Hindawi
Complexity
Volume 2020, Article ID 1047369, 21 pages
https://doi.org/10.1155/2020/1047369

Review Article

Decentralizing Coordination in Open Vehicle Fleets for Scalable and Dynamic Task Allocation

Marin Lujak[1], **Stefano Giordani**[2], **Andrea Omicini**[3] and **Sascha Ossowski**[4]

[1]*CERI Numeric Systems, IMT Lille Douai, 59508 Douai, France*
[2]*Dip. Ingegneria Dell'Impresa, University of Rome "Tor Vergata", 00133 Rome, Italy*
[3]*Department of Computer Science and Engineering (DISI) Alma Mater Studiorum—Università di Bologna, 47522 Cesena, Italy*
[4]*Centre for Intelligent Information Technologies (CETINIA), University Rey Juan Carlos, 28933 Madrid, Spain*

Correspondence should be addressed to Marin Lujak; marin.lujak@imt-lille-douai.fr

Received 25 May 2019; Revised 21 January 2020; Accepted 24 June 2020; Published 16 July 2020

Academic Editor: Daniela Paolotti

One of the major challenges in the coordination of large, open, collaborative, and commercial vehicle fleets is dynamic task allocation. Self-concerned individually rational vehicle drivers have both local and global objectives, which require coordination using some fair and efficient task allocation method. In this paper, we review the literature on scalable and dynamic task allocation focusing on deterministic and dynamic two-dimensional linear assignment problems. We focus on multiagent system representation of open vehicle fleets where dynamically appearing vehicles are represented by software agents that should be allocated to a set of dynamically appearing tasks. We give a comparison and critical analysis of recent research results focusing on centralized, distributed, and decentralized solution approaches. Moreover, we propose mathematical models for dynamic versions of the following assignment problems well known in combinatorial optimization: the assignment problem, bottleneck assignment problem, fair matching problem, dynamic minimum deviation assignment problem, Σ_k-assignment problem, the semiassignment problem, the assignment problem with side constraints, and the assignment problem while recognizing agent qualification; all while considering the main aspect of open vehicle fleets: random arrival of tasks and vehicles (agents) that may become available after assisting previous tasks or by participating in the fleet at times based on individual interest.

1. Introduction

Open collaborative vehicle fleets composed of autonomous self-interested system participants are ever more widespread. However, even though the drivers are autonomous and self-interested, the authority and the ownership of these systems today remain centralized in terms of management, control, and access. The trend seems to be an ever-increasing access to mobility and last-mile services for the average person at the cost of relying on just a few (centralized) worldwide enterprises. The state-of-the-art algorithms for the allocation of tasks to vehicle fleets solve customer requests in very large fleets in almost near real time, but they seem to be limited to centralized systems. Centralization here can be a source of failure (a single bottleneck of the system), obsolete information due to significant computation delay while processing ever-increasing quantity of data, privacy evasion, and mistrust if the interests of the enterprise mismatch the users' interest.

Distributed decision-making (DDM) obviously resolves the drawbacks of centralized systems. The multitude of the connected smart devices of the vehicles' drivers and customers makes it possible to combine their potential and to coordinate fleets at a scale exceeding spatial and computational boundaries. This potential can be exploited for the benefit of the fleet system as a whole as well as for the interest of individual vehicle drivers and customers.

The decision-making authority in the DDM is distributed throughout a system, and the decisions are taken locally based on the local and shared global information and the interactions of an individual with the rest of the system and with the environment. Here, each fleet participant is

modelled as an autonomous collaborative individually rational software agent installed on a user's smart device. The agent has only a local vision of the fleet and it needs to cooperate with other agents in order to find the allocation of dynamically appearing tasks faced by the whole fleet.

The behaviour of the fleet as a whole is a result of intervehicle coordination. Distributed task allocation strongly contributes to the shift of knowledge and power from the individual (fleet owner) to the collective (vehicles composing the fleet). A desired behaviour of the fleet emerges from the identifiable interest of its participating vehicles, their beliefs, and collective actions and, as such, is a shift away from the hierarchical organizational paradigm (see, e.g., [1]). A major challenge is the identification of a right decision-maker for each part of the problem, timely exchange of relevant and up-to-date information among vehicle agents, and modelling of complex relations in such a multiagent system. A trade-off between the amount of computation and the quality of the solution is often necessary. Moreover, minimizing the overhead of communication required to converge to a desirable global solution is desirable.

Decentralized coordination algorithms may be the means to obtain scalability for task allocation in the context of large-scale open fleets. Here, each self-concerned (vehicle, driver, or courier) agent aims at achieving a desired local objective based on a limited local information and by communicating with the rest of the fleet and interacting with the environment. Due to the limited local information, one of the drawbacks of decentralization is lack of control of the emerging fleet behaviour that cannot be predicted with certainty. Moreover, to facilitate cooperation, assuming individually rational agents, we have to consider efficiency and fairness. How to balance decentralization and centralization to improve system performance is much investigated but still not a completely solved question.

1.1. Contribution. In this work, we present a survey on multiagent system (MAS) coordination mechanisms for computationally complex dynamic (one-on-one) task allocation problem (DTAP) and its variations for open vehicle fleet applications. These problems may be modelled by a variety of deterministic and dynamic two-dimensional linear assignment problems, i.e., the problems regarding the assignment of two sets that may be referred to as "agents" and "tasks" with at most one task per agent and one agent per task, where the tasks appear dynamically and the task assignment is fully determined by the (cost, profit, or revenue) parameter values and the initial conditions. We extend mathematical models of the variations of the static task assignment problem to their dynamic counterparts in open vehicle fleet scenarios considering, among others, self-interested and individually rational vehicle drivers, time restrictions, fairness, agent qualification, and personal rank.

We identify some of the main scalable solution methods, i.e., coordination mechanisms, that can be put at work to solve these problems. We investigate the theoretical scalability of these approaches and introduce a taxonomy to classify them in terms of the level of interdependence in decision-making available to individual vehicles and customers during the coordination process (centralized, distributed, and decentralized coordination). Our intention here is not to perform an exhaustive search nor to identify the most scalable solution procedure. Contrarily, we identify and mathematically model the variations of the dynamic task assignment problem applicable to the studied fleet task allocation contexts and provide general scalability characteristics of their solution approaches. Our intention is to make it easier for a researcher to solve some variation of the task allocation problem in large-scale open vehicle fleets by describing state-of-the-art solutions and their theoretical scalability results.

Even though some works exist that include reviews of the state of the art in multiagent-task allocation (see, e.g., [2–6]) and in vehicle fleet coordination (see, e.g., [7–9]) or ride-sharing optimization (see, e.g., [10, 11]), none of them addresses one-on-one dynamic task assignment problems in open vehicle fleets. In addition, a few approaches apply methods of multiagent-task allocation to the field of vehicle fleet coordination (see, e.g., [12]) but, to the best of our knowledge, there is no systematic survey combining both fields.

The paper is organized as follows. In Section 2, we discuss some relevant concepts in the context of coordination for dynamic task allocation in open systems with the focus on distribution and decentralization of decision-making. In Section 3, we present mathematical models of various static and dynamic task assignment problems applicable in the open vehicle fleet context. Centralized, distributed, and decentralized state-of-the-art solution methods and mechanisms for the problems presented in Section 3 are discussed in Section 4. We conclude the paper emphasizing open issues and challenges for possible future research directions in Section 5.

2. Coordination in Open Vehicle Fleets

In this section, we introduce some key concepts and characteristics of the target domains related to decentralizing coordination for scalable and dynamic task allocation. The coordination problem arises due to the distributed nature of the control exercised by the fleet's vehicles.

Generally, coordination may be defined as "the process of organizing people or groups so that they work together properly and well" (https://www.merriam-webster.com/dictionary/coordination). By the coordination in open vehicle fleets for task allocation, we refer to the organization and management of decision-making within the fleet with the aim to improve given key performance indicators of a fleet's task allocation.

The topics of coordination and task allocation are the object of studies in multiple disciplines, e.g., operations research, economics, and computer science. The corresponding definitions and related concepts may vary based on the specific discipline at hand. In the so-called field of coordination models and languages, for instance, the focus is on the general-purpose abstractions (so-called *coordination*

media) that can be generally used to model and engineer the patterns of interaction between computational agents—with no specific reference to a particular application scenario or coordination problem. In our survey, and in the following, we focus on the specific issues of dynamic task allocation and distributed/decentralized coordination, with a particular emphasis on open vehicle fleets.

2.1. Fleet Coordination. We consider the context with cooperative vehicles in a large vehicle fleet, which functions as an organization that constrains the cooperation schemes within it. The coordination problem here can be tackled from a bottom-up point of view, considering the emergence of global properties from the interfleet direct vehicle-to-vehicle communication and fleet-environment interaction.

For simplicity and without loss of generality, we consider a two-dimensional space in which tasks may appear randomly at any location in space and time while the vehicles circulate through a transportation network within the space to reach them. Each vehicle can have three states: *idle*, in which a vehicle is waiting for the assignment of a task, *assigned* in which a vehicle is assigned to a task but has still not reached the task, and *assisting* in which the vehicle has reached its assigned task and is assisting it. Only idle and assigned vehicles can be assigned or reassigned from one task to another. Once assigned, the vehicles start moving towards their assigned task. A task is considered completed once when it is reached and assisted by a vehicle.

Given a dynamically changing set (fleet) of idle and assigned vehicles, a dynamically changing set of randomly appearing tasks, and a cost function of the assignment of each task to every idle and assigned vehicle (e.g., the distance or time traveled or a given execution cost), the objective is to dynamically assign these vehicles to tasks in a given time horizon reaching a globally minimum cost assignment considering that each task must be performed by exactly one vehicle.

Coordinating the vehicles in this respect requires that they find the globally best allocation in a distributed or decentralized way and resolve conflicts that violate local constraints. An efficient strategy in this context is a dynamic (re-)assignment of the vehicles in the fleet to the tasks as they appear. The vehicles require continuous communication and processing for task allocation. The coordination system must ensure a balanced use of shared resources, e.g., vehicle-to-cloud (V2C) communication bandwidth and vehicle processing capacities.

V2C communication is limited in bandwidth and latency, so is the vehicle processing capacity. Coordination strategies that ignore these communication and computation constraints may fail to find a fleet's action plan in close to real time and thus may be inapt for the application in real-time fleets (see, e.g., [13]). These fleets require both autonomous and collaborative behaviours since vehicles have localized viewpoints, knowledge, and control and lack the overview of the global data integrated from various locations beyond their local capabilities. Such a dynamic context requires for coordination fault detection that indicates if the coordination exists within the fleet (see, e.g., [14]). Once a coordination fault is detected, a coordination recovery process can begin in which cooperation can be rebuilt.

Vehicle fleets that rely on one-on-one vehicle task assignment are, for example, rescue fleets (see, e.g., [15]), ride-hailing and taxi service (see, e.g., [16]), ambulance assistance of urgent out-of-hospital patients (see, e.g., [17]), and home-delivered restaurant hot meal services (see, e.g., [18]). Ride-hailing and restaurant hot meal delivery services are examples of open vehicle fleets that use online on-demand service platforms (see, e.g., [19]) to allocate in real-time customers and independent private vehicle owners, drivers, or couriers, using their personal vehicles. These platforms usually exploit sensor and GPS data to track the delivery process in real time [20].

Our focus is on the dynamic scenario with nonrecurring prearranged and spontaneously requested single-rider (customer), single-driver trips with at most one pickup and delivery for each rider and driver. Dynamically appearing riders (customers) should be allocated to drivers in a one-on-one manner. Before the allocation, in ride-hailing, a customer chooses the driver based on the time of arrival and the price of the ride. In case of hot meal delivery, the system gives an estimated delivery time to the customer and assigns a courier that meets such an estimate.

Coordination here is the key issue, including the stages of communication, resource allocation, and agreement. The allocation of the dynamically appearing customers over time needs to be performed in real time and it fails if not completed within a specified deadline relative to an arrival of a customer; deadlines must always be met, regardless of the system load. Conventionally, the matching is based just on the rider's personal preferences and the nearby drivers' availabilities. Reallocation of already matched drivers to riders that are awaiting the service is not possible even if a more efficient matching exists. At the end of each trip, every driver is available for a new rider allocation.

Speedy meal delivery services are constrained in geographic availability and timing. Usually, restaurants, riders, and customers have access to the system through an app. A customer detects his/her location and displays restaurants that participate in the platform in the region of interest and are open at the time. Couriers participate in this open fleet context by delivering whenever they choose and they may get paid on the individual delivery basis. Once a customer requests a meal from a restaurant via his/her app, the corresponding delivery is assigned to a courier available nearby. The courier picks up the delivery from the restaurant and delivers it to the customer. After the delivery, a courier is available for new deliveries.

The allocation of a courier to the customer is conventionally done based on the shortest arrival time to the restaurant (first-come-first-served strategy) and the availability of the courier; reallocation is not possible once the courier is allocated. The challenge here is to assign couriers to dynamically appearing pickups and deliveries in order to maximize customer satisfaction (which can be measured in different ways, as explored in [20]) without violating delivery times agreed at the time of the customer's hot meal request.

Task allocation problem in open vehicle fleets considers both providers of transportation services (vehicle drivers) and their customers and thus both of them may be considered active participants in the transportation process. In the ride-hailing scenario, drivers are usually modelled as agents and riders as tasks, while in the hot meal delivery scenario, couriers are agents while meal deliveries are tasks.

