Neural Computing and Applications (2023) 35:17599–17618
https://doi.org/10.1007/s00521-023-08631-9

ORIGINAL ARTICLE

Best-response planning for urban fleet coordination

Pasqual Martí[1] · Jaume Jordán[1] · Vicente Julian[1]

Received: 21 December 2022 / Accepted: 24 April 2023 / Published online: 16 May 2023

Abstract
The modeling of fleet vehicles as self-interested agents brings a realistic perspective to open fleet transportation research. This feature allows us to model the fleet operation from a non-cooperative point of view. In this work, we study parcel delivery in a city with limited resources (roads and charging stations). We designed and implemented a system composed of a multi-agent planner and a game-theoretic coordination algorithm: a Best-Response Fleet Planner. The system allows for the self-organization of the transportation system by coordinating a fleet of self-interested electric vehicles. The system's operation is optimized together with resource usage while preserving the agents' private interests, allowing each agent to plan its actions. The results show that our system has higher scalability than similar approaches, allowing it to function for a considerable number of agents in settings that feature congestion and conflicts. Additionally, overall solution quality is improved compared to other coordination systems, reducing congestion and avoiding unnecessary waiting times.

Keywords Intelligent agents · Transportation · Self-interest · Best response · Nash equilibrium · Coordination

1 Introduction

A city can be seen as a non-cooperative or competitive scenario. Many of its resources, like road networks or petrol stations, may get congested if too many users want to use them simultaneously. As users (generally drivers) act selfishly and uninformedly, resource management tends to be poor. This translates into traffic congestion, higher waiting times to refuel, and, in general, more air pollution and less quality of service (for transportation service users).

Optimization techniques can improve systems by identifying and minimizing inefficiencies, reducing waste, and maximizing output. These techniques use mathematical and computational models to analyze data and identify areas of improvement [1, 2], such as minimizing production costs, reducing delivery times, or improving quality. In transportation systems, for instance, optimization techniques can be used to optimize routing, minimize fuel consumption, and reduce transportation time, resulting in improved delivery performance and reduced costs. One of the main limitations, however, when it comes to traffic optimization, is gathering the necessary information to coordinate every user's actions and make intelligent decisions. Connecting all services and infrastructure would be beneficial for such a complex task. Smart city technology would allow data collection and exchange among these services. These data could be used in research on improving urban traffic and applied to develop solutions.

The aforementioned technologies can be applied to develop an intelligent, self-organizing transportation service. This work focuses on *open* delivery fleets, dynamic fleets whose number of vehicles can increase or decrease according to the demand. In contrast with traditional fleets, the drivers are autonomous; they choose the passenger or delivery to serve and obtain a benefit accordingly. Although drivers belong to the same fleet, they act according to their benefits. Therefore, when reproducing such a fleet, we must ensure that the agents make their own decisions and are not coordinated by a centralized entity. The transports that compose the delivery service must be able to self-organize themselves according to their private

Pasqual Martí, Jaume Jordán and Vicente Julian have contributed equally to this work.

✉ Pasqual Martí
pasmargi@vrain.upv.es

Jaume Jordán
jjordan@dsic.upv.es

Vicente Julian
vjulian@upv.es

[1] Valencian Research Institute for Artificial Intelligence (VRAIN), Universitat Politècnica de València, Camino de Vera s/n, 46022 Valencia, Spain

goals, but taking into account they all coexist in the same scenario, and thus it is in their best interest not to cause congestion. Considering these features, agent-based modeling (ABM) and game theory are applied to reproduce this type of fleet. ABM [3] is a computational modeling technique used to simulate complex systems consisting of multiple interacting agents. In ABM, each agent in the system is programmed with a set of rules that govern their behavior and decision-making processes. Game theory [4], on the other hand, is defined as the study of mathematical models of strategic interaction among *rational decision-makers*, i.e., agents who make decisions based on personal benefit. Game theory provides the tools to coordinate the fleet's autonomous transports taking into account the actions of each other.

Our work presents a practical application to coordinating self-interested vehicles of a fleet. In addition, this coordination considers the resources of the urban area where the fleet operates to optimize its use and avoid congestion. To this end, we have, on the one hand, designed and implemented an ad hoc optimal planning algorithm that enables each fleet's individual vehicles to plan their actions according to their interest. Moreover, the planner considers every other vehicle's plans to obtain the optimal plan with respect to every other agent's plan. This, in turn, implies the avoidance of congestion and conflict resolution. On the other hand, we have implemented a game-theoretic coordination algorithm (best-response dynamics) which converges to an equilibrium: A collection of agent plans from which no vehicle is incentivized to change. The fleet's operation that describes the equilibrium ensures the vehicles perform their services to maximize their benefits, implying that their private interests are preserved.

The main differences between our approach and other fleet coordination techniques are the following. On the one hand, decentralized coordination is provided. Generally, fleets are coordinated by a central entity that decides the actions of each vehicle. In contrast, our fleet vehicles have the autonomy to make their decisions. In addition, vehicle coordination may occur even if a member fails to communicate, thus being appropriate to model an open fleet. Finally, our approach enables each vehicle to keep its goals private, which is useful when coordinating agents in a non-cooperative scenario. On the other hand, using game theory techniques allows us to define the use of the city's resources as a congestion game which, in turn, shows the agents (vehicles) that it is in their best interest to make better use of them. With these features, we achieve the optimization of the whole system together with the preservation of the agent's autonomy, which is generally lost in other coordination approaches.

Our research explores the limitations of the proposed system through extensive experimentation. We show the extent to which the ad hoc planner can return optimal plans in a reasonable time according to problem complexity and the number of agents. The results indicate that the system overcomes similar approaches in terms of computation power, taking into account the advantage of having an ad hoc planner. Our system proves the viability of simulating realistic scenarios, with a significant number of agents, in a game-theoretic environment. Finally, we assess the quality of the returned solutions, which are better than those obtained by greedy coordination.

The remainder of the paper is organized as follows. Section 2 reviews related work. Then, in Sect. 3, we present an overview of the entire proposed system. Next, Sect. 4 specifies the urban mobility planning domain that reflects the problem to be solved. Section 5 describes in detail the developed ad hoc planner. Following, the best response fleet planning (BRFP) process with which the whole game is developed to reach an equilibrium solution is explained in Sect. 6. The experimental results of the proposed work are presented in Sect. 7. Section 8 discusses urban transportation challenges, how our system applies to other problems, and its limitations. Finally, Sect. 9 draws the conclusions of this work and presents possible future research directions.

2 Related work

The proposed system is related to three fields within artificial intelligence: Multi-agent systems, automated planning, and game theory. In this case, techniques of each branch are applied to an urban mobility domain intending to optimize the operation of delivery fleets. Multi-agent systems and their simulation allow us to reproduce the behavior of human drivers in a software world and study their actions and any synergies that may arise. Moreover, automated planning algorithms ensure that given an agent's current knowledge, they are able to compute their best course of action, considering each possible path at each computational step. Finally, knowing they are part of a competitive environment and assuming rationality on all other participants, game theory gives the basis to reach agreements and equilibria among the actions of all agents in the scenario.

2.1 Multi-agent simulation

Multi-agent simulation (MAS) is a computational modeling technique that allows the simulation of complex systems composed of multiple interacting agents. Applied to urban fleet management, MAS is used to model and

analyze the behavior of a fleet of vehicles in a city, considering various factors such as transportation demand, traffic conditions, and resource availability. MAS can help improve the efficiency and sustainability of the transportation system by allowing fleet managers to test different scenarios and strategies in a safe and controlled environment before implementing them in the real world. MAS has been widely used to model and simulate vehicle fleets [5–7]. An urban mobility domain must define many different interactions among the various elements of the scenarios. MAS help achieve that as we can represent each element through an agent (vehicles, pedestrians, charging stations, etc.) and define appropriate behaviors for them. In [8], authors presented a MAS-based simulator specialized in the representation of urban fleets of different kinds. Later, in [9], the aforementioned simulator was extended to include new types of fleets, such as carsharing. Using simulators enables us to explore the effect of different coordination paradigms on the operation of a fleet without having to implement changes in the real world.

In recent years, new agent-based simulators have appeared that facilitate the development of different strategies for fleet management in the urban environment. One of the tools is SUMO [10], an open-source traffic simulator that can be used for route choice, communication between agents and infrastructure, traffic management, and autonomous driving. SUMO uses an origin/destination matrix to assign movement between city zones. Another tool is MATSim [11], a framework for demand modeling and traffic flow simulations. SIMmobility [12] is another simulation tool that focuses on mobility demand impact prediction for smart shipment services. Finally, commercial tools like VISSIM [13] offer an array of technologies to address multiple mobility and transportation problems.

2.2 Fleet coordination and game theory

Regarding vehicle fleet coordination, the degree of freedom given to each vehicle is crucial. Such a degree indicates how much the self-interest of the vehicle (or its driver) can influence its actions. Authors assess this topic in [14], where a taxonomy of autonomous vehicle coordination problems is presented. According to the degree of freedom given to each vehicle, the coordination approaches vary from fully *centralized*, where an external entity imposes actions on every fleet vehicle, to fully *emergent*, where its self-interest guides all of the vehicle's actions.

There is no direct involvement of the agents (vehicles, drivers) in any coordination protocol in emergent coordination approaches. Agents behave according to their goals and aim to maximize their actions' utility. These features give rise to the use of game-theoretic techniques, where each agent assumes the rationality of the others and determines its actions based on the information it knows or can guess about other participants.

For instance, the work in [15] presents a distributed approach for coordinating the charging of a large fleet of plug-in electric taxis in a city, aiming to reduce charging costs, improve charging station utilization, and balance charging requests for the power grid. The approach involves a two-stage decision process with a thresholding method for charging time slot selection and a game-theoretical approach for charging station selection, as validated by extensive numerical simulations. Similarly, the paper [16] discusses the problem of fleet configuration for unmanned vehicles, focusing on optimizing the fleet for minimum costs. The proposed approach involves transforming the fleet configuration activity into an optimization problem using game-theoretic techniques, with the aim of achieving interoperability among different organizations involved in fleet provision through distributed and decentralized planning.

