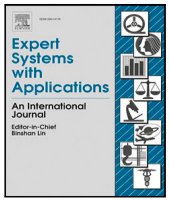
Expert Systems With Applications 251 (2024) 123964

Contents lists available at ScienceDirect

Expert Systems With Applications

journal homepage: www.elsevier.com/locate/eswa



Dynamic, fair, and efficient routing for cooperative autonomous vehicle fleets

Aitor López Sánchez [a,b], Marin Lujak [a,*], Frédéric Semet [b], Holger Billhardt [a]

[a] CETINIA, University Rey Juan Carlos, Calle Tulipán s/n, 28933 Mostoles (Madrid), Spain
[b] Univ. Lille, CNRS, Inria, Centrale Lille, UMR 9189 CRIStAL, F-59000 Lille, France

ARTICLE INFO

Keywords:
Multi-agent coordination
Distributed optimization
Column generation
VRP with time windows
Fairness
Egalitarian welfare
Cooperatives

ABSTRACT

This paper addresses challenges in agricultural cooperative autonomous fleet routing through the proposition, modeling, and resolution of the Dynamic Vehicle Routing Problem with Fair Profits and Time Windows (DVRP-FPTW). The aim is to dynamically optimize routes for a vehicle fleet serving tasks within assigned time windows, emphasizing fair and efficient solutions. Our DVRP-FPTW accommodates unforeseen events like task modifications or vehicle breakdowns, ensuring adherence to task demand, vehicle capacities, and autonomies. The proposed model incorporates mandatory and optional tasks, including optional ones in operational vehicle routes if not compromising the vehicles' profits. Including asynchronous and distributed column generation heuristics, the proposed Multi-Agent-based architecture DIMASA for the DVRP-FPTW dynamically adapts to unforeseen events. Systematic Egalitarian social welfare optimization is used to iteratively maximize the profit of the least profitable vehicle, prioritizing fairness across the fleet in light of unforeseen events. This improves upon existing dynamic and multi-period VRP models that rely on prior knowledge of demand changes. Our approach allows vehicle agents to maintain privacy while sharing minimal local data with a fleet coordinator agent. We propose publicly available benchmark instances for both static and dynamic VRP-FPTW. Simulation results demonstrate the effectiveness of our DVRP-FPTW model and our multi-agent system solution approach in coordinating large, dynamically evolving cooperative autonomous fleets fairly and efficiently in close to real-time.

1. Introduction

In agriculture cooperatives, where individual rationality meets mutual competition among farmers managing fragmented and dispersed lands with precision farming, an ever-growing need arises for fair and efficient coordination of shared agriculture vehicles. This demand becomes pronounced as farmers aim to enhance their competitiveness across the industry and is particularly evident in sharing expensive autonomous tractor and agricultural robot (hereafter referred to as vehicle) fleets.

Agricultural cooperatives, on the other hand, as member-owned and controlled organizations, pursue common goals for both individual and mutual economic or social benefits. These organizations operate on principles that promote democratic control, active member participation, and equitable distribution of benefits for all members and, thus, inherently balance collective efforts and individual profit motives. In agricultural cooperatives, while the primary goal is collaborative work, an inherent tension exists between the collective efforts of the group and each farmer's pursuit of maximizing individual profits within the cooperative's regulations and value system. This inherent discord is addressed through the foundational values statement, which encapsulates the cooperative's core principles and code of ethics. Given the natural conflict between collective objectives and individual profit motives, farmers, driven by individual rationality, choose to cooperate within the cooperative framework when the benefits outweigh those of competition. Furthermore, they are inclined to retain autonomy and control of their operational and financial decisions rather than handing over such authority to external entities.