Even though the ownership of most of the open fleet systems today is centralized, not only customers but also drivers with vehicles may appear dynamically and spontaneously in time and space influenced by a variety of factors unknown in advance such that it is reasonable to assume that they appear randomly. In this *dynamic task allocation* context, available vehicles are assigned to pending customers as they appear. Each agent and task is assumed to be characterized by a set of attributes that influences the cost or profit resulting from an agent-task allocation. In this way, the *task allocation* problem that assigns tasks to agents in time is simplified to *task assignment* problem focusing on the one agent-one task allocation at the time (see, e.g., [17, 21]). Optimized and dynamic task (re-)assignment may considerably improve the performance of the fleet while considering individual fairness and efficiency (see, e.g., [21]). If dynamic courier (rider) reallocation is allowed, a substantial increase in efficiency may be observed, as in the case of ambulance allocation to out-of-hospital patients (see, e.g., [8, 21, 22]).

2.2. Coordination Models for Open Vehicle Fleets. Based on the ownership of the fleet, its structure, and the level of decentralizing coordination that we want to achieve in the fleet task allocation, we can design the following models:

A *centralized coordination model*, where the task allocation problem is solved in a single block by only one decision-maker (e.g., a single enterprise) having total control over and complete information about the vehicle fleet.

A *distributed coordination model*, where the global task allocation problem is decomposed such that each customer is represented by an autonomous decision-maker (agent) that may solve its own subproblem only with its own local decision variables and parameters. The allocation of a limited number of vehicles (global constraints) is done through the interaction between competing customer agents and a vehicle fleet owner (a single autonomous agent) having available all the fleet information. Customer agents that compete for the resources are not willing to disclose their complete information but will share a part of it if it facilitates achieving their local objectives. The vehicle fleet owner agent is responsible for achieving globally efficient resource allocation by interacting with customer agents usually through an auction. The problem decomposition here is done to gain computational efficiency since customer agents can compute their bids in parallel. However, the resource allocation decisions are still made by a single decision-maker (vehicle fleet owner) with the requirement on synchronous bidding of customer agents (see, e.g. [23–25]).

A *decentralized coordination model*, which further decentralizes the distributed model by allowing for multiple resource owner (vehicle) agents, multiple competing customer agents requesting the transportation service, and asynchrony in decision-making. Customer agents compete for fleet's vehicles held by multiple resource owners while each customer and resource owner agent has access only to its local information with no global information available. Therefore, they must negotiate resource allocation by running localized algorithms while exchanging relevant (possibly obsolete) information. Localized algorithms make the achievement of a desired global objective easier through simple local interactions of agents with their environment and other agents, with no need for a central decision-maker. The decisions specifying these interactions emerge from local information. Fairness in resource allocation here plays a major role. The same as in the distributed model, an agent is not willing to disclose its complete information but will share a part of it if it facilitates achieving its local objective. Resource allocation here is achieved by the means of a decentralized protocol.

Generally speaking, coordination is distributed when complex behaviour within a system does not emerge due to the control of the system owner, but through interactions and communication of individual agents operating on local information, while sharing globally relevant knowledge. This form of control is typically known as *distributed control*, that is, control where each agent is equally responsible for contributing to the global, complex behaviour by acting properly on local information. Agents are implicitly aware of the interaction rules through mechanisms that are based on the agent's interaction with other agents and the environment. The system behaviour is then an emergent property of distributed coordination mechanisms (algorithms) that act upon agents, rather than the result of a control mechanism of a centralized system owner. In decentralized algorithms, no global clock is assumed, no agent has complete information about the systems' state, every agent takes decisions based only on local information, and failure of one agent does not prevent the system to continue running. An example is Bitcoin: Instead of one central server owned and operated by a single entity, Bitcoin's ledger is distributed across the globe making it impossible to shut down, break in, or hack as there is no single central bottleneck of the system.

Let us notice the main difference between distributed and decentralized coordination models. Distributed coordination relies on local and shared (global) parameters and variables. Local parameters and variables are private, whereas shared and global parameters and variables need to be shared among two or more agents—even among all the agents of the system. If we assume self-concerned agents, resource owner can manipulate these parameters and variables or deceive agents in communicating their values to influence the individual decision-making of each one of

them and thus obtain the behaviour of the system the resource owner wants. This can be prevented by ensuring individual agent access to nonobsolete and truthful information—using, e.g., blockchain technology. Reaching a globally optimal solution with quality of solution guarantees is then possible, contrary to the decentralized coordination case. In the latter case, due to the lack of the global nonobsolete and truthful information, quality of solution guarantees generally do not exist. In general, solution approaches for decentralized coordination concentrate on finding a feasible (admissible) solution without quality of solution guarantees. Contrary to the distributed case most often studied in the operations research field where the emphasis is on the method's optimality gap, decentralized coordination methods are mostly approximate heuristics-based methods without quality of solution guarantees but with proven completeness, soundness, and termination.

Open vehicle fleets are intrinsically distributed systems since they comprise a multitude of geographically distributed and mutually communicating customers' and vehicle drivers' apps. Traditionally, distributed systems refer to systems consisting of sequential processes (each one with an independent thread of control, possibly located on geographically distributed processors) that coordinate their actions by exchanging messages to meet a common goal (see, e.g., [26, 27]). The common goal in this context is an efficient and cost-effective transportation service of the vehicle fleet while considering individual rationality, preferences, and constraints whether it is of drivers, riders, or hot meal delivery customers. Quality of solution guarantees play a crucial role of sustainable competitive advantage in any transportation network company.

Distributed open vehicle fleets exhibit some clear strong points over their centralized counterparts. First of all, they are more robust than their centralized counterparts because they can rely on their intrinsic built-in redundancy. They can operate at a larger scale and assist more customers at once since they are aggregating vehicle capacity and customer throughput across all their individual vehicle drivers. However, distributed open vehicle fleets also have to deal with intervehicle communication and coordination overhead that can sometimes make them slower or more difficult to control than their centralized counterparts. Applying trustless distributed systems that are meant to operate in an adversarial environment, such as Bitcoin, in open fleets entails an additional overhead.

3. Task Assignment Models for Open Vehicle Fleets

Assignment problems (APs) are among the earliest optimization problems studied in the operations research field. They involve optimally matching the elements of two or more sets, where the dimension of the problem refers to the number of sets to be matched [28]. For example, in two-dimensional assignment problems, given is a set of agents A and a set of tasks T and we have to match (assign) tasks to agents. Tasks are assumed atomic, i.e., each task cannot be decomposed into subtasks and it can be completed by a single vehicle. In general, two-dimensional assignment problems can be solved in polynomial time, while d-dimensional assignment problems, with $d > 2$, in general are NP-hard (see, e.g., [29]).

We distinguish between the static and dynamic assignment problems (see, e.g., [30]). The former refers to the assignment of a set of tasks to a set of agents in a given static environment in which the problem data does not change during the planning horizon, while in the dynamic task assignment problems, both agents and tasks may appear and disappear dynamically over time. In the open vehicle fleet setting, agents can be in one of the following three states: *idle*, *assigned* without still having reached the customer, or *assisting a customer*, and only idle and assigned agents that have still not reached their customers can be (re)assigned to unassisted tasks. In general, agents are assumed renewable, i.e., after completing a task, an agent's state changes from *assisting a customer* to *idle* and it becomes assignable again to customers (tasks) that have not been assisted yet. This is a special case of a more general computationally complex dynamic vehicle routing problem (DVRP) in which, for each (vehicle) agent, we find a minimum cost route that visits a dynamically changing set of tasks (customers) [31]. Due to the high computational complexity, myopic algorithms are the most usual solution approaches for DVRP. For simplicity, we can assume that agents are nonrenewable, i.e., an agent can be assigned only to one task; if, after completing a task, it is still available for new task assignment, it appears as a new agent.

The static and deterministic AP is a computationally easy problem, which allows us (in theory) to find an optimal solution in close to real time (in the nonrenewable agent case). Dynamic AP can be solved by (suboptimal) myopic approaches that consider only the information available at the present time with no consideration for future events and possibly reassign tasks among idle and already assigned agents to improve the system's efficiency (see, e.g., [8, 17, 21, 22]). However, in the case where tasks are not randomly appearing, this approach can be significantly improved by considering future developments.

3.1. Static Task Assignment. Based on the categorization of the AP models presented in [28], in this section, we consider the classic assignment problem and its variations relevant in the open fleet vehicle task assignment considering self-interested and individually rational vehicle users whose tasks can be performed simultaneously: the classic linear assignment problem (LAP), assignment problem recognizing agent qualification (APRAQ), the bottleneck assignment problem (BAP), the fair matching problem (FMP), the minimum deviation assignment problem (MDAP), the Σ_k-assignment problem (Σ_k-AP), the semiassignment problem (SAP), and the assignment problem with side constraints (APSC). In Figure 1, we give a framework for easier understanding of the characteristics of both the static and dynamic version of these problems.

For self-completeness of this article, we bring in the following the descriptions of these problems. Considering that the number of publications concerning assignment problems is

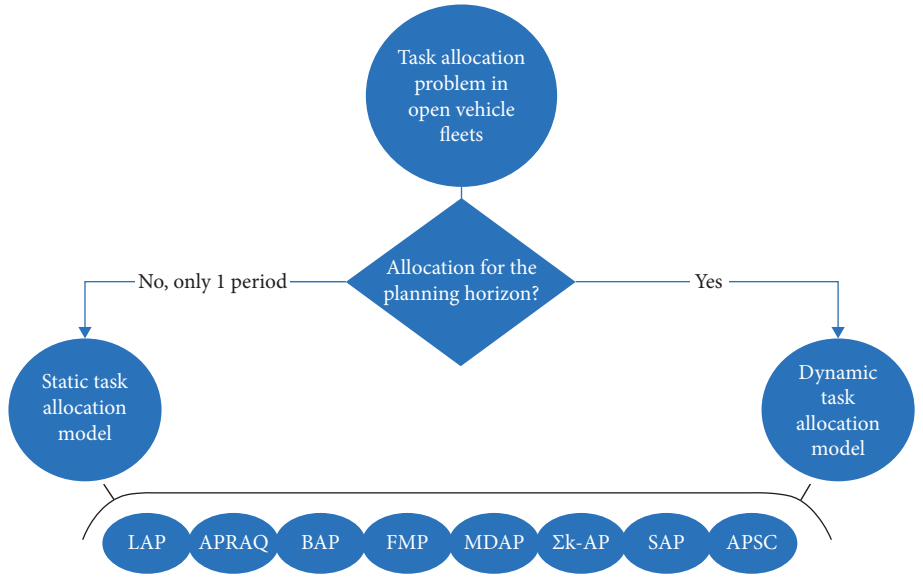

Figure 1: Static and dynamic task assignment problems in open vehicle fleets.

enormous, the references in this section constitute only a very limited part of them. For the details and other assignment problem variations, the reader is referred to [28].

3.1.1. Classic (Linear) Assignment Problem (LAP). The static classic linear assignment problem involves two sets of the same size and consists of finding, in a weighted complete bipartite graph, a perfect matching in which the sum of weights of the matched edges is as low as possible, i.e., a *minimum-weight perfect matching*. Perfect weighted matching implies that each node must be matched to some other node by minimizing the total cost of the arcs in the (perfect) matching.

The classic linear assignment problem (LAP) can be defined as follows: given a weighted complete bipartite graph $G = (A \cup T, E)$ with two vertex sets A and T, with $n = |A| = |T|$, and an edge set $E = A \times T$, with edge weights c_{ij} on edge $(i, j) \in E$, find a minimum-weight perfect matching of G, i.e., a perfect matching among vertices in A and vertices in T such that the sum of the costs of the matched edges is minimum. An edge $(i, j) \in E$ is matched if two extreme vertices i and j are mutually matched, and a matching is perfect if every vertex i of A is matched (assigned) exactly to one vertex j of T, and vice versa. The LAP is equivalent to the weighted bipartite matching, since we may assume that the bipartite graph is always complete by letting the weights of the edges that are missing being sufficiently large. If $|A| \neq |T|$, we can add a number of dummy nodes to the set with lower cardinality and connect them by dummy arcs of zero cost to the other set. The number of dummy nodes should be sufficient to balance the cardinalities of the two sets.

The LAP is equivalent to the maximum weighted bipartite matching (with edge weights $w_{ij} \geq 0$), since we may assume that the bipartite graph is always complete by letting the weights of the edges that are missing being sufficiently large. Furthermore, also in this case, we can assume that the two vertex sets of the bipartite graph have the same size. At this point, we can reformulate the problem as a minimization problem by considering costs $c_{ij} = W - w_{ij}$, where W is larger than the maximum of the w_{ij}, and hence, this problem corresponds to the LAP.

The LAP is a special case of the transportation problem assuming an equal number of supplier agents and customer agents and each one with their unitary supply and unitary demand, respectively. The transportation problem is one of the special cases of the minimum cost flow problem together with, e.g., the shortest path problem and the max flow problem. While it is possible to solve this problem using the simplex algorithm, specialized algorithms take advantage of its special network structure and are thus more efficient.

From the multiagent systems' point of view, in the assignment problem, a number of agents need to be assigned to a number of tasks based on the given cost of agent-task assignment. In general, each agent can be assigned to any task. In case an agent is not capable of performing a task, a given agent-task assignment cost is modelled as a very large number. All tasks should be performed with the objective to minimize the total cost of the assignment such that exactly one agent is assigned to each task and exactly one task to each agent. The mathematical formulation of the problem is as follows:

$$\min \sum_{i,j} c_{ij} x_{ij}, \tag{1}$$

subject to

$$\sum_{i=1}^{n} x_{ij} = 1, \quad \forall j \in T, \tag{2}$$

$$\sum_{j=1}^{n} x_{ij} = 1, \quad \forall i \in A, \tag{3}$$

$$x_{ij} \in \{0,1\}, \quad \forall i \in A, j \in T. \tag{4}$$

Constraints (2) ensure that every task is assigned to only one agent and constraints (3) ensure that every agent is assigned to only one task.