Therefore, emergent coordination is generally applied to fleets composed of independent vehicles, in other words, non-cooperative fleets. In these fleets, like those of Uber, Lyft, or Glovo, each driver obtains benefits thanks to his/her work. Even if they belong to the same fleet, the different drivers do not tend to cooperate, although that does not imply that they are competitive either. For our work, we will assume non-cooperative (agents only care about maximizing their utility) and non-strictly competitive (agents do not actively look to reduce the utility of other agents) self-interested agents.

2.3 Automated planning

Self-interested agents must be able to plan their actions according to their private benefits. Because of that we introduced automated planning to the system. A planner generally looks for a feasible, somewhat optimized solution to a problem. This applied to a fleet would imply centralized coordination, as the planner would define each vehicle's actions. Nevertheless, the planning goal can be distributed into different tasks, allowing each agent to plan, by itself, how to carry out their task. When planning is applied to MAS, we perform multi-agent planning (MAP) [17].

In recent years, there has been significant research on cooperative MAP, where agents join their efforts to achieve a common goal. Cooperative MAP is used to solve tasks that cannot be performed by a single agent or are better solved when several agents work together [18]. In some cases, agents with different abilities must cooperate to solve a planning task [19]. However, we focus on types of MAP where game theoretic techniques may be applied. These are the coalitional MAP, where establishing

alliances benefits groups of agents [20]; adversarial MAP, which features self-interested agents with opposed goals and, consequently, takes place in strictly competitive scenarios; and finally, non-cooperative MAP, in which agents are not strictly competitive; and therefore, they are prone to follow a collaborative strategy and resolve conflicts.

The coordination of self-interested agents in non-cooperative settings is generally performed through a game. In this game, the agent strategies are their plans, the actions they intend to do. These plans will be adapted to other agents' plans to avoid conflicts. Finally, an equilibrium is obtained: a union of agent plans (joint plan) that ensures no agent will deviate from it. The equilibrium, in addition, must solve the goal of the MAP task. The works in [21, 22] introduce FENOCOP, an approach to solving non-cooperative planning problems. In this approach, agents have a limited set of plans. The final joint plan is built in two phases or games: first, a Nash Equilibrium [23] is obtained from the many combinations of agent plans. Then, a scheduling process delays specific actions to obtain an executable outcome, avoiding conflicts. This approach can obtain Pareto optimal and fair equilibria, an extra-quality measure for the solutions. However, the methods lack scalability because of their exponential complexity.

Another work, presented in [24], describes the so-called *Better-response Planning Strategy (BRPS)*, a game-theoretic algorithm to solve *congestion games* [25]. In congestion games, the scenario features a series of resources that agents will use. When too many agents use a resource simultaneously, it gets congested, and its cost increases. Congestion games can significantly represent urban mobility domains, as these contain many resources (roads, charging stations) in which we wish to avoid congestion. In a best-response process, an equilibrium is reached through an iterative process in which the participant agents propose, in turn, a plan which is a *best-response* to every other agent's plan. This process finally converges when no agent is incentivized to change its plan. The Better-response Planning Strategy of [24] allows agents to propose not their best plan but simply a plan that improves the utility of its previously proposed plan. This avoids the need for optimal planning that a best-response process requires, which is computationally more costly than satisficing planning in practice [26].

Our approach is inspired by the Better-response Planning Strategy but uses a best-response process, as we can perform optimal planning in a fast manner thanks to the design and implementation of our ad hoc planner. We apply these methods to coordinate the operation of an open delivery vehicle fleet, ensuring optimal delivery routes and resource congestion avoidance.

3 System overview

The work described in this paper is motivated by the research on rational, self-interested agents. An agent with those features has its private objectives, which, in practice, translates to its unique utility function. Our goal is to explore the coordination of urban fleets composed of self-interested agents, particularly electric delivery vehicles. Such vehicles may belong to a fleet, thus serving customers' delivery requests and getting compensated by it. Introducing delivery vehicles in a city with limited resources creates a competitive scenario where agents compete to deliver their parcels as soon as possible. However, we must ensure that the aforementioned scenario (delivery service) is solvable, avoiding the conflicts that generally arise between agents. For that, we model the operation of the agents as a MAP task, precisely a non-cooperative MAP task in a non-strictly competitive setting, one in which agents do not create coalitions to solve the global goal of the task but instead, the task is solved by coordinating how agents solve their goals. In this way, we aim to obtain a functioning of the agents which preserves their individuality, not imposing any action but allowing them to determine their actions by themselves and simultaneously avoid conflicts with the rest of the agents of the scenario.

In a delivery fleet with the aforementioned modeling, the global goal would be to complete all delivery tasks, thus solving the scenario. On the other hand, the agents that compose it will aim to maximize their utility, dropping off the parcels as soon as possible while following the most efficient route: the one that involves less traveled distance and power consumption. To avoid conflicts, the actions of each agent are decided by a game-theoretic process: A Best-Response Fleet Planning (BRFP) process (Sect. 6). The process begins by creating a congestion game equivalent to the scenario to solve. For this game, the moves or strategies of the players will be their actions, i.e., an ordered list of the actions they will do. This list of actions is a plan, being the plan's goal to get the highest possible utility out of the actions. Therefore, transport (delivery) agents act as players whose strategies are plans built according to their interests.

To compute such individual plans, we developed an ad hoc planner (Sect. 5). It is ad hoc as it is designed to solve problems set in the Urban Mobility Planning Scenario (Sect. 4). We take advantage of the domain characteristics to speed up the search. All in all, given a scenario and a transport agent in that scenario, the ad hoc planner builds the optimal plan for such an agent, taking into account both the state of the scenario and the plans of every other agent,

always obtaining a plan that is the best response to all other agents' plans.

Once the congestion game is established, it is developed by the BRFP process, in which the agents propose different strategies (plans), always in the best response, improving each turn (if possible) their previously proposed plans to (1) avoid conflicts with other agents and (2) minimize their costs. The process converges to an executable solution (joint plan), a pure-strategy Nash equilibrium (PNE) [27], guaranteeing that no agent will deviate from it (change its strategy). In this way, the BRFP obtains a solution that indirectly achieves the global goals of the fleet (all transport agent's delivery tasks are served) by capitalizing on the agents' own incentive to maximize its benefits, which it does by completing the tasks following the optimal route and avoiding congestion. As the global goals are satisfied, the obtained joint plan also coordinates the fleet's operation, which could be simulated.

The diagram in Fig. 1 shows the operation of the BRFP. It can be seen how the agents use the planner to propose their best strategy. In one iteration, every agent has to propose a new best plan (if the previous one was not in best-response already), updating the joint plan. If, after a whole iteration, no agent has changed their plan, the process has converged, and the joint plan, the union of every agent's individual plan, is returned. The joint plan describes a solution to the congestion game, which is an equilibrium and, in addition, ensures the lack of conflicts among agents of the scenario. In the following sections, we describe the planner and BRFP algorithm and the urban mobility planning domain in which the executions occur.

4 Urban mobility planning scenario

The planning problems of this work are set in an urban mobility scenario that models a real-world smart-city urban area. The scenario contains three types of elements: *parcels*, with an associated initial position and final destination; electric delivery vehicles or *transport agents*, with an initial position and a current travel capacity (electric power), expressed in kilometers; and finally, electric *charging stations*, which have a certain number of charging poles for the transports to recharge their batteries and an electric power which determines the speed at which agents charge in them.

We are modeling a delivery service with non-fixed pick-up or drop-off locations. Transport agents have two basic behaviors: complete a delivery task, which involves moving to the parcel location, picking it up, driving to its destination, and recharging their batteries by driving to a charging station. Transports can carry a single (1) parcel at a time. Consequently, our system considers the following four types of actions:

1. PICK-UP: Move to a parcel's position and pick it up.
2. MOVE-TO-DEST: Move to the carried parcel's destination and drop it off.
3. MOVE-TO-STATION: Move to a charging station and wait for the charge.
4. CHARGE: Begin charging until the travel capacity is full.

Actions 1 and 2 constitute a *delivery service* while actions 3 and 4 constitute a *charging service*. These services must be executed without interruption; consequently, during the construction of the individual plan of a transport agent, actions of types 1 and 2 will always appear consecutively. The same applies to actions of types 3 and 4. A scenario will be solved once the delivery tasks assigned to every transport are completed. In practice, a transport agent with a preassigned number of delivery tasks will aim to follow the optimal order to complete them so that delivery time and power expenses are minimized.

We have chosen to apply this system to a delivery fleet. Nevertheless, our approach can be used for other applications in the field of urban mobility, such as the coordination of fully autonomous vehicles. In addition, it could be adapted to manage the operation of other distributed systems in which the self-interest of each part must be considered.

4.1 Transport agent's utility

Transport agents are modeled as rational, self-interested agents that act according to their private interests. Such an interest is to maximize their utility. Transports must complete all their delivery tasks regardless of the cost involved. Because of this, an agent's utility is equivalent to the negative value of its costs. Thus, transports are motivated to complete their tasks minimizing their total cost.

The costs arise from two main sources: customer waiting time or *waiting cost* and resource congestion. Regarding the former, transport agents have their costs incremented by a fixed amount every time instant a delivery task assigned to them remains uncompleted. Concerning the latter, roads and charging stations represent resources whose use incurs costs. These resources may get congested if too many agents use them simultaneously (in overlapping time intervals). If a congested resource is used, the cost of such usage will be higher than expected. Resource congestion costs are identified as *road_cong* for road network congestion and *power_cong* for electric power network congestion in Eq. 1.

