A Real-Time Heuristic for Large-Scale Workforce Task Assignment, Multi-Vehicle Routing, and Scheduling with Time-Dependent Profits

Diego Deplano, *Member, IEEE*, Carla Seatzu, *Senior, IEEE*, Mauro Franceschelli, *Senior, IEEE*

***Abstract*—This paper presents a mixed-integer linear programming formulation for a dynamic and heterogeneous multi-depot vehicle routing problem with time windows, which accounts for the following complexities: customers profitability and deadlines, skills requirements, shifting technician availability, flexible and partial scheduling, multi-day planning. Building on a gossip-based strategy, we propose a heuristic that: (i) assigns tasks to technicians based on skills compatibility and expected profitability; (ii) plans near-optimal routes within shift and time constraints; (iii) continuously updates assignments and routes in real-time (within minutes), to absorb stochastic delays, unforeseen impediments, and dynamically generated tasks; (iv) maintains private the real-time geolocation of the technicians; (v) is parallelizable in the sense that it can run simultaneously on multiple processors to speed up the execution. This paper is inspired by the real case study of DEDEM S.p.A., an international company that manages hundreds of technicians charged with refurbishing and repairing thousands of photo-booth machines spread across a wide geographic area, whose objective is to maximize the net profit by balancing per-task revenue against travel and labor costs. The performance of the proposed heuristic is demonstrated through numerical simulations over both synthetic data and real operational data provided by DEDEM S.p.A., saving up to 40% of profit loss and increasing up to 100% the net profit by efficiently handling stochastic delays.**

***Note to Practitioners*—This work is motivated by the operational challenges involved in managing a nationwide workforce responsible for carrying out tasks distributed across the country. Traditional scheduling methods typically rely on static, centralized optimization or simple proximity-based assignment, often resulting in suboptimal solutions with imbalanced workloads and poor adaptability to real-time disruptions. We provide a real-time, parallelizable algorithm that assigns and schedules tasks based on their profitability and deadlines, on technician availability, and on skill compatibility, while dynamically adjusting routes and schedules in response to delays or new task requests, maintaining private the technicians geolocation. Empirical validation using real company data demonstrates an overall increase in the net profit and the number of completed tasks when employing the proposed heuristic, preserving service quality under uncertainty. The algorithm is computationally tractable for practical problem sizes as each technician only needs to coordinate with nearby colleagues, enabling scalability to hundreds of workers and thousands of tasks and more. Companies can autonomously integrate this low-cost algorithm with their existing workforce management systems as a decision-support tool, especially when privacy concerns prevent continuous location tracking of technicians.**

***Index Terms*—Real-Time Scheduling, Multi-Vehicle Routing, Heterogeneous Vehicles, Time-dependent Profits, Gossip Algorithms, Distributed Optimization, Orienteering Problem.**

Diego Deplano, Carla Seatzu, and Mauro Franceschelli are with the DIEE, University of Cagliari, 09123 Cagliari, Italy. E-mails: *diego.deplano@unica.it*, *carla.seatzu@unica.it*, *mauro.franceschelli@unica.it*. Mauro Franceschelli is the corresponding author.

I. Introduction

In this paper we address a challenging instance of large-scale workforce management that can be abstracted as a dynamic heterogeneous multi-depot vehicle routing problem with time windows (MDVRPTW). Our motivating case study, provided by DEDEM S.p.A., involves assigning and routing a dispersed workforce of over $1,000$ technicians to refurbish and repair photo-booths across Italy (Fig. 1). At peak, each technician handles up to 20 tasks per day, resulting in $20,000$ daily tasks executed by the workforce.

In the DEDEM S.p.A. scenario, each technician starts and ends their shift at their designated home depot, visiting a sequence of photo-booths spread across Italy. Some machines generate hourly revenue once repaired, while others carry contractual deadlines with fixed payments; tasks should only be assigned if their expected revenue exceeds travel and labor costs while meeting the deadlines. Additionally, not every technician can service every machine, as skill certification requirements create compatibility constraints. Workloads should be planned over a multi-day horizon to balance daily throughput with future opportunities, and the schedule should be continuously updated in real-time (within minutes) to dynamically deal with traffic delays, sudden unavailability of technicians, new task requests, and so on. Overall, the problem combines (i) assignment (matching customers to vehicles), (ii) routing (sequencing visits to maximize net profit), and (iii) scheduling (respecting time windows and shift durations).

Motivated by these practical needs, we address a variant of the dynamic heterogeneous MDVRPTW incorporating the following complexities:

(a) each vehicle departs from one of the depots, travels to reach and execute a set of customers, and goes back to the same depot by the end of the working shift;
(b) each customer may follow two different revenue schemes: an expected hourly revenue (e.g.., machines that generate revenue over time after being serviced) or a fixed ex-



Fig. 1. The "Icona" photobooth operated by DEDEM S.p.A. (Source: https://www.wired.it/gallery/icona-cabina-fototessera-pininfarina/)

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/ DOI: 10.1109/TASE.2026.3730532

pected revenue tied to a strict deadline (e.g., mandated refurbishments with contractual completion dates);
(c) customers may remain unscheduled when their total cost—including travel and labor—exceeds their expected revenue, resulting in negative net profit;
(d) vehicles may remain unassigned when they cannot be assigned profitably to any customer;
(e) each vehicle possesses a specific set of skills and can only be assigned to customers for which they meet all the required qualifications;
(f) the planning horizon covers multiple days, which allows to generate day-by-day routes balancing immediate profit with future opportunities (which falls in the so-called *multi-trip* variation);
(g) the vehicle fleet operates under stochastic delays, unforeseen disruptions (e.g., traffic or cancellations), and a dynamically evolving customer set, all of which necessitate real-time reassignment and route optimization.

This paper offers the following original contributions:

- A novel MILP formulation for the MDVRPTW that incorporates hourly or fixed revenues for customers, flexible scheduling for both vehicles and customers, heterogeneous vehicles and customers with different skills sets, as well as multi-day planning.
- A real-time[1], paralleizable heuristic that uses randomized gossip, local optimizations between pairs of vehicles' schedules to reassign tasks and update routes, which make the heuristic scalable for large vehicle fleets and high number of tasks. Another important feature of the proposed heuristic is that it does not require continuous GPS tracking of vehicles' location.
- Empirical validation of the proposed heuristic is conducted on real DEDEM data, which demonstrate an increase in net profit ($> 6\,\%$) and in performed tasks ($> 12\,\%$) compared to the real, actual planning carried out by DEDEM. Also, it can be observed that the proposed heuristic robustly copes with unexpected delays and technician unavailability, preventing any profit loss.
- Monte Carlo simulations of the proposed heuristic were conducted on synthetic data reflecting realistic scenarios with up to 100 technicians and 500 daily tasks. On average, the results demonstrate high performance by recovering up to 40% of profit loss and increasing the net profit by as much as 100% compared with static planning, by efficiently handling stochastic delays.

Structure of the paper: Section II recalls a standard MILP formulation for the MDVRPTW, while Section III introduces the time-dependent-profit variant considered in this manuscript and motivated by the DEDEM S.p.A. case study. Section IV presents the proposed offline and online heuristic approaches for the heterogeneous MDVRPTW. Sections V and VI report, respectively, numerical simulations based on real DEDEM S.p.A. data and Monte-Carlo simulations assessing the benefits of the proposed solution. Section VII concludes the paper.

[1]In the context of our application, the term "real-time" refers to the ability to update the schedule within a few minutes after triggering events, such as delays, technician unavailability, new tasks, and similar operational changes.

A. Literature review

The MDVRPTW is a well-established, challenging combinatorial optimization problem in logistics and transportation. It involves designing routes for a fleet of heterogeneous vehicles originating from multiple depots to serve geographically dispersed customers within specified time windows, while respecting vehicle capacity and operational constraints [16]–[18]. This problem generalizes the classical Vehicle Routing Problem (VRP) [19], which itself extends the Traveling Salesman Problem (TSP) [20], spurring a vast body of heuristic and exact methods (over 1,000 articles from 1959–2008 alone) [21] due to its NP-hard complexity [22], [23]. MDVRPTW arises in numerous practical applications, such as workforce scheduling, distribution networks, and service logistics, where multiple depots are operated simultaneously to improve efficiency and customer service [17], [18]. Recent studies in autonomous multi-robot pick-and-deliver [24], [25], collision-free shared-space routing [26]–[28], and precision agriculture [29]–[32] rely on Mixed-Integer Linear Programming (MILP) formulations to ensure fair workload distribution and efficiency.

The complexity of MDVRPTW stems from the interplay between multiple depots, heterogeneous vehicle fleets, and strict time window constraints, which significantly increase the combinatorial search space compared to single-depot problems [17], [18]. Exact solution methods, such as branch-and-cut and branch-price algorithms, are effective only on small to medium-sized instances due to exponential growth in computational requirements [18], [33]. Consequently, heuristic and metaheuristic algorithms have become the predominant approaches for solving large-scale MDVRPTW instances, balancing solution quality and computational efficiency [17], [33]–[36]. Of particular interest are distributed approaches over networks [37] where local, smaller computations allows to each the optimum of a global objective function [38]–[40], such as gossip-based algorithms applied to distributed load-balancing [41], [42], task assignment [43], [44], and heterogeneous routing [36]. These methods guarantee constant-factor approximations of the global objective by iteratively exchanging and updating local routes. Recent advances in heuristic methods include adaptive large neighborhood search, variable neighborhood search (VNS), and hybrid metaheuristics that incorporate problem-specific neighborhood structures and local search strategies [17], [33]. [33] propose a two-phase decomposition combining trip generation and route assignment via a VNS-based heuristic, showing strong performance on real-world multi-depot, multi-trip instances with time windows [33], [45]–[48]. [34] develop a hybrid genetic algorithm with adaptive local search, improving travel cost reduction by enabling flexible depot assignments.

Another important branch of research has focused on variants of the vehicle routing problem that incorporate customer-associated profits [49]. Within this literature, the vast majority of studies assume that the profit obtained from serving each customer is constant over time and fully collected upon completion of service. Only a small subset of works has addressed the more challenging case of time-dependent profits, as can be seen from an examination of the most recent and

Licensed under CC BY 4.0. DOI: 10.1109/TASE.2026.3730532

TABLE I
COMPARISON OF FEATURES IN RELATED WORKS WITH TIME-DEPENDENT PROFITS.

Reference	Multi-vehicle	Multi-depot	Multi-trip	Time windows	Skills	Deadlines	Time-dependent profit
Arrival-time-dependent variations							
[1] Erkut and Zhang (1996)	✗	✗	✗	✗	✗	✗	✓
[2] Afsar and Labadie (2013)	✓	✗	✗	✗	✗	✗	✓
[3] V.F. Yu et al. (2019)	✓	✗	✗	✓	✗	✗	✓
[4] Peng et al. (2019)	✗	✗	✓	✓	✗	✗	✓
[5] Kim et al. (2020)	✗	✗	✗	✗	✗	✗	✓
[6] Kim and Kim (2022)	✗	✗	✗	✗	✗	✗	✓
Service-time-dependent variations							
[7] Tang et al. (2007)	✓	✗	✓	✗	✓	✗	✓
[8] Pietz and Royset (2013)	✓	✗	✗	✓	✗	✗	✓
[9] Erdoğan and Laporte (2013)	✗	✗	✗	✗	✗	✗	✓
[10] Gunawan et al. (2018)	✓	✗	✗	✗	✗	✗	✓
[11] Q. Yu et al. (2019)	✗	✗	✗	✗	✗	✗	✓
[12] Khodadadian et al. (2022)	✗	✗	✗	✓	✗	✗	✓
[13] Marzal and Sebastia (2022)	✓	✗	✗	✓	✗	✗	✓
Other variations							
[14] Q. Yu et al. (2022)	✓	✗	✗	✗	✗	✗	✓
[15] Li et al. (2024)	✓	✗	✗	✓	✗	✗	✓
This work	✓	✓	✓	✓	✓	✓	✓

comprehensive surveys on the topic [50]–[52]. While the authors in [51] do not refer to any work addressing time-dependent profits, they explain that the VRP with profits is often referred to as the *Orienteering Problem* (OP), whose multi-vehicle variant is called the *Team* OP (TOP) [53]. The review [50] mentions only the work of Khodadadian et al. [12] as one addressing the orienteering problem with time-dependent profit, which, however, is limited to the single-vehicle scenario. On the other hand, the review [52] mentions a few works dealing with time-dependent profits, some limited to OP [1], [9] and some addressing TOP [2], [7], [10]. Upon our own investigation, other relevant works addressing time-dependent profits exist and can be categorized in: "arrival-time-dependent profits" where the profit depends on the arrival time of the vehicle at the customer location [1]–[6]; "service-time-dependent profits" where the profit depends on the time at which customer service starts and/or ends [7]–[13]; and other variations [14], [15]. It is also worth mentioning other relevant studies that address variants of the problem in which profits vary with respect to metrics other than time [54]–[56].