The structure of the problem, i.e., the total unimodularity of the constraint matrix, makes the binary requirements on the variables unnecessary. In fact, in this case, it can be proven that the linear relaxation has always an optimal binary solution (see, e.g., [32, 33]) and, therefore, the LAP is a linear programming (LP) problem.

3.1.2. The Classic Assignment Problem Recognizing Agent Qualification (APRAQ). Caron et al. in [34] propose a mathematical model in which not every agent is qualified to do every task, and the objective is utility maximization:

$$\max \sum_{i,j} p_{ij} x_{ij}, \tag{5}$$

subject to

$$\sum_{i \in A} q_{ij} x_{ij} \leq 1, \quad \forall j \in T, \tag{6}$$

$$\sum_{j \in T} q_{ij} x_{ij} \leq 1, \quad \forall i \in A, \tag{7}$$

$$x_{ij} \in \{0,1\}, \quad \forall i \in A, j \in T, \tag{8}$$

where parameter $q_{ij} = 1$ if agent i is qualified to perform task j, 0 otherwise, parameter p_{ij} is the utility of assigning agent i to task j (with $p_{ij} = 0$ if $q_{ij} = 0$), and variable $x_{ij} = 1$ if agent i is assigned to task j, 0 otherwise. Constraints (6) ensure that no more than one qualified agent is assigned to any task, while constraints (7) guarantee that each agent is assigned to not more than one task.

The classic assignment problem does not consider fairness. The solution of classic AP (1)–(4) maximizes utilitarian social welfare (see, e.g., [35]), but it may be unfair and unsatisfactory since there may be one or more agents with a much higher task cost than the rest. This is why it is best applied to centralized open vehicle fleets with a single owner of the fleet's vehicles that is interested in the minimization of the overall cost of the fleet's operation costs but not in how they are distributed among the vehicles.

3.1.3. Bottleneck Assignment Problem (BAP). To resolve the issues with fairness and workload distribution, we may minimize maximum cost among the individual agent-task assignments and thus maximize the system's egalitarian social welfare (see, e.g., [36]). The mathematical program for the BAP is as follows: minimize $\max_{i,j}\{c_{ij} x_{ij}\}$ or minimize $\max_{i,j}\{c_{ij} \mid x_{ij} = 1\}$ subject to constraints (2)–(4) and definitions of the LAP.

Note that here the integrality requirements cannot be relaxed. Contrary to the classic AP model, the BAP model pursues the objective of fairness among agents. It is based on the optimization of the worst-off performance and provides a good solution when the minimum requirements of all agents should be satisfied. However, only the most costly agent-task assignment influences the objective function, while the contribution of the rest of the agents is ignored. For this reason, this approach deteriorates the system efficiency and thus the system's utilitarian social welfare.

3.1.4. The Fair Matching Problem (FMP). The fair matching problem minimizes the difference between the maximum and minimum assignment values [37]: minimize $\max_{i,j}\{c_{ij} \mid x_{ij} = 1\} - \min_{i,j}\{c_{ij} \mid x_{ij} = 1\}$ subject to the same constraints and definitions as in the classic AP.

This formulation of fairness is not unique. Sun and Yang in [38] study the concept of fair and optimal allocations. They define an allocation to be fair and optimal if it is envy-free and the sum of compensations is maximized, subject to the compensation assigned to each object is less than or equal to the maximum compensation limit. They prove that fair and optimal allocations exist and demonstrate that the fair and optimal allocation mechanism achieves efficiency, fairness, and strategy-proofness simultaneously. Andersson [39] demonstrates that it is also coalitionally strategy-proof, i.e., it is not possible for any agent or any coalition of agents to successfully manipulate the allocation rule.

3.1.5. The Minimum Deviation Assignment Problem (MDAP). The objective here is to minimize the difference between the maximum and average assignment costs:

$$\text{minimize } \min\{n,m\} \times \max_{p,q}\{c_{pq} x_{pq}\} - \sum_{i=1}^{n}\sum_{j=1}^{m} c_{ij} x_{ij}, \tag{9}$$

or to minimize the difference between the average and minimum assignment profit:

$$\text{minimize} \sum_{i=1}^{n}\sum_{j=1}^{m} p_{ij} x_{ij} - \min\{n,m\} \times \min_{s,t}\{p_{st} x_{st}\}, \tag{10}$$

subject to constraints (2)–(4). Here, n is the cardinality of agent set A, and m of task set T, and other definitions are the same as in the LAP [40, 41].

3.1.6. The Σ_k-Assignment Problem (Σ_k-AP). Since there may be generally multiple different sets of assignments with the same minimum value for $\max\{c_{ij} x_{ij}\}$, the objective here is to find a set of assignments for which the sum of the k largest values is minimized. The BAP and LAP can be viewed as special cases of Σ_k-AP with $k = 1$ and $k = n$, respectively.

A recent study on generic mixed integer problem with Σ_k optimization is done by Filippi et al. [42].

3.1.7. The Semiassignment Problem (SAP). This is the version of the assignment problem where every agent or task may not be unique. This results in a constraint matrix containing a number of rows or columns with equal coefficients. Kennington and Wang in [43] show examples of such a problem in workforce and project planning and

scheduling as use case examples. Here, constraints (2) from the classic LAP are substituted by

$$\sum_{i=1}^{m} x_{ij} = d_j, \quad \forall j, \tag{11}$$

everything else being the same as in the classic LAP for the situation in which there are n agents and m task categories. Here, $m \leq n$, and d_j is the number of tasks in task group j with $\sum_j d_j = n$.

Note that if also the agents are not unique and are clustered into agent groups, with q_i agents in each group i, where $\sum_j d_j = \sum_i q_i$, the problem is equivalent to the transportation problem.

3.1.8. The Assignment Problem with Side Constraints (APSC). Classic assignment problem can be solved by multiple centralized and efficient polynomial algorithms. However, by introducing side constraints, generally, this problem becomes NP-hard. Side constraints may include budgetary limitations, degree of technical training of personnel, the rank of personnel, or time restrictions that limit the assignment of agents to tasks.

Aggarval [44] introduces to the classical LAP problem an additional knapsack-type constraint:

$$\sum_{i,j} r_{ij} x_{ij} \leq b, \tag{12}$$

where r_{ij} is the amount of resource used if agent i is assigned to task j and b is the amount of a resource available. Adding constraint (12) to LAP results in a resource-constrained assignment problem (RCAP), which is a knapsack problem under perfect matching over a bipartite network. Constraint (12) deranges the unimodularity of the LAP set of constraints so that the optimal solution of the linear relaxation of the problem is no more always within the values $\{0, 1\}$ and, hence, integrality constraints cannot be relaxed. The resulting problem belongs to the class of NP-complete problems for which no polynomially bounded algorithm is likely to exist (see, e.g., [44]).

Mazzola and Neebe [45] present a general model for the assignment problem with side constraints that generalizes the general assignment problem (GAP) (see, e.g., [46]) and adds the following constraints to either the classic LAP model or the classic LAP recognizing agent qualifications:

$$\sum_{i,j} r_{ijk} x_{ij} \leq b_k, \quad \forall k, \tag{13}$$

where r_{ijk} is the amount of resource k used if agent i is assigned to task j and b_k is the amount of resource k available.

By side constraints, we can model drivers that belong to different seniority classes and customers that have different priority levels. Seniority constraints impose for the solution to be such that no unassigned agent can be assigned to a task unless an assigned agent with the same or higher seniority becomes unassigned, while priority constraints specify that the solution must be such that no unassigned task can become assigned without a task with the same or higher priority becoming unassigned [34].

3.2. Dynamic Task Assignment. In this section, we propose extensions of the static assignment problem models presented previously to the dynamic versions in which new agents and tasks may enter the system in each time period and the costs or profits of agent-task assignment are updated in (close to) real time. This problem is similar to the online bipartite matching problem, in which tasks that appear in sequence should be assigned to the agents immediately as they appear. Relating to the previously presented terminology of the static AP, a set of available (idle and assigned) agents A (that are not assisting any customer) is known in the given weighted bipartite graph $G = (A \cup T, E)$. Tasks in T (along with their incident edges) arrive online. Upon the arrival of a task $j \in T$, we must assign it to one of agents $i \in A$ with an existing edge $(i, j) \in E$. At all times, the set of matched edges must form a (feasible) matching, i.e., each agent should be matched with at most one task and vice versa. In case of different cardinalities of the two sets, to balance the two, dummy elements are added to the set with lower cardinality.

We assume random arrivals of customer demands (tasks) over time. In open fleets, we also assume that agents (drivers and couriers) either become available randomly after assisting previous tasks (customers) or by entering and leaving the fleet based on personal interest, available time, and/or other individual constraints and preferences. Given are attribute parameters both for agents and tasks that define their main characteristics in terms of the assignment.

We consider deterministic on-demand task allocation where the (re-)assignment of vehicles (agents) to tasks is performed as soon as a new vehicle or task enters the system. Close to real-time reassignment is beneficial here since the parameters and variables of the assignment problem are perfectly known.

Spivez and Powell [30] propose a Markov decision process model for the dynamic assignment problem. In this paper, inspired by their work, we propose mathematical programming models for the variations of the static task assignment described in the previous section while respecting agent-task taxonomy used previously in this paper.

The decisional variables in the dynamic AP receive a third index such that

$$x_{ij\tau} = \begin{cases} 1, & \text{if task } j \in T \text{ is assigned to agent } i \in A \text{ at period } \tau \in \mathcal{T}, \\ 0, & \text{otherwise.} \end{cases} \tag{14}$$

Moreover, we introduce two additional binary variables $\alpha_{\tau i}$ and $\beta_{\tau j}$, for all $i \in A$, $j \in T$ defined as follows:

$$\alpha_{i\tau} = \begin{cases} 1, & \text{if agent } i \in A \text{ is known and available for assignment in period } \tau, \\ 0, & \text{otherwise,} \end{cases}$$
$$\beta_{j\tau} = \begin{cases} 1, & \text{if task } j \in T \text{ is known and available for assignment in period } \tau, \\ 0, & \text{otherwise.} \end{cases} \tag{15}$$

Let $\mathcal{T}$ be a set of consecutive time periods of the planning time horizon. The mathematical formulation of the deterministic and dynamic LAP problem considering utility maximization is then given by

$$Z = \max \sum_{\tau \in \mathcal{T}} \sum_{i \in A} \sum_{j \in T} p_{ij\tau} x_{ij\tau}, \tag{16}$$

subject to

$$\sum_{j \in T} x_{ij\tau} \le \alpha_{i\tau}, \quad \forall i, \tau \tag{17}$$

$$\sum_{i \in A} x_{ij\tau} \le \beta_{j\tau}, \quad \forall j, \tau, \tag{18}$$

$$\alpha_{i,\tau+1} = \alpha_{i\tau} - \sum_{j \in T} x_{ij\tau} + \widehat{A}_{i,\tau+1}, \quad \forall i, \forall \tau \in \{1, \ldots, |\mathcal{T}| - 1\}, \tag{19}$$

$$\beta_{j,\tau+1} = \beta_{j\tau} - \sum_{i \in A} x_{ij\tau} + \widehat{T}_{j,\tau+1}, \quad \forall j, \forall \tau \in \{1, \ldots, |\mathcal{T}| - 1\}, \tag{20}$$

$$\alpha_{i,1} = \widehat{A}_{i,1}, \quad \forall i, \tag{21}$$

$$\beta_{j,1} = \widehat{T}_{j,1}, \quad \forall j, \tag{22}$$

$$x_{ij\tau} \in \{0, 1\}, \quad \forall i \in A, j \in T, \tau \in \mathcal{T}, \tag{23}$$

$$\alpha_{i\tau} \in \{0, 1\}, \quad \forall i \in A, \tau \in \mathcal{T}, \tag{24}$$

$$\beta_{j\tau} \in \{0, 1\}, \quad \forall j \in T, \tau \in \mathcal{T}, \tag{25}$$

where $p_{ij\tau}$ is the utility of assigning agent i to task j at period τ (note that it may vary through time) and $\widehat{A}$ and $\widehat{T}$ are given parameters such that

$$\widehat{A}_{i\tau} = \begin{cases} 1, & \text{if agent } i \in A \text{ enters into set } A \text{ (the fleet) in period } \tau, \\ 0, & \text{otherwise.} \end{cases}$$
$$\widehat{T}_{j\tau} = \begin{cases} 1, & \text{if task } j \in T \text{ becomes known in period } \tau, \\ 0, & \text{otherwise.} \end{cases} \tag{26}$$

Moreover, based on the assumption of nonrenewable agents and tasks, we assume that $\sum_{\tau \in \mathcal{T}} \widehat{A}_{i\tau} \le 1$ and $\sum_{\tau \in \mathcal{T}} \widehat{T}_{j\tau} \le 1$, i.e., every agent and task are unique and enter into the fleet and thus become available for assignment only once.

The aim is maximizing the total utilitarian social welfare over the planning time horizon, which is achieved by maximizing the assignment utility (16) over all agent-task assignments in all periods of the planning time horizon. Constraints (17) guarantee that each available agent at time period τ is assigned to at most one task while unavailable agents cannot be assigned to any task. Constraints (18) ensure that at most one agent is assigned to any available task while no agent can be assigned to any unavailable task.