The exact formulas that describe congestion costs can be configured by the user. Generally, a resource will have a

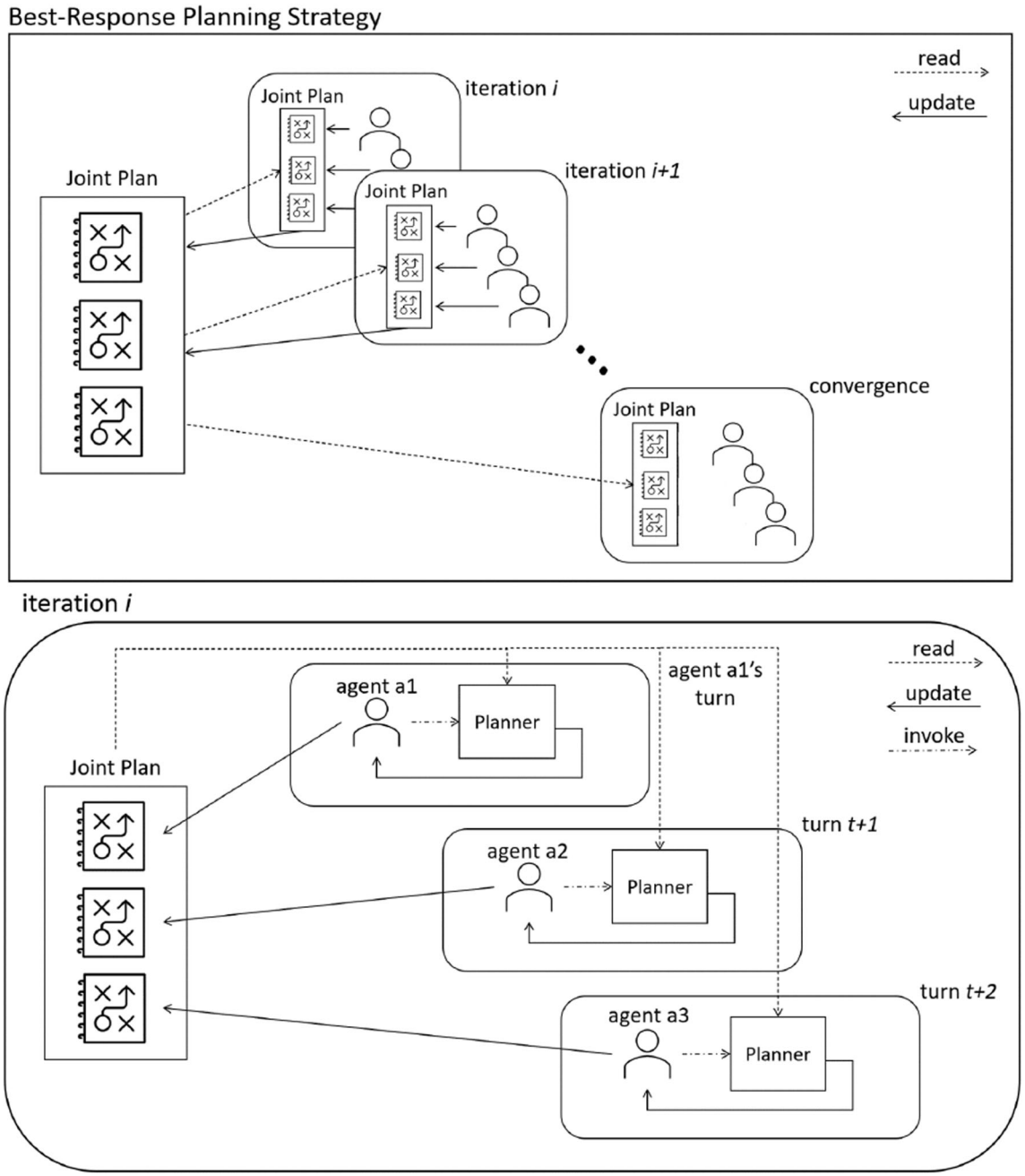

Fig. 1 Graphic of the functioning of the best-response fleet planner. The image at the top shows how the Joint plan is updated each iteration: All agents compute their plans and update a copy of the joint plan sequentially. The process converges when no agent changes their plan. The image at the bottom shows an iteration in detail; each agent invokes the ad hoc planner during its turn to propose their best plan, updating the Joint plan if necessary. The planner considers all other agents' plans by reading the joint plan each time it is invoked

certain *resource bound* defining the number of simultaneous uses it can withstand. Once that bound is surpassed, the resource's cost increases proportionally to how many agents use it. Our modeling defines two resource bounds: *power bound*, for power network congestion, and *road bound*, for road network congestion. Let us define a bound, for instance, of 0.5. For power network congestion, that would indicate the network gets congested once 50% of its power is drawn at a time. In contrast, road network congestion would indicate a road is congested once 50% of its capacity is used at a time.

The *total cost* of a transport agent is used to evaluate its plan. It is computed as the addition of its *waiting cost* and the *costs derived from resource congestion*, if any. In addition, every type of equation cost is pondered by a *weight*: w_w, w_r, and w_p for waiting, road congestion, and

power congestion costs, respectively. The utility of an agent, described in Eq. 1, is equal to $-(\textit{total cost})$, which, in time, is the utility associated with its plan.

$$\begin{aligned} U &= -(\text{total\_cost}) \\ &= -(\text{waiting\_cost} \cdot w_w + \text{road\_cong} \cdot w_r \\ &\quad + \text{power\_cong} \cdot w_p) \end{aligned} \tag{1}$$

With this modeling of costs, we achieve transport agents interested in completing their assigned delivery tasks in an order that involves less delivery time, fewer power expenses, and avoids congestion when it is profitable.

4.2 Conflicts

A shared scenario, populated by self-interested agents and with a limited number of resources, may give rise to conflicts among agent plans. A conflict makes the involved agents' plans unfeasible. Our domain presents *charging station conflicts* when two or more agents plan to recharge in the same charging station during overlapping periods, and *the station does not have enough available charging poles to serve all transports simultaneously*. In this situation, the agent that arrives at the station the soonest has its charging spot ensured. Therefore, the conflict resolution falls to the rest of the agents, who will have to choose between waiting in line at the station for their turn to charge or charging at another station.

Conflict resolution always involves an increase in the agent's delivery time and, therefore, costs. However, there is no way in which conflicts could be permitted. To ensure agents avoid and/or resolve conflicts, any agent whose plan is in conflict will be penalized with a great increment of its costs.

4.3 Individual plans and the joint plan

Agent actions are reflected in plans. A plan consists of a list of entries arranged in ascending order according to their start time. Every plan entry corresponds to one action and presents it with its attributes and initial and end time in seconds. An example of a joint plan (the union of agents' individual plans) in our domain is shown in Table 1.

We must differentiate between two types of plans: *individual* or agent plans, and the *joint* plan. Individual plans are the ones planned and executed by a single agent. The joint plan, in contrast, is the union of every individual plan. Individual plans are computed guiding the planning only by the agent's private interests, i.e., minimizing its costs. However, when part of the joint plan, the individual plan may have actions in conflict. All the conflicts must be resolved for a joint plan to be executable.

Table 1 Visual representation of a joint plan

Init time	Actions	End time
0.00	(Agent_A, MOVE-TO-STATION, station1)	4.62
0.00	(Agent_B, PICK-UP, parcel1)	9.81
4.62	(Agent_A, CHARGE, station1)	9.97
8.52	(Agent_C, PICK-UP, parcel3)	17.51
9.81	(Agent_B, MOVE-TO-DEST, parcel1)	16.07
9.97	(Agent_A, PICK-UP, parcel2)	19.01
17.51	(Agent_C, MOVE-TO-DEST, parcel3)	25.41

Each row corresponds to a plan entry. On the left column, the initial time instant of the action is presented in seconds. In the middle one, the action with all its attributes. On the right column, the time instant in which the action finishes is indicated, also in seconds

5 Ad hoc planner

Considering the characteristics of the urban mobility domain defined above, we decided to implement an ad hoc planner. Agents invoke an instance of the planner to obtain optimal individual plans that solve the problem scenario while ensuring their actions avoid conflicts. The current world state is represented by the transport agent's knowledge at the moment of planning. This includes its current position and travel capacity as well as its uncompleted tasks. In addition, because of the associated best-response process, the agent will have complete information on the plan in the joint plan of every other agent in the scenario. The planner uses this to avoid conflicts.

In this section, we describe our planner's elements, its search tree's components, and the procedure used to build and explore it.

5.1 Best-response planning

Our planner is meant to be used by the agents participating in a best-response process to obtain and propose their best strategy, that is, the best possible plan with respect to every other agent's plan. Because of that, our planner performs optimal planning, which is reasonable given the restrictions of our domain. Hence, the individual plan returned from the planning process is always the best response to the plan of every other transport agent. When a plan is returned, the agent proposes it and is added to the joint plan, updating it.

Each planner instance has its own *station usage table*, a data structure containing, for every charging station in the scenario, the agents that planned to use it together with the time instants they arrive at it and start and finish the charge. An example of such a structure is shown in Table 2. This data structure is used to detect charging conflicts at the end

Table 2 Station usage table example

Station	Agent	Arrival	Charge start	Charge end
Station1	Agent1	17.38	17.38	21.38
	Agent3	19.56	21.38	26.38
Station2				
Station3	Agent2	5.54	5.54	15.54

It reflects the information regarding charges that appear in the joint plan. For every station, it contains a list with the agents charging in it and the time instants in which they arrive at the station, begin and finish charging

of a best-response turn and makes agents avoid them in their subsequent planning processes.

The best-response process will eventually converge to a pure-strategy Nash equilibrium (PNE) [27]. Once the best-response process converges, the joint plan is guaranteed to be a PNE, conflictless and, thus, executable.

5.2 Partial plan search tree and exploration algorithm

Our planner searches for the optimal plan by building and expanding a search tree of partial plans following an A* algorithm. During this process, the most promising nodes are expanded depending on both the utility of the partial plan developed so far and the potential (optimistic) utility that the rest of the plan could develop from that node. The nodes of the tree contain partial plans. Nodes expand and generate children, which inherit their plan and add new actions.

Nodes can be of two types: *parcel* or *charge* nodes. A *parcel* node is created for each uncompleted delivery task of the agent when expanding the parent node. *Charge* nodes are created whenever the agent's travel capacity is not maxed out in the parent node. One charging node is created per reachable charging station in the scenario. The information of each station is accessed through the *station usage table* (Sect. 5.1). With it, the planner sets the time instants at which the agent will reach the station and start charging, according to the available poles and the charge duration.

