. Logic, Language & Information 9 (4) (2000) 447–466.
[63] S. Legg, M. Hutter, Universal intelligence: A definition of machine intelligence, Minds and Machines 17 (4) (2007) 391–444.
[64] J. Hernández-Orallo, D. L. Dowe, Measuring universal intelligence: Towards an anytime intelligence test, Artificial Intelligence 174 (18) (2010) 1508–1539.
[65] J. Insa-Cabrera, D. L. Dowe, S. España-Cubillo, M. V. Hernández-Lloreda, J. Hernández-Orallo, Comparing humans and AI agents, in: J. Schmidhuber, K. Thórisson, M. Looks (Eds.), Artificial General Intelligence, Vol. 6830, LNAI, Springer, 2011, pp. 122–132.
[66] D. L. Dowe, J. Hernández-Orallo, How universal can an intelligence test be?, Adaptive Behavior 22 (1) (2014) 51–69.
[67] J. Hernández-Orallo, On the computational measurement of intelligence factors, in: A. Meystel (Ed.), Performance metrics for intelligent systems workshop, National Institute of Standards and Technology, Gaithersburg, MD, U.S.A., 2000, pp. 1–8.
[68] J. Hernández-Orallo, D. L. Dowe, S. España-Cubillo, M. V. Hernández-Lloreda, J. Insa-Cabrera, On more realistic

environment distributions for defining, evaluating and developing intelligence, in: J. Schmidhuber, K. Thórisson, M. Looks (Eds.), Artificial General Intelligence, Vol. 6830, LNAI, Springer, 2011, pp. 82–91.
[69] J. Hernández-Orallo, J. Insa, D. L. Dowe, B. Hibbard, Turing Tests with Turing machines, in: A. Voronkov (Ed.), Turing-100, Vol. 10, EPiC Series, 2012, pp. 140–156.
[70] J. Hernández-Orallo, D. L. Dowe, M. V. Hernández-Lloreda, Universal psychometrics: Measuring cognitive abilities in the machine kingdom, Cognitive Systems Research 27 (2014) 5074.
[71] J. E. Laird, N. Derbinsky, M. Tinkerhess, A case study in integrating probabilistic decision making and learning in a symbolic cognitive architecture: Soar plays dice, in: AAAI Fall Symposium: Advances in Cognitive Systems, 2011, pp. 162–169.
[72] P. Langley, J. E. Laird, S. Rogers, Cognitive architectures: Research issues and challenges, Cognitive Systems Research 10 (2) (2009) 141–160.
[73] R. Sun, The importance of cognitive architectures: An analysis based on CLARION, Journal of Experimental and Theoretical Artificial Intelligence 19 (2) (2007) 159–193.
[74] D. R. Hofstadter, M. Mitchell, The copycat project (1984).
[75] M. Hutter, Universal Artificial Intelligence: Sequential Decisions based on Algorithmic Probability, Springer, 2005.
[76] B. Goertzel, S. V. Bugaj, AGI preschool: a framework for evaluating early-stage human-like AGIs, in: Proc. of the 2nd Intl. Conf. on Artificial General Intelligence (AGI-09), 2009, pp. 31–36.
[77] A. Lovett, K. Forbus, J. Usher, A structure-mapping model of Ravens Progressive Matrices, in: Proceedings of CogSci, Vol. 10, 2010, pp. 2761–2766.
[78] C. Lebiere, Profile: The FMS Cognitive Modeling Group at Carnegie Mellon university, IEEE Intelligent Informatics Bulletin 12 (1) (2011) 1–3.
[79] E. Nyamsuren, N. Taatgen, Set as an instance of a real-world visual-cognitive task, Cognitive Science 37 (1) (2013) 146–175.
[80] D. Gentner, K. Holyoak, B. Kokinov, The analogical mind: Perspectives from cognitive science, MIT Press, Cambridge, MA, 2000.
[81] H. Prade, G. Richard, Computational Approaches to Analogical Reasoning: Current Trends, Springer, 2014.
[82] C. Eliasmith, O. Trujillo, The use and abuse of large-scale brain models, Current Opinion in Neurobiology 25 (2014) 1–6.
[83] J. E. Laird, A. Newell, P. S. Rosenbloom, SOAR: An architecture for general intelligence, Artificial Intelligence 33 (1) (1987) 1–64.