Although these studies address specific variants of the multi-vehicle routing problem with time-dependent profits, they do not consider several additional complexities examined in this manuscript, which are tailored by the specific real-case study of DEDEM S.p.A.. These complexities are described below and compared with those of other time-dependent profit studies in Table I, which provides a comprehensive comparison of the features considered in each work and highlight the necessity and relevance of our study. The MILP formulation of the problem and the gossip-based heuristic proposed in this manuscript are amenable to ad hoc, non-trivial variations to cope with different scenarios in other relevant contexts, thus offering potential for replicability in other complex real-world routing problems with time-dependent profits.

II. A STANDARD MILP FORMULATION FOR THE MDVRP

We define the multi-depot vehicle routing problem (MDVRP) on a loop-free directed graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$. The set $\mathcal{V}$ of nodes is partitioned into a set $\mathcal{D}$ of technicians' depots (i.e., the vehicles) and a set $\mathcal{T}$ of tasks (i.e., the customers). Direct travel between two depots is not allowed, that is, no arcs between pairs of depots exist, so

$$\mathcal{E} = \{(i,j) \mid i,j \in \mathcal{V} \text{ where } i \neq j \text{ and } \{i,j\} \cap \mathcal{T} \neq \emptyset)\}.$$

We partition the set of links $\mathcal{E}$ into the set of arcs $\mathcal{E}^T$ between tasks' positions, namely $\mathcal{E}^T = \{(i,j) \mid i,j \in \mathcal{T}, i \neq j\}$; the set of arcs $\mathcal{E}_s^D$ starting from depots $d \in \mathcal{D}$, namely $\mathcal{E}_s^D = \{(d,j) \mid d \in \mathcal{D}, j \in \mathcal{T}\}$; the set of arcs $\mathcal{E}_t^D$ terminating at depots $d \in \mathcal{D}$, namely $\mathcal{E}_t^D = \{(j,d) \mid d \in \mathcal{D}, j \in \mathcal{T}\}$.

We associate a non-negative traversing cost c_{ij} with each arc $(i,j) \in \mathcal{E}$. The objective of MDVRP is to find a minimum cost set of exactly $|\mathcal{D}|$ disjoint circuits on graph $\mathcal{G}$ such that each depot and each task belongs to exactly one route. We use binary variables X_{ij} that equal to 1 if the path $(i,j) \in \mathcal{E}$ belongs to a circuit and equal to 0 otherwise. An integer linear programming formulation for the MDVRP is given as follows:

$$\min \quad \sum_{(i,j)\in\mathcal{E}} c_{ij} X_{ij} \tag{1}$$

$$\text{s.t.} \quad X_{ij} = 0, \quad \forall (i,j) \notin \mathcal{E}, \tag{2}$$

$$\sum_{i\in\mathcal{V}} X_{ji} = 1, \quad \forall j \in \mathcal{V}, \tag{3}$$

$$\sum_{i\in\mathcal{V}} X_{ij} = 1, \quad \forall j \in \mathcal{V}, \tag{4}$$

$$\{(i,j) \in \mathcal{E} \mid X_{ij} = 1\} \text{ contains no circuit with zero depots,} \tag{5}$$

$$\{(i,j) \in \mathcal{E} \mid X_{ij} = 1\} \text{ contains no circuit with two or more depots.} \tag{6}$$

Licensed under CC BY 4.0. DOI: 10.1109/TASE.2026.3730532

The constraints in eqs. (5)-(6) are usually called "subtour elimination constraints" and "path elimination constraints", respectively, and ensure route validity in the multi-depot vehicle routing problem. Specifically, constraint in eq. (5) prevents closed loops that include only tasks and no depot, while constraint in eq. (6) ensures that routes do not form loops passing through more than one depot. Together, these constraints ensure that each task is executed by exactly one technician. The two following subsections present these constraints in more detail.

A. Subtour elimination: A Multi-Commodity Flow formulation

A wide variety of formulations for the subtour elimination constraints exist in the literature for the standard routing problem [57], some of which have been adapted to the MDVRP in [58]. Here we adapt another formulation[2] provided by Gouveia and Pires in [59]. Our choice to use such formulation as a base start for the development of the problem formulation in our scenario stems not only from the fact that it is equivalent (in terms of tightness of the linear programming relaxation) to the seminal formulation provided by Dantzig et. al [60] (see [57, Figure 2]) and that it enjoys a polynomial number of constraints instead of an exponential number (see [57, Table 1]), but also for its more convenient interpretation of the variables that are more suitable for our derivations in future sections.

To get the idea behind the multi-commodity flow formulation, one should imagine that when passing through a position $k \in \mathcal{T}$ a commodity is collected (e.g., a receipt) and it must be brought back to a depot. The multi-commodity flow formulation of Gouveia and Pires [59] makes use of the so-called "*flow variables*" $F_{ijk} \in [0,1]$ representing the flow of commodity $k \in \mathcal{V} \setminus \mathcal{D}$ between $i,j \in \mathcal{V}$. More precisely, variables F_{ijk} may be greater than 0 only if $j \neq k$ (a commodity can not return to the task from which has been collected) and $i \notin \mathcal{D}$ (a commodity can not leave a depot), thus we have

$$F_{ikk} = 0, \qquad \forall k \in \mathcal{V} \setminus \mathcal{D}, \tag{7}$$

$$F_{ijk} = 0, \qquad \forall k \in \mathcal{V} \setminus \mathcal{D}, \forall (i,j) \notin \mathcal{E} \setminus \mathcal{E}_s^D. \tag{8}$$

The following system of inequalities models the generic subtour elimination constraints in eq. (5):

$$\sum_{i \in \mathcal{V}} F_{jik} = \sum_{i \in \mathcal{V} \setminus \mathcal{D}} F_{ijk}, \qquad \forall j,k \in \mathcal{V} \setminus \mathcal{D}, j \neq k \tag{9}$$

$$\sum_{i \in \mathcal{V}} F_{kik} = 1, \qquad \forall k \in \mathcal{V} \setminus \mathcal{D} \tag{10}$$

$$0 \leq F_{ijk} \leq X_{ij}, \qquad \forall k \in \mathcal{V} \setminus \mathcal{D}, i,j \in \mathcal{V}. \tag{11}$$

Constraints (9) ensure that one commodity k entering a task $j \neq k$ also leaves that task. Constraints (10) ensure that a commodity leaves the task to which is associated. Lower and upper bounds for the amount of flow on each arc are defined by constraints (11). In so doing, connectivity between the depots and every task is ensured, preventing the existence of subtours.

[2]For reference, the formulation we adapt for the multi-depot scenario is known as "MCF" (multi-commodity flow) in [59], and as "GP6" (after Gouveia and Pires) in [57].

B. Path elimination: Task-Depot formulation

As path elimination constraints are much less studied in the literature compared to their subtour elimination counterparts, Bektas et al in [58] have reviewed the most effective formulations and presented some new ones. According to their numerical results, one of the best formulations – in terms of LP relaxation – to pair with "*commodity flow*" formulations is what they call "*task-depot assignment*", i.e., tasks are associated with a depot by means of two-index variables V_{dk} representing the assignment of commodity $k \in \mathcal{V} \setminus \mathcal{D}$ toward depot $d \in \mathcal{D}$. The following system of inequalities models path elimination constraints in eq. (6):

$$X_{dk} \leq V_{dk}, \qquad \forall k \in \mathcal{V} \setminus \mathcal{D}, d \in \mathcal{D} \tag{12}$$

$$X_{kd} \leq V_{dk}, \qquad \forall k \in \mathcal{V} \setminus \mathcal{D}, d \in \mathcal{D} \tag{13}$$

$$\sum_{d \in \mathcal{D}} V_{dk} = 1, \qquad \forall k \in \mathcal{V} \setminus \mathcal{D} \tag{14}$$

$$X_{ij} + X_{ji} + V_{di} - V_{dj} \leq 1, \quad \forall d \in \mathcal{D}, \forall i,j \in \mathcal{V} \setminus \mathcal{D}, i \neq j. \tag{15}$$

Constraints (12)-(13) ensure that task $k \in \mathcal{V} \setminus \mathcal{D}$ is included in the circuit of depot $d \in \mathcal{D}$ if either arc (d,i) or (i,d) are used. Constraints (14) guarantee that every client is in one and only one circuit. Constraints (15) state that if a task $j \in \mathcal{V} \setminus \mathcal{D}$ is in a circuit of a depot $d \in \mathcal{D}$ and arc $(i,j) \in \mathcal{E}^T$ is used, then task $j \in \mathcal{V} \setminus \mathcal{D}$ must also be in the circuit of depot d.

III. A MILP FORMULATION FOR THE MDVRPTW WITH TIME-DEPENDENT PROFITS INSPIRED BY DEDEM S.P.A.

In this section we extend the basic formulation for the standard heterogeneous multi-depot vehicle routing problem with time windows (MDVRPTW) with objective as in eq. (1) and constraints in eqs. (2)-(4) and eqs. (7)-(15) to deal with the real-world scenario provided by DEDEM S.p.A., which requires managing a large workforce of technicians responsible for refurbishing and repairing numerous photo booths across the entire Italian territory.

We anticipate next the high-level formulation of the MILP, whose objective function and constraints are detailed in the following subsections:

$$\min \quad \text{FUEL (17)} + \text{TECH (26)} - \text{REW (24)} + \text{RISK (25)} \tag{16a}$$

$$\text{s.t.} \quad \text{eqs. (2), (7)-(9), (11)-(13), (15), (18)-(23), (27)-(33)} \tag{16b}$$

where the objective function consists of the following terms:

- FUEL is the cost incurred from fuel consumption;
- TECH is the cost associated with employing technicians;
- REW is the profit earned by completing tasks;
- RISK is the cost reflecting both the risk and the actual penalty of not completing tasks on time;

and where the constraints are related to:

- subtour elimination constraints in eqs. (7)-(9), (11), (30);
- path elimination constraints in eqs. (12),(13), (15), (31);
- multi-day constraints in eqs. (18)-(20);
- time-window constraints in eqs. (21)-(23);
- partial assignment constraints in eqs. (27)-(29);
- skill constraints in eq. (32);
- calendar constraints in eq. (33).

Licensed under CC BY 4.0. DOI: 10.1109/TASE.2026.3730532

In the following Subsections III-A-III-H, we provide a detailed explanation of the derivation of each term in the objective function and of each constraint. Finally, in Subsection III-I we prove the feasibility of the problem in eq. (16).

A. Fuel cost

The first main modification we do on the basic formulation affects the objective, where the cost now depends on the fuel consumption cost to cover the routes. To each viable path $(i,j) \in E$ we associate a distance $d_{ij} \geq 0$ (*km*) and a fuel cost $f_{ij} \geq 0$ (*eur/km*), which takes into account the average speed on that path. This allows us to reformulate the cost of the routes as follows:

$$\text{FUEL} := \sum_{(i,j)\in\mathcal{E}} f_{ij} d_{ij} X_{ij}. \tag{17}$$

B. Multi-day scheduling

To model routes that span multiple working days, such that each technician must return to its depot at the end of every day, we create a copy of each depot in $\mathcal{D}$ for every night of the planning horizon. Let $\eta \geq 0$ be an integer denoting the total number of nights. For each night $\ell = 1, \cdots, \eta$, we thus create a set $\mathcal{N}_\ell$ that contains a copy of every original depot in $\mathcal{D}$. To maintain notation consistency, we also define $\mathcal{N}_0 = \mathcal{D}$ so that $\mathcal{N}_0$ corresponds to the set of original depots and we denote by $\mathcal{N}$ the complete set of night depots, namely,

$$\mathcal{N} := \textstyle\bigcup_{\ell=1}^{\eta} \mathcal{N}_\ell, \quad \mathcal{N}_0 := \mathcal{D}.$$

For each night depot $d \in \mathcal{N}_\ell$, we define:

- $\underline{d} \in \mathcal{N}_{\ell-1}$ as the associated depot of the preceding night;
- $d_0 \in \mathcal{N}_0 = \mathcal{D}$ as the corresponding original depot.