By creating parcel and charge-type children, plans are built adding two actions to the parent node's partial plan in each step. Using this method, we only consider the addition of necessary and feasible actions every time. Consequently, we are avoiding search tree ramifications that would eventually be discarded either because of conflicts or a low utility value.

5.3 Plan evaluation

The value of a plan is tied to the utility it reports to the transport agent that executes it and, therefore equivalent to the -(*total cost*) described in Sect. 4.1. Globally, a joint plan is not evaluated, as only its feasibility is relevant. The planner (the individual instance of an agent planner) evaluates *partial* plans during the plan-building process and *complete* plans to return the best solution. Any congestion in which the agent might be involved is considered during the plan evaluation, increasing its cost accordingly.

To evaluate a partial plan n Eq. 2 is used, where $g(n)$ is its cost, $h(n)$ is an optimistic calculus of the expected cost that completing every non-completed goal would yield, and $h^*(n)$ is the optimal cost to reach all non-completed goals from node n.

$$f(n) = g(n) + h(n), \quad h(n) \leq h^*(n) \tag{2}$$

The heuristic function h is a relaxation of the problem constraints. It assumes that from a particular partial plan, the remaining delivery tasks can be completed as efficiently as possible without the need to charge. This is done by computing the *best permutation*; the order in which to attend the remaining delivery tasks that minimize costs. The node's heuristic value will estimate the minimum cost of completing the rest of a plan. The heuristic estimate would only match the actual cost of a plan if such a plan was completed without charge actions and the agent was not involved in any congestion.

The value of a complete plan n is equal to $g(n)$. When an agent proposes its plan, it gets integrated into the joint plan. Such a plan may present conflicts as a part of a joint plan. If the integration does not cause any conflicts, the plan's value will be the same as it had when proposed. However, when a plan causes any conflict, its cost is highly increased, forcing the planner to change it in the following planning turn.

5.4 Search tree pruning

Planning is a computationally hard task. Our planner implements mechanisms that aid in speeding up the plan search process and lower memory consumption.

Best solution prune. Once the first solution node is found, its plan is extracted, evaluated, and its utility saved as the *best solution value* found so far. This value is updated as new solution nodes are reached. If the value of a child node is worse than the best solution value, it will be discarded. The partial plan of an open node with an f-value below the best solution value has no potential to evolve into a better solution, so the planner can avoid wasting computational power expanding it.

Previous plan utility bound. When an instance of the planner is created, the invoking agent's previous plan (found in the previous best-response iteration) can be passed to it. If there is a previous plan and the utility it reports is higher than 0, such a value will define a lower bound value for the planning process. When a node is evaluated, it will be discarded if its value is below the lower bound. In this way, solution nodes that contain worse plans (or partial plans with no potential to improve) than the previously obtained ones are not considered, speeding up the process. This technique is only applied if the previous plan of an agent is not causing any conflicts.

6 Best-response planning

This section explains how the best-response planning process is developed. First, it describes the iterative process in which the agents propose their best plan given the plans of the other agents. Then, the way to resolve conflicts that may arise between agents during this process is explained. Finally, we explain how to build an initial joint plan with a greedy algorithm.

6.1 BRFP process

The BRFP is a process in which an agent a iteratively looks for a plan π^a which is in best response to every other plan in the joint plan Π. At the beginning of the process, an arbitrary order is defined among all participant agents, and an empty joint plan $\Pi = \emptyset$ is created. Alternatively, the process can begin from an initial joint plan $\Pi = \langle \pi^{1'}, \pi^{2'}, \ldots, \pi^{n'} \rangle$, where $\pi^{a'}$ is a non-optimal plan created following a greedy strategy (see Sect. 6.2). This provides the agents with a lower utility bound (see Sect. 5.4), speeding up the planning during the first BRFP iteration.

During the process execution, agents must best respond in each iteration. A planning process is used for that, which can return either a new plan, the same plan as the previous iteration, or nothing if there is no solution. If the same plan is returned, the agent will preserve it since it means that it is still in the best response to every other plan. When no agent modifies its plan in a complete iteration, the BRFP has converged to a joint plan that is a PNE.

From an agent's perspective, the BRFP works as follows:

- An arbitrary order between agents is established. Following such order, an initial joint plan is built incrementally using the individual planner of the agent or following a greedy strategy: $\Pi = \langle \emptyset, \ldots, \emptyset \rangle$, $\Pi = \langle \pi^1, \emptyset, \ldots, \emptyset \rangle$, $\Pi = \langle \pi^1, \pi^2, \ldots, \emptyset \rangle$, $\ldots$, $\Pi = \langle \pi^1, \pi^2, \ldots, \pi^n \rangle$.
- In one iteration i, agent a executes these steps:
  1. Analyze the utility of its current plan π^a_{i-1} in the joint plan, defining a lower bound for the following search.
  2. Start a planning process to search for a new plan π^a_i which is in best response to every plan in the joint plan.
  3. If a new plan is returned, update the joint plan:

     $$\Pi = \langle \ldots, \pi^a_{i-1}, \ldots, \rangle \rightarrow \Pi' = \langle \ldots, \pi^a_i, \ldots, \rangle$$

     In case no plan with higher utility than the lower bound can be found, the agent keeps its previous plan π^a_{i-1}, since it is still in best response.
- When no participant agent changes its plan in a complete iteration, the process has converged, and the current joint plan is a PNE.

6.2 Initial greedy joint plan

The complexity of our planning scenarios is proportional to the number of parcels and charging stations that they include. The planning during the first iteration of the BRFP process is considerably slower. The absence of previous individual plans implies not being able to use the previous plan utility bound (Sect. 5.4). We implemented a greedy method that creates an initial plan for every agent to palliate this. Such a greedy plan provides the first best-response iteration with a utility value to prune the search tree. The greedy plan is built as follows:

While there are uncompleted delivery tasks:

1. Select the parcel with a pick-up location closest to the agent's current location.
2. Check if the agent has enough travel capacity to complete the delivery.
   (a) If it does, go to (3).
   (b) If it does not, the agent goes to the closest station and charges. Then goes back to (1).
3. Complete the delivery task of the selected parcel (pick-up and drop-off).

The cost of the initial greedy plan will be higher or equal to that of the optimal plan but never lower. The creation of an initial greedy joint plan has proved to be very effective, significantly reducing the amount of generated nodes during the first planning process. However, it influences the BRFP process, as it guides it toward certain equilibria, avoiding others that cannot be reached with the method's restrictions.

7 Experimental results

The described solution has been implemented with Python 3.7. Among the employed Python modules, Shapely, Geopy, and Geojson stand out, as they were employed to reproduce real-life road networks, calculating travel distances and times over the city area where the vehicle fleet is deployed. Transport routing is solved by the open-source routing machine [28], a routing service that calculates, among others, the fastest route between any two given points. Each problem configuration was encoded in JSON format, indicating the attributes of each of the actors (agents, resources) of the problem together with their location in the city. Finally, the multi-agent simulator SimFleet [8] was used to load and visualize the problem configurations, although it had nothing to do with their resolution.

To test our system, we defined a set of 13 problem configurations, presented in Table 3, with different levels of complexity. The complexity of our problem is defined by the number of transport agents, the number of delivery tasks or parcels an agent must complete, and the number of charging stations. The number of parcels per agent (P/A) increases planning variability. Charging stations have the same effect. Therefore, as those values increase, the complexity does too. The number of agents mainly affects the performance of the best-response process, as more agents imply longer iterations and more conflicts to resolve. All charging stations belong to the same power network, whose maximum power is the addition of each station's power.

The main area of the city of Valencia, Spain, was chosen as the scenario for all the problems, and the agents used its road network. Distances among scenario elements are determined by their location in the city and expressed in meters. The speed of every transport agent is fixed. Figure 2 shows a visual representation of a problem in our urban mobility domain. The initial position of transport agents and parcel positions are defined according to a probability distribution computed from various city data, including population, traffic intensity per road, and geolocalized social network activity. Please note that each transport has been assigned its packages already. Thus, the problem we are dealing with is the coordination of their delivery.

Such a heterogeneous set of problems aims to show both our system's capability and limits. Therefore, we first compare the performance of our planner against other similar approaches. Then, we address the quality of our solutions to show how our approach optimizes the urban traffic system. Finally, we demonstrate the interest in modeling resource congestion and how self-interested agents can be incentivized to avoid it.

Unless otherwise indicated, the default values of the different variables affecting the agents' utility are those presented in Table 4. The base price of power is established, as well as the standard power consumption for electric vehicles. In addition, a unit cost for waiting time is defined. Finally, the congestion bounds of the different resources are indicated following their modeling in Sect. 4.1.

7.1 General performance

In the first set of experiments, we show the performance of our planner in terms of time to achieve a solution. For that, we solved the problems presented in Table 3 and measured the number of iterations the best-response algorithm

Table 3 Problem instances used for experimentation

Problem	Transport agents	Parcels	P/A	Stations	Charging poles
p20–60	20	60	3	20	40
p20–80	20	80	4	20	40
p20–100	20	100	5	20	40
p50–150	50	150	3	20	40
p50–200	50	200	4	20	40
p50–250	50	250	5	20	40
p100–200	100	200	2	30	60
p100–300	100	300	3	30	60
p100–400	100	400	4	30	60
p150–300	150	300	2	30	60
p150–450	150	450	3	30	60
p200–400	200	400	2	30	60
p500–1000	500	1000	2	30	60

Row values indicate the number of transport agents, parcels, parcels per agent (P/A), stations, and charging poles in the scenario, respectively