[84] J. E. Laird, Extending the Soar cognitive architecture, in: P. Wang, S. Franklin (Eds.), Artificial General Intelligence 2008: Proceedings of the First AGI Conference, IOS Press Inc, 2008, pp. 224–235.
[85] B. Falkenhainer, K. D. Forbus, D. Gentner, The structure-mapping engine: Algorithm and examples, Artificial Intelligence 41 (1989) 1–63.
[86] S. Weller, U. Schmid, Solving proportional analogies by E-generalization, in: KI 2006: Advances in Artificial Intelligence, 29th Annual German Conference on AI, KI 2006, Proceedings, LNAI, Springer, Heidelberg, 2006, pp. 64–75.
[87] D. D. Salvucci, J. R. Anderson, Integrating analogical mapping and general problem solving: the path-mapping theory, Cognitive Science 25 (1) (2001) 67–110.
[88] S. Itzhaky, S. Gulwani, N. Immerman, M. Sagiv, Solving geometry problems using a combination of symbolic and numerical reasoning, in: Logic for Programming, Artificial Intelligence, and Reasoning, Springer, 2013, pp. 457–472.
[89] M. J. Hosseini, H. Hajishirzi, O. Etzioni, N. Kushman, Learning to solve arithmetic word problems with verb categorization, in: Conference on Empirical Methods in Natural Language Processing (EMNLP), 2014.
[90] M. J. Seo, H. Hajishirzi, A. Farhadi, O. Etzioni, Diagram understanding in geometry questions, in: Twenty-Eighth AAAI Conference on Artificial Intelligence, 2014.
[91] N. Kushman, Y. Artzi, L. Zettlemoyer, R. Barzilay, Learning to automatically solve algebra word problems, ACL (1) (2014) 271–281.
[92] R. J. Solomonoff, Some recent work in artificial intelligence, Proceedings of the IEEE 54 (12) (1966) 1687–1697.
[93] A. Newell, Information processing language-V manual, Prentice-Hall, 1961.
[94] D. R. Hofstadter, The architecture of Jumbo, in: Proceedings of the International Machine Learning Workshop, University of Illinois Press, 1983, pp. 161–170.
[95] J. C. Raven, J. H. Court, J. Raven, Manual for Raven's Progressive Matrices and Vocabulary Scale, San Antonio, TX: Psychological Corporation, 1992.
[96] T. Simon, D. Klahr, A. Newell, SCSoar: Pattern induction in series completion problem solving, in: European Soar Workshop, Cambridge, 1991, p. 17.
[97] C. Spearman, General Intelligence, Objectively Determined and Measured, The American Journal of Psychology 15 (2) (1904) 201–92.
[98] D. Wechsler, Wechsler Adult Intelligence Scale-Revised, San Antonio, TX: Psychological Corporation, 1981.
[99] E. Tomai, A. Lovett, K. Forbus, J. Usher, A structure mapping model for solving geometric analogy problems, in: Proceedings of the 27th Annual Conference of the Cognitive Science Society, Cognitive Science Society, Inc, Stressa, Italy, 2005, pp. 2190–2195.
[100] K. D. Forbus, J. Usher, Sketching for knowledge capture: A progress report, in: Proceedings of the 7th International Conference on Intelligent User Interfaces, IUI '02, ACM, New York, NY, USA, 2002, pp. 71–77.
[101] H. E. Foundalis, Phaeaco: A cognitive architecture inspired by Bongard's problems, Doctoral thesis, Indiana University, Bloomington.
[102] M. M. Bongard, Pattern Recognition, Spartan Books, 1970.
[103] A. Lovett, E. Tomai, K. Forbus, J. Usher, Solving geometric analogy problems through two-stage analogical mapping,

Cognitive Science 33 (7) (2009) 1192–1231.
[104] A. Lovett, K. Forbus, Modeling multiple strategies for solving geometric analogy problems, in: Proceedings of the 34th Annual Conference of the Cognitive Science Society, 2012, pp. 701–706.
[105] A. Lovett, K. Forbus, J. Usher, Analogy with qualitative spatial representations can simulate solving Ravens Progressive Matrices, in: Proceedings of the 29th Annual Conference of the Cognitive Society, Vol. 30, 2007, p. 34.