In other words, each night depot d is linked to its predecessor night depot $\underline{d}$ and to its original depot d_0, allowing us to model multi-day routes consistently. Because night depots are logical copies used only for routing over multiple days, their geographic location does not change. Thus, distances and fuel costs are inherited from the original depot:

$$d_{dk} = d_{d_0 k}, \quad f_{dk} = f_{d_0 k}, \quad \forall d \in \mathcal{N}, k \in \mathcal{T}.$$

The MDVRP on multiple days is then defined as a directed graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ where $\mathcal{V} = \mathcal{D} \cup \mathcal{N} \cup \mathcal{T}$ and where $\mathcal{E}$ is partioned into:

- $\mathcal{E}^T = \{(i,j) \mid i,j \in \mathcal{T}, i \neq j\}$;
- $\mathcal{E}^D_s = \{(d,j) \mid d \in \mathcal{D}, j \in \mathcal{T}\}$;
- $\mathcal{E}^D_t = \{(j,d) \mid d \in \mathcal{D}, j \in \mathcal{T}\}$;
- $\mathcal{E}^N_s = \{(d,j) \mid d \in \mathcal{N}, j \in \mathcal{T}\}$;
- $\mathcal{E}^N_t = \{(j,d) \mid d \in \mathcal{N}, j \in \mathcal{T}\}$;
- $\mathcal{E}^N = \cup_{\ell=1}^{\eta} \{(\underline{d}, d) \mid d \in \mathcal{N}_\ell\} \cup \{(d, d_0) \mid d \in \mathcal{N}_\eta\}$.

Here, the last set of edges $\mathcal{E}^N$ models transitions between night depots: each night depot $d \in \mathcal{N}_\ell$ can only be reached from its predecessor night depot $\underline{d} \in \mathcal{N}_{\ell-1}$ and direct travel to the original depot $d_0 \in \mathcal{D}$ is allowed only from the night depot $d \in \mathcal{N}_\eta$. To ensure that there are no circuits involving a depot $d \in \mathcal{D}$ and passing through night depots associated with depots different from d, we impose that:

$$F_{ijd} = 0, \quad \begin{array}{l} \forall d \in \mathcal{N},\ \forall i,j \in \mathcal{D} \cup \mathcal{N} \\ \text{s.t. } (i,j) \in \mathcal{E} \wedge (i_0 \neq d_0 \vee j_0 \neq d_0), \end{array} \tag{18}$$

where $d_0, i_0, j_0 \in \mathcal{D}$ are the depots uniquely associated with $d \in \mathcal{N}$ and $i,j \in \mathcal{D} \cup \mathcal{N}$.

To ensure that the night depots $d \in \mathcal{N}$ are included in the circuits we add the following set of constraints:

$$\sum_{i\in\mathcal{V}} X_{id} = 1, \quad \forall d \in \mathcal{N}, \tag{19}$$

$$\sum_{i\in\mathcal{V}} X_{di} = 1, \quad \forall d \in \mathcal{N}. \tag{20}$$

C. Time windows for tasks and technicians

In this section we introduce the necessary notation to introduce timings within the formulation. First of all, we associate with each viable path $(i,j) \in \mathcal{E}$ a travel time $t_{ij} \geq 0$ (*hour*). Next, we also describe duration and time windows of the tasks and the depots.

We associate with each task $k \in \mathcal{T}$ a duration $\delta_k > 0$ (*hour*) denoting the amount of time needed to execute the task, as well as a starting time $T^s_k \in [0,24]$ (*hour*) and an ending time $T^e_k \in [0,24]$ (*hour*) describing the time window within which the task in that position can be executed. We associate with each depot $d \in \mathcal{D}$ a starting time $T^s_d \in [0,24]$ (*hour*) and an ending time $T^e_d \in [0,24]$ (*hour*) describing the time window $[T^s_d, T^e_d]$ within which the technicians can work every day. Thus, we associate with each night depot $d \in \mathcal{D}_\ell$ a unique depot $d_0 \in \mathcal{D}$ describing the time window $[T^s_d, T^e_d] = [T^s_{d_0}, T^e_{d_0}]$ (*hour*). By convention, we let $\delta_d = 0$ for all $d \in \mathcal{D} \cup \mathcal{N}$.

To formulate appropriate constraints, define the variables U_{ij} for all $i,j \in \mathcal{V}$ with the following interpretation: if the path (i,j) is not traversed, i.e., $X_{ij} = 0$, then U_{ij} is set to 0. Instead, if the path is traversed, i.e., $X_{ij} = 1$, then U_{ij} represents the time when position i is left. The following set of constraints imposes that all time windows are met, meaning that technicians leave/return their depots within the working time and tasks are executed when available:

$$(T^s_i + \delta_i) X_{ij} + 24 \sum_{\ell\in\mathcal{N}} F_{ij\ell} \leq U_{ij}, \quad \forall i,j \in \mathcal{V}, \tag{21}$$

$$\min\{T^e_i, T^e_j - t_{ij} - \delta_j\} + 24 \sum_{\ell\in\mathcal{N}} F_{ij\ell} \geq U_{ij}, \quad \forall i,j \in \mathcal{V}. \tag{22}$$

Since $\sum_{\ell\in\mathcal{N}} F_{ij\ell}$ equals the number of nights already elapsed when arc (i,j) is traversed, the quantity $24 \sum_{\ell\in\mathcal{N}} F_{ij\ell}$—where 24 is the number of hours in a day—represents the time shift ahead for the starting time U_{ij} from the start of the planning horizon. These constraints ensure executing a task and traveling between tasks use up the correct amount of time:

$$\sum_{j\in\mathcal{V}} U_{ij} \geq \sum_{j\in\mathcal{V}} \left(U_{ji} + (t_{ji} + \delta_i) X_{ji}\right), \qquad i \in \mathcal{V} \setminus \mathcal{D}. \tag{23}$$

D. Time-dependent profits and deadlines for tasks' execution

To prioritize task execution, we assign each task $k \in \mathcal{T}$ a profit, which can be either an hourly profit g_k or a fixed profit q_k with a deadline and a penalty; the two parameters are distinct: tasks with hourly profit have $g_k > 0$ and $q_k = 0$, while tasks with fixed profit have $g_k = 0$ and $q_k > 0$.

Licensed under CC BY 4.0. DOI: 10.1109/TASE.2026.3730532

An hourly profit $g_k > 0$ (*eur/hour*) starts accruing from the moment the task is visited and executed and continues thereafter. This is modeled by including in the objective function the following reward for executing a task with an hourly profit:

$$\text{REW_H} = \sum_{k\in\mathcal{T}} g_k \left(\sum_{i\in\mathcal{V}} [24\cdot(\eta+1)\cdot X_{ki})] - U_{ki} \right).$$

A fixed profit $q_k > 0$ (*eur*), on the other hand, is a one-time reward given if the task is completed before the deadline, expressed in days $\sigma_i \geq 0$ left to complete the task. Two cases may occur:

- $\sigma_j \leq \eta$, i.e., the deadline falls within the planning horizon. In this case, if the task $j \in \mathcal{T}$ is not scheduled, the full penalty ρ_j is incurred, because there is no opportunity to schedule the task after the planning horizon while meeting the deadline.
- $\sigma_j > \eta$, i.e., the deadline is beyond the planning horizon. In this case, if the task $j \in \mathcal{T}$ is not scheduled, a fraction of the penalty ρ_j is incurred, since it is still possible to schedule the task after the planning horizon while meeting the deadline. In particular, the scale factor we consider is given by $1/(1+\sigma_j-\eta)$.

By means of the following notation

$$[x]_\alpha = \max\{x, \alpha\}, \qquad \forall \alpha \in \mathbb{R},$$

we modify the objective function by including the following term to account for both the profit generated by assigned tasks and the risk-related penalty incurred from unassigned tasks:

$$\text{REW_F} = \sum_{j\in\mathcal{T}} \left(q_j \sum_{i\in\mathcal{V}} X_{ij} - \frac{\rho_j}{1+[\sigma_j-\eta]_0} \left(1 - \sum_{i\in\mathcal{V}} X_{ij} \right) \right).$$

This formulation promotes the assignment of tasks that are both profitable and time-sensitive, striking a balance between maximizing expected profit and minimizing the risk associated with delayed or unfulfilled tasks. We sum the hourly profit and the fixed profit into one single term as follows:

$$\text{REW} = \text{REW_H} + \text{REW_F} \tag{24}$$

We also incorporate a risk-based cost term designed to incentivize the early scheduling of tasks that are prone to failure over time. Specifically, the probability of failure for a task increases as its scheduling day approaches the deadline and becomes equal to one if the scheduling day is beyond the deadline. By letting $p_k \in (0,1)$ denote the probability of failure, we model the cost of the risk as follows:

$$\text{RISK} = \sum_{j\in\mathcal{T}} (\rho_j + q_j) \sum_{i\in\mathcal{V}} \left[p_j^{[\sigma_j-\eta+1]_0} \sum_{k\in\mathcal{D}} F_{ikj} + \cdots \right. \tag{25}$$

$$\left. + \sum_{\ell\in[1,\eta]} \left(p_j^{[\sigma_j-\eta+\ell+1]_0} - p_j^{[\sigma_j-\eta+\ell]_0} \right) \sum_{k\in\mathcal{N}_{\eta+1-\ell}} F_{ikj} \right],$$

This term captures the marginal risk increase over time, weighted by both the expected penalty ρ_j and the profit loss q_j if the task fails; so together, $\rho_j + q_j$ expresses the total loss from not completing task j, i.e., both direct (penalty) and indirect (lost revenue) losses. To model the fact that the marginal risk increases as the day of scheduling approaches the deadline, the term within the square brackets is constructed in a way that intermediate terms "telescope", that is, they cancel each other out when summed over all possible nights. As a result, the cumulative risk for each task ultimately depends only on the probability $p_j^{[\sigma_j-\eta+\ell+1]_0}$, where ℓ represents how many days before the end of the planning horizon the task is scheduled. Two cases may occur:

- $\sigma_j < \eta-\ell$, i.e., the task is scheduled after the deadline. In this case, the probability of failure is equals one, namely, $p_j^{[\sigma_j-\eta+\ell+1]_0} = 1$.
- $\sigma_j = \eta - \ell$, i.e., the task is scheduled exactly on its deadline. In this case, the probability of failure is exactly equal to the base probability, namely $p_j^{[\sigma_j-\eta+\ell+1]_0} = p_j$.
- $\sigma_j > \eta-\ell$, i.e., the task is scheduled before the deadline. In this case, the probability of failure is strictly reduces proportionally with the number of remaining days to schedule within the deadline, namely $p_j^{[\sigma_j-\eta+\ell+1]_0} < p_j$.

The sooner a task is scheduled, the smaller the risk term becomes due to the larger exponent; this reflects the lower risk as there is an ample opportunity to reschedule. Conversely, the later a task is scheduled, the higher the risk term becomes due to the smaller exponent; this reflects the high risk as there is no so much time (or none) to reschedule. Thus, the risk cost dynamically adapts to both the timing of execution and the flexibility available, incentivizing the completion of urgent tasks earlier and robustly reducing exposure to last-minute failures to mitigate both financial loss and operational risk.

E. Cost for technicians' employment

The employment of a technician $d \in \mathcal{D}$ for a day of work has a fixed cost $q_d > 0$ (*eur*). The total cost of the workforce of the technician is given by:

$$\text{TECH} = \sum_{d\in\mathcal{D}} q_d \sum_{k\in\mathcal{T}} F_{kdk} + \sum_{d\in\mathcal{N}} q_{d_0} \sum_{k\in\mathcal{T}} F_{kdk}, \tag{26}$$

where $d_0 \in \mathcal{D}$ is the depot uniquely associated with $d \in \mathcal{N}$, with the first term being the cost of the first day and the second term being the cost of subsequent days.

F. Tasks and technicians may remain unassigned

We now relax the fundamental assumption in the standard MDVRP that all technicians must be employed and all tasks must be visited. Since tasks generate profit upon completion, they should only be executed if they result in a net profit, considering the costs of the route, such as fuel and technician expenses. Similarly, not all technicians need to be employed if a smaller number is sufficient to complete the tasks. Thus, we relax constraints in eqs. (3)-(4) as follows:

$$\sum_{i\in\mathcal{V}} X_{ji} \leq 1, \qquad \forall j \in \mathcal{V}, \tag{27}$$

$$\sum_{i\in\mathcal{V}} X_{ij} \leq 1, \qquad \forall j \in \mathcal{V}, \tag{28}$$

$$\sum_{i\in\mathcal{V}\setminus\mathcal{D}} (X_{ji} - X_{ij}) = 0, \qquad \forall j \in \mathcal{V}\setminus\mathcal{D}, \tag{29}$$

Licensed under CC BY 4.0. DOI: 10.1109/TASE.2026.3730532

where the last constraint ensures that if a position $i \in \mathcal{V}$ is visited, then it is also left: this was not necessary before because all positions were forced to be visited. Similarly, we relax the constraints in eq. (10) and (14) as follows:

$$\sum_{i \in \mathcal{V}} F_{kik} = \sum_{i \in \mathcal{V}} X_{ki}, \qquad \forall k \in \mathcal{V} \setminus \mathcal{D}, \tag{30}$$

$$\sum_{d \in \mathcal{D}} V_{dk} \leq 1, \qquad \forall k \in \mathcal{V} \setminus \mathcal{D}. \tag{31}$$

G. Skills required for the tasks' execution

Let $\mathcal{S}$ be the set of skills and denote by:

- $\mathcal{S}_D \subset \mathcal{D} \times \mathcal{S}$ be the set of pairs (j, s) such that technician $j \in \mathcal{D}$ has the skill $s \in \mathcal{S}$;
- $\mathcal{S}_T \subset \mathcal{T} \times \mathcal{S}$ be the set of pairs (k, s) such that task $k \in \mathcal{T}$ requires the skill $s \in \mathcal{S}$.