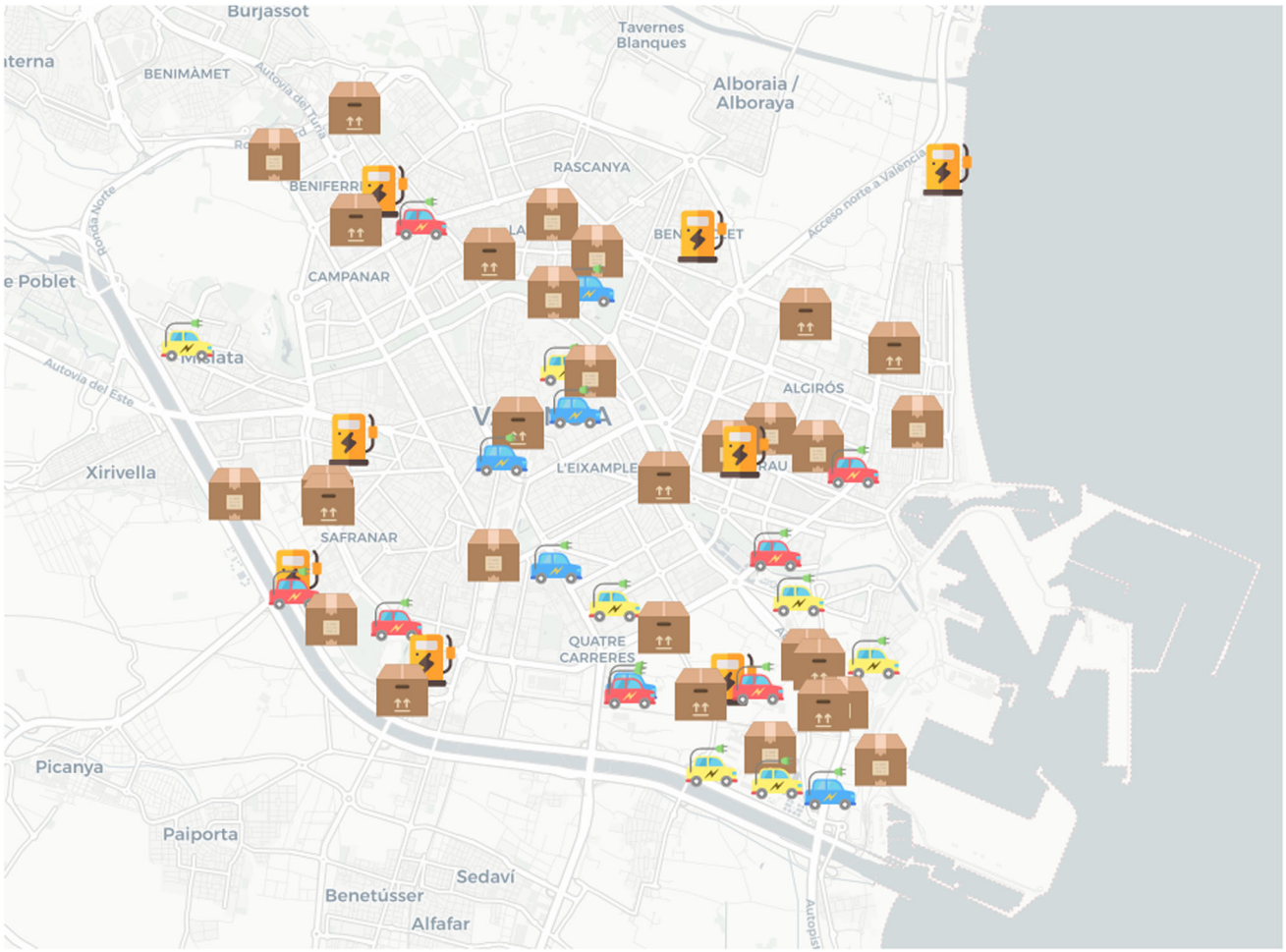

Fig. 2 Visualization of a problem configuration using the SimFleet software. The experimentation takes place in the main area of the city of Valencia, Spain. Different icons represent the location of electric vehicles, parcels, and charging stations

Table 4 Default values of different problem variables

Price per KWh	0.3 €
Power consumption per Km	0.14 KWh
Power price per Km	0.3 · 0.14
Time penalty (waiting cost)	1
Road network cong. bound	0.3
Power network cong. bound	0.5

These variables affect the numeric value of the agent's costs and the application of congestion

needed to converge, the total running time of the BRFP, and the average time that each individual planner instance took to return a solution. We also indicate the time per iteration since it is helpful to estimate the total running time of problems with a similar level of complexity.

This process was repeated on five instances of the problems: problems with the same complexity magnitude (number of agents, parcels, and stations) but with the elements positioned differently within the scenario. This was done to palliate the irregularity among problems caused by element positioning. Averages of the results of the five instances are presented in Table 5,[1] where time is expressed in seconds.

As can be seen, the number of parcels per agent (P/A) is closely related to the increase in planning time. An agent with more parcels will have more ways to deliver them; thus, it will have to explore every order to find the optimal one. The standard deviation of the planning time also increases with the problem's complexity. Regarding the total time, the best-response process is expected to last longer with the more participants it has. With problems such as p500-1000, even though the P/A number is only 2, the high number of agents makes the process too time-consuming.

[1] All the tests were conducted on a single machine with an Intel Core i7-7700 CPU at 3.60GHz and 16 GB RAM.

Table 5 Average performance of five repetitions of the problem set

Problem	P/A	Iterations	Total time	Time/iter.	Planning time
p20–60	3	3.2	22.2	6.82	0.45±0.21
p20–80	4	3.6	72.8	20.06	1.33±0.83
p20–100	5	4.0	234.4	58.59	3.82±3.11
p50–150	3	3.4	150.1	43.61	1.12±0.60
p50–200	4	4.2	523.7	125.08	3.14±2.09
p50–250	5	5.5	2693.0	489.25	11.79±10.81
p100–200	2	3.0	206.7	68.90	0.89±0.52
p100–300	3	3.8	820.2	216.06	2.72±1.42
p100–400	4	4.8	3638.7	757.45	9.28±6.96
p150–300	2	3.0	528.7	176.24	1.53±0.92
p150–450	3	3.8	2102.2	555.08	4.73±2.47
p200–400	2	3.0	929.1	309.69	2.01±1.21
p500–1000	2	3.0	6462.8	2154.27	5.75±3.48

Times indicated in seconds. *P/A* parcels per agent, *iterations* number of iterations the best-response algorithm took to converge, *total time* time until a solution was obtained, *time/iter.* total time/iterations, *planning time* average time that each individual planner instance needed to return a solution

Even though our work focuses on our particular problem and domain, we want to compare it with a similar approach. The research in [24] approaches fleet coordination through the so-called *Better-response* dynamics. As its name hints, such an algorithm is developed in a very similar way to best-response but with the agents proposing plans that *improve* the utility of their previously proposed plan, not necessarily being the current *best* plan. As the authors prove, the convergence to an equilibrium in such a case is guaranteed as with best-response dynamics. With this, the need for optimal planning is avoided. This enables the authors to use a general-purpose satisficing planner that can be applied to different domains. However, in contrast with our planner, the plan computation for complex scenarios is computationally more costly, as our planner is refined for the specific planning scenario. Regarding the problem modeling, we are applying the BRFP to a more realistic application using the real road network and a routing service (OSRM). In contrast, their modeling solves an electric autonomous taxi problem in simple networks with a concrete number of junctions.

Finally, assessing the experimentation results of both approaches, we can see that our system can solve scenarios with a higher level of relative complexity. The problem complexity in these scenarios depends on the number of agents to be routed and the number of junctions, increasing the planning process's ramifications. The most complex experiments performed in [24] and [29] [Section 6.4] include six agents (which can be interpreted as 18 since each company agent manages three taxis carrying customers) and between 8 and 12 junctions, depending on the case. Therefore, our simplest problem (p20–60) is already orders of magnitude above the aforementioned ones that require almost 1800 s of computation time to reach an equilibrium, which makes it unfair to compare planning and total times directly. Our approach's most significant benefit (with its ad hoc planner) brings the system's ability to manage up to 500 agents in a considerable amount of time. Nevertheless, most of our configurations reach an equilibrium in less than 15 min, except in the most complex cases where between 15 min and an hour is required, even for problems like p50–250, where the planning process is especially complex.

Our system's major limitation comes from the complexity of planning, which is PSPACE-complete [30] or even harder in practice for optimal planning [26]. The computation time increases exponentially with the problem complexity, which means that our planner would stop returning solutions in a reasonable time for a certain number of agents or problem variability. However, given the nature of the type of problems we are dealing with, which include both the self-interest of the agents, thus requiring a game-theoretic approach, and their planning capabilities to perform a set of tasks optimally, the theoretical complexity cannot be reduced except by improving at the practical level the computation time, as we have done using an ad hoc planner for the restricted domain we have defined.

From the point of view of game theory, our system introduces a number of agents, which is orders of magnitude above the norm. The computation of equilibria is costly; therefore, most applications cannot bear to compute them for games with such a significant number of participants. Even though our approach computes only a single equilibrium, it can do so for up to 500 participants in

complex planning scenarios that feature congestion and conflicts.

7.2 Comparison of solution quality

Users of urban traffic systems, especially drivers, tend to act selfishly, only concerned about their goals, whether those are to reach their destination fast, follow their preferred route, etc. With the introduction of game theory techniques, we can turn selfishness into competitiveness and the latter into optimization. If every user acts in the best way with respect to every other user in the system, their experiences will improve.

Decentralized coordination such as the one presented in this paper may seem inadequate to optimize a system globally. Nevertheless, having agents follow selfish strategies in a competitive (or non-cooperative) scenario will generally improve the agents' utilities and some of the system's metrics.

In this section, we compare the agent plans obtained by our BRFP with those obtained by a greedy strategy that aims to reproduce the behavior of an uninformed, selfish driver. We analyze the agents' costs from a global perspective, showing that both agents' utilities and system metrics are optimized by following the BRFP. A solution with improved agent utility implies, in turn, global benefits such as optimal delivery of the parcels, both in order and time, reduced energy spending, and less resource congestion. Although there are related works such as [22] in which the quality of Nash equilibrium solutions is compared using additional criteria such as Pareto optimality or fairness, in such a case, it is necessary to list all Nash equilibria of the game, which is unfeasible in complex, realistic scenarios such as ours. This is why we have preferred to compare with a greedy solution preserving the complexity of the scenario.

The experimentation has been carried out by obtaining solutions to the first instance of the previous 13 problems (Table 3). Those problems were solved using the BRFP and the so-called Greedy Solver (GS). The GS builds an agent's plan in two steps: Greedy plan-building and conflict-solving.