Constraints (19) and (20) represent the dynamics of dependent variables $\alpha_{\tau i}$ and $\beta_{\tau j}$, assuming that both agents and tasks disappear from the system at the end of the period when they are assigned. Furthermore, constraints (21) and (22) represent the initial conditions of the problem, while the variable ranges are given by (23)–(25).

We can also consider cost minimization problem where we substitute (16) with the following objective function:

$$Z = \min \sum_{\tau \in \mathcal{T}} \sum_{i \in A} \sum_{j \in T} c_{ij\tau} x_{ij\tau}, \tag{27}$$

subject to

$$\sum_{i \in A} \sum_{j \in T} \sum_{\tau \in \mathcal{T}} x_{ij\tau} = n, \tag{28}$$

and (17)–(25). Constraint (28) guarantees the assignment of all the tasks and/or agents in the planning time horizon, depending on the relative size of these two sets.

3.2.1. The Dynamic Classic Assignment Problem Recognizing Agent Qualification. Here, the objective function is again the utility maximization (16), while constraints (17) and (18) are substituted by the following ones, everything else remaining the same as in the dynamic LAP:

$$\sum_{j \in T} q_{ij\tau} x_{ij\tau} \le \alpha_{\tau i}, \quad \forall i, \tau, \tag{29}$$

$$\sum_{i \in A} q_{ij\tau} x_{ij\tau} \le \beta_{\tau j}, \quad \forall j, \tau, \tag{30}$$

where parameter $q_{ij\tau} = 1$ if agent i is qualified to perform task j at period τ, 0 otherwise, parameter $p_{ij\tau}$ is the utility of assigning agent i to task j at period τ (with $p_{ij\tau} = 0$ if $q_{ij\tau} = 0$), and variable $x_{ij\tau} = 1$ if agent i is assigned to task j at period τ, 0 otherwise. Constraints (29) guarantee that no more than one qualified agent is assigned to any task, while constraints (30) ensure that each agent is assigned to not more than one task. Instead of the profit maximization, here, we can introduce cost minimization by substituting (16) with (27) and introducing (28) into the constraint set.

3.2.2. The Dynamic Bottleneck Assignment Problem (DBAP). The objective function of the DBAP problem can be formulated as follows: at each period $\tau \in \mathcal{T}$, maximize

$Z = \min_{i,j}\{p_{ij\tau}x_{ij\tau}\}$ or maximize $Z = \min_{i,j}\{p_{ij\tau} \mid x_{ij\tau} = 1\}$. This maxmin problem can be expressed by maximizing an additional variable L that is a lower bound for each of the individual values $\{p_{ij\tau} \mid x_{ij\tau} = 1\}$ as follows: max L subject to constraints $L \leq \sum_{j\in T} p_{ij\tau}x_{ij\tau}$ for all $i \in A_\tau$, $\tau \in \mathcal{T}$, and (17)–(25) and definitions of the dynamic LAP.

3.2.3. The Dynamic Fair Matching Problem (DFMP). Here, at each period $\tau \in \mathcal{T}$, we minimize the objective function $\max_{i,j}\{c_{ij\tau} \mid x_{ij\tau} = 1\} - \min_{i,j}\{c_{ij\tau} \mid x_{ij\tau} = 1\}$ and subject to constraints (17)–(25). Similarly, we can minimize the difference between the maximum and minimum profit obtained among agents, i.e., minimize $(\max_{i,j}\{p_{ij\tau} \mid x_{ij\tau} = 1\} - \min_{i,j}\{p_{ij\tau} \mid x_{ij\tau} = 1\}$ and subject to constraints (17)–(25).

3.2.4. The Dynamic Minimum Deviation Assignment Problem (DMDAP). At each period $\tau \in \mathcal{T}$, the objective function is as follows:

$$\text{minimize} \quad \min\{n, m\} \times \max_{p,q}\{c_{pq\tau}x_{pq}\} - \sum_{i\in A}\sum_{j\in T} c_{ij\tau}x_{ij\tau}, \tag{31}$$

or

$$\text{minimize} \quad \sum_{i=1}^{n}\sum_{j=1}^{m} p_{ij}x_{ij} - \min\{n, m\} \times \min_{s,t}\{p_{st}x_{st}\}, \tag{32}$$

subject to constraints (17)–(25) and definitions of the minimum deviation assignment problem.

3.2.5. The Dynamic Σ_k-Assignment Problem ($D\Sigma_k$-AP). Given parameter k, objective function (16) is modified to

$$Z = \max \sum_{\tau\in\mathcal{T}}\sum_{i=1}^{k}\sum_{j\in T} p_{ij\tau}x_{ij\tau}, \tag{33}$$

subject to constraints (17)–(25) and definitions of the dynamic LAP.

3.2.6. The Semiassignment Problem. Here, constraints (18) from the dynamic LAP are substituted by

$$\sum_{i=1}^{m} x_{ij\tau} = d_j\beta_{j\tau}, \quad \forall j, \tau, \tag{34}$$

everything else being the same as in the dynamic LAP for the situation in which there are n agents and m task categories, where $m \leq n$.

3.2.7. The Assignment Problem with Side Constraints. Side constraints (13) here include also the time index:

$$\sum_{j\in T} r_{ij\tau}x_{ij\tau} \leq b_{\tau i}, \quad \forall k, \tau, \tag{35}$$

where $r_{ijk\tau}$ is the amount of resource k used if agent i is assigned to task j at period τ and $b_{k\tau}$ is the amount of resource k available at period $\tau \in \mathcal{T}$. Constraints (35) are simply added to the formulation of the dynamic LAP.

3.3. Bottom Line. To sum up, in Table 1, we give the overview of the characteristics of the treated (static and dynamic) task assignment problems related to (i) the kind of the social welfare they optimize (utilitarian, egalitarian, elitist, or a difference between them), (ii) whether agents are qualified to perform only certain tasks or not, (iii) including fairness or not, (iv) whether the agents are considered homogeneous or not, (v) time restrictions, (vi) personal ranking, and (vii) technical training.

Note that once we introduce additional constraints to the classic assignment problem, the resulting model is, generally, no more resolvable in polynomial time and is highly computationally expensive. Additionally, *we consider tasks and agents that may be known both at some future time period and at the first period of the planning time horizon*. Therefore, we can use this model to coordinate task allocation for planned tasks and agents that schedule their appearance in advance for some future time period, but also for the tasks and agents that need to be allocated on short notice or immediately as they get known and enter the system. To this aim, we must use highly computationally efficient close to real-time solution approaches and, generally, exact methods do not suffice for this purpose. Therefore, we are obliged to use heuristic-based approximations.

4. Coordination Approaches in Task Allocation to Fleet's Vehicles

In this section, we recall the main (coordination) solution methods for the task allocation problem in open vehicle fleets in general and the treated assignment problems in particular, categorizing them in centralized, distributed, and decentralized (Figure 2), with special attention to those with the best time complexity. Recall that the static classic assignment problem consists in finding the minimum cost perfect matching of a complete bipartite graph $G = (A \cup T, E)$, with $E = A \times T$ and $n = |A| = |T|$.

4.1. Centralized Coordination Approaches. There are a huge number of algorithms for the linear assignment problem (LAP). They can be subdivided into *primal*, *dual*, and *primal-dual* algorithms. The worst-case time complexity of the best algorithms is $O(n^3)$.

We preliminary recall the mathematical formulation of the dual problem of the linear formulation of the LAP:

$$\max \sum_{i=1}^{n} u_i + \sum_{j=1}^{n} v_j, \tag{36}$$

subject to

$$u_i + v_j \leq c_{ij}, \forall i, j \in \{1, \ldots, n\}, \tag{37}$$

Table 1: Characteristics of the discussed task assignment models.

Model	Soc. welfare	Agent qualif.	Fairness	Unique ag./tasks	Time restr.	Pers. rank	Tech. train.
LAP	Util.	No	No	Yes	No	No	No
APRAQ	Util.	Yes	No	Yes	No	No	No
BAP	Egal.	No	No	Yes	No	No	No
FMP	El. – Eg.	No	No	Yes	No	No	No
MDAP	El. – ut.	No	No	Yes	No	No	No
	Ut. – el.	No	No	Yes	No	No	No
Σ_k-AP	Egal.	No	Yes	No	No	No	No
SAP	Util.	No	No	Yes	No	No	No
APSC	Util.	No	No	No	Yes	Yes	Yes

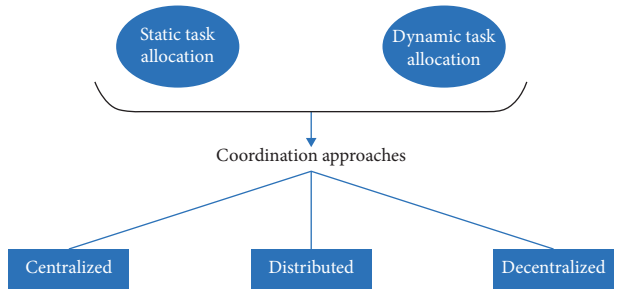

Figure 2: Coordination approach framework for task allocation.

where u_i and v_j are the (dual) variables.

4.1.1. Primal Algorithms. Primal algorithms are in general special implementations of the network simplex algorithm: one of the best primal algorithms is proposed in [47] and runs in $O(n^3)$ time.

4.1.2. Dual Algorithms. Dual algorithms are iterative algorithms which at each iteration maintain a feasible dual solution, and only at the final iteration, they come up with a primal solution (i.e., a feasible assignment). In this regard, also the primal-dual algorithms can be viewed as special dual algorithms. Typical dual algorithms are those based on successive shortest paths, signature, pseudoflow, interior point, and auction methods. In the following, we concentrate on the auction methods because from the latter, one can easily derive distributed versions of the same. For additional details, the reader is referred to [29, 36].

For a short survey on the above solution algorithms for the LAP, the reader is referred to a not so recent but detailed experimental comparison of some of the algorithms in [48]. Another survey on the state-of-the-art algorithms for the LAP is provided in [36].

4.1.3. Auction Algorithms. The first auction algorithm for the LAP was given by Bertsekas [49] and successively improved by Bertsekas and Eckstein [50] through a scaling technique providing an algorithm that runs in $O(n^3 \log(nC))$, where $C = \max\{|c_{ij}|\}$. A survey of iterative combinatorial auction algorithms for task allocation in multiagent systems can be found in, e.g., [4, 51–53].

The auction algorithm proposed by Bertsekas in [49] is an iterative algorithm that at each iteration maintains a triple $(x, (u, v))$ of primal and dual solutions that satisfy the complementary slackness conditions such that the dual solution is feasible. The algorithm terminates when also the corresponding primal solution is feasible. At each iteration, the dual solution is updated and the corresponding primal solution (with respect to complementary slackness conditions) is found.

In particular, given a dual vector v, the optimal (feasible) dual vector u can be obtained by considering $u_i = \min_j\{c_{ij} - v_j\}$, and, hence, the dual problem can be rewritten as

$$\max q(v) = \sum_{i=1}^{n} \min_j\{c_{ij} - v_j\} + \sum_{j=1}^{n} v_j. \tag{38}$$

Denoting with $j_i = \arg - \min_j\{c_{ij} - v_j\}$, the primal solution x, with $x_{i,j_i} = 1$ and 0 for $j \neq j_i$, with $i = 1, \ldots, n$, satisfies the complementary slackness conditions.

The dual problem has a nice economical interpretation. Assume that $p_j = -v_j$ represents the price that any agent will pay for being assigned to task j and u_i is the utility for agent i for being assigned to a task. The dual assignment problem consists in determining u_i and p_j (i.e., $-v_j$) maximizing the agents' total net utility, such that agents' net utilities cannot be greater than the costs c_{ij} they face. LP duality theory states that the maximum agents' total net utility equals the total assignment cost. At optimum, each task is assigned exactly to one agent, and the LP duality theory and complementary slackness conditions in particular assure that each agent i is assigned to the most profitable task j_i, which guarantees that agent net utility $u_i - p_{j_i}$ is exactly equal to the assignment cost c_{i,j_i}.

From the LP duality theory applied to the AP, we can derive the following auction algorithm [51]. Assume that agents are assigned to tasks through a market mechanism, with agent i acting according to its own best interest. Assume that task prices $p_j = -v_j$ are given. The total agent utility $(\sum_j u_j)$ is maximized if we set each u_j to its largest value allowed by the dual constraints, that is, $u_i = \min_j\{c_{ij} + p_j\}$. From the complementary slackness conditions, it follows that each agent i will bid for the most profitable task j_i, i.e., with $c_{ij_i} + p_{j_i} = u_i$ in order to be assigned to it. If no task is bid by more than one agent, we reach an equilibrium and the assignment is optimal; otherwise, we may change (increase) task prices p_j in order to discourage agents to bid for the

same task. This mechanism may be regarded as a naive auction algorithm that proceeds in rounds and halts if we get an equilibrium. We call it naive because it contains a flaw (as we will show next), but it motivates a more sophisticated and correct algorithm.

At each round of the naive auction algorithm, we start with a partial assignment and a given set of task prices and repeat the following two steps until all agents are assigned to their desired task (when we are at the equilibrium):

(1) Bidding step: given task prices p_j and a partial assignment of agents to tasks, (i) each unassigned agent i bids for its most profitable task $j_i = \arg - \min_j \{c_{ij} + p_j\}$ with an offer equal to $p_{j_i} + \gamma_i$, with $\gamma_i = \beta_i - \alpha_i$, where $\alpha_i = \min_j \{c_{ij} + p_j\}$ and $\beta_i = \min_{j \neq j_i} \{c_{ij} + p_j\}$, while (ii) each already assigned agent still submits the previous winning bid (without changing their bid offers).