With this notation, task $k \in \mathcal{T}$ cannot be executed by technician $j \in \mathcal{D}$ if there is a skill $s \in \mathcal{S}$ such that $(k, s) \in \mathcal{S}_T$ and $(j, s) \notin \mathcal{S}_D$, yielding the constraints:

$$\sum_{i \in \mathcal{V} \setminus \mathcal{D}} F_{ijk} = 0, \quad \forall s, j, k : (k, s) \in \mathcal{S}_T \wedge (j, s) \notin \mathcal{S}_D. \tag{32}$$

H. Calendar of technicians' availability

Let $\mathcal{U} \subset \mathcal{D} \times \{0, \cdots, \eta\}$ be the set of pairs (j, ℓ) such that technician $j \in \mathcal{D}$ is unavailable on day $\ell \in \{0, \cdots, \eta\}$. For the sake of clarity, in this section only, let $d_{j,0} \in \mathcal{N}_0$ denote the original depot of technician $j \in \mathcal{D}$, and let $d_{j,r} \in \mathcal{N}_r$, $r = 1, \ldots, \eta$, denote the copy of the same depot at night r. Thus, the nodes $d_{j,0}, d_{j,1}, \ldots, d_{j,\eta}$ correspond to the same physical depot of technician j, but to different time layers. With this notation, technician $j \in \mathcal{D}$ cannot execute any task on the ℓ-th day if $(j, \ell) \in \mathcal{U}$, which can be modeled by the following set of constraints:

$$\begin{cases} X_{d_{j,\ell} d_{j,\ell+1}} = 1 & \text{if } \ell \leq \eta - 1, \\ X_{d_{j,\eta} d_{j,0}} = 1 & \text{if} \ell = \eta, \end{cases} \qquad \forall (j, \ell) \in \mathcal{U}. \tag{33}$$

The constraints $X_{d_{j,\ell} d_{j,\ell+1}} = 1$ enforces the selection of a dummy idle arc from the depot copy of technician j at night ℓ to the depot copy of the same technician at night $\ell + 1$. Since these two nodes represent the same physical depot in two consecutive time layers, this means that technician j remains at the depot during day ℓ and, hence, no task is executed on that day. A similar argument applies to the constraints $X_{d_{j,\eta} d_{j,0}} = 1$, which enforces the dummy idle arc from the last night depot to the original depot and, therefore, prevents technician j from executing tasks on the last day.

I. Feasibility analysis and geolocation privacy

We now prove that the proposed optimization problem in eq. (16) always admits a feasible solution.

Proposition 1: *The heterogeneous multi-depot vehicle routing problem with time windows defined by the MILP in* (16) *is guaranteed to admit at least one feasible solution.*

Proof: The key feature of the MILP formulation in eq. (16) is that it allows for tasks and technicians to be left unassigned (see constraints (27)-(31) described in Section III-F). It is thus always possible to construct a trivial solution by assigning no tasks to any technician, i.e., $X_{ij} = F_{ijk} = U_{ij} = 0$ for all $i, j \in \mathcal{V}$ and $k \in \mathcal{V} \setminus \mathcal{D}$, which satisfies constraints (16b). □

Remark 1: *Algorithm 1 preserves the privacy of technicians' real-time geolocation by approximating each technician's true position as the location of their ongoing task under execution. Specifically, at Line 9, the algorithm assigns each technician's location to that of their ongoing task. Subsequently, at Line 11, the Delaunay graph—which connects each technician to their geographically nearest peers—is updated based on these approximated positions. This design avoids the need for continuous monitoring of precise technician locations and restricts the sharing of geospatial information to coarse, task-associated points only, thereby ensuring that technicians' location privacy is maintained throughout the scheduling process during the execution of Algorithm 1.*

IV. The Proposed Solution: A Heuristic Gossip-Based Algorithm

In this section, we present a heuristic, sub-optimal approach designed to tackle the problem described in Section III, particularly when the problem size makes exact methods computationally infeasible. The proposed method can operate both *offline*, to compute an initial feasible solution, and in *real-time*, to iteratively improve it in response to dynamic, real-world events such as:

- Delays or disruptions caused by traffic, unexpected complications during task execution, or the sudden unavailability of technicians. In such cases, the algorithm can re-optimize schedules to mitigate potential loss;
- Arrival of new technicians or tasks, prompting the algorithm to update the current solution in real time to maximize overall profit.

Our approach, detailed in Algorithm 1, is inspired by *gossiping*, a parallelizable computation paradigm in which information propagates across a network through pairwise exchanges. In our context, technicians are modeled as nodes in a communication graph, and edges represent potential pairs for local schedule optimization. We denote by $\mathcal{E}_A \subseteq \mathcal{D} \times \mathcal{D}$ the subset of technicians pairs that can be selected for possible optimization by Algorithm 1, and we denote each pair in $\mathcal{E}_A$ as an "active edge". At each iteration, the algorithm randomly selects an active edge (i.e., a pair of technicians) within $\mathcal{E}_A$ and solves a localized, smaller optimization problem involving their currently assigned tasks (if any) along with a subset of unassigned tasks. More precisely, condition (i) restricts the technician set to $\mathcal{D}_{loc} = \{i^\star, j^\star\}$; condition (ii) limits the task set to $\mathcal{T}_{loc} = \mathcal{T}_\star \cup \mathcal{T}_m$, where $\mathcal{T}_\star = \{t \in \mathcal{T} : t \text{ is assigned to either } i^\star \text{ or } j^\star\}$ and $\mathcal{T}_m \subseteq \{t \in \mathcal{T} : t \text{ is unassigned}\}$ with $|\mathcal{T}_m| \leq m$; condition (iii) fixes the routing variables X_{ij} and timing variables U_{ij} of already completed or ongoing tasks to their current values, while there is no need to fix $F_{ij\ell}$, whose value is uniquely determined by constraints (11) and (22). The algorithm is parallelizable in the sense that multiple instances can be executed simultaneously, with each instance selecting a different active edge (assuming

Licensed under CC BY 4.0. DOI: 10.1109/TASE.2026.3730532

no two selected edges share a node) and performing local optimization in parallel. If the new assignment/routing/schedule improves the objective, the change is accepted and propagated; otherwise, the edge is temporarily deactivated, i.e., removed from $\mathcal{E}_A$). The process continues until no further improvements can be made through any technician pair, i.e., $\mathcal{E}_A$ is empty). This method is particularly efficient for two main reasons: (i) each local problem is significantly smaller and more tractable than the global one, and (ii) local optimizations for disjoint technician pairs can be performed in parallel, enabling scalable, real-time computation.

Since the number of possible pairs among $n \in \mathbb{N}$ technicians grows quadratically with n, we restrict our gossip exchanges to a sparse subset of edges determined by the technicians' geographic distribution. To this end, we employ the Delaunay graph [61], [62], an example of which is depicted in Fig. 2.

Definition 1: *Let $\mathcal{V}$ be any finite set of points in the plane, and let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ be a triangulation of those points. $\mathcal{G}$ is the* Delaunay graph *of $\mathcal{V}$ if and only if no point of $\mathcal{V}$ lies inside the circumcircle of any triangle in $\mathcal{G}$.*

The Delaunay graph offers two key advantages (Fig. 2):

- *Locality.* Each node is connected only to its nearest neighbors, reflecting geographic proximity.
- *Sparsity.* For $n \geq 3$, it has at most $3n - 6$ edges, thus growing linearly instead of quadratically with n.

In Algorithm 1, we build and maintain a Delaunay graph over the technicians' current positions. Initially (offline), each technician's position is its depot; during the day (real-time), whenever a technician departs for a new task or falls behind schedule, its position is updated to that of the current task under execution, and the graph is recomputed. To avoid redundant local optimizations, each edge carries an *active* flag:

- *Initialization:* all edges are active.
- *After each local optimization on edge $(i^\star, j^\star)$:*
  - If the schedule of $i^\star$ or $j^\star$ changed, all edges incident to that node are reactivated.
  - In every case, edge $(i^\star, j^\star)$ is deactivated.
- *When the Delaunay graph changes:* edges that persist maintain their status; newly created edges are activated.
- *Upon any technician's delay or task change:* all edges incident to that technician's node are reactivated.

This edge-activation logic ensures that, once no active edges remain, no further local improvements are possible and the algorithm becomes idle until some edges are reactivated.

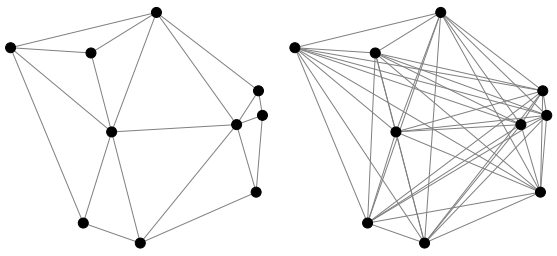


Fig. 2. Comparison between the number of arcs in the Delaunay graph (left) versus the complete graph (right).

Algorithm 1 Real-Time Task Assignment, Routing, and Scheduling for a Large-Scale Workforce

Input: Sets of technicians $\mathcal{D}$ and tasks $\mathcal{T}$, Delaunay graph $\mathcal{G}_D = (\mathcal{D}, \mathcal{E}_D)$, number of nights $\eta \geq 0$, and integer $m \geq 0$.

1: Initialize active edges: $\mathcal{E}_A \leftarrow \mathcal{E}_D$
2: **repeat** indefinitely steps 2-22:
3: **while** $\mathcal{E}_A = \emptyset$
4: **if** technician $i^\star \in \mathcal{D}$ starts a task
5: Mark incident edges of agent $i^\star$ as active: $\mathcal{E}_A \leftarrow \mathcal{E}_A \cup \{(i,j) \in \mathcal{E}_D \mid i = i^\star \text{ or } j = i^\star\}$
6: **end if**
7: Update the set of technicians $\mathcal{D}$ and tasks $\mathcal{T}$
8: **end while**
9: Update technician positions based on ongoing tasks
10: Save old Delaunay graph's edges: $\mathcal{E}_D^- \leftarrow \mathcal{E}_D$
11: Update Delaunay graph $\mathcal{G}_D = (\mathcal{D}, \mathcal{E}_D)$
12: Mark old edges as inactive: $\mathcal{E}_A \leftarrow \mathcal{E}_A \setminus \mathcal{E}_D^-$
13: Mark new edges as active: $\mathcal{E}_A = \mathcal{E}_A \cup (\mathcal{E}_D \setminus \mathcal{E}_D^-)$
14: Select a random edge $(i^\star, j^\star) \in \mathcal{E}_A$
15: Solve the local optimization (16) involving:
(i) only technicians $i^\star, j^\star$
(ii) their assigned tasks + m unassigned tasks
(iii) fixed scheduling of past/current tasks
16: **if** the new solution improves the objective of (16)
17: **for** agent in $\ell \in \{i^\star, j^\star\}$ who changed assignment
18: Mark incident edges of agent ℓ as active: $\mathcal{E}_A \leftarrow \mathcal{E}_A \cup \{(i,j) \in \mathcal{E}_D \mid i = \ell \text{ or } j = \ell\}$
19: **end for**
20: **return** updated assignment and route for $i^\star, j^\star$
21: **end if**
22: Mark edge $\{(i^\star, j^\star)$ inactive: $\mathcal{E}_A \leftarrow \mathcal{E}_A \setminus \{(i^\star, j^\star)\}$
23: Update the set of technicians $\mathcal{D}$ and tasks $\mathcal{T}$

Theorem 1: *Consider any point in time during the execution of Algorithm 1 and assume that from this point onward, no technician starts a new task and that the sets of technicians and tasks remain fixed. Then, the algorithm converges in finite number of iterations to a feasible solution where no technician can improve its local objective function by exchanging tasks with neighboring technicians.*

Proof: Algorithm 1 maintains a set of active edges $\mathcal{E}_A$ (see Line 5) representing pairs of technicians whose combined routes can be potentially improved. Consider any point in time during the execution of Algorithm 1 and, without loss of generality, assume that $\mathcal{E}_A \neq \emptyset$. An edge $(i,j) \in \mathcal{E}_A$ becomes inactive (i.e., it is removed from $\mathcal{E}_A$ as in Line 22) after a local optimization involving technicians i and j is executed, and it is reactivated (i.e., it is added to $\mathcal{E}_A$ as in line 18) only if the assignment of either technician i or j changes (under the assumption that no technician starts a new task, and the sets of technicians and tasks remain unchanged). Also, a technician can change assignment only if a new local assignment is found with a strictly improved (decreased) objective function. Since the number of tasks and technicians is finite, then the number of possible assignments is finite as well. Thus, there exists a constant lower bound to the minimum change to the global

Licensed under CC BY 4.0. DOI: 10.1109/TASE.2026.3730532

objective function in eq. (16) due to a local optimization and, in turn, there exists only a finite number of possible improvements to eq. (16). Eventually, no improvements are possible and all edges become inactive, which means that the set of active edges $\mathcal{E}_A$ becomes empty, i.e., $\mathcal{E}_A = \emptyset$. Algorithm 1 then waits for new active edges indefinitely (see Lines 3-8) because by assumption no technician starts a new task and the set of technicians remain fixed. This means that the algorithm terminates in a finite number of iterations. □

V. Empirical performance evaluation: a case study by DEDEM S.p.A.

This section presents a numerical evaluation of Algorithm 1, based on operational data provided by DEDEM S.p.A., an international company specialized in photo booth services since 1962, when it installed its first machine in Rome, Italy. The analysis is conducted using a regional subset of DEDEM S.p.A.'s operational data, consisting of 18 technicians (out of over 1,000) performing approximately 1,000 maintenance tasks (out of up to 20,000 daily across the full national operation) over a 5-day workweek, covering five Italian regions: Lazio, Campania, Toscana, Umbria, and Abruzzo.