The greedy plan is built with the following strategy: The agent tries, at each time, to complete the delivery task whose pick-up location is the closest to its current position. If, at some point, the agent's travel capacity is not enough to complete the selected delivery task, it will instead drive to the closest station and recharge its batteries. After that, the delivery task selection will begin again. This process finishes once the agent has completed every task, thus obtaining a complete plan. Then, any action of the plan that causes a conflict with another agent is delayed until the conflict no longer exists. Ultimately, we obtain a feasible joint plan composed of individual greedy plans.

The results of this experimentation are presented in Table 6, where a comparison of the problems solved by the GS and the BRFP can be seen. The metrics that define the quality of a solution are the mean total cost of the delivery fleet operation (according to each agent's utility function described in Sect. 4.1); and the number of agents that experienced either a road or a power network congestion. The BRFP shows lower values for the mean total cost and the number of congested agents.

Table 7 further analyzes the comparison by showing the percentage in which every mean cost is reduced by the BRFP (with respect to the GS's solutions). As can be seen, the average total cost is reduced between 3.23% and 10.43%. As the problem complexity increases, the reduction is higher. It can be observed how the number of parcels per agent (P/A) affects the decrease in total cost. For 2 P/A, the cost reduction does not overcome 3.81%. For 3 P/A, the reduction reaches a value of 7.81%. Finally, for problems with 4 or 5 P/A, the reduction can surpass 10%. Even though the total cost reductions are not high, there is a significant decrease in the number of congested agents (see Table 7, columns under "congested agents"). This shows how our approach, despite the selfish strategy of the agents, in a competitive environment can lead to socially better solutions. In problems p20–80 and p50–150, the results show an increase in agents which suffered road congestions (values – 200% and – 700%) in favor of a high reduction in those that suffered power (or charge) congestions. This occurs mainly because, for this problem's configuration, the cost increment associated with power congestion is higher than the one associated with road congestion. Consequently, when the planner only finds plans which involve either one or the other, road congestion will generally be preferred.

Ultimately, these results confirm the usefulness of our approach and show how the use of self-interested agents improves not only their benefits, but also brings global improvements. Our system increases customer satisfaction, reducing the time it takes to deliver all parcels. Also, the sustainability of the urban traffic system is enhanced, firstly, by reducing traffic and power congestion and, secondly, by decreasing vehicle operating times, which entails fewer kilometers traveled and, therefore, less energy consumption. In addition, this type of system also studies and promotes the implementation of electric vehicles as the standard in urban environments.

Ideally, we would compare our system with one that solves problems in the same domain but using a centralized approach. However, because of our ad hoc design, there is no other application we could fairly compare it with. Even so, the relevance of our decentralized planning and the

Table 6 Solution quality comparison for problems solved with the Greedy Solver and the BRFP

		Greedy solver			BRFP		
			Congested agents			Congested agents	
Problem	P/A	Mean total cost	Road	Power	Mean total cost	Road	Power
p20–60	3	77.07	0	0	71.45	0	0
p20–80	4	138.39	0	16	123.95	2	8
p20–100	5	200.33	0	17	182.42	0	14
p50–150	3	69.81	1	10	66.30	8	0
p50–200	4	126.96	2	27	114.65	0	14
p50–250	5	203.89	5	44	182.70	3	30
p100–200	2	30.62	0	0	29.63	0	0
p100–300	3	70.90	0	19	65.51	0	0
p100–400	4	131.03	0	70	118.10	0	38
p150–300	2	33.04	0	0	31.78	0	0
p150–450	3	72.99	0	33	67.29	0	0
p200–400	2	31.70	0	0	30.58	0	0
p500–1000	2	32.07	2	0	31.00	2	0

The problems present different values for the average total cost and the number of congested agents. The mean total costs of the fleet's transports as well as the number of congested agents are generally lower with the BRFP

Table 7 Greedy solver versus BRFP costs reduction percentages

				Congested agents		
Problem	P/A	Total cost (%)	Traveled kms	Road	Power	Waiting cost
p20–60	3	7.29%	4.04%	0.00%	0.00%	7.35%
p20–80	4	10.43%	3.54%	−200.00%	50.00%	10.46%
p20–100	5	8.94%	5.18%	0.00%	17.65%	8.91%
p50–150	3	5.03%	2.10%	−700.00%	100.00%	5.08%
p50–200	4	9.70%	3.43%	100.00%	48.15%	9.67%
p50–250	5	10.39%	5.51%	40.00%	31.82%	10.36%
p100–200	2	3.23%	1.48%	0.00%	0.00%	5.46%
p100–300	3	7.60%	3.70%	0.00%	100.00%	7.66%
p100–400	4	9.87%	5.44%	0.00%	45.71%	9.75%
p150–300	2	3.81%	2.59%	0.00%	0.00%	3.85%
p150–450	3	7.81%	5.06%	0.00%	100.00%	7.79%
p200–400	2	3.53%	2.00%	0.00%	0.00%	3.56%
p500–1000	2	3.34%	1.98%	0.00%	0.00%	3.39%

Columns 5 and 6 show a reduction in the number of congested agents (instead of costs)

best-response algorithm is preserving the agents' private interests. No entity imposes actions on the agents; they decide for themselves their best possible actions, guided by data from the urban traffic system.

7.3 Effect of congestion

The modeling of resource congestion and how the agents can be aware of it to avoid it gives our system the potential to test interesting scenarios. Higher congestion cost increments will drive agents to avoid congestion with greater interest. In the experimentation presented so far, in Sects. 7.1 and 7.2, the impact of congestion cost increments on an agent's total cost was minimal and did not strongly influence the agent's plan. In those cases, the customer's waiting time was the main parameter to optimize, as it entailed a much higher cost.

In this section, we analyze the changes in agents' costs and behavior when *resource bounds* vary and greater *congestion costs* are introduced to the system. Therefore, for the following experiments, the power network congestion costs were multiplied by a hundred, whereas the

road network congestion ones were multiplied by fifty. With this, we achieve congestion costs whose order of magnitude is comparable to the waiting cost of customers.

7.3.1 Resource bound variation

For the first round of experiments, we executed problem p20–100 with values for the resource bounds ranging from 0 (any simultaneous use congests the resource) to 1 (the resource will only be congested if all agents are using it simultaneously). The relevant results are presented in Table 8. As can be seen, with a bound of 0, most agents get involved in congestion at some point in their plans. The number of congested agents decreases as the bound is incremented until no agent gets congested. The cost increments associated with congestion are higher according to the number of agents involved. That is reflected in both the congestion cost and the total cost, which are reduced as the bound increases and fewer agents get congested.

On the other hand, it can also be observed in Table 8 that there is much more congestion on the power network than on the road network when the bounds are between 0.1 and 0.25. This occurs because transports can coincide in time using the power network more easily than the roads since there is only one power network, while the roads that agents can take are less likely to coincide.

Table 8 Average costs and congested agents variation according to resource bound

Power network congestion						
	Total cost		Power congestion			
Problem	Mean	Std	Count	Mean	Std	Power bound
p20–100	199.7	26.3	19.0	8.1	4.4	0
p20–100	196.7	26.9	18.0	5.9	3.2	0.1
p20–100	192.1	27.0	12.8	3.4	1.2	0.25
p20–100	189.4	26.3	0.8	1.9	0.0	0.5
p20–100	188.5	26.0	0.0	0.0	0.0	> 0.5

Road network congestion						
	Total cost		Road congestion			
Problem	Mean	Std	Count	Mean	Std	Route bound
p20–100	207.4	29.3	20.0	15.3	7.6	0
p20–100	190.3	25.3	7.7	7.1	6.5	0.1
p20–100	186.2	25.5	0.3	0.3	0.0	0.25
p20–100	186.0	25.5	0.0	0.0	0.0	> 0.25

count columns indicate the number of agents involved in congestion. *mean* columns indicate the average cost increment and *std* columns show the standard deviation

7.3.2 Agent behavior analysis

Following the trend of researching the effect of congestion, in this experiment, we analyze the change in agent behavior (reflected in their plans) once higher congestion costs are introduced to the system. With our base modeling, the plan building is mainly motivated by the agent's waiting cost. In other words, agents prioritize on-time delivery and thus reduce the customer's waiting time. Congestion costs may appear, and the agent will be inclined to avoid them when possible. However, when avoiding congestion involves an increase in waiting cost that overcomes the congestion cost increment, the agent will decide to assume the congestion in favor of faster delivery since it implies a lower cost.

The behavior described above is intended and achieved thanks to the higher value of waiting costs with respect to congestion cost increments. For the following experiment, we cause a change in agent behavior by increasing congestion costs, making them overcome waiting costs. With this, the agent's primary motivation will be to avoid congestion. To study such a setting, we built a small problem configuration, with ten transport agents, three delivery tasks each, and five charging stations, all belonging to the same power network, with two charging poles each. The transport agents had an initial travel capacity of 20 km out of a maximum travel capacity of 30 km. Because of that some agents will need to recharge their batteries at some point in their execution to complete their delivery tasks.

The BRFP has solved the aforementioned configuration in three different ways: (1) without considering charging congestion costs ($w_p = 0$ in Eq. 1), (2) with the default charging congestion cost increments ($w_p = 1$), and finally, (3) with significantly higher charging congestion cost increments ($w_p = 100$). The power network bound was fixed at 0.5, its default value for the previous experiments.

Our results are presented both in Table 9 and in Figs. 3, 4, and 5, where the charging intervals of the agents are represented on a timeline. For simplicity, the continuous time has been discretized in the representations. Bear in mind that as all charging stations are fed by the same power network, the specific station in which agents are charging is not relevant. Because of the same reason, any overlapping interval indicates an increment in the power demand to the power network. Power congestion will arise when such an increment exceeds the power network bound.