[106] A. Lovett, K. Lockwood, K. Forbus, A computational model of the visual oddity task, in: Proceedings of the 30th Annual Conference of the Cognitive Science Society. Washington, DC, Vol. 25, 2008, p. 29.
[107] A. Lovett, K. Forbus, Cultural commonalities and differences in spatial problem-solving: A computational analysis, Cognition 121 (2) (2011) 281–287.
[108] K. Forbus, J. Usher, A. Lovett, K. Lockwood, J. Wetzel, Cogsketch: Open-domain sketch understanding for cognitive science research and for education, in: Proceedings of the Eurographics Workshop on Sketch-Based Interfaces and Modeling, 2008.
[109] J. Sinapov, A. Stoytchev, The odd one out task: Toward an intelligence test for robots, in: Development and Learning (ICDL), 2010 IEEE 9th International Conference on, IEEE, 2010, pp. 126–131.
[110] K. McGreggor, M. Kunda, A. Goel, A fractal analogy approach to the Ravens test of intelligence, in: AAAI workshops at the 24th AAAI conference on Artificial Intelligence, 2010, pp. 69–75.
[111] M. Kunda, K. McGreggor, A. Goel, Taking a look (literally!) at the Raven's intelligence test: Two visual solution strategies, in: Proc. 32nd Annual Meeting of the Cognitive Science Society, Portland, 2010, pp. 1691–1696.
[112] M. Kunda, K. McGreggor, A. Goel, Reasoning on the Ravens Advanced Progressive Matrices test with iconic visual representations, in: 34th Annual Conference of the Cognitive Science Society, Portland, OR, 2012, pp. 1828–1833.
[113] M. Kunda, K. McGreggor, A. Goel, A computational model for solving problems from the Ravens Progressive Matrices intelligence test using iconic visual representations, Cognitive Systems Research (2013) 22–23, 47–66.
[114] K. McGreggor, A. Goel, Finding the odd one out: a fractal analogical approach, in: Proceedings of the 8th ACM conference on Creativity and cognition, ACM, New York, NY, USA, 2011, pp. 289–298.
[115] K. McGreggor, A. K. Goel, Fractally finding the odd one out: An analogical strategy for noticing novelty, in: AAAI Fall Symposium: Advances in Cognitive Systems, 2011, pp. 224–231.
[116] M. Klenk, K. Forbus, E. Tomai, H. Kim, Using analogical model formulation with sketches to solve Bennett mechanical comprehension test problems, Journal of Experimental & Theoretical Artificial Intelligence 23 (3) (2011) 299–327.
[117] G. K. Bennett, Bennett mechanical comprehension test, Psychological Corporation, 1969.
[118] K. D. Forbus, D. Gentner, Qualitative mental models: Simulations or memories, in: Proceedings of the Eleventh International Workshop on Qualitative Reasoning, 1997, pp. 3–6.
[119] P. D. Turney, Analogy perception applied to seven tests of word comprehension, Journal of Experimental & Theoretical Artificial Intelligence 23 (3) (2011) 343–362.
[120] P. E. Ruiz, Building and solving odd-one-out classification problems: A systematic approach, Intelligence 39 (5) (2011) 342–350.
[121] M. Ragni, A. Klein, Predicting numbers: an AI approach to solving number series, in: KI 2011: Advances in Artificial Intelligence, Springer, 2011, pp. 255–259.
[122] M. Ragni, S. Neubert, Solving Ravens IQ-tests: An AI and cognitive modeling approach, in: ECAI, IOS Press, 2012, pp. 666–671.
[123] M. Ragni, S. Neubert, Analyzing ravens intelligence test: Cognitive model, demand, and complexity, in: H. Prade, G. Richard (Eds.), Computational Approaches to Analogical Reasoning: Current Trends, Vol. 548 of Studies in Computational Intelligence, Springer, 2014, pp. 351–370.
[124] M. Bayoudh, H. Prade, R. G., Evaluation of analogical proportions through kolmogorov complexity, Knowledge-Based Systems 29 (0) (2012) 20–30, artificial Intelligence 2010.
[125] H. Prade, G. Richard, Analogy-making for solving IQ tests: A logical view, Case-Based Reasoning Research and Development (2011) 241–257.