Table II summarizes the key operational parameters used to configure the simulation framework, as described in Section III. The actual technician routes are shown in Fig. 3, which include 831 tasks (out of 940 tasks, i.e., $\approx 88.4\%$ of the total) completed within five working days. The expected profit, calculated according to eq. (16a) over a full week (including the weekend), is equal to € 365,793. The optimized technician routes are shown in Fig. 4, which include 937 tasks (representing approximately 99.7% of the total) completed within five working days. The expected profit, calculated according to eq. (16a) over a full week (including the weekend), is equal to € 389,100, with an improvement of € 23,307 ($> 6\%$), while almost all tasks have been scheduled.

Fig. 3 shows that DEDEM currently assigns tasks to technicians primarily based on geographic proximity. Each technician is exclusively assigned to machines within their designated area, and is not permitted to service machines assigned to others. In contrast, Fig. 4 shows that Algorithm1 allows routes without such constraints. This enables more tasks to be scheduled within the same time window (in this case, a working week), as technicians can assist with tasks in geographical areas that are not nearby, while still maintaining a natural preference for those closer to their location.

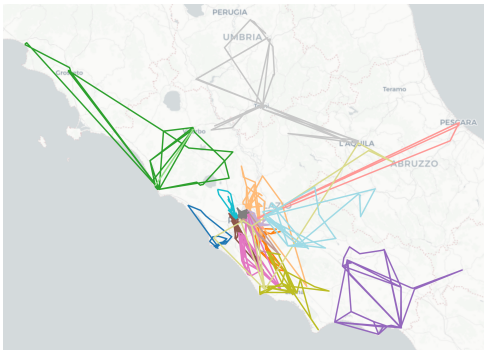


Fig. 3. Real routes retrieved from DEDEM S.p.A.'s data.

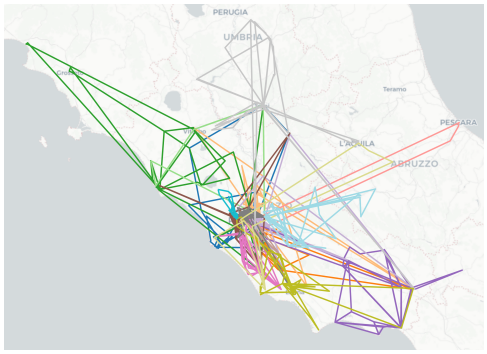


Fig. 4. Routes obtained by Algorithm 1.

TABLE II
Real operational data provided by DEDEM S.p.A. that are used to configure the simulation framework of Section V. The table describe both the real-world value of such parameters (middle column) and the corresponding value for the parameters of the optimization problem in eq. (16) (right column).

Description	Real-World Value	Optimization Parameter
Length of the optimization horizon	5 days	$\eta = 4$
Number of technicians	18 technicians	$\lvert\mathcal{D}\rvert = 18$
Number of tasks	940 tasks	$\lvert\mathcal{T}\rvert = 940$
Number of skills	4 skills	$\lvert\mathcal{S}\rvert = 4$
Geographical area	Square of 300 km side	$d_{ij} \in [0, 450]$, $\forall i, j \in \mathcal{V}$
Travel times	From Google Maps APIs	$t_{ij} \in [0, 5]$, $\forall i, j \in \mathcal{V}$
Fuel cost	€ 0.10 per km for all roads	$f_{ij} = 0.10$, $\forall i, j \in \mathcal{V}$
Technician cost	€ 50 per day	$q_i = 50$, $\forall i \in \mathcal{V} \setminus \mathcal{T}$
Technicians working hours	From 07:00 AM to 6:30 PM	$T_i^s = 7$, $T_i^e = 18.5$, $\forall i \in \mathcal{V} \setminus \mathcal{T}$
Task execution window	Fully availbale within working hours	$T_i^s = 7$, $T_i^e = 18.5$, $\forall i \in \mathcal{T}$
Task duration	Between 1 minute and 5 hours	$d_i \in [0.02, 5]$, $\forall i \in \mathcal{T}$
Task profit (hourly)	Between € 1 and € 20 per hour	$g_i \in \{0\} \cup [0.5, 20]$, $\forall i \in \mathcal{T}$
Task profit (fixed)	Between € 350 and € 2,000	$q_i \in \{0\} \cup [350, 2,000]$, $\forall i \in \mathcal{T}$
Deadlines	Spread across the horizon	$\sigma_i\{0, 1, 2, 3, 4\} \cup \{\infty\}$, $\forall i \in \mathcal{T}$
Penalties for missed deadlines	Between € 90 and € 600	$\rho_i \in \{0\} \cup [90, 600]$, $\forall i \in \mathcal{T}$
Probability of failure	5% each day	$p_i \in \{0\} \cup \{0.05\}$, $\forall i \in \mathcal{T}$

Licensed under CC BY 4.0. DOI: 10.1109/TASE.2026.3730532

The following Section V-A discusses the offline solution computed by means of Algorithm 1, starting from the initial assignment given by the actual routes, which took about 5 hours to compute. Subsequently, Section V-B discusses the benefit of using Algorithm 1 in real-time to reduce the bad impact of unexpected, disrupting events.

A. Heuristic offline solution

The complete optimization problem—considering all 18 technicians and 940 tasks—becomes really hard to solve, as it contains roughly a billion variables, including a million boolean ones. In contrast, Algorithm 1 manages complexity by restricting each local optimization to just two technicians, their assigned tasks, and up to m unassigned tasks (with $m = 10$ in our simulations). Although new tasks are gradually introduced with each iteration, every step still focuses only on two technicians and a small subset of tasks. This distributed approach makes early optimizations easier to solve and allows complexity to grow gradually as more tasks accumulate, maintaining it much lower than that of the whole problem. On average, by the end of the process, each local optimization does not contain more than a million variables (including about ten thousands boolean variables), which is only about 0.1% of the total problem size. Moreover, the solver leverages the previous iteration's solution as a warm start. By reusing the incumbent basis and variable values, it immediately has a feasible (though not optimal) solution. This reduces the number of simplex iterations and branch-and-bound nodes needed, leading to significantly faster convergence time.

We implemented Algorithm 1 in Python, using Gurobi 12 (Academic License) as the optimization solver. To compute a solution for the upcoming workweek, we simulate the execution of Algorithm 1 over the weekend. We impose the following constraints on each local optimization:

- an optimality gap of 10%;
- a time limit of 15 minutes;
- an horizon time of 5 days.

The algorithm ran from Saturday at 8 AM to Saturday at 1 PM (5 hours in total). The routes of the heuristic solution are illustrated in Fig. 4, encompassing 937 tasks ($\approx 99.7\%$ of the total available) completed over a span of 5 workdays. The expected profit, computed as per eq. (16a) over a complete week (considering also the weekend), amounts to € 389.100. When compared with the real routes, the solution found by the proposed heuristic allows to perform 106 more tasks ($> 12\%$) with an extra profit of € $23,307$ ($> 6\%$). These results are summarized together with the real-time results in Table III.

B. Heuristic real-time solution

We now consider the scenario in which an unexpected event does not allow the normal execution of the tasks in the schedule and show how the real-time execution of Algorithm 1 allows to mitigate the negative effects of the event, i.e., it allows to reduce the loss of profit for the company. The unfortunate event taken into consideration is the sudden unavailability of one technician on the first day, to which were assigned 15 tasks, which would cause a profit loss of € $6,472$ if no action is taken.

TABLE III
PERFORMANCE OF ALGORITHM 1 ON REAL DEDEM S.P.A. DATA. IN THE OFFLINE SCENARIO, THE BASELINE IS DEDEM'S CURRENT PROXIMITY-BASED PLAN. IN THE REAL-TIME SCENARIO (ONE TECHNICIAN UNAVAILABLE, 15 TASKS AFFECTED), THE BASELINE IS THE STATIC PLAN WITH NO REASSIGNMENT.

Scenario	Metric	Baseline	Algorithm 1
Offline	Tasks completed	831/940	937/940
	Expected profit (€)	365,793	389,100
	Profit change (€)	—	+23,307 (>6%)
Real-time (disruption)	Tasks reassigned	0 / 15	14 / 15
	Profit change (€)	−6,472 (>3.5%)	+1,752 (>0.5%)

We simulate the real-time execution of Algorithm 1 during the first day of the workweek, where the 15 tasks originally assigned to the technician who is no longer available are now available for assignment to other technicians, provided it results in increased profit. The objective is that of trying to re-assign these tasks to other technicians during the working day, possibly substituting less profitable tasks. We impose the following constraints on each local optimization:

- an optimality gap of 0%;
- a time limit of 5 minutes;
- an horizon time of 2 days.

Within the first hour, the algorithmperformed 60 local optimizations (one optimization per minute on average) and assigned 14 of the 15 tasks. It operated in real time alongside the technicians, incrementally updating their future schedules by exchanging or adding tasks whenever a local pair of technicians was optimized. On average, Algorithm 1 completed one local optimization and added up to m unassigned tasks (with $m = 5$ in our simulations). After the first hour, no additional tasks were introduced, but the algorithm continued to refine the routes for the remainder of the day whenever doing so was beneficial. We note that whenever Algorithm 1 computes new routes for a pair of technicians whose positions in the graph have changed during the optimization (because they completed a task and started a new one), the computed solution is discarded, as it is no longer feasible; such case is, in practice, quite unlikely. By day's end, the initial profit loss was fully recovered, and the expected total profit rose by € $1,752$ ($> 0.5\%$). This improvement stems from two main factors: 1) Reassigning tasks when an employee became unexpectedly unavailable on the first day; 2) Local optimizations for new technician pairs, guided by a time-varying Delaunay graph. These results are summarized in Table III.

C. Algorithm tuning and management insights

The performance of Algorithm 1 is mainly influenced by two parameters: the number of nights η included in each local optimization and the number of unassigned tasks m introduced at each iteration. Extending the planning horizon enlarges the search space and eases daily congestion by allowing tasks to shift across days, typically improving solution quality but at the cost of higher computational time; shorter horizons reduce complexity but limit flexibility. Likewise, increasing m enhances the chance of finding profitable exchanges while making each local improved solution longer to be found.

Licensed under CC BY 4.0. DOI: 10.1109/TASE.2026.3730532

DEDEM's experience indicates that a horizon of five days with about ten unassigned tasks offers an effective balance for offline planning, while a shorter two-day horizon with roughly five tasks is preferable in real time, enabling quick reactions to disruptions without sacrificing improvement potential. These choices illustrate a broader tradeoff between exploration and responsiveness: larger horizons and task sets support reshaping weekly schedules, whereas smaller ones facilitate fast local corrections. As operational conditions evolve, such as changes in task density, deadline patterns, or technician availability, these parameters may be revisited.

From a managerial perspective, the case study also highlights that the heuristic naturally promotes cross-regional collaboration, suggesting that relaxing strict territorial boundaries can alleviate workload imbalances and increase throughput. Because profitability parameters directly shape task prioritization, maintaining accurate estimates of revenues and service durations is essential. Even moderate inaccuracies can cause the algorithm to misjudge which tasks are worthwhile or underestimate the time required for certain routes. Investing in reliable data collection—through updated profit models, consistent technician feedback, and periodic validation of tasks duration—therefore has a clear impact on scheduling quality. On the other hand, the benefits of real-time re-optimization depend strongly on how early disruptions are reported. Timely communication about delays, cancellations, or technician unavailability preserves the flexibility needed to reassign tasks, avoid emerging bottlenecks, and coordinate nearby technicians efficiently. Late reporting greatly narrows this window of opportunity, reducing the algorithm's corrective power and limiting the portion of lost profit that can be recovered.