Comparing executions (1) (Fig. 3) and (2) (Fig. 4), it can be seen how the introduction of a mild congestion cost can cause some agents to opt for a plan which avoids it. In this case, agent 7 moved its charge action to the beginning of its plan. With such a change, it is still available to

Table 9 Mean costs and congested agent number for the different executions of the problem configuration

Instance	Total cost	Congested agents	Cong. cost	Waiting cost
(1)	71.1 ± 16.2	–	–	69.9 ± 16.0
(2)	71.4 ± 16.3	6	0.42 ± 0.05	70.0 ± 16.0
(3)	74.5 ± 18.1	4	3.60 ± 0.71	72.0 ± 17.7

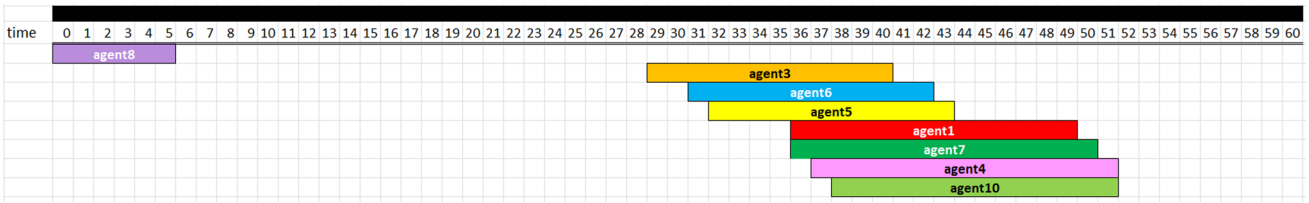

Fig. 3 Timeline of transport agent charging intervals with no charging congestion costs. Agents are represented by colored rectangles whose length indicates the duration of the charge. Most of the agents recharge their batteries between the 29th and the 51st time units, as they have no incentives not to overcharge the power network

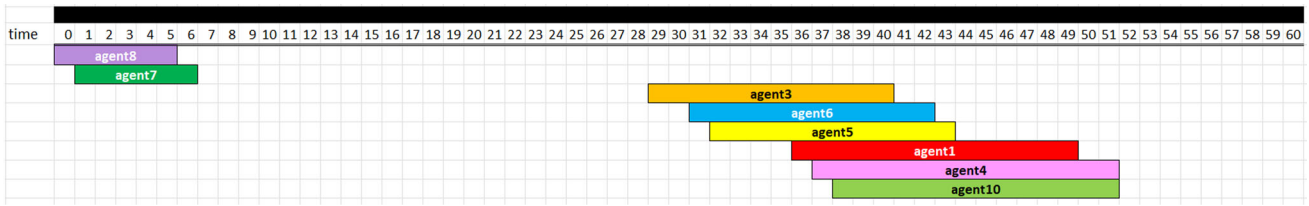

Fig. 4 Timeline of transport agent charging intervals with default charging congestion costs. Agents are represented by colored rectangles whose length indicates the duration of the charge. The charging congestion cost is not enough to motivate the majority of the agents not to overcharge the power network, resulting in most agents recharging between the 29th and the 51st time units

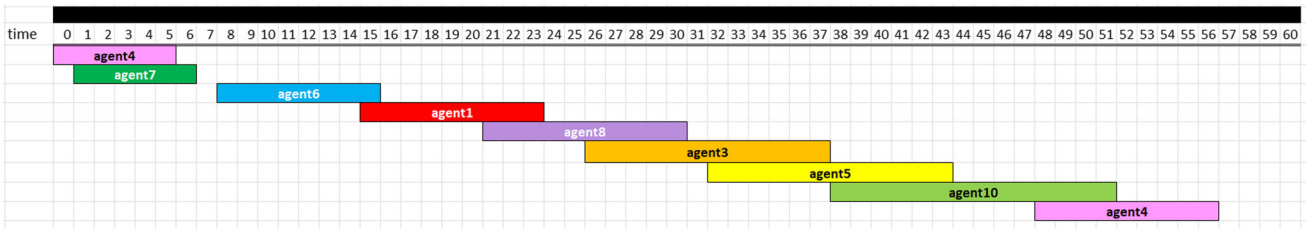

Fig. 5 Timeline of transport agent charging intervals high charging congestion costs. Agents are represented by colored rectangles whose length indicates the duration of the charge. The high congestion costs motivate the agents to schedule their recharge so as not to overcharge the power network, thus splitting their charging intervals uniformly over the simulation time

complete every delivery task with only one charge, and, at the same time, it avoids charging in the period in which the network is overused. However, it is also clear that for most agents, the cost increment of congestion is not high enough to induce a change of plan. Agents 1, 3, 4, 5, 6, and 10 prefer to keep their charge schedule even though all of them are affected by congestion (see instance (2) in Table 9), which increases the price of their charge. This is because, as we commented above, the rescheduling of their charge would entail an increment in the customer waiting time, which is the factor that contributes to the total cost the most.

With a high congestion cost increment (Fig. 5), there is an evident change in behavior, as agents are now interested in avoiding congestions and, in case of being unable to do so, minimizing the overlap of their charge with the charge intervals of other agents. It can be seen how agent 4 decides to charge two times, only to avoid congestion. Also, agent 8 delays its charge, as currently charging at the start would provoke congestion with agents 4 and 7. In this case, agent 8 gets involved in congestion, but it only partially overlaps with two agents (1 and 3), so the cost increment is not too high.

As it can be seen in Table 9, the customer waiting time slightly increases for (3), as the charge actions are now scheduled mainly to avoid congestion (in contrast with (1) and (2), in which they were scheduled to reduce waiting time). Nevertheless, the amount of congested agents is

reduced by 2, and, what is more relevant, the pressure on the power network is evenly divided along with the execution of the agents' plans. Figures 6 and 7 present a visualization of the power network usage, showing with colorized intervals concurrent charges. Darker colors indicate a higher number of overlapping charges. With the default charging congestion costs (Fig. 6), the maximum amount of overlapping charges is 7, whereas with high congestion costs (Fig. 7) it is only 2.

In the proposed system, a cost variation can significantly influence the agents' actions, as they are mainly motivated to reduce costs. These experiments show how the system can be tuned to achieve solutions with higher global customer satisfaction, as in (2), or reduce the simultaneous use of a resource, such as the power network, in (3).

8 Discussion

This section enumerates current challenges in the field of urban transportation and optimization. Then, it discusses the application of the described system to other smart areas and different problem domains, and describes the limitations of our system.

8.1 Urban transportation challenges

The research field of urban transportation and optimization faces various challenges. One of the most prominent issues is traffic congestion, resulting from increasing urbanization and the rising number of vehicles on the road. The adverse effects of traffic congestion include significant economic and environmental losses and a decline in the quality of life for city residents. Another challenge is to make transportation more sustainable and reduce greenhouse gas emissions and air pollution. The development of intelligent transportation systems has led to the use of sensors, data analytics, and other technologies to optimize traffic flow and reduce congestion, but their implementation can be complex and costly. Autonomous vehicles have the potential to revolutionize urban transportation, but safety concerns, infrastructure requirements, and public acceptance present significant challenges. Lastly, the growth of e-commerce has created a major challenge for urban areas regarding last-mile delivery. Addressing these challenges and finding ways to optimize delivery routes and reduce the environmental impact of delivery vehicles is a crucial area of research in urban transportation and optimization.

Our work addresses several of the aforementioned issues. Specifically, we contribute toward reducing traffic congestion and improving sustainability by modeling fleet coordination as a congestion game. This approach motivates each agent to optimize their use of city resources such as roads and charging stations. Our approach is also relevant to developing intelligent transportation systems, as it relies on data estimates and sensor technology to improve fleet operations and enhance the quality of life of city residents. Additionally, our coordination principles of decentralization, privacy, and agent autonomy contribute to the development of autonomous transportation. Finally, our work is applicable to the challenge of last-mile delivery, as the domain we have developed addresses a problem within this category.

8.2 Applicability of the proposal to other domains

The main area of the city of Valencia, Spain, has been chosen to illustrate the operation of the proposed system as well as to perform its evaluation. From a general perspective, our system offers a solution for the coordination of open fleets composed of autonomous, self-interested agents, independent of the agents' goals and the concrete area where they are deployed. The coordination by means of game theory, however, requires complete information for the agents to make a strategic decision. In terms of the deployment area, this implies having access to real-time data and the computation of estimations. For the chosen domain, parcel delivery, those estimations would be traffic congestion, traveling times, and speed. Because of that the presented system can be applied to any smart area that fulfills its need for estimated data.

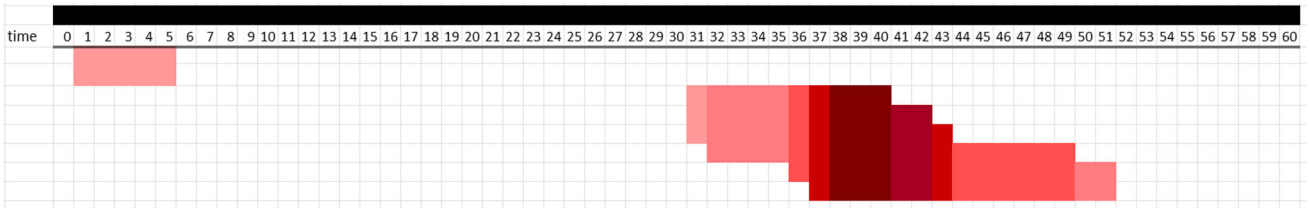

Fig. 6 Timeline of power network usage with default charging congestion costs. Painted intervals illustrate concurrent charges. Darker colors indicate a higher number of concurrent charges. The power network is congested between the 32nd and the 51st time units, showing a major congestion between 38th and the 40th

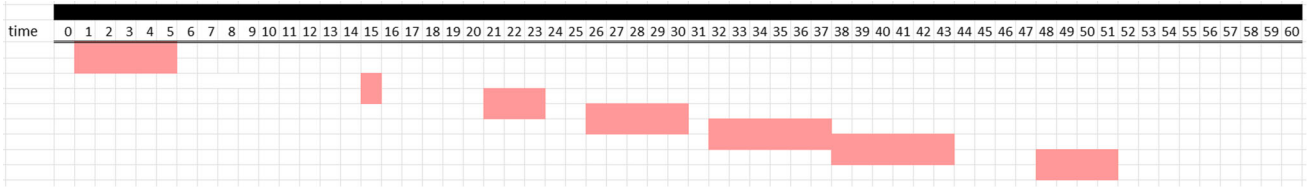

Fig. 7 Timeline of power network usage with high charging congestion costs. Painted intervals illustrate concurrent charges. The network does not get congested, as the maximum number of concurrent charges is two

Regarding the application domain, the present work assesses traffic optimization by coordinating open delivery fleets. For that, a so-called "ad hoc" planner is designed, as commented in Sect. 5. The term "ad hoc" refers to the design of the planner according to the domain; this is, according to the specific actions, conflicts and utility functions defined for the parcel delivery problem, described in Sect. 4. If we separate the system from the domain chosen in this work, it offers a general solution for coordinating autonomous self-interested agents, regardless of the agents' concrete goals. The only essential requirement for the described system to operate is the existence of a mechanism that allows the agents to plan their actions according to their objectives, hence the creation of the ad hoc planner. Ultimately, this implies that with a few adjustments, it is possible to modify the presented planner, adapt it to a new domain, and thus develop a solution for that domain without changing the workflow of the entire system.