(2) Pricing step: each task j is assigned to the highest offering bidder (agent) for that target. The price p_j of each task j receiving a new (greater) bid is increased to the highest received offer, i.e., the new price value will be equal to $p_j + \gamma_i$.

Unfortunately, this naive auction mechanism does not always work. It gets trapped in a cycle when (a) there is at least one unassigned agent and (b) each new winner bidder i submitted an offer for its preferred task j_i at its given target price p_{j_i}, i.e., $\gamma_i = 0$, meaning that its first and second best choices have the same cost.

In order to avoid this to happen, we need to keep rising the prices of tasks receiving new bids by at least a small amount $\epsilon > 0$. Therefore, we assume that agent i will bid for its preferred task j_i by offering $p_{j_i} + \gamma_i + \epsilon$.

This means that agent i desires to be assigned to task j_i if $c_{ij_i} + p_{j_i} \leq \min_j \{c_{ij} + p_j\} + \epsilon = \alpha_i + \epsilon$, which therefore is not necessarily its best choice. The above condition is known as ϵ-complementary slackness (see, e.g., [51]).

With this correction, the auction algorithm works ending in a finite number of rounds (depending on ϵ), with each task receiving a bid. At the end, we are almost at an equilibrium with agent i assigned to its almost desired task j_i. In general, this corresponds to an almost optimal solution for the assignment problem, since complementary conditions are only almost satisfied, while primal and dual complementary solutions are both feasible. It can be proved that if the cost c_{ij} are integers and $0 < \epsilon < 1/n$, then the (corrected) auction algorithm ends with an optimal solution for the assignment problem (see, e.g., [51]).

Without loss of generality, let us assume that $c_{ij} \geq 0$, and let $C = \max_{ij} \{c_{ij}\}$. In this case, it can be proved that the auction algorithm runs in $O(n^3 (C/\epsilon))$ time (see, e.g., [51]). Then, choosing $0 < \epsilon < 1/n$, the algorithm returns an optimal solution in $O(n^4 C)$ time. By using the scaling technique, Bertsekas and Eckstein in [50] proposed a modified version of the above-described auction algorithm that runs in $O(n^3 \log(nC))$ time. In real-world vehicle networks, the quality of solution in localized algorithms for task assignment is related to the communication network quality and range of communication. In [54], the influence of the communication range and different strategies of movement on the task assignment value in the auction algorithm was evaluated in simulations in mobile (robot) agent-task allocation scenarios.

4.1.4. Primal-Dual Algorithms. Primal-dual algorithms start from a dual feasible solution (u, v). From this solution, a restricted primal problem is defined and solved, consisting in finding the maximum cardinality matching on the bipartite subgraph $G' = (A \cup T, E')$, where $E' = \{(i, j) \in E \mid c_{ij} - u_i - v_j = 0\}$. If the optimal matching has a size equal to n, we are done; otherwise, the dual solution is improved (the dual objective function is increased), while assuring that also the size of E' is increased, and the procedure is repeated.

Note that also the auction algorithms for LAP consider simultaneously primal and dual solutions but, differently from primal-dual algorithms, they can improve as well as worsen both the primal and the dual cost through the intermediate iterations, although at the end, the optimal assignment is found (see, e.g., [51]).

4.1.5. Hungarian Algorithm. In particular, the Hungarian algorithm proposed by Munkres [55] is a primal-dual algorithm. The original version of the algorithm runs in $O(n^4)$ time and was improved to $O(n^3)$ by Lawler in 1976 (see, e.g., [32]) by using successive shortest path technique when finding a new maximum cardinality matching after having updated the dual variables.

In the following, we give some insights of the Hungarian algorithm that will be also useful for describing a decentralized version of the same. The Hungarian algorithm proceeds as follows:

Start with any feasible dual solution (u, v) and any matching $M \subseteq E' = \{(i, j) \in E \mid c_{ij} - u_i - v_j = 0\}$. For the starting dual solution, we can consider $v_j = \min_j \{c_{ij}\}$, with $j = 1, \ldots, n$, and $u_i = \min_i \{c_{ij} - v_j\}$, with $i = 1, \ldots, n$.

While M is not perfect, repeat the following:

(1) Given M and $G' = (A \cup T, E')$, find an alternating augmenting path P (i.e., a sequence of an odd number of edges that alternate edges of $E' \setminus M$ and edges of M, starting and ending with nonmatched edges); augment the matching by considering the new matching $M\prime = M \setminus P \cup P \setminus M$. Note that $|M\prime| = |M| + 1$. Update the matching M (with $M\prime$) and repeat until no new alternating augmenting path exists. M is the maximum cardinality matching of G'.

(2) If M is not perfect, update the dual solution such that at least a new edge is added to the set of (admissible) edges $E' = \{(i, j) \in E \mid c_{ij} - u_i - v_j = 0\}$, and continue with a new iteration. In particular, we can achieve this result by updating the values of u_i with $u_i + \delta$ and the values of v_j with $v_j - \delta$, where

$\delta = \min\{c_{ij} - u_i - v_j \mid i \in A\prime, j \in T\prime\}$ with $A\prime$ and $T\prime$ being the subsets of the vertices incident to the edges of the matching.

Searching for the alternating augmenting path can be done by a graph visiting algorithm that identifies a forest of alternating trees of G'. Note that in each step of the loop, we will either be increasing the size of M or the size of E' so this process must terminate. Furthermore, when the process terminates, M will be a perfect matching of $G' = (A \cup T, E')$, whose edge set E' is defined according to a feasible dual solution (u, v). Since the matching is perfect also for the complete bipartite graph G, the former represents a feasible primal solution for the assignment problem, respecting complementary constraints (by construction of E'); therefore, the primal and dual solutions are optimal.

4.1.6. Parallel Primal-Dual Algorithms. A certain number of parallel algorithms for the linear assignment problem have been proposed. They are parallelized versions of primal-dual algorithms based on shortest path computations, of the auction algorithm, and of primal simplex-based methods. Among the most efficient parallel algorithms for the LAP is the one proposed by Orlin and Stein [56] that adopting the cost scaling technique solves the problem using $\Omega(n^4)$ processors in $O(\log^3 n \cdot \log(\max\{cij\}))$ time. For a review, the reader is referred to [36, 51, 57].

4.1.7. Algorithms for the Bottleneck Assignment Problem. The bottleneck assignment problem can be solved in polynomial time, for example, by the so-called *threshold algorithm* that alternates two phases (see, e.g., [36, 58]). In the first one, a threshold value $\overline{c}_{ij}$ is chosen, and in the second phase, it is checked if the bipartite graph $G' = (A \cup T, E')$ admits a perfect matching or not, where $E' = \{(i, j) \in E \mid c_{ij} \leq \overline{c}_{ij}\}$.

One possible way to implement the first phase is applying a binary search. This leads to a threshold algorithm that runs in $O(T(n)\log n)$ time, where $O(T(n))$ is the time complexity for perfect matching checking. One of the best time complexity algorithms is by Punnen and Zhang (see, e.g., [59, 60]) that runs in $O(m\sqrt{n \log n})$, where m is the number of finite entries of the cost matrix $\{c_{ij}\}$.

4.1.8. Algorithms for the Fair Matching Problem. The balanced assignment problem can be solved in polynomial time, for example, by means of an iterative algorithm based on a feasibility subroutine that runs in $O(kT(n))$ (see, e.g., [37]), where $k \leq n^2$ is the number of distinct values of c_{ij} and $O(T(n))$ is the time required to test if there is a feasible assignment on a subset $\overline{E} \subseteq E$ of the edges of the complete bipartite graph $G = (A \cup T, \overline{E})$. Testing if there is a feasible assignment on $\overline{E}$ corresponds to check if the bipartite graph $\overline{G} = (A \cup T, \overline{E})$ admits a perfect matching that can be done by solving the maximum cardinality matching of $\overline{G}$, e.g., in $O(n^{2.5})$ time [61]. Hence, since $k \leq n^2$, the overall algorithm runs in $O(n^{4.5})$ time. Martello et al. in [37] improved the algorithm time complexity to $O(n^4)$ with a special refinement of the same.

4.1.9. Algorithms for an Online Bipartite Matching. Karp et al. in [62] evaluate an online algorithm for bipartite matching by comparing its performance by the worst-case ratio of its profit to that of the optimal offline algorithm. They propose an optimal online $1 - 1/e$ competitive simple randomized online algorithm to maximize the size of the matching in an unweighted bipartite graph. The best approximation algorithm for this problem is presented in [63] that applies the power of two choices paradigm, i.e., compute two offline matchings and use them to guide the adaptive online solutions.

Haeupler et al. in [64] study the unrestricted weighted problem in the stochastic arrival model and present the first approximation algorithms for it. They improve $1 - 1/e$-approximation for the online stochastic weighted matching problem to a 0.667-approximation. Moreover, they apply a discounted LP technique to give an improved competitive algorithm for the online stochastic matching problem and use the dual of the tightened LP to obtain a new upper bound on the optimal solution with a competitive ratio of 0.684. Via pseudomatching, they obtain an algorithm with a competitive ratio of 0.7036. They also present simple adaptive online algorithms to solve the online (weighted) stochastic matching problem optimally for the union of two matchings.

In [65], at each time step, a task is sampled independently from the given distribution and it needs to be matched upon its arrival to an agent. The goal is to maximize the number of allocations. An online algorithm is presented for this problem with a competitive ratio of 0.702. A key idea of the algorithm is to collect statistics about the decisions of the optimum offline solution using Monte Carlo sampling and use these statistics to guide the decisions of the online algorithm. The algorithm achieves a competitive ratio of 0.705 when the rates are integral.

4.1.10. Summary. While it is possible to solve most of these problems using the simplex algorithm, each AP variation has specialized more efficient algorithms designed to take advantage of its special structure.

Many centralized algorithms have been developed for solving the assignment problem in polynomial time (see, e.g., [36]). One of the first such algorithms was the Hungarian algorithm [55]. Other solution approaches include augmenting path methods (see, e.g., [66, 67]), adaptations of the primal simplex method (see, e.g., [68]), relaxation methods and auction algorithms (see, e.g., [51]), and signature methods (see, e.g., [69]).

The complexity of the Hungarian method by using Fibonacci heaps is $O(mn + n^2 \log n)$ [70]. Duan and Su's approach in [71] give an algorithm whose running time for integer weights is $O(m\sqrt{n} \log N)$, where m and n are the number of edges and vertices and N is the largest weight magnitude. Sankowski in [72] gave an $\widetilde{O}(Wn^{\omega})$ ($\widetilde{O}$ denotes the so-called "soft O" notation) time, where ω is the matrix

multiplication exponent and W is the highest edge weight in the graph.

Duan and Pettie in [73] find an $O(m\epsilon^{-1}\log\epsilon^{-1})$ running time algorithm that computes $(1-\epsilon)$-approximate maximum weight matching for any fixed ϵ.

Dell'Amico and Toth in [48] consider the classic linear assignment problem with a min-sum objective function, and the most efficient and easily available sequential codes for its solution that include shortest path algorithms APC, CTCS, and LAPm; shortest augmenting path algorithm with reduction transfer procedure JV, naive auction and sequential shortest path algorithm NAUCTION SP, two different implementations of the auction method, AFLP and AFR, and pseudoflow cost scaling algorithm CSA. Based on the results of the computational experiments obtained on dense instances containing both randomly generated and benchmark problems, it is not possible to obtain a precise ranking of the eight algorithms. However, APC is the fastest code for the two cost class and has a behaviour, on average, similar to that of CTCS for the other classes. Algorithm LAPm is the winner for the uniform random and the geometric classes and for the instances from the OR library. No dominance with respect to NAUCTION SP, CTCS, and APC exists for the remaining classes. Code JV has a good and stable average performance for all the classes, and it is the best algorithm for the uniform random (together with LAPm) and for the single-depot class. CSA performance strongly depends on the class, and it wins for no-wait flow-shop classes.

4.2. Distributed Coordination Approaches. By distributed, we consider the algorithms that combine the concepts of centralized and decentralized coordination, and principally *market-based approaches*, where solutions are built based on a bidding-auctioning procedure between the bidders (agents) and coordinators that play the role of auctioneers for allocating tasks to agents. There may be one or more coordinator agents as intermediaries in the task assignment process. The most known such algorithm is the auction algorithm that is presented in the following.

In this section, we recall two distributed solution approaches, respectively, based on auction algorithm and on primal-dual Hungarian method.

The Bertsekas auction algorithm (see, e.g., [51]) can be naturally implemented in a decentralized fashion. Zavlanos et al. [23] provide a distributed version of the auction algorithm proposed by Bertsekas for the considered networked systems with the lack of global information due to the limited communication capabilities of the agents. Updated prices necessary for accurate bidding can be obtained in a multihop fashion only by local exchange of information between adjacent agents. No shared memory is available, and the agents are required to store locally all the pricing information. This approach calculates the optimal solution in $O(\Delta n^3 C)$ time, with $\Delta \leq n-1$ being the maximum network diameter of the communication network.

Another market-based algorithm has been proposed more recently by Liu and Shell in [74] that instead of auctioning via a series of selfish bids from customers (agents) adopts a mechanism from the perspective of a merchant. The algorithm is capable of producing a solution (equilibrium) that satisfies both merchant and customers and is globally optimal; its running time is $O(n^3\log n)$.