VI. Benchmarking and Monte Carlo Simulations on Synthetic Data

This section presents two complementary numerical studies, both conducted on synthetic problem instances. Section VI-A compares Algorithm 1 against state-of-the-art centralized solvers on small-to-medium instances (up to 10 technicians and 240 tasks), averaging results over 10 randomly generated instances per configuration. Sections VI-B–VI-C then present a comprehensive Monte Carlo simulation study on larger instances (up to 100 technicians and 500 tasks), whose setup is described in Section VI-B and whose results are discussed in Section VI-C; the code to reproduce these simulations is available at https://github.com/diegodeplano/mc_tase.

A. Benchmarking Algorithm 1 Against a state-of-the-art solver

We compare in Table IV the performance of Algorithm 1 with a state-of-the-art centralized solver, which addresses the entire problem at once. Specifically, we use Gurobi 12 (with Academic License) as the optimization solver under different tuning settings:

- Using standard meta-heuristics[3] for a total of 5% of the total running time, called hereafter *Heur5*.
- Using the proprietary Gurobi heuristic[4] known as the *No-Relaxation (NoRel) heuristic* for the entire running time.

We note that all three methods are heuristic, as none guarantees global optimality. Algorithm 1 is solver-agnostic and solves each local subproblem—involving only two technicians—with any MILP solver; in our implementation we use Gurobi. Instead, Gurobi (NoRel) and Gurobi (Heur5) use Gurobi directly to the full global MILP.

TABLE IV
Performance comparison between Algorithm 1, the Gurobi proprietary heuristic that skips the root relaxation (NoRel), and the standard Gurobi solver equipped with its full suite of meta-heuristics run for 5% of the total running time (Heur5). All solvers are warm-started with a proximity-based solution, and the table reports the percentage improvement of the expected profit defined in (16) with respect to this initial solution.

Size		Time	Percentage improvement (%)								
			Algorithm 1			Gurobi (NoRel)			Gurobi (Heur5)		
Emp	Tasks	(min)	Min	Avg	Max	Min	Avg	Max	Min	Avg	Max
3	30	3	2.7	14.9	52.8	2.4	14.9	52.9	2.2	14.2	52.7
4	60	3	3.7	8.9	16.3	0.4	4.7	12.9	0.0	0.0	0.0
5	90	6	7.3	11.7	19.8	0.3	2.4	12.8	0.0	0.0	0.0
6	120	10	10.0	17.0	25.4	0.0	0.4	2.1	0.0	0.0	0.0
7	150	12	11.3	15.4	21.3	0.0	0.0	0.0	0.0	0.0	0.0
8	180	13	8.8	13.1	18.0	0.0	0.0	0.0	0.0	0.0	0.0
9	210	20	7.9	13.2	16.9	0.0	0.0	0.0	0.0	0.0	0.0
10	240	21	10.3	16.3	29.4	0.0	0.0	0.0	0.0	0.0	0.0

Each experimental configuration included:

- *Number of employees*: $n \in \{3, 4, 5, 6, 7, 8, 9, 10\}$;
- *Number of tasks*: $30 \times (n - 2)$ to ensure scalability;
- *Number of instances per configuration*: 10
- *Algorithms compared*: Algorithm 1, NoRel, Heur5.

Results averaged over 10 instances are presented in Table IV. The table shows the percentage improvement achieved by the three methods with respect to a proximity-based baseline solution (see Subsection VI-C for a detailed explanation) used as a warm start. For fairness, the Gurobi solvers have been allotted the same runtime required by Algorithm 1 to terminate in any configuration (reported in the "Time" column of Table IV). For Gurobi (Heur5) with $n \geq 4$, this budget was insufficient to complete the LP relaxation or find improved integer solutions thereafter; for Gurobi (NoRel) with $n \geq 7$, the problem proved too large even bypassing the LP relaxation.

The results show that Algorithm 1 is the only method that consistently improves the given solution across all problem sizes, with average improvements ranging from 8.9% to 17.0%. In contrast, the centralized methods provided by the Gurobi solvers scale poorly as the problem size increases. Both methods perform as well as Algorithm 1 for the smallest instances considered, i.e., with 3 employees and 30 tasks. For slightly larger instances, namely with 4 employees and 60 tasks, the Heur5 method fails to improve the initial solution, while the NoRel method finds improved solutions that are still worse than those obtained by Algorithm 1. For even larger instances, up to 6 employees and 120 tasks, the NoRel method continues to find only slightly improved solutions,

[3]They include relaxation induced neighborhood search (RINS), local branching, relaxation-based rounding, feasibility pump, zero-objective heuristics, sub-MIP improvement heuristics, and solution polishing procedures (see docs.gurobi.com/projects/optimizer/en/current/reference/parameters.html.)

[4]The NoRel heuristic attempts to generate feasible solutions directly without relying on the root LP relaxation, a strategy that is particularly useful for very large problem instances in which solving the LP relaxation is highly time-consuming.

Licensed under CC BY 4.0. DOI: 10.1109/TASE.2026.3730532

whose quality deteriorates rapidly. Finally, for instances with 7 employees and 150 tasks or more, it fails to find any improved solutions.

To compare the three algorithms fairly across multiple problem instances, we applied the Friedman test, a non-parametric statistical test suitable when the same set of instances is evaluated by all algorithms and when performance measures may not follow a normal distribution [63]. For each instance, the algorithms are ordered from best to worst based on their percentage improvement over the proximity baseline; these orders are converted into ranks (1 = best, 3 = worst), and the test evaluates whether the average ranks differ more than expected by chance. The null hypothesis is that all algorithms perform equally, meaning their average ranks are the same; statistical significance is assessed using the following metrics:

- χ^2 (chi-squared) statistics, which measures how large are the differences between algorithms, with higher values indicating stronger differences and vice versa;
- p-value, which measures the statistical significance, with values below 5% indicating differences that are statistically significant (not due to random chance), and values above 0.05 denoting differences that could be due to random chance.

The Friedman test was implemented using Python's SciPy library. Table V reports, for each problem size, the Friedman chi-squared statistic, the corresponding p-value, an indicator of whether the result is significant at the 5% level, and the algorithm with the best average rank. Across all sizes, the p-values are small enough to reject the null hypothesis, indicating statistically significant differences among the algorithms. Table VI then lists the average ranks themselves, where lower values indicate better performance; these averages offer an intuitive summary of how often each algorithm outperforms the others on a per-instance basis. Overall, the results confirm that solutions provided by Algorithm 1 outperforms those provided by the centralized solvers of Gurobi.

TABLE V
FRIEDMAN TEST RESULTS

(Emp.,Tasks)	χ^2 statistic	p-value	Significant	Best method
(3,30)	6.20	0.0450	Yes	Algorithm 1
(4,60)	20.00	0.0000	Yes	Algorithm 1
(5,90)	20.00	0.0000	Yes	Algorithm 1
(6,120)	18.73	0.0001	Yes	Algorithm 1
(7,150)	20.00	0.0000	Yes	Algorithm 1
(8,180)	20.00	0.0000	Yes	Algorithm 1
(9,210)	20.00	0.0000	Yes	Algorithm 1
(10,240)	20.00	0.0000	Yes	Algorithm 1

TABLE VI
AVERAGE RANKS (LOWER IS BETTER)

(Emp.,Tasks)	Algorithm 1	Gurobi (NoRel)	Gurobi (Heur5)
(3,30)	1.5	1.9	2.6
(4,60)	1.0	2.0	3.0
(5,90)	1.0	2.0	3.0
(6,120)	1.0	2.35	2.65
(7,150)	1.0	2.5	2.5
(8,180)	1.0	2.5	2.5
(9,210)	1.0	2.5	2.5
(10,240)	1.0	2.5	2.5

B. Set-up description

We consider all combinations across a number of employees equal to $|\mathcal{D}| \in \{10, 50, 100\}$ and a number of tasks equal to $|\mathcal{T}| \in \{100, 250, 500\}$. Each instance is defined by a user-specified number of employees (interpreted as vehicles or depots) and tasks (customer requests). The values of these parameters are chosen to enable scalability analysis, ranging from small-scale to large-scale scenarios. The spatial distribution of both employees and tasks is modeled by sampling their coordinates uniformly at random over a $50\,\mathrm{km} \times 50\,\mathrm{km}$ grid. This choice captures the geographical dispersion typical of regional field service operations, while ensuring that each location is unique.

Time windows are incorporated to reflect practical scheduling constraints. Employee shifts are fixed, starting at 7:00 and ending at 17:00, with a pause for lunch break of 2 hours, corresponding to a standard 8-hours workday. For each task, the earliest and latest allowable service times are sampled independently from uniform distributions: the earliest start is drawn from 7:00 to 13:00, and the latest finish from 11:00 to 17:00. This approach produces a diverse set of overlapping time windows, representative of real-world service requests. Task durations are sampled from a normal distribution with a mean of 15 minutes and a standard deviation of 5 minutes, truncated to ensure strictly positive values. This reflects the empirical variability observed in job lengths for maintenance and repair activities.

Tasks are further differentiated by their profit structure: some yield ongoing hourly profit after completion (such as maintenance tasks that restore revenue-generating machines), while others offer a fixed profit subject to a strict deadline, with penalties for late or missed service. Hourly profits are sampled from a normal distribution with mean € 5 per hour and standard deviation € 5 per hour, rounded to two decimal places. These tasks do not have a fixed profit component and are not subject to deadlines or penalties. Instead, fixed profits are chosen equal to € 200. These tasks do not have an hourly profit but are associated with a deadline (sampled uniformly within the time horizon of 5 days) and a penalty of € 100 for late or missed completion, reflecting their higher urgency and contractual obligations. Also, a probability of failure is assigned to each of these tasks to model the risk of unsuccessful completion due to unforeseen circumstances such as technical issues, client unavailability, or external disruptions. In the simulation, this probability is chosen equal to 5% for every task, which is a realistic choice because even though most tasks are expected to be completed successfully, there remains a non-negligible chance of failure, especially in dynamic and uncertain operational environments. The inclusion of this risk parameter allows the simulation to capture the impact of uncertainty on scheduling and routing decisions, as well as on the overall system performance.

Skill constraints are modeled by distinguishing between unexperienced and senior technicians through a single skill. Instances are generated in a way that most technicians are as-

Licensed under CC BY 4.0. DOI: 10.1109/TASE.2026.3730532

signed as senior, while only a small fraction are unexperienced. Conversely, the majority of tasks can be performed by any technician, but a minority specifically require a senior qualification. This asymmetric assignment of qualifications and requirements reflects real-world workforce structures, where most employees are broadly qualified, but some tasks demand higher expertise, introducing realistic assignment constraints into the problem.

Travel distances between all pairs of locations are computed using the Euclidean metric. To account for varying traffic conditions, travel speeds are independently sampled for each arc from a normal distribution with mean $0.5\,\mathrm{km/min}$ and standard deviation $0.15\,\mathrm{km/min}$. Similarly, fuel costs are independently assigned to each arc as random variables with a mean price of €0.10 per kilometer and a standard deviation of €0.60 per kilometer. This variability is intended to introduce heterogeneity in travel conditions across the network, so that higher fuel costs may reflect, for example, less efficient routes, and vice versa. This independent sampling allows us to introduce a broad variability in both travel times and costs, capturing the diversity of road types, traffic conditions, and operational expenses encountered in practice. While this approach provides a challenging benchmark, it is worth noting that, in real-world settings, travel speed and cost per kilometer are often correlated, with slower routes typically incur higher costs due to factors such as congestion or difficult terrain.

For reproducibility, the random seed for each generated instance is set as a deterministic function of the number of employees, number of tasks, and the instance index. This ensures that identical parameter choices always yield the same problem instance, supporting transparent benchmarking and comparison.

C. Discussion of the results

Figure 5 presents a comprehensive performance evaluation of Algorithm 1 (the proposed heuristic) compared to a baseline proximity-based approach, across a range of synthetic problem instances that vary in both the number of technicians $|\mathcal{D}|$ and tasks $|\mathcal{T}|$. The figure is structured to show three key metrics as functions of the problem size:

- Expected profit (in thousands of euros);
- Expected number of executed tasks;
- Expected number of free workdays (in total).

For each metric, results are reported for three workforce sizes (10, 50, and 100 technicians), and for increasing numbers of tasks (100, 250, 500). Each panel contrasts four scenarios:

- *Proximity* (offline): Tasks are assigned to the nearest technician, followed by standard routing and scheduling. This serves as a baseline and reflects the current practice at DEDEM S.p.A.
- *Algorithm 1* (offline): The proposed heuristic is run offline, initialized from the Proximity solution, to optimize assignment, routing, and scheduling.