In this context, we study a variant of the capacitated Vehicle Routing Problem with Profits (VRPP) (Aksen & Aras, 2006; Archetti et al., 2014), focusing on the deployment of autonomous vehicle fleets for the efficient execution of geographically dispersed farming operations or tasks. We explore both heterogeneous vehicle fleets where each

* Corresponding author.
E-mail addresses: aitor.lopez@urjc.es (A.L. Sánchez), marin.lujak@urjc.es (M. Lujak), frederic.semet@centralelille.fr (F. Semet), holger.billhardt@urjc.es (H. Billhardt).

https://doi.org/10.1016/j.eswa.2024.123964
Received 2 January 2024; Received in revised form 7 March 2024; Accepted 8 April 2024
Available online 12 April 2024

autonomous vehicle is compatible with specific tasks and homogeneous advanced agricultural machinery, including multi-functional autonomous tractors (agrirobots) capable of handling a variety of tasks. The tasks in this context encompass agricultural activities like plowing, harvesting, pesticide, and herbicide spraying, among others, with each task corresponding to a 'customer' in traditional VRPP scenarios. Notably, each task is linked to revenue collected by the vehicle responsible for its execution. Vehicle profit is the difference between the revenue earned from the performed tasks and travel costs. The pivotal question we address in this paper is how to dynamically optimize a cooperative fleet's profits. The challenge lies in balancing efficiency and fairness, considering both the cooperative's commitment to equitable distribution and operational efficiency and the occurrence of unexpected events during fleet operation. Such events (encompassing, e.g., task alterations, vehicle additions, breakdowns, or in-operation repairs) may necessitate prompt and flexible route adjustments.

Therefore, building upon López-Sánchez et al. (2023a), the study at hand proposes, models, and solves the Dynamic Vehicle Routing Problem with Fair Profits and Time Windows (DVRP-FPTW) in this complex context, addressing the challenge of dynamically optimizing fleet profits while accommodating unforeseen events. It enforces vehicle capacity and autonomy constraints, preventing any vehicle from exceeding the capacity and maximum allowed distance between two depot visits, while also meeting task demands within specified time windows.

The proposed DVRP-FPTW model incorporates mandatory and optional tasks. Mandatory tasks are those which, before a disruption event, were assigned to vehicles that remain operational after the event, while optional tasks are those left unassigned following the event. Generally speaking, models in the literature do not oblige to visit each customer, e.g., (Archetti et al., 2014). As Gendreau et al. (1997), our proposed model ensures the serving of all mandatory customers (farming operations or tasks). It incorporates optional tasks into operational vehicle routes only when their inclusion does not negatively impact the profit of the vehicles.

The DVRP-FPTW is a computationally complex and intrinsically distributed problem; thus, we decompose it and apply a column generation method that initiates with a small set of variables corresponding to an initial set of routes. It iteratively generates additional variables (columns) when necessary to enhance the current solution. This approach is seamlessly integrated into the Distributed Multi-Agent System (MAS) Architecture proposed in this paper, which we name DIMASA, comprising a fleet coordinator and vehicle agents, to produce a scalable and computationally efficient solution suitable for real-world application. The DIMASA architecture enables flexible and distributed decision-making in an intrinsically decentralized cooperative fleet environment. Each vehicle finds its best routes autonomously adapting to dynamically occurring local events in real time. This adaptation is driven by the vehicle's local information and the information received from the coordinator. Among these routes, the coordinator proposes the best one for the whole fleet. In fact, the proposed DIMASA architecture fosters cooperation among different vehicle agents as they collectively work towards a shared objective. While all vehicles contribute to the common goal of completing tasks, they may obtain varying profits depending on their individual routes.

To ensure both fairness and efficiency in dynamic environments, we propose a systematic egalitarian social welfare optimization approach that aims to iteratively maximize the profit of vehicles' routes in a non-decreasing order of vehicle profitability across the fleet. In addition, the column generation method maintains the solution quality and guarantees finding an optimal solution if we integrate it into a Branch and Price method. The proposed DIMASA architecture allows for a distributed, parallel, and asynchronous computation, particularly useful for routing large-scale real-world cooperative vehicle fleets in close-to real-time, and avoids the need to share private vehicle data. Leveraging distributed computation and real-time information exchange among vehicle agents, the DIMASA architecture mitigates single points of failure, enhancing fleet robustness against individual vehicle breakdowns and contingencies.