8.3 System limitations

The described approach, despite being highly configurable, has a series of limitations that must be commented. On the one hand, the planning of agent actions is performed statically before the agents' execution. This requires that each agent estimates their utility function. For the domain of parcel delivery, the estimation includes traffic congestion and traveling time and speed. Because of that the most realistic application area of our system would be in a smart, highly monitored closed area, where all components are constantly sharing data. The implementation in an open area with components or agents that are not willing to share data would worsen the estimations and, thus, the operation of the fleet.

On the other hand, the best-response coordination requires that participants are willing to share their actions with other agents. While the concrete goals may remain private, the course of action toward such goals must be disclosed. Since our system is thought to coordinate agents in a non-cooperative but non-strictly competitive scenario, some participants may have reservations about sharing their plans. Notice, however, that the final objective of the coordination is the optimization of the whole operation and the usage of the area's resources, and thus sharing information would ultimately benefit all involved parties.

9 Conclusions

In this paper, we have presented a system for coordinating urban fleets of self-interested agents using best-response dynamics and multi-agent planning. The problem addressed has been defined as an urban mobility domain in which a set of transport agents, which may represent electric vehicles, have to carry the parcels assigned to them from an origin to their destination. To do so, each agent can strategically decide the order in which it makes the deliveries, as well as when and where to recharge the vehicle's batteries. These strategic decisions are made by each agent, in particular, depending on the strategies (plans) of the other agents to obtain the highest possible utility avoiding the congestion of both the power network and the roads, as well as conflicts due to lack of free poles in the power stations.

To resolve this mobility problem with self-interested agents, an ad hoc planner has been developed for this domain. Each transport agent has its own instance of the planner to obtain a plan that is the best response to the plan of the other agents, i.e., the current joint plan. Thus, the resolution of the complete problem is approached by a best-response algorithm in which each agent, in turn, proposes its best plan with respect to the current joint plan. This iterative process ends when no agent changes its plan during a complete iteration, in which case, the resulting joint plan is guaranteed to be a pure-strategy Nash equilibrium.

We have tested our system's performance for different levels of complexity through extensive experimentation. Using our own ad hoc planner is an advantage over similar systems with general-purpose planners. The optimal plans' obtention is achieved relatively quickly, even for the most complex settings. However, one must take note of the restrictions of our domain, which also help speed up the search. In addition, we have also compared the quality of the solutions obtained with our approach, which are Nash equilibria, versus solutions obtained with a greedy

approach. In this sense, the solutions of our system are better (around 7% on average, and more than 10% in several cases) from a global point of view by avoiding congestion and unnecessary waits. Moreover, being equilibrium solutions for self-interested agents, it can be ensured that none of the agents is incentivized to change its plan instead of other types of coordination solutions that agents might not respect, causing conflict situations.

Our system, because of its characteristics, is not able to adapt to new delivery tasks as fast as online approaches would. In general, any change in the initial conditions of the problem would require a new equilibrium, that is, a new execution of the best-response algorithm. However, our system would be adequate for problems in which the parcels to be delivered are not updated every few minutes since delivery windows of 15, 30, or 60 min could be assumed, depending on the needs. If the system were to be implemented as a smart city solution, the computing power would be much higher than that used in our tests with a conventional computer. This would imply that solutions could be computed in seconds or a few minutes.

In the future, we aim to overhaul the described system by including a mechanism that detects and deals with incorrect data estimations. One of the limitations of the system, as discussed, is the reliance on data estimates. With this improvement, we would increase system reliability for use in real-world scenarios. Also, in the line of fair and optimized coordination of vehicle fleets, we would like to explore the implementation of a task allocation algorithm that follows the principles of privacy and decentralization established for this work. Finally, a future development of greater magnitude would be the development of a digital twin representing the deployment area of the open fleet together with its resources and with the possibility to include new and different smart services. Such work would bring a tool for significant research in smart cities and their optimization.

Acknowledgements This work is partially supported by Grant PID2021-123673OB-C31 funded by MCIN/AEI/10.13039/501100011033 and by "ERDF A way of making Europe." Pasqual Martí is supported by Grant ACIF/2021/259 funded by the "Conselleria de Innovación, Universidades, Ciencia y Sociedad Digital de la Generalitat Valenciana". Jaume Jordán is supported by Grant IJC2020-045683-I funded by MCIN/AEI/10.13039/501100011033 and by "European Union NextGenerationEU/PRTR".

Funding Open Access funding provided thanks to the CRUE-CSIC agreement with Springer Nature.

Data Availability Data sharing not applicable to this article as no datasets were generated or analyzed during the current study.

Declarations

Conflict of interest The authors declare no conflict of interest. The funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript, or in the decision to publish the results.

References

1. Rao Z et al (2022) Machine learning enabled high-entropy alloy discovery. Science 378(6615):78–85
2. Chen Y, Lu C, Yan J, Feng J, Sareh P (2022) Intelligent computational design of scalene-faceted flat-foldable tessellations. J Comput Des Eng 9(5):1765–1774
3. Railsback SF, Grimm V (2019) Agent-based and individual-based modeling: a practical introduction. Princeton University Press, Princeton
4. von Neumann J, Morgenstern O (1944) Theory of games and economic behavior. Princeton University Press, Princeton
5. Jordán J, Palanca J, Martí P, Julian V (2022) Electric vehicle charging stations emplacement using genetic algorithms and agent-based simulation. Expert Syst Appl 197:116739
6. Utomo DS, Gripton A, Greening P (2022) Designing mixed-fleet of electric and autonomous vehicles for home grocery delivery operation: an agent-based modelling study. IEEE, pp 1401–1412
7. Davidsson P, Holmgren J, Persson JA, Ramstedt L (2008) Multi agent based simulation of transport chains. In: IFAAMAS
8. Palanca J, Terrasa A, Carrascosa C, Julián V (2019) Simfleet: a new transport fleet simulator based on mas. Springer, pp 257–264
9. Martí P, Jordán J, Palanca J, Julian V, Analide C, Novais P, Camacho D, Yin H (2020) Free-floating carsharing in simfleet. In: Analide C, Novais P, Camacho D, Yin H (eds) Intelligent data engineering and automated learning—IDEAL 2020. Springer, Cham, pp 221–232
10. Lopez PA et al (2018) Microscopic traffic simulation using sumo. IEEE, pp 2575–2582
11. Axhausen WK, Horni A, Nagel K (2016) The multi-agent transport simulation MATSim. Ubiquity Press, New York
12. Adnan M, et al (2016) Simmobility: a multi-scale integrated agent-based simulation platform, vol 2. The National Academies of Sciences, Engineering, and Medicine Washington, DC
13. Fellendorf M, Vortisch P (2010) Microscopic traffic flow simulator vissim. Fundam Traff Simul 63–93
14. Mariani S, Cabri G, Zambonelli F (2021) Coordination of autonomous vehicles: taxonomy and survey. ACM Comput Surv (CSUR) 54(1):1–33
15. Yang Z, Guo T, You P, Hou Y, Qin SJ (2019) Distributed approach for temporal-spatial charging coordination of plug-in electric taxi fleet. IEEE Trans Industr Inf 15(6):3185–3195
16. Gigante G et al (2018) Game-theoretic approach for the optimal configuration computing of an interoperable fleet of unmanned vehicles. Expert Syst 35(5):e12293
17. Weerdt MD, Clement B (2009) Introduction to planning in multiagent systems. Multiagent Grid Syst 5(4):345–355

18. Durfee EH (2001) Distributed problem solving and planning. Springer, pp 118–149
19. Torreño A, Onaindia E, Sapena Ó (2014) Fmap: distributed cooperative multi-agent planning. Appl Intell 41(2):606–626
20. Dunne PE, Kraus S, Manisterski E, Wooldridge M (2010) Solving coalitional resource games. Artif Intell 174(1):20–50
21. Jordán J, Onaindía E (2015) Game-theoretic approach for non-cooperative planning, pp 1357–1363
22. Jordán J, Torreño A, de Weerdt M, Onaindia E (2021) A non-cooperative game-theoretic approach for conflict resolution in multi-agent planning. Group Decis Negot 30(1):7–41
23. Nash J (1951) Non-cooperative games. Ann Math 54(2):286–295
24. Jordán J, Torreño A, De Weerdt M, Onaindia E (2018) A better-response strategy for self-interested planning agents. Appl Intell 48(4):1020–1040
25. Rosenthal RW (1973) A class of games possessing pure-strategy Nash equilibria. Int J Game Theory 2(1):65–67
26. Aghighi M, Bäckström C (2016) A multi-parameter complexity analysis of cost-optimal and net-benefit planning
27. Monderer D, Shapley LS (1996) Potential games. Games Econom Behav 14(1):124–143
28. Luxen D, Vetter C (2011) Real-time routing with openstreetmap data, GIS '11. ACM, 513, pp 513–516
29. Jordán J (2017) Non-cooperative games for self-interested planning agents. Ph.D. thesis, Universitat Politècnica de València
30. Bylander T (1994) The computational complexity of propositional strips planning. Artif Intell 69(1):165–204