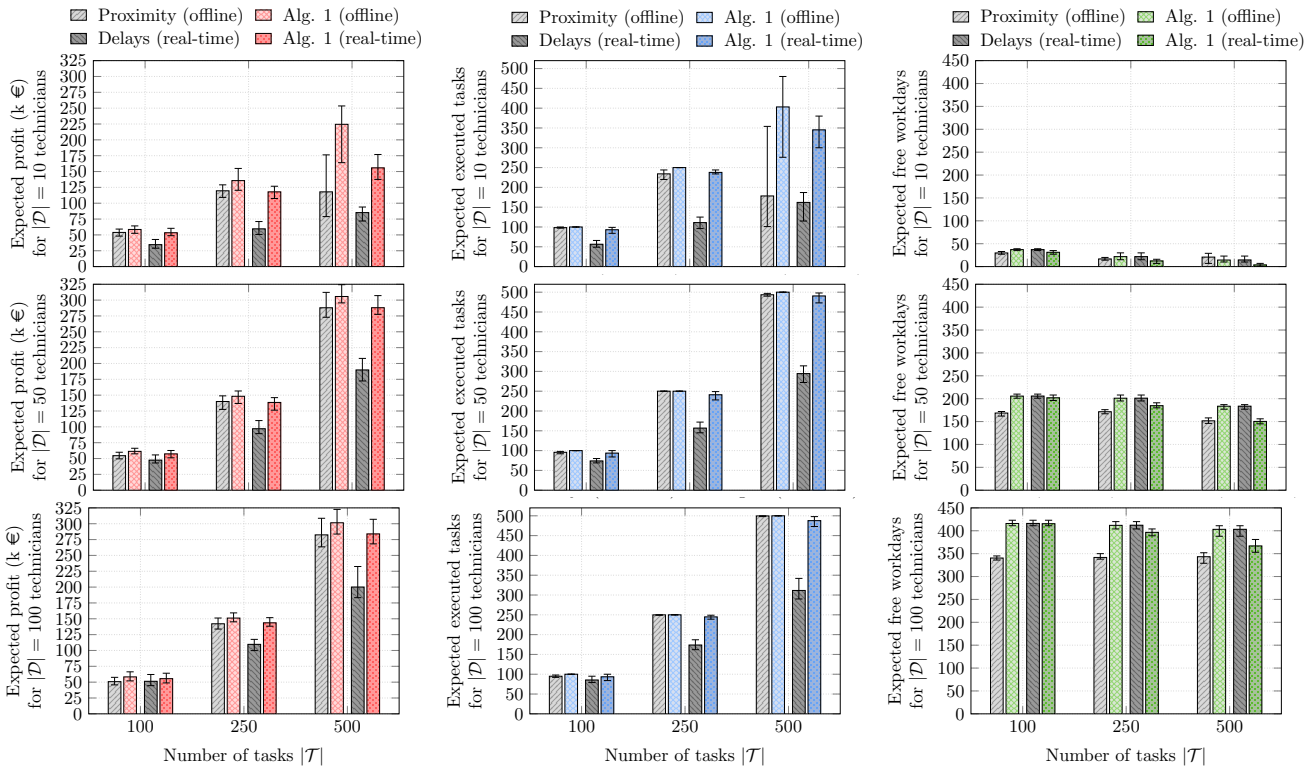


Fig. 5. Performance comparison of Algorithm 1 under offline and real-time conditions across synthetic problem instances with varying numbers of technicians and tasks. In the offline setting, "Proximity" denotes a baseline heuristic assigning tasks to the nearest technicians, serving as a warm start for Algorithm 1. In the real-time setting, "Delays" represents simulations incorporating stochastic task execution delays without any dynamic compensation logic, contrasted against Algorithm 1's real-time dynamic reassignment, routing, and scheduling capabilities. The figure illustrates how Algorithm 1 consistently improves expected profit, the number of expected executed tasks, and the number of free workdays in static/dynamic environments.

Licensed under CC BY 4.0. DOI: 10.1109/TASE.2026.3730532

- *Delays* (real-time): The offline solution provided by Algorithm 1 is simulated in real-time subjected to stochastic delays without any dynamic compensation.
- *Algorithm* 1 (real-time): The offline solution provided by Algorithm 1 is simulated in real-time subjected to stochastic delays with the dynamic compensation provided by the real-time execution of Algorithm 1 which dynamically reassign/reschedule tasks in response to delays while computing new optimized routes.

The results show that Algorithm 1 consistently improves the solution quality in both offline and real-time settings.

Expected Profit: Algorithm 1 consistently outperforms the baseline across all tested scenarios, demonstrating robust performance improvements with strong statistical significance.

In the offline scenario, the "Proximity" heuristic—which assigns tasks to the nearest technician and then computes routes and schedules—serves as a baseline reflecting current practice at DEDEM S.p.A. When Algorithm 1 is initialized from this solution, it achieves profit improvements ranging from a minimum of 3% to as much as 20%, with an average gain of approximately 10%. The performance gains exhibit a clear correlation with problem complexity and task/employees ratio. For smaller instances (e.g., $|\mathcal{D}| = 10$) and for low task/employees ratios (e.g., ≤ 5), the improvement averages 6.6%, while for larger instances (e.g., $|\mathcal{D}| = 100$) and especially for high ratios (e.g., ≥ 10), gains averages reach up to 13.7%. This scaling behavior indicates that Algorithm 1's optimization becomes increasingly valuable as the solution space grows exponentially.

In real-time scenarios, the "Delays" baseline—which does not adapt to stochastic disruptions—experiences significant profit losses. In contrast, Algorithm 1 dynamically reassigns and reschedules tasks in response to delays, recovering up to 40% of profit loss and increasing average net profit by up to 100%. The greatest improvements are observed in more complex problems with many tasks and few technicians. This underscores the value of real-time adaptability in operational environments subject to uncertainty.

Expected number of executed tasks: The number of completed tasks is a direct measure of operational efficiency. Algorithm 1 consistently increases task completion rates compared to the baseline, particularly when technician availability is tight. In large-scale scenarios (e.g., $|\mathcal{D}| = 100$, $|\mathcal{T}| = 500$), Algorithm 1 achieves completion rates above 90% when the workforce is sufficient, while the baseline's performance degrades more rapidly as the system becomes congested. Under stochastic delays, the baseline leaves a significant fraction of tasks unexecuted. Algorithm 1's real-time adjustments enable rapid reassignment, maintaining high completion rates even in the presence of disruptions.

Expected number of free workdays: This metric reflects workload balance and technician utilization. Algorithm 1 achieves a higher number of free workdays per technician compared to the baseline, indicating a more balanced distribution of workload and reduced risk of overcommitment. This is particularly important for workforce well-being and long-term sustainability.

D. Scalability and Robustness

The results show diminishing returns from simply increasing the workforce size: beyond a certain point, adding more technicians does not proportionally increase profit or task completion. Instead, the largest gains are realized through effective coordination and dynamic scheduling, as enabled by Algorithm 1. Algorithm 1 demonstrates strong scalability, maintaining robust performance as both the number of technicians and tasks increase. Its local, pairwise optimization structure allows it to efficiently handle large problem instances that are computationally intractable for exact methods. The ability to operate both offline (for initial planning) and in real time (for adaptive rescheduling) makes it highly suitable for real-world, dynamic environments such as DEDEM S.p.A.

VII. Conclusions

This paper presents a real-time heuristic algorithm designed to address a problem encompassing task assignment, route planning, and scheduling for a large workforce. The algorithm accounts for unique complexities arising from a real-world case study provided by DEDEM S.p.A., aimed at maximizing profit. It operates in real-time during tasks execution, automatically adjusting scheduling in response to new tasks, technician unavailability, or delays. Numerical simulations validate the effectiveness of the proposed approach, demonstrating improved expected profit compared to the company's current scheduling and highlighting its ability to mitigate potential profit loss.

Acknowledgments: We gratefully acknowledge and thank DEDEM s.p.a. for the data and support.

References

[1] E. Erkut and J. Zhang, "The maximum collection problem with time-dependent rewards," *Naval Research Logistics (NRL)*, vol. 43, no. 5, pp. 749–763, 1996.

[2] H. Murat Afsar and N. Labadie, "Team orienteering problem with decreasing profits," *Electronic Notes in Discrete Mathematics*, vol. 41, pp. 285–293, 2013.

[3] F. Y. Vincent, P. Jewpanya, S.-W. Lin, and A. P. Redi, "Team orienteering problem with time windows and time-dependent scores," *Computers & Industrial Engineering*, vol. 127, pp. 213–224, 2019.

[4] G. Peng, R. Dewil, C. Verbeeck, A. Gunawan, L. Xing, and P. Vansteenwegen, "Agile earth observation satellite scheduling: An orienteering problem with time-dependent profits and travel times," *Computers & Operations Research*, vol. 111, pp. 84–98, 2019.

[5] H. Kim, B.-I. Kim, and D.-j. Noh, "The multi-profit orienteering problem," *Computers & Industrial Engineering*, vol. 149, p. 106 808, 2020.

[6] H. Kim and B.-I. Kim, "Hybrid dynamic programming with bounding algorithm for the multi-profit orienteering problem," *European Journal of Operational Research*, vol. 303, no. 2, pp. 550–566, 2022.

[7] H. Tang, E. Miller-Hooks, and R. Tomastik, "Scheduling technicians for planned maintenance of geographically distributed equipment," *Transportation Research Part E: Logistics and Transportation Review*, vol. 43, no. 5, pp. 591–609, 2007.

[8] J. Pietz and J. O. Royset, "Generalized orienteering problem with resource dependent rewards," *Naval Research Logistics (NRL)*, vol. 60, no. 4, pp. 294–312, 2013.

[9] G. Erdoğan and G. Laporte, "The orienteering problem with variable profits," *Networks*, vol. 61, no. 2, pp. 104–116, 2013.

[10] A. Gunawan, K. M. Ng, G. Kendall, and J. Lai, "An iterated local search algorithm for the team orienteering problem with variable profits," *Engineering Optimization*, vol. 50, no. 7, pp. 1148–1163, 2018.

Licensed under CC BY 4.0. DOI: 10.1109/TASE.2026.3730532

[11] Q. Yu, K. Fang, N. Zhu, and S. Ma, "A matheuristic approach to the orienteering problem with service time dependent profits," *European Journal of Operational Research*, vol. 273, no. 2, pp. 488–503, 2019.

[12] M. Khodadadian, A. Divsalar, C. Verbeeck, A. Gunawan, and P. Vansteenwegen, "Time dependent orienteering problem with time windows and service time dependent profits," *Computers & Operations Research*, vol. 143, p. 105 794, 2022.

[13] E. Marzal and L. Sebastia, "Team orienteering problem with time windows and variable profit," in *2022 17th Conference on Computer Science and Intelligence Systems (FedCSIS)*, 2022, pp. 347–350.

[14] Q. Yu, Y. Adulyasak, L.-M. Rousseau, N. Zhu, and S. Ma, "Team orienteering with time-varying profit," *INFORMS Journal on Computing*, vol. 34, no. 1, pp. 262–280, 2022.

[15] J. Li, J. Zhu, G. Peng, J. Wang, L. Zhen, and E. Demeulemeester, "Branch-price-and-cut algorithms for the team orienteering problem with interval-varying profits," *European Journal of Operational Research*, vol. 319, no. 3, pp. 793–807, 2024.

[16] G. Laporte, "Fifty years of vehicle routing," *Transportation Science*, vol. 43, no. 4, pp. 408–416, 2009.

[17] K. Braekers, K. Ramaekers, and I. Van Nieuwenhuyse, "The vehicle routing problem: State of the art classification and review," *Computers & industrial engineering*, vol. 99, pp. 300–313, 2016.

[18] J.-F. Cordeau, G. Laporte, and A. Mercier, "A unified tabu search heuristic for vehicle routing problems with time windows," *Journal of the Operational research society*, vol. 52, no. 8, pp. 928–936, 2001.

[19] P. Toth and D. Vigo, *Vehicle routing: problems, methods, and applications*. SIAM, 2014.

[20] K. L. Hoffman, M. Padberg, G. Rinaldi, *et al.*, "Traveling salesman problem," *Encyclopedia of operations research and management science*, vol. 1, pp. 1573–1578, 2013.

[21] B. Eksioglu, A. V. Vural, and A. Reisman, "The vehicle routing problem: A taxonomic review," *Computers & Industrial Engineering*, vol. 57, no. 4, pp. 1472–1483, 2009.

[22] N. Christofides, "Worst-case analysis of a new heuristic for the travelling salesman problem.," DTIC Document, Tech. Rep., 1976.

[23] L. Few, "The shortest path and the shortest road through n points," *Mathematika*, vol. 2, no. 2, pp. 141–144, 1955.

[24] A. Camisa, A. Testa, and G. Notarstefano, "Multi-robot pickup and delivery via distributed resource allocation," *IEEE Transactions on Robotics*, vol. 39, no. 2, pp. 1106–1118, 2022.

[25] L. Abbatecola, M. P. Fanti, G. Pedroncelli, and W. Ukovich, "A distributed cluster-based approach for pick-up services," *IEEE Transactions on Automation Science and Engineering*, vol. 16, no. 2, pp. 960–971, 2018.