The paper is structured as follows. In Section 2, we give an overview of the state of the art. Section 3 presents the DVRP-FPTW problem formulation where we decompose the problem using a column generation (also called pricing) approach to incorporate it into the proposed DIMASA architecture. This new architecture together with our proposed distributed and scalable solution approach for the DVRP-FPTW problem is presented in Section 4. In Section 5, we give a use-case example and assess the performance of our approach on a set of publicly available benchmark instances that we propose in this paper for the static and dynamic versions of the VRP-FPTW. Here, we present the setup and results of simulation experiments with related discussion. We end the paper in Section 6 with conclusions and future work.

2. Related work

In Lujak et al. (2021), we introduced the Agriculture Fleet Vehicle Routing Problem (AF-VRP) that, to the best of our knowledge, differs from any previously studied versions of the Vehicle Routing Problem. Uniquely characterized by its intrinsic dynamism and decentralization, and with a specific focus on agriculture cooperatives, our approach necessitates the consideration of fundamental concepts such as efficiency, fairness and equity.

2.1. Fair vehicle routing problem

In the domain of collaborative vehicle routing problems, the sole pursuit of maximum overall benefit for the fleet is complemented by considerations for service quality, equity, and fairness, e.g., (Gansterer & Hartl, 2018, 2020; Guajardo & Rönnqvist, 2016).

Soriano et al. (2023) investigate how to solve the Multi-Depot Vehicle Routing Problem (MDVRP) with added profit fairness constraints. They consider each depot as an independent partner with its fleet and propose a bi-objective optimization problem that minimizes the total cost and maximizes the fairness of profit distribution. A multi-objective optimization approach focusing on max–min fairness was proposed by Liu and Papageorgiou (2018), aiming to maximize the level of satisfaction for the least satisfied owner. Furthermore, Lee and Ahn (2017) introduce a novel vehicle routing problem with vector profits, where each customer's revenue is linked to multiple stakeholders, aiming at maximizing the total profit while prioritizing the least satisfied stakeholder. The authors adopted a column-generation approach to solve the problem.

Fairness between customers is another important consideration in addition to the fairness between vehicles. Stavropoulou et al. (2019) study a Vehicle Routing Problem (VRP) with profits and consistency constraints considering a mixed set of mandatory and optional customers to visit. They determine profitable vehicle routes that maximize the net profit while satisfying vehicle capacity, route duration, and consistency constraints. Mancini et al. (2021) and Rodríguez-Martín et al. (2019) considered the distribution of customers' requirements over multiple days with added driver consistency constraints where customers must be served by the same vehicle over time.

2.2. Dynamic and robust vehicle routing problem

The Dynamic Vehicle Routing Problem (DVRP) is a widely studied problem which adapts to demand and traffic conditions that vary over time (Pillac et al., 2013; Psaraftis et al., 2016). The solution methods to DVRP can be classified into heuristics, exact methods, and hybrid methods. Most exact methods are based on Mixed Integer Linear Programming (MILP) models, which find an optimal solution for the current state. However, they may not remain optimal or even feasible

as conditions evolve. Therefore, exact methods generally solve small instances or are used in conjunction with heuristics to obtain good-quality solutions for more complex problems (Fonseca-Galindo et al., 2022; Li et al., 2009; Monroy-Licht et al., 2017; Steever et al., 2019; Ulmer et al., 2019, 2017; Ulmer & Streng, 2019).