[126] W. Correa, H. Prade, G. Richard, When intelligence is just a matter of copying, in: Proc. 20th Europ. Conf. on Artificial Intelligence, Montpellier, Aug, 2012, pp. 27–31.
[127] H. Prade, G. Richard, From analogical proportion to logical proportions: A survey, in: H. Prade, G. Richard (Eds.), Computational Approaches to Analogical Reasoning: Current Trends, Vol. 548 of Studies in Computational Intelligence, Springer Berlin Heidelberg, 2014, pp. 217–244.
[128] M. Siebers, U. Schmid, Semi-analytic natural number series induction, in: KI 2012: Advances in Artificial Intelligence, Springer, 2012, pp. 249–252.
[129] C. Schenck, Intelligence tests for robots: Solving perceptual reasoning tasks with a humanoid robot, Master's thesis, Iowa State University (2013).
[130] H. Prade, G. Richard, Picking the one that does not fit - A matter of logical proportions, in: Proceedings of the 8th conference of the European Society for Fuzzy Logic and Technology, EUSFLAT-13, September 11-13, 2013, 2013.
[131] C. Strannegård, M. Amirghasemi, S. Ulfsbücker, An anthropomorphic method for number sequence problems, Cognitive Systems Research 2223 (2013) 27–34.
[132] C. Strannegård, A. Nizamani, A. Sjöberg, F. Engström, Bounded Kolmogorov complexity based on cognitive models, in: K. U. Kühnberger, S. Rudolph, P. Wang (Eds.), Artificial General Intelligence, Vol. 7999 of Lecture Notes in Computer Science, Springer Berlin Heidelberg, 2013, pp. 130–139.
[133] C. Strannegård, S. Cirillo, V. Ström, An anthropomorphic method for progressive matrix problems, Cognitive Systems Research 2223 (0) (2013) 35–46.
[134] S. Ohlsson, R. H. Sloan, G. Turán, A. Urasky, Verbal IQ of a four-year old achieved by an AI system, in: COMMONSENSE

2013, 11th International Symposium on Logical Formalizations of Commonsense Reasoning, 2013, p. 6.
[135] J. Hofmann, E. Kitzelmann, U. Schmid, Applying inductive program synthesis to induction of number series a case study with IGOR2, in: KI 2014: Advances in Artificial Intelligence, Springer, 2014, pp. 25–36.
[136] J. Burghardt, E-generalization using grammars, Artificial Intelligence 165 (1) (2005) 1–35.
[137] C. Schenck, A. Stoytchev, The object pairing and matching task: Toward montessori tests for robots, in: Ugur, E., Nagai, Y., Oztop, E., and Asada, M. (Eds) Proceedings of Humanoids 2012 Workshop on Developmental Robotics: Can developmental robotics yield human-like cognitive abilities?, 2012, pp. 7–13.
[138] C. Schenck, J. Sinapov, A. Stoytchev, Which object comes next? Grounded order completion by a humanoid robot, Cybernetics and Information Technologies 12 (3) (2012) 5–16.
[139] A. Sjberg, S. Sjberg, K. Forssn, Predicting Job Performance, Assession Intenational, Stockholm, 2006.
[140] R. Amthauer, B. Brocke, D. Liepmann, A. Beauducel, Struktur-Test 2000 R, 2001.
[141] P. Singh, The public acquisition of commonsense knowledge, Proceedings of AAAI Spring Symposium: Acquiring (and Using) Linguistic (and World) Knowledge for Information Access (2001).
[142] P. Flener, U. Schmid, An introduction to inductive programming, Artificial Intelligence Review 29 (1) (2008) 45–62.
[143] S. Gulwani, J. Hernández-Orallo, E. Kitzelmann, S. H. Muggleton, U. Schmid, B. Zorn, Inductive programming meets the real world, communications of the ACM, to appear (2015).
[144] U. Schmid, E. Kitzelmann, Inductive rule learning on the knowledge level, Cognitive Systems Research 12 (3) (2011) 237–248.
[145] T. G. Holzman, J. W. Pellegrino, R. Glaser, Cognitive dimensions of numerical rule induction., Journal of Educational Psychology 74 (3) (1982) 360–373.