Otte et al. in [75] study various auction algorithms for task assignment in the multirobot context and study how lossy communication between the auctioneer and bidders affects solution quality. They demonstrate both analytically and experimentally that even though many auction algorithms have similar performance when communication is perfect, they degrade in different ways as communication quality decreases from perfect to nonexistent. They compare six auction algorithms including standard implementations of the sequential auction, parallel auction, combinatorial auction; a generalization of the prim allocation auction called G-Prim; and two multiround variants of a repeated parallel auction. Variants of these auctions are also considered in which award information from previous rounds is rebroadcast by the auctioneer during the later round. They conclude that the best performing auction changes based on the reliability of the communication between the bidders and the auctioneer.

Giordani et al. in [24, 25] propose a distributed version of the Hungarian method for solving the LAP, based on the concept of alternating augmenting paths that are searched by maintaining a forest of alternating trees that is updated during the execution of the algorithm. In particular, given the current bipartite subgraph $G' = (A \cup T, E')$, where $E' = \{(i,j) \in E \mid c_{ij} - u_i - v_j = 0\}$, and A and T are agent and task vertices, respectively, the algorithm maintains forest F_1 of all the alternating trees rooted at free task vertices. Moreover, it maintains forest F_2 of the alternating trees of G' rooted at agent vertices containing all the agent/task vertices not contained in F_1. Clearly, the alternating trees in F_2 are not connected with vertices in F_1.

The algorithm involves root agents that initiate message exchange with other agents in the network via a depth-first search and synchronize the decision rounds (iterations, each containing multiple communication hops) across all agents. Through autonomous calculations and the communication with the (agent) neighbors, with respect to the position of the vertex representing the agent in the spanning alternating forests, agents get and share the information about the position of each task vertex (whether in F_1 or F_2), the values of dual variables related to tasks, the value of δ for the dual variables' update, the new admissible edge entering in a set of admissible edges of G' due to the dual variables' update, and the root agents $r(F_1)$ and $r(F_2)$ of forests F_1 and F_2, respectively. All these data are locally stored by each agent. In this way, there is no common coordinator or a shared memory of the agent's system. The agents, depending on the positions of the related vertices in the forests, change their roles and accordingly execute some of the steps of the distributed Hungarian algorithm. The total computational time is $O(n^3)$ as well as the total number of messages exchanged by the robots; nonetheless, the computational time required to perform the local calculation by each robot is $O(n^2)$. Regarding the robustness of the proposed method, if the agent during the execution of the algorithm stops

responding, it is considered erroneous and is eliminated from the further calculations. In the case where the agent was unmatched in forest F_2, the calculation continues without any modifications, ignoring the agent in question. Otherwise, the algorithm starts from the beginning excluding the same.

Chopra et al. in [76] propose a novel distributed version of the Hungarian method for solving the LAP that does not use any coordinator or shared memory. Specifically, each agent runs a local routine to execute ad hoc substeps of the centralized Hungarian method and exchanges estimates of the solution with neighboring robots. The authors show that with their approach, all agents converge to a common optimal assignment in a finite number ($O(n^3)$) of communication rounds if agents act synchronously. The overall performance of their approaches in terms of running time is only evaluated experimentally.

Eiselt and Marianov in [77] propose a model for the task assignment to employees with heterogeneous capabilities and multiple goals. Employees and tasks are mapped into the skill space where, after finding feasible matchings, they are assigned to each other by minimizing employee-task distance to minimize assignment cost, boredom, and unfairness between employees' workloads.

Peters and Zelewski in [78] develop two goal programming models for the employee assignment to workplaces according to both their competencies and preferences and the workplace requirements and attributes to ensure effective and efficient task performance. A review and classification of the literature regarding workforce planning problems incorporating skills can be found in [79].

The bottleneck assignment problem can be solved in polynomial time, for example, by the so-called *threshold algorithm* that alternates two phases (see, e.g., [36, 58]). In the first one, a threshold value $\overline{c}_{ij}$ is chosen, and in the second phase, it is checked if the bipartite graph $G' = (A \cup T, E')$ admits a perfect matching or not, where $E' = \{(i, j) \in E \mid c_{ij} \leq \overline{c}_{ij}\}$.

One possible way to implement the first phase is applying a binary search. This leads to a threshold algorithm that runs in $O(T(n)\log n)$ time, where $O(T(n))$ is the time complexity for perfect matching checking. One of the best time complexity algorithms by Punnen and Zhang (see, e.g., [59, 60]) that runs in $O(m\sqrt{n \log n})$, where m is the number of finite entries of the cost matrix $\{c_{ij}\}$. Efrat et al. in [80] propose algorithms that, assuming planar objects, run in roughly $O(n^{1.5}\log n)$ time. Pothen and Fan in [81] propose a parallel algorithm with $O(nm)$ time complexity, which is currently among the best practical serial algorithms for maximum matching. However, its performance is sensitive to the order in which the vertices are processed for matching.

In [82], Azad et al. study the performance improvement of augmentation-based parallel matching algorithms for bipartite cardinality matching on multithreaded machines over serial algorithms and report extensive results and insights on efficient multithreaded implementations of three classes of algorithms based on their manner of searching for augmenting paths: breadth-first search, depth-first search, and a combination of both.

In [80], algorithms for the balanced assignment problem and minimum deviation assignment are presented that run in roughly $O(n^{10/3})$ and, as such, are more efficient than the algorithms in [37, 41] that run in $O(n^4)$ time on general bipartite graphs. Kennington and Wang in [43] present a shortest augmenting path algorithm for solving the semi-assignment problem in which each iteration during the final phase of the procedure (also known as the endgame) obtains an additional assignment.

4.3. Decentralized Coordination Approaches. In contrast to centralized and distributed coordination approaches to task allocation where full knowledge of global information is assumed available to every relevant decision-maker (central decision-maker or fleet coordinator (fleet owner) and (vehicle) bidder agents), in the decentralized task assignment approaches, there is no coordinator and each vehicle agent disposes only of its local (possibly incomplete and imperfect) information and finds its local assignment based exclusively on this information and the communication with the rest of the agents and interaction with its environment.

In general, decentralized approaches have several advantages, i.e., real-time property, robustness, and scalability. These characteristics are in general absent in centralized and distributed approaches that outperform decentralized approaches in terms of efficiency especially for large-scale instances. The decentralized decision-making does not include any intermediary. In case of imperfect communication, conflicts may occur. This is why the related literature in decentralized multivehicle cooperative control is related to consensus, i.e., the agreement of all vehicles on some common features by negotiating with their local neighbors. General consensus issues are related to, e.g., positions, velocities, and attitudes. In the following, we analyze localized, scalable, and decentralized heuristic algorithms for coordination of deterministic and dynamic task assignment in open vehicle fleets. We concentrate on the approaches resulting in both task assignment feasibility and efficiency even though these approaches usually have no quality of solution guarantees.

Decentralized task assignment approaches have been mostly developed in the multirobot and unmanned aerial vehicle (UAV) coordination domain. The most known ones are sequential auction-based or consensus and negotiation-based algorithms (e.g., [83–85]).

One of the most known approaches for the decentralized task assignment in the coordination of a fleet of unmanned vehicles when all-to-all intervehicle communication is not possible is the consensus-based auction algorithm (CBAA) and its more general version that allows for the assignment of bundles of tasks to each agent called the consensus-based bundle algorithm (CBBA) [84].

The CBAA is a polynomial time market-based decentralized task selection agreement protocol running in two phases: in the first phase, each vehicle places a bid on a task asynchronously with the rest of the fleet, and in the second, consensus phase, conflicting assignments are identified and resolved through local communication between neighboring

agents within certain predefined rules to avoid task conflicts. The agents use a consensus strategy to converge on the list of winning bids and use that list to determine the winner and associated winning scores. The list accounts for inconsistent information among agents guaranteeing a conflict-free assignment for all. This allows conflict resolution over all tasks that are robust to inconsistencies in the situational awareness across the fleet and the changes in the communication network topology. If the resulting scoring scheme satisfies a diminishing marginal gain property (i.e., the value of a task does not increase as other tasks are assigned to the same agent before it), a feasible, conflict-free solution is guaranteed.

Provided that the scoring function abides by the principle of diminishing marginal gains, the CBBA has convergence guarantees. In a synchronized conflict resolution phase over a static communication network, it produces the same solution as the sequential greedy algorithm sharing across the fleet the corresponding winning bid values and winning agent information. Moreover, the convergence time is bounded from above and it does not depend on the inconsistency in the situational awareness over the agent set.

In [84], it is analytically shown that CBAA produces the same solution as some centralized sequential greedy procedures, and this solution guarantees 50% optimality. Segui-Gasco et al. [86] propose a decentralized algorithm for multirobot task allocation with a constant factor approximation of 63% for positive-valued monotone submodular utility functions and of 37% for general positive-valued submodular utility functions. Therefore, the authors improve the approximation guarantee of Choi et al. [84] for monotone positive-valued submodular utility functions from 50% to 37%.

The CBBA has also been extended to consider coupled constraints [87, 88]. Choi et al. in [87] extended CBBA for heterogeneous task allocation to UAV agents with different qualifications and various cooperation constraints. The CBBA was extended with task decomposition and a scoring modification to allow for soft constraints related to cooperation preferences and a decentralized task elimination protocol that ensures the satisfaction of the hard constraints related to cooperation requirements. The performance of the algorithms was analyzed in Monte Carlo simulations in some randomly generated experiments.

The CBBA was also extended in [88] to consider the assignment of tasks with assignment constraints and also with different types of coupled and temporal constraints, where it was assumed that assigned tasks are executed in the order defined by their temporal precedence.

The temporal sequential single-item (TeSSI) auction algorithm [83] allocates tasks with time windows to cooperative robot agents using a variant of the sequential single-item auction algorithm. Contrary to the CBBA algorithm that does not let the change of the start time of the tasks once they are allocated and thus reduces the number of tasks that the algorithm allocates, the TeSSI algorithm overcomes this limitation by allowing tasks' start times to change, which results in higher allocation rates.

The main features of the TeSSI algorithm are a fast and systematic processing of temporal constraints and two bidding methods that optimize either completion time or a combination of completion time and distance. The main objective function used in the TeSSI algorithm is the makespan (the time the last task is finished) even though it is also combined with the total distance traveled. Each robot maintains the temporal consistency of its allocated tasks using a simple temporal network. The authors show that TeSSI outperforms a baseline greedy algorithm and the CBBA through random experiments and related work datasets.

Ponda et al. in [89] further extend the CBBA to tasks with time windows and address replanning in dynamic environments and consider agents with different capabilities. Agents obtain new plans based on the changes in the environment considering new tasks while pruning older or irrelevant ones.

One of the drawbacks of the CBBA algorithm is that it relies on global synchronization mechanisms which are hard to enforce in decentralized environments. Johnson et al. [85] proposed the asynchronous CBBA (ACBBA) for agents that communicate asynchronously. To allow for asynchrony in communication, the ACBBA contains a set of local deconfliction rules that do not require access to the global information. In ACBBA, agents locally replan their actions that, possibly, affect only a limited number of agents.

Johnson et al. [90] propose a situational awareness algorithm for task assignment when agents predict the bids of their neighbors, in order to obtain more informed decisions in a cooperative way.

To respond to the problem with local information consistency assumption that reduces optimization capabilities compared to global information assumption approaches, Johnson et al. [91] proposed a bid warped consensus-based bundle algorithm that converges for all deterministic objective functions and has nontrivial performance guarantees for submodular and some non-submodular objective functions. They analyze the convergence and performance of the algorithm and show its efficiency compared with some other relevant local and global information approaches.

Another extension to the CBBA is provided by Binetti et al. [92] that consider the decentralized surveillance problem by a team of robots. Tasks are assigned to each robot with the additional constraint that a subset of the tasks called critical tasks must be assigned. The authors use the CBBA incorporating hard constraints in order to ensure that the critical tasks are not left unassigned.

In [93], Garcia and Casbeer present a robust task assignment algorithm that reduces communication between vehicles in uncertain environments. Piece-wise optimal decentralized allocation of tasks is considered for a group of unmanned aerial vehicles. They present a framework for multiagent cooperative decision-making under communication constraints. Each vehicle estimates the position of all other vehicles in order to assign tasks based on these estimates, and it also implements event-based broadcasting strategies that allow the multiagent system representing the

vehicle fleet to use communication resources more efficiently. The agents implement a simple decentralized auction scheme in order to resolve possible conflicts.

Cui et al. in [94] investigate game theory-based negotiation for task allocation in the multirobot task assignment context. Tasks are initially allocated using an approach based on contract net (see [95]), after which a negotiation approach employing the utility functions to select the negotiation robot agents and construct the negotiation set is proposed. Then, a game theory-based negotiation strategy achieves the Pareto-optimal solution for the task reallocation. Extensive simulation results demonstrate the efficiency of such a task assignment approach.

Yet another extension of the consensus-based bundle algorithm (CBBA) allowing for the fast allocation of new tasks without a full reallocation of existing tasks is CBBA with partial replanning (CBBA-PR) [96]. The algorithm enables the multiagent system to trade-off between convergence time and increased coordination by resetting a portion of their previous allocation at every round of bidding on tasks. By resetting the last tasks allocated by each agent, the convergence of the MAS to a conflict-free solution is assured. CBBA-PR can be further improved by reducing the team size involved in the replanning, further reducing the communication burden of the team and runtime of CBBA-PR.

In [97], Sayyaadi and Moarref investigate a proportional task assignment problem in which it is desired for (robot) agents to have an equal duty to capability ratios, i.e., the agents with more capability should perform more tasks. They address this problem as a combination of deployment and consensus problems in which agents should reach consensus over the value of their duty to capability ratios. They propose a distributed, asynchronous and scalable algorithm for this problem in the continuous time domain.