[26] S. Karaman and E. Frazzoli, "Vehicle routing problem with metric temporal logic specifications," in *2008 47th IEEE conference on decision and control*, 2008, pp. 3953–3958.

[27] C. I. Vasile and C. Belta, "An automata-theoretic approach to the vehicle routing problem.," in *Robotics: Science and Systems*, Berkeley, CA, 2014, pp. 1–9.

[28] D. Deplano, M. Franceschelli, S. Ware, S. Rong, and A. Giua, "A discrete event formulation for multi-robot collision avoidance on pre-planned trajectories," *IEEE Access*, vol. 8, pp. 92 637–92 646, 2020.

[29] M. Lippi and A. Marino, "A mixed-integer linear programming formulation for human multi-robot task allocation," in *2021 30th IEEE International Conference on Robot & Human Interactive Communication (RO-MAN)*, 2021, pp. 1017–1023.

[30] M. Lippi, J. Gallou, A. Gasparri, and A. Marino, "An optimal allocation and scheduling method in human-multi-robot precision agriculture settings," in *2023 31st Mediterranean Conference on Control and Automation (MED)*, 2023, pp. 541–546.

[31] A. Furchì, M. Lippi, R. F. Carpio, and A. Gasparri, "Route optimization in precision agriculture settings: A multi-steiner tsp formulation," *IEEE Transactions on Automation Science and Engineering*, vol. 20, no. 4, pp. 2551–2568, 2023.

[32] A. Furchì, M. Franceschelli, and A. Gasparri, "A milp scheduling problem for multi-robot logistics systems in a precision farming application and a polynomial time optimal algorithm," in *2024 32nd Mediterranean Conference on Control and Automation (MED)*, 2024, pp. 664–669.

[33] M. Cavecchia, T. A. de Queiroz, R. Lancellotti, G. Zucchi, M. Iori, *et al.*, "A real-world multi-depot, multi-period, and multi-trip vehicle routing problem with time windows," in *Proceedings of the 14th International Conference on Operations Research and Enterprise Systems (ICORES 2025)*, vol. 1, 2025, pp. 112–122.

[34] J. Li, Y. Li, and P. M. Pardalos, "Multi-depot vehicle routing problem with time windows under shared depot resources," *Journal of combinatorial optimization*, vol. 31, pp. 515–532, 2016.

[35] S. Chopra and M. Egerstedt, "Heterogeneous multi-robot routing," in *IEEE American Control Conference*, 2014, pp. 5390–5395.

[36] M. Franceschelli, D. Rosa, C. Seatzu, and F. Bullo, "Gossip algorithms for heterogeneous multi-vehicle routing problems," *Nonlinear Analysis: Hybrid Systems*, vol. 10, pp. 156–174, 2013.

[37] N. Bastianello, D. Deplano, M. Franceschelli, and K. H. Johansson, "Robust online learning over networks," *IEEE Transactions on Automatic Control*, vol. 70, no. 2, pp. 933–946, 2025.

[38] D. Deplano, N. Bastianello, M. Franceschelli, and K. H. Johansson, "A unified approach to solve the dynamic consensus on the average, maximum, and median values with linear convergence," 2023, pp. 6442–6448.

[39] D. Deplano, M. Franceschelli, and A. Giua, "Dynamic max-consensus with local self-tuning," 13, vol. 55, 2022, pp. 127–132.

[40] D. Deplano, M. Franceschelli, and A. Giua, "Lyapunov-free analysis for consensus of nonlinear discrete- time multi-agent systems," vol. 2018-December, 2018, pp. 2525–2530.

[41] M. Franceschelli, A. Giua, and C. Seatzu, "A gossip-based algorithm for discrete consensus over heterogeneous networks," *IEEE Transactions on Automatic Control*, vol. 55, no. 5, pp. 1244–1249, 2010.

[42] E. Gravelle and S. Martínez, "An anytime distributed load-balancing algorithm satisfying capacity and quantization constraints," *IEEE Transactions on Control of Network Systems*, vol. 4, no. 2, pp. 279–287, 2015.

[43] M. Franceschelli, A. Giua, and C. Seatzu, "Fast discrete consensus based on gossip for makespan minimization in networked systems," *Automatica*, vol. 56, pp. 60–69, 2015.

[44] A. Miele, M. Lippi, and A. Gasparri, "A distributed framework for integrated task allocation and safe coordination in networked multi-robot systems," *IEEE Transactions on Automation Science and Engineering*, 2025.

[45] J. C. S. Brandão and A. Mercer, "The multi-trip vehicle routing problem," *Journal of the Operational research society*, vol. 49, no. 8, pp. 799–805, 1998.

[46] D. Cattaruzza, N. Absi, and D. Feillet, "The multi-trip vehicle routing problem with time windows and release dates," *Transportation Science*, vol. 50, no. 2, pp. 676–693, 2016.

[47] T. Calamoneri, F. Corò, and S. Mancini, "Management of a post-disaster emergency scenario through unmanned aerial vehicles: Multi-depot multi-trip vehicle routing with total completion time minimization," *Expert Systems with Applications*, vol. 251, p. 123 766, 2024.

[48] D. Deplano, C. Seatzu, and M. Franceschelli, "A distributed online heuristic for a large-scale workforce task assignment and multi-vehicle routing problem," in *2024 IEEE 20th International Conference on Automation Science and Engineering (CASE)*, 2024, pp. 3174–3180.

[49] F. Stavropoulou, P. P. Repoussis, and C. D. Tarantilis, "The vehicle routing problem with profits and consistency constraints," *European Journal of Operational Research*, vol. 274, no. 1, pp. 340–356, 2019.

[50] T. Adamo, M. Gendreau, G. Ghiani, and E. Guerriero, "A review of recent advances in time-dependent vehicle routing," *European Journal of Operational Research*, vol. 319, no. 1, pp. 1–15, 2024.

[51] C. Archetti, L. Coelho, M. Speranza, and P. Vansteenwegen, "Beyond fifty years of vehicle routing: Insights into the history and the future," *European Journal of Operational Research*, 2025.

[52] C. F. Dursunoglu, O. Arslan, S. M. Demir, B. Y. Kara, and G. Laporte, "A unifying framework for selective routing problems," *European Journal of Operational Research*, vol. 320, no. 1, pp. 1–19, 2025.

[53] I.-M. Chao, B. L. Golden, and E. A. Wasil, "The team orienteering problem," *European journal of operational research*, vol. 88, no. 3, pp. 464–474, 1996.

[54] K. D. Mukhina, A. A. Visheratin, and D. Nasonov, "Orienteering problem with functional profits for multi-source dynamic path construction," *PloS one*, vol. 14, no. 4, e0213777, 2019.

[55] H. Jandaghi, A. Divsalar, and S. Emami, "The categorized orienteering problem with count-dependent profits," *Applied Soft Computing*, vol. 113, p. 107 962, 2021.

[56] A. Santini and C. Archetti, "The hazardous orienteering problem," *Networks*, vol. 81, no. 2, pp. 235–252, 2023.

[57] T. Öncan, I. K. Altınel, and G. Laporte, "A comparative analysis of several asymmetric traveling salesman problem formulations," *Computers & Operations Research*, vol. 36, no. 3, pp. 637–654, 2009.

[58] T. Bektaş, L. Gouveia, and D. Santos, "Compact formulations for multi-depot routing problems: Theoretical and computational comparisons," *Computers & Operations Research*, vol. 124, p. 105 084, 2020.

Licensed under CC BY 4.0. DOI: 10.1109/TASE.2026.3730532

[59] L. Gouveia and J. M. Pires, "The asymmetric travelling salesman problem: On generalizations of disaggregated miller–tucker–zemlin constraints," *Discrete Applied Mathematics*, vol. 112, no. 1-3, pp. 129–145, 2001.
[60] G. Dantzig, R. Fulkerson, and S. Johnson, "Solution of a large-scale traveling-salesman problem," *Journal of the operations research society of America*, vol. 2, no. 4, pp. 393–410, 1954.
[61] D. P. Dobkin, S. J. Friedman, and K. J. Supowit, "Delaunay graphs are almost as good as complete graphs," *Discrete & Computational Geometry*, vol. 5, pp. 399–407, 1990.
[62] J.-D. Boissonnat, O. Devillers, and S. Hornus, "Incremental construction of the delaunay triangulation and the delaunay graph in medium dimension," in *Proceedings of the twenty-fifth annual symposium on Computational geometry*, 2009, pp. 208–216.
[63] J. Demšar, "Statistical comparisons of classifiers over multiple data sets," *Journal of Machine learning research*, vol. 7, no. Jan, pp. 1–30, 2006.



Diego Deplano (S'17–M'21) is Assistant Professor (RTD-A) at the Department of Electrical and Electronic Engineering, University of Cagliari, Italy. He received the B.S., M.S., and PhD degrees in Electronic Engineering "cum laude" from the University of Cagliari, Italy, respectively in '15, '17, and '21. He was awarded the best PhD Thesis defended in the area of Systems and Control Engineering at an Italian University by SIDRA. He spent visiting periods at the Nanyang Technological University (NTU), Singapore, at the Centre National de la Recherche Scientifique (CNRS), Grenoble, France, and at the University of Toronto (UofT), Toronto, Canada. He has lectured seminars at KTH in Stockholm, Sweden, and at Peking University in Beijing, China. He is a member of the Conference Editorial Board (CEB) for the IEEE Control Systems Society (CSS) and serves as Associate Editor for the IEEE Conference on Decision and Control (CDC) and the IEEE American Control Conference (ACC) since 2025. He also serves as an Associate Editor for the IEEE Conference on Automation Science and Engineering (CASE) since 2024. His research interests include open multi-agent systems, distributed optimization and learning, dynamic consensus problems, monotone dynamical systems, contraction theory.



Carla Seatzu (M'05–SM'16) is Full Professor of Automatic Control at the University of Cagliari, Cagliari, Italy. Her research interests are: discrete event systems, Petri nets, hybrid systems, networked control systems, manufacturing and transportation systems. She is the author/co-author of 300+ publications, including 100+ articles in international journals and one textbook. She is the editor of two international books. In Scopus she has 6230 citations and an h-index equal to 41. Prof. Seatzu is Department Editor of Discrete Event Dynamic Systems; she is Senior Editor of the IEEE Transactions on Automation Science and Engineering and of the International Journal of Robotics and Research. She has been Senior Editor of the IEEE Control Systems Letters and Associate Editor of many international journals including the IEEE Transactions on Automatic Control. She is General co-Chair of the 11th Int. Conference on Control, Decision and Information Technologies; was Program Chair of the 20th IEEE International Conference on Automation and Science Engineering, and of the 23rd IEEE International Conference on Emerging Technologies and Factory Automation in 2018; Workshop Chair of the 55th IEEE Conference on Decision and Control in 2016, and General Co-Chair of the 18th IEEE International Conference on Emerging Technologies and Factory Automation in 2013. Since 2020 she is Chair of the IFAC Technical Committee TC1.3 on Discrete Event and Hybrid Systems. She has been a Visiting Professor in universities in Spain, (Zaragoza), USA (Atlanta), Mexico (Guadalajara), and China (Xi'an, Hangzhou).



Mauro Franceschelli (M'11–SM'22) is Associate Professor at the Department of Electrical and Electronic Engineering, University of Cagliari, Italy. He received the Laurea degree in Electronic Engineering "cum laude" in 2007 and the PhD degree in 2011 from the University of Cagliari. He spent visiting periods at the Georgia Institute of Technology (GaTech), and the University of California at Santa Barbara (UCSB), USA. In 2013 he received a fellowship from the National Natural Science Foundation of China (NSFC), at Xidian University, Xi'an, China. In 2015 he was awarded a position of Assistant Professor (RTD-A) funded by the Italian Ministry of Education, University and Research (MIUR) under the 2014 call "Scientific Independence of Young Researchers" (SIR). He is currently an Associate Editor for the IEEE Transactions on Automatic Control (TAC) and IFAC Nonlinear Analysis: Hybrid Systems (NAHS). He is a member of the Conference Editorial Board (CEB) for the IEEE Control Systems Society (CSS) since 2019. He serves as Associate Editor for the IEEE Conference on Automation Science and Engineering (CASE) since 2015. He was an Associate Editor for the IEEE Transactions on Automation Science and Engineering (T-ASE) 2021-2023 and Guest Associate Editor for the IEEE Open Journal of Control Systems (OJ-CSYS) in 2024. He is IEEE Senior Member and Officer (vice-Chair) for the IEEE Italy Section Chapter of the IEEE Control Systems Society since 2025. His research interests include consensus problems, open multi-agent systems, multi-robot systems, networks of energy systems, resilient networks, distributed optimization and learning.

Licensed under CC BY 4.0. DOI: 10.1109/TASE.2026.3730532