[146] A. Newell, Productions systems : models of control structures, Computer Science Department. Paper 2034, http://repository.cmu.edu/compsci/2034l.
[147] M. A. Just, P. A. Carpenter, The psychology of reading and language comprehension., Allyn & Bacon, 1987.
[148] J. R. Anderson, ACT: A simple theory of complex cognition, American Psychologist 51 (4) (1996) 355–365.
[149] D. Gentner, Structure-mapping: A theoretical framework for analogy, Cognitive Science 7 (2) (1983) 155–170.
[150] S. Kuehne, K. Forbus, D. Gentner, B. Quinn, Seql: Category learning as progressive abstraction using structure mapping, in: Proceedings of the 22nd Annual Meeting of the Cognitive Science Society, 2000, pp. 770–775.
[151] D. Gentner, J. Loewenstein, Relational language and relational thought, Language, literacy, and cognitive development: The development and consequences of symbolic communication (2002) 87–120.
[152] K. D. Forbus, D. Gentner, A. B. Markman, R. W. Ferguson, Analogy just looks like high level perception: Why a domain-general approach to analogical mapping is right, JETAI 10 (1998) 231–257.
[153] S. Büttcher, C. L. Clarke, Efficiency vs. effectiveness in terabyte-scale information retrieval., in: TREC, 2005.
[154] M. G. Bellemare, Y. Naddaf, J. Veness, M. Bowling, The arcade learning environment, J. Artificial Intelligence Res 47 (2012) 253–279.
[155] M. Ghallab, C. Knoblock, D. Wilkins, A. Barrett, D. Christianson, M. Friedman, C. Kwok, K. Golden, S. Penberthy, D. E. Smith, et al., Pddl-the planning domain definition language.
[156] J. You, Beyond the turing test, Science 347 (6218) (2015) 116–116.
[157] R. M. French, Moving beyond the turing test, Communications of the ACM 55 (12) (2012) 74–77.
[158] R. F. Amthauer, B. Brocke, D. Liepmann, A. Beauducel, Intelligenz-Struktur-Test 2000: IST 2000, Hogrefe & Huber, 1999.
[159] N. J. A. Sloane, The on-line encyclopedia of integer sequences, Notices of the AMS 50 (8) (2003) 912–915.
[160] L. L. Thurstone, T. G. Thurstone, American Council on Education for college freshmen: Manual of Instructions, Cooperative Test Division, Educational Testing Service, 1947.
[161] T. Zentall, D. Hogan, J. Holder, Comparison of two oddity tasks with pigeons, Learning and Motivation 5 (1) (1974) 106–117.
[162] T. R. Zentall, D. E. Hogan, C. A. Edwards, Oddity learning in the pigeon: Effect of negative instances, correction, and number of incorrect alternatives, Animal Learning & Behavior 8 (4) (1980) 621–629.
[163] S. Dehaene, V. Izard, P. Pica, E. Spelke, Core Knowledge of Geometry in an Amazonian Indigene Group, Science 311 (5759) (2006) 381–384.
[164] D. R. Hofstadter, Gödel, Escher, Bach: An Eternal Golden Braid, Basic Books, Inc., New York, NY, USA, 1979.
[165] A. Linhares, A glimpse at the metaphysics of Bongard problems, Artificial Intelligence 121 (1) (2000) 251–270.
[166] L. De Raedt, W. Van Laer, Inductive constraint logic, in: Algorithmic Learning Theory, Springer, 1995, pp. 80–94.
[167] P. A. Flach, C. Giraud-Carrier, J. W. Lloyd, Strongly typed inductive concept learning, in: Inductive Logic Programming, Springer, 1998, pp. 185–194.
[168] K. Kersting, An inductive logic programming approach to statistical relational learning, in: Proceedings of the 2005 conference on An Inductive Logic Programming Approach to Statistical Relational Learning, IOS Press, 2005, pp. 1–228.
[169] K. Kersting, L. De Raedt, Basic principles of learning bayesian logic programs, in: Probabilistic Inductive Logic Programming, Springer, 2008, pp. 189–221.
[170] D. R. Hofstadter, Fluid concepts and creative analogies: Computer models of the fundamental mechanisms of thought, Basic Books, 2008.
[171] M. Montessori, The Montessori method, Frederick A. Stokes Co., New York City, USA, 1912.