Duran et al. in [98] study the problem of finding the list of solutions with strictly increasing cost for the semiassignment problem. Four different algorithms are described and compared. The results show that they find the exact list of solutions and considerably reduce the computation times in comparison with the other exact approaches.

Spivey et al. in [99] propose a distributed, flexible, and scalable control scheme that evenly allocates tasks. Dynamic load balancing exploits feedback information about the status of tasks and vehicles with the objective to keep a balanced task load and, thus, force cooperation in the solution of the randomized bottleneck task assignment problem.

In summary, most of the state-of-the-art decentralized and deterministic coordination approaches for task allocation are heuristic algorithms developed for multirobot or UAV task allocation scenarios that often include both operational and tactical constraints of a vehicle fleet and its environment. Even though their adaptation for the use in open vehicle fleets does not seem difficult, it remains an open challenge, especially if we consider task allocation efficiency, the key performance indicator of commercial open fleets.

5. Challenges in Open Vehicle Fleet Coordination

In this paper, we proposed new mathematical programming models of dynamic versions of the following assignment problems well known in combinatorial optimization and applicable in open vehicle fleets: the assignment problem, bottleneck assignment problem, fair matching problem, dynamic minimum deviation assignment problem, $\sum_k$−assignment problem, the semiassignment problem, the assignment problem with side constraints, and the assignment problem while recognizing agent qualification. The goal of the studied problems is finding an optimal (minimum cost or maximum profit) assignment to the (vehicle) agents of the tasks that are known at the time of decision-making. These approaches do not take into account unknown tasks that may appear once when the current tasks are completed.

With the long-term objective of decentralizing and democratizing shared mobility, we categorized solution approaches for static and dynamic task assignment problems applicable in open vehicle fleets into centralized, distributed, and decentralized and discussed their main characteristics. The presented distributed and decentralized task assignment methods are applicable in distributed and decentralized open vehicle fleets, respectively. In case of decentralized fleets, the issues related to privacy, trust, and control intrinsic to centralized systems are gone.

We focused on homogeneous vehicle agents and tasks, i.e., each vehicle agent is able to complete each task with equal efficiency but varying cost or profit. In the real world, that might not be the case since in open vehicle fleets, the vehicles tend to be heterogeneous. The proposed mathematical programs can easily be adapted to this case by varying the agent-task assignment cost/profit depending on the performance efficiency of an agent; in case of an agent inapt to perform a task, its agent-task assignment cost is assigned a very large value.

With fully decentralized scalable coordination of task allocation, there is no need to put limits to the size of the system. However, even though scalable task allocation and related coordination mechanisms are essential for efficiently managing large-scale open vehicle fleet systems, it should be noticed that, for real-world applications, they need to be complemented with scalable and efficient solution approaches to other combinatorial optimization problems depending on the context, e.g., dial-a-ride problem and traveling salesperson problem, etc.

We dealt with the deterministic and dynamic assignment problem where real-time reassignment is beneficial since both the variables and parameters of the optimization problem are perfectly known at each period. However, when dealing with real-world stochastic environments with increased sensor noise, a too high frequency of task reassignment may result in a churning effect in the assignment and may lead to increased human errors. Thus, a chosen coordination method must consider churning in this context

to obtain good overall task allocation performance (see, e.g., [100]).

A truly open vehicle fleet system should work also based on heterogeneous software agents produced by multiple producers. The agent software could be an open source and/or there may be multiple proprietary software companies working on a common open fleet coordination standard. The Agreement Technologies (AT) paradigm [101] identifies and relates various such technologies. It provides a sandbox of mechanisms to support coordination among (heterogeneous) autonomous software agents, which focuses on the concept of agreement between them. To this respect, AT-based systems not only support the interactions for reaching an agreement in a coordinated manner (e.g., as part of a distributed or decentralized algorithm) but are also endowed with means to specify and govern the "space" of agreements that can be reached, as well as monitoring agreement execution. In particular, in truly open vehicle fleet systems where there may be a multitude of (possibly heterogeneous) software providers, semantic mismatches among vehicle agents need to be dealt with through the alignment of ontologies, so that vehicle agents can reach a common understanding on the elements of agreements.

Furthermore, (weak) constraints on agreement and agreement processes (often also called *norms*) need to be defined and represented in a declarative manner, so autonomous agents can decide as to whether they will adopt them, determine as to how far they are applicable in a certain situation, dynamically generate priorities among conflicting norms depending on the context, etc. In addition, trust and reputation models are necessary for keeping track of whether the agreements reached, and their executions, respect the requirements put forward by norms and organizational constraints. So, norms and trust can be conceived as a priori and a posteriori approaches, respectively, to support the security in relation to the coordination process. How to find seamless and effective means of integrating the different distributed and decentralized algorithms outlined in this paper in such a framework is still an open issue that we will treat in our future work.

The presented distributed and decentralized coordination methods for dynamic task assignment may be applied to semiautonomous and autonomous vehicles and are a necessary part of reaching full vehicle fleet autonomy. They may not fix the mobility concerns, but they will definitely improve them as they are directly related to giving a higher control both to an individual driver (or to an autonomous vehicle) and to a customer (rider). Intrinsically, these methods aid in changing the hierarchical tree structure of the transportation networks to a more horizontal one. Indirect benefits of such coordination methods, among others, include higher efficiency, smaller carbon footprints, and less traffic jams. In the long run, they will facilitate more decentralized, autonomous, and transparent open vehicle fleets, but above all, they will further the task allocation efficiency and fair rewards and benefits of vehicles, drivers, customers, and riders proportional to their participation in large and open fleets.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work has been partially supported by E-Logistics project financed by the French Agency for Environment and Energy Management (ADEME) and "IntelliEDGE" project "RTI2018-095390-B-C33" funded by Spanish Ministry of Science, Innovation and Universities.

References

[1] B. Horling and V. Lesser, "A survey of multi-agent organizational paradigms," *The Knowledge Engineering Review*, vol. 19, no. 4, pp. 281–316, 2004.

[2] Y. Chevaleyre, P. E. Dunne, U. Endriss et al., "Issues in multiagent resource allocation," *Informatica*, vol. 30, no. 1, Article ID 03505596, 2006.

[3] P. Ahuja, P. Sethi, D. Juneja, and S. Mukherjee, "Task allocation strategy for multi agent: a review," *Journal of Network Communications and Emerging Technologies (JNCET)*, vol. 7, no. 1, 2017.

[4] S.-Y. Tang, Y.-F. Zhu, Q. Li, and Y.-L. Lei, "Survey of task allocation in multi agent systems," *Systems Engineering and Electronics*, vol. 32, no. 10, pp. 2155–2161, 2010.

[5] Y. Jiang and Z. Huang, "The rich get richer: preferential attachment in the task allocation of cooperative networked multiagent systems with resource caching," *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans*, vol. 42, no. 5, pp. 1040–1052, 2012.

[6] J. Jiang, B. An, Y. Jiang, C. Zhang, Z. Bu, and J. Cao, "Group-oriented task allocation for crowdsourcing in social networks," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, pp. 1–16, 2019.

[7] S. Mariani, G. Cabri, and F. Zambonelli, "Coordination of autonomous vehicles: taxonomy and survey," 2020, https://arxiv.org/abs/2001.02443.

[8] H. Billhardt, A. Fernández, L. Lemus et al., "Dynamic coordination in fleet management systems: toward smart cyber fleets," *IEEE Intelligent Systems*, vol. 29, no. 3, pp. 70–76, 2014.

[9] M. Bielli, A. Bielli, and R. Rossi, "Trends in models and algorithms for fleet management," *Procedia—Social and Behavioral Sciences*, vol. 20, pp. 4–18, 2011.

[10] N. Agatz, A. Erera, M. Savelsbergh, and X. Wang, "Optimization for dynamic ride-sharing: a review," *European Journal of Operational Research*, vol. 223, no. 2, pp. 295–303, 2012.

[11] M. Furuhata, M. Dessouky, F. Ordóñez, M.-E. Brunet, X. Wang, and S. Koenig, "Ridesharing: the state-of-the-art and future directions," *Transportation Research Part B: Methodological*, vol. 57, pp. 28–46, 2013.

[12] H. Billhardt, A. Fernández, M. Lujak et al., "Towards smart open dynamic fleets," in *Multi-Agent Systems and Agreement Technologies*, pp. 410–424, Springer, Berlin, Germany, 2015.

[13] C. Zhu, G. Pastor, Y. Xiao, Y. Li, and A. Ylae-Jaeaeski, "Fog following me: latency and quality balanced task allocation in vehicular fog computing," in *Proceedings of the 2018 15th Annual IEEE International Conference on Sensing, Communication, and Networking (SECON)*, pp. 1–9, IEEE, Hong Kong, China, June 2018.

[14] M. Lindner, M. Kalech, and G. A. Kaminka, "A representation for coordination fault detection in large-scale multi-agent systems," *Annals of Mathematics and Artificial Intelligence*, vol. 56, no. 2, pp. 153–186, 2009.

[15] M. Pujol-Gonzalez, J. Cerquides, P. Meseguer, and J. A. Rodríguez-Aguilar, "Efficient inter-team task allocation in robocup rescue," in *Proceedings of the International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, pp. 413–421, 2015.

[16] H. Billhardt, A. Fernandez, M. Lujak et al., "Coordinating open fleets. a taxi assignment example," *AI Communications*, vol. 30, no. 1, pp. 37–52, 2017.

[17] M. Lujak, H. Billhardt, and S. Ossowski, "Distributed coordination of emergency medical service for angioplasty patients," *Annals of Mathematics and Artificial Intelligence*, vol. 78, no. 1, pp. 73–100, 2016.

[18] M. W. Ulmer, B. W. Thomas, A. M. Campbell, and N. Woyak, "The restaurant meal delivery problem: dynamic pick-up and delivery with deadlines and random ready times," Transportation Science, 2017.

[19] T. A. Taylor, "On-demand service platforms," *Manufacturing & Service Operations Management*, vol. 20, no. 4, pp. 704–720, 2018.

[20] H. Dai, L. Ge, and Y. Liu, "Information matters: an empirical study of the efficiency of on-demand services," *Information Systems Frontiers*, vol. 22, no. 4, pp. 815–827, 2020.

[21] H. Billhardt, M. Lujak, V. Sánchez-Brunete, A. Fernández, and S. Ossowski, "Dynamic coordination of ambulances for emergency medical assistance services," *Knowledge-Based Systems*, vol. 70, pp. 268–280, 2014.

[22] M. Lujak and H. Billhardt, "Coordinating emergency medical assistance," in *Agreement Technologies*, pp. 597–609, Springer, Berlin, Germany, 2013.

[23] M. M. Zavlanos, L. Spesivtsev, and G. J. Pappas, "A distributed auction algorithm for the assignment problem," in *Proceedings of the 2008 47th IEEE Conference on Decision and Control*, pp. 1212–1217, IEEE, Cancun, Mexico, December 2008.

[24] S. Giordani, M. Lujak, and F. Martinelli, "A distributed algorithm for the multi-robot task allocation problem," in *Trends in Applied Intelligent Systems*, pp. 721–730, Springer, Berlin, Germany, 2010.

[25] S. Giordani, M. Lujak, and F. Martinelli, "A distributed multi-agent production planning and scheduling framework for mobile robots," *Computers & Industrial Engineering*, vol. 64, no. 1, pp. 19–30, 2013.

[26] S. Ghosh, *Distributed Systems: An Algorithmic Approach*, Chapman and Hall/CRC, London, UK, 2014.

[27] A. S. Tanenbaum and M. Van Steen, *Distributed Systems: Principles and Paradigms*, Prentice-Hall, Upper Saddle River, NJ, USA, 2007.

[28] D. W. Pentico, "Assignment problems: a golden anniversary survey," *European Journal of Operational Research*, vol. 176, no. 2, pp. 774–793, 2007.

[29] R. E. Burkard and E. Cela, "Linear assignment problems and extensions," in *Handbook of Combinatorial Optimization*, pp. 75–149, Springer, Berlin, Germany, 1999.

[30] M. Z. Spivey and W. B. Powell, "The dynamic assignment problem," *Transportation Science*, vol. 38, no. 4, pp. 399–419, 2004.

[31] V. Pillac, M. Gendreau, C. Guéret, and A. L. Medaglia, "A review of dynamic vehicle routing problems," *European Journal of Operational Research*, vol. 225, no. 1, pp. 1–11, 2013.

[32] E. L. Lawler, *Combinatorial Optimization: Networks and Matroids*, Courier Corporation, Chelmsford, MA, USA, 2001.

[33] C. H. Papadimitriou and K. Steiglitz, *Combinatorial Optimization: Algorithms and Complexity*, Courier Corporation, Chelmsford, MA, USA, 1982.

[34] G. Caron, P. Hansen, and B. Jaumard, "The assignment problem with seniority and job priority constraints," *Operations Research*, vol. 47, no. 3, pp. 449–453, 1999.

[35] H. Moulin, *Fair Division and Collective Welfare*, MIT press, Cambridge, MA, USA, 2004.

[36] B. Raine, M. Dell'Amico, and S. Martello, *Assignment Problems*, Springer, Berlin, Germany, 2009.

[37] S. Martello, W. R. Pulleyblank, P. Toth, and D. De Werra, "Balanced optimization problems," *Operations Research Letters*, vol. 3, no. 5, pp. 275–278, 1984.

[38] N. Sun and Z. Yang, "A general strategy proof fair allocation mechanism," *Economics Letters*, vol. 81, no. 1, pp. 73–79, 2003.

[39] T. Andersson, "A general strategy-proof fair allocation mechanism revisited," *Economics Bulletin*, vol. 29, no. 3, pp. 1717–1722, 2009.

[40] C. W. Duin and A. Volgenant, "Minimum deviation and balanced optimization: a unified approach," *Operations Research Letters*, vol. 10, no. 1, pp. 43–48, 1991.

[41] S. K. Gupta and A. P. Punnen, "Minimum deviation problems," *Operations Research Letters*, vol. 7, no. 4, pp. 201–204, 1988.

[42] C. Filippi, W. Ogryczak, and M. G. Speranza, "Bridging k-sum and cvar optimization in milp," *Computers & Operations Research*, vol. 105, pp. 156–166, 2019.

[43] J. Kennington and Z. Wang, "A shortest augmenting path algorithm for the semi-assignment problem," *Operations Research*, vol. 40, no. 1, pp. 178–187, 1992.

[44] V. Aggarwal, "A lagrangean-relaxation method for the constrained assignment problem," *Computers & Operations Research*, vol. 12, no. 1, pp. 97–106, 1985.

[45] J. B. Mazzola and A. W. Neebe, "Resource-constrained assignment scheduling," *Operations Research*, vol. 34, no. 4, pp. 560–572, 1986.

[46] D. G. Cattrysse and L. N. Van Wassenhove, "A survey of algorithms for the generalized assignment problem," *European Journal of Operational Research*, vol. 60, no. 3, pp. 260–272, 1992.

[47] M. Akgül, "A genuinely polynomial primal simplex algorithm for the assignment problem," *Discrete Applied Mathematics*, vol. 45, no. 2, pp. 93–115, 1993.

[48] M. Dell'Amico and P. Toth, "Algorithms and codes for dense assignment problems: the state of the art," *Discrete Applied Mathematics*, vol. 100, no. 1-2, pp. 17–48, 2000.

[49] D. P. Bertsekas, "A new algorithm for the assignment problem," *Mathematical Programming*, vol. 21, no. 1, pp. 152–171, 1981.

[50] D. P. Bertsekas and J. Eckstein, "Dual coordinate step methods for linear network flow problems," *Mathematical Programming*, vol. 42, no. 1–3, pp. 203–243, 1988.

[51] D. P. Bertsekas, "Auction algorithms," *Encyclopedia of Optimization*, Springer, Berlin, Germany, pp. 128–132, 2009.

[52] Y. Jiang, "A survey of task allocation and load balancing in distributed systems," *IEEE Transactions on Parallel and Distributed Systems*, vol. 27, no. 2, pp. 585–599, 2015.

[53] T. Scheffel, A. Pikovsky, M. Bichler, and K. Guler, "An experimental comparison of linear and nonlinear price combinatorial auctions," *Information Systems Research*, vol. 22, no. 2, pp. 346–368, 2011.

[54] M. Lujak and S. Giordani, "On the communication range in auction-based multi-agent target assignment," in *Self-Organizing Systems Of LNCS*, C. Bettstetter and C. Gershenson, Eds., vol. 6557, pp. 32–43, Springer, Berlin, Germany, 2011.

[55] J. Munkres, "Algorithms for the assignment and transportation problems," *Journal of the Society for Industrial and Applied Mathematics*, vol. 5, no. 1, pp. 32–38, 1957.

[56] J. B. Orlin and C. Stein, "Parallel algorithms for the assignment and minimum-cost flow problems," *Operations Research Letters*, vol. 14, no. 4, pp. 181–186, 1993.

[57] D. P. Bertsekas, *Network Optimization: Continuous and Discrete Methods*, vol. 8, Athena Scientific, Nashua, NH 03060, USA, 1998.

[58] H. Luss, *Equitable Resource Allocation: Models, Algorithms and Applications*, vol. 101, John Wiley & Sons, Hoboken, NJ, USA.

[59] J. LaRusic and A. P. Punnen, "The asymmetric bottleneck traveling salesman problem: algorithms, complexity and empirical analysis," *Computers & Operations Research*, vol. 43, pp. 20–35, 2014.

[60] A. P. Punnen and R. Zhang, "Bottleneck flows in unit capacity networks," *Information Processing Letters*, vol. 109, no. 6, pp. 334–338, 2009.

[61] J. E. Hopcroft and R. M. Karp, "An $n\{5/2\}$ algorithm for maximum matchings in bipartite graphs," *SIAM Journal on Computing*, vol. 2, no. 4, pp. 225–231, 1973.

[62] R. M. Karp, U. V. Vazirani, and V. V. Vazirani, "An optimal algorithm for on-line bipartite matching," in *Proceedings of the Twenty-Second Annual ACM Symposium on Theory of Computing—STOC '90*, pp. 352–358, ACM, Baltimore, MA, USA, 1990.

[63] V. H. Menshadi, S. O. Gharan, and A. Saberi, "Online stochastic matching: online actions based on offline statistics," *Mathematics of Operations Research*, vol. 2, no. 3, pp. 559–674, 2012.

[64] B. Haeupler, V. S. Mirrokni, and M. Zadimoghaddam, "Online stochastic weighted matching: improved approximation algorithms," in *International Workshop on Internet and Network Economics*, pp. 170–181, Springer, Berlin, Germany, 2011.

[65] V. H. Manshadi, S. O. Gharan, and A. Saberi, "Online stochastic matching: online actions based on offline statistics," *Mathematics of Operations Research*, vol. 37, no. 4, pp. 559–573, 2012.

[66] R. Jonker and A. Volgenant, "A shortest augmenting path algorithm for dense and sparse linear assignment problems," *Computing*, vol. 38, no. 4, pp. 325–340, 1987.

[67] G. A. Mills-Tettey, A. Stentz, and M. B. Dias, "The dynamic hungarian algorithm for the assignment problem with changing costs," Technical Report CMU-RI-TR-07-27, Robotics Institute, Pittsburgh, PA, USA, 2007.

[68] J. B. Orlin, "A polynomial time primal network simplex algorithm for minimum cost flows," *Mathematical Programming*, vol. 78, no. 2, pp. 109–129, 1997.

[69] M. L. Balinski, "Signature methods for the assignment problem," *Operations Research*, vol. 33, no. 3, pp. 527–536, 1985.

[70] M. L. Fredman and R. E. Tarjan, "Fibonacci heaps and their uses in improved network optimization algorithms," *Journal of the ACM (JACM)*, vol. 34, no. 3, pp. 596–615, 1987.

[71] R. Duan and H.-H. Su, "A scaling algorithm for maximum weight matching in bipartite graphs," in *Proceedings of the Twenty-Third Annual ACM-SIAM Symposium on Discrete Algorithms, Society for Industrial and Applied Mathematics*, pp. 1413–1424, Kyoto, Japan, 2012.

[72] P. Sankowski, "Maximum weight bipartite matching in matrix multiplication time," *Theoretical Computer Science*, vol. 410, no. 44, pp. 4480–4488, 2009.

[73] R. Duan and S. Pettie, "Linear-time approximation for maximum weight matching," *Journal of the ACM*, vol. 61, no. 1, pp. 1–23, 2014.

[74] L. Liu and D. A. Shell, "Optimal market-based multi-robot task allocation via strategic pricing," Edited by P. Newman, D. Fox, and D. Hsu, Eds., in *Proceedings of the Robotics: Science and Systems*, vol. IX, pp. 1–8, New York, NY, USA, 2013.

[75] M. Otte, M. J. Kuhlman, and D. Sofge, "Auctions for multi-robot task allocation in communication limited environments," *Autonomous Robots*, vol. 44, no. 3-4, pp. 1–38, 2019.

[76] S. Chopra, G. Notarstefano, M. Rice, and M. Egerstedt, "A distributed version of the Hungarian method for multirobot assignment," *IEEE Transactions on Robotics*, vol. 33, no. 4, pp. 932–947, 2017.

[77] H. A. Eiselt and V. Marianov, "Employee positioning and workload allocation," *Computers & Operations Research*, vol. 35, no. 2, pp. 513–524, 2008.

[78] M. L. Peters and S. Zelewski, "Assignment of employees to workplaces under consideration of employee competences and preferences," *Management Research News*, vol. 30, no. 2, pp. 84–99, 2007.

[79] P. De Bruecker, J. Van den Bergh, J. Beliën, and E. Demeulemeester, "Workforce planning incorporating skills: state of the art," *European Journal of Operational Research*, vol. 243, no. 1, pp. 1–16, 2015.

[80] A. Efrat, A. Itai, and M. J. Katz, "Geometry helps in bottleneck matching and related problems," *Algorithmica*, vol. 31, no. 1, pp. 1–28, 2001.

[81] A. Pothen and C.-J. Fan, "Computing the block triangular form of a sparse matrix," *ACM Transactions on Mathematical Software (TOMS)*, vol. 16, no. 4, pp. 303–324, 1990.

[82] A. Azad, M. Halappanavar, S. Rajamanickam, E. G. Boman, A. Khan, and A. Pothen, "Multithreaded algorithms for maximum matching in bipartite graphs," in *Proceedings of the 2012 IEEE 26th International Parallel and Distributed Processing Symposium*, IEEE, Washington, DC, USA, pp. 860–872, 2012.

[83] E. Nunes and M. Gini, "Multi-robot auctions for allocation of tasks with temporal constraints," in *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*, 2015.

[84] H.-L. Choi, L. Brunet, and J. P. How, "Consensus-based decentralized auctions for robust task allocation," *IEEE Transactions on Robotics*, vol. 25, no. 4, pp. 912–926, 2009.

[85] L. Johnson, S. Ponda, H.-L. Choi, and J. How, "Asynchronous decentralized task allocation for dynamic environments," in *Proceedings of the AIAA Infotech@ Aerospace 2011*, pp. 1–12, AIAA, St. Louis, MI, USA, June 2011.

[86] P. Segui-Gasco, H.-S. Shin, A. Tsourdos, and V. Segui, "Decentralised submodular multi-robot task allocation," in *Proceedings of the 2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 2829–2834, IEEE, Hamburg, Germany, October 2015.

[87] H.-L. Choi, A. K. Whitten, and J. P. How, "Decentralized task allocation for heterogeneous teams with cooperation constraints," in *Proceedings of the 2010 American Control Conference*, pp. 3057–3062, IEEE, Baltimore, MD, USA, June 2010.

[88] A. K. Whitten, H.-L. Choi, L. B. Johnson, and J. P. How, "Decentralized task allocation with coupled constraints in

complex missions," in *Proceedings of the 2011 American Control Conference*, pp. 1642–1649, IEEE, San Francisco, CA, USA, July 2011.

[89] S. Ponda, J. Redding, H.-L. Choi, J. P. How, M. Vavrina, and J. Vian, "Decentralized planning for complex missions with dynamic communication constraints," in *Proceedings of the 2010 American Control Conference*, pp. 3998–4003, IEEE, Baltimore, MD, USA, July 2010.

[90] L. B. Johnson, H.-L. Choi, and J. P. How, "Hybrid information and plan consensus in distributed task allocation," in *Proceedings of the AIAA Guidance, Navigation, and Control (GNC) Conference*, p. 4888, Boston, MA, USA, 2013.

[91] L. B. Johnson, H.-L. Choi, S. S. Ponda, and J. P. How, "Decentralized task allocation using local information consistency assumptions," *Journal of Aerospace Information Systems*, vol. 14, no. 2, pp. 103–122, 2017.

[92] G. Binetti, D. Naso, and B. Turchiano, "Decentralized task allocation for surveillance systems with critical tasks," *Robotics and Autonomous Systems*, vol. 61, no. 12, pp. 1653–1664, 2013.

[93] E. Garcia and D. W. Casbeer, "Cooperative task allocation for unmanned vehicles with communication delays and conflict resolution," *Journal of Aerospace Information Systems*, vol. 13, no. 2, pp. 1–13, 2015.

[94] R. Cui, J. Guo, and B. Gao, "Game theory-based negotiation for multiple robots task allocation," *Robotica*, vol. 31, no. 6, pp. 923–934, 2013.

[95] R. G. Smith, "The contract net protocol: high-level communication and control in a distributed problem solver," *IEEE Transactions on Computers*, vol. C-29, no. 12, pp. 1104–1113, 1980.

[96] N. Buckman, H.-L. Choi, and J. P. How, "Partial replanning for decentralized dynamic task allocation," in *Proceedings of the AIAA Scitech 2019 Forum*, San Diego, CA, USA, 2019.

[97] H. Sayyaadi and M. Moarref, "A distributed algorithm for proportional task allocation in networks of mobile agents," *IEEE Transactions on Automatic Control*, vol. 56, no. 2, pp. 405–410, 2010.

[98] D. Rosa Medina, E. Malaguti, and C. Duran-Faundez, "Techniques for finding a list of solutions with increasing costs for the semi-assignment problem," in *Proceedings of the 2017 CHILEAN Conference on Electrical, Electronics Engineering, Information and Communication Technologies (CHILECON)*, pp. 1–5, IEEE, Pucon, Chile, October 2017.

[99] M. Z. Spivey, "Asymptotic moments of the bottleneck assignment problem," *Mathematics of Operations Research*, vol. 36, no. 2, pp. 205–226, 2011.

[100] M. Alighanbari and J. P. How, "A robust approach to the UAV task assignment problem," *International Journal of Robust and Nonlinear Control*, vol. 18, no. 2, pp. 118–134, 2008.

[101] S. Ossowski, V. Botti, and C. Sierra, "Agreement technologies – a computing perspective," in *Agreement Technologies, No. 8 in Law, Governance and Technology*, pp. 3–16, Springer, Dordrecht, Netherlands, 2013.