Robust Vehicle Routing Problems address uncertainties in customer demands, travel times, and service times (Bertsimas et al., 2011). Agra et al. (2013) apply mathematical programming to ensure route feasibility within time windows across all travel times. Duan et al. (2021) employ robust multi-objective particle swarm optimization to handle disturbances. The periodic VRP with service time uncertainty is addressed using a Variable Neighbourhood Search algorithm based on the worst case (Qi et al., 2018). Guo et al. (2016) propose a two-phase method including optimal robust routes for all customers in the first phase and a multi-objective particle swarm optimization for dynamically appearing customers in the second phase.

The previous approaches do not account for the possibility of vehicle breakdowns, which, in real-world scenarios, can significantly disrupt the routing plans and result in delays, missed appointments, and additional costs. Seyyedhasani and Dvorak (2018) propose a re-optimization algorithm for the DVRP to address unexpected changes in field conditions or machinery management, transitioning from a single-depot to a multi-depot VRP to accommodate in-progress vehicle locations and capabilities. More specifically, they change the initial VRP with one depot to a Multi-Depot VRP where each new depot corresponds to the current position of the vehicles when something goes wrong. A similar idea is used in this article, but with a distributed approach where the coordinator does not know the exact location of vehicles. This information is handled only by the vehicles themselves, and they calculate their routes based on their current characteristics.

2.3. Multiagent approaches to vehicle routing problems

Distributed solution approaches to the DVRP generally utilize multi-agent systems (MAS) where autonomous agents interact to achieve a common goal (Alipour et al., 2022; Fonseca-Galindo et al., 2022). These systems enable decentralized dynamic vehicle routing, where autonomous agents coordinate decisions in response to the evolving conditions of the problem. While MAS approaches to the DVRP can manage large instances and dynamic scenarios, solution quality is only sometimes guaranteed (Alipour et al., 2022; Bono et al., 2020; Fonseca-Galindo et al., 2022).

Barbucha (2016, 2019) proposes a multi-agent approach to the DVRP where agents address continuous customer requests by reallocating vehicles using a Variable Neighborhood Search method. Lujak et al. (2020) confront dynamic task allocation in large, open, and collaborative vehicle fleets, using MAS to assign agent-represented vehicles to dynamically appearing tasks. In addition to these methods, auction-based methods address larger-scale problems (Los, Schulte, Gansterer et al., 2022; Los, Schulte, Spaan, & Negenborn, 2022). For example, customers' demands can be bundled and auctioned off to carriers who bid for the right to serve them. While these approaches do not provide quality of solution guarantees, they are known for their efficient resource allocation in large-scale scenarios within a limited computation time.

Contribution of the paper. Led by the open challenges identified in agriculture fleet vehicle routing presented by Lujak et al. (2021), in this paper, we propose the dynamic VRP-FPTW model, which extends the static model of López-Sánchez et al. (2023a) by incorporating mechanisms to adjust predetermined routes in response to unforeseen events. The proposed problem and related distributed multi-agent-based solution approach are a continuation of our previous works (López-Sánchez et al., 2023a, 2022, 2023b), where we proposed deterministic offline models for balancing fairness and efficiency of vehicle routes for the (static) multiple Travelling Salesman Problem and the static VRP-FPTW.

3. Dynamic VRP-FPTW

In this Section, we first provide a short description of the static VRP-FPTW, followed by a description of its dynamic counterpart. The objective is to develop a decomposed approach for the Dynamic VRP-FPTW that can be integrated into the DIMASA architecture composed of a fleet coordinator and vehicle agents, which we propose in this paper.

3.1. Static VRP-FPTW

Consider a complete arc-weighted digraph $G = (V, A)$, where $V = \{0, \dots, |V| - 1\}$ is the vertex set of size $|V|$, and A is the set of arcs $(i, j) \in V \times V$ with $i \neq j$. Vertex $0 \in V$ is the depot. The task (farming operation) vertices, included in set $N = \{1, \dots, |V|-1\}$, are the vertices to be visited (served). The arcs correspond to the shortest paths between any two distinct vertices i and j in a transportation network embodied by graph G. Let d_{ij} denote the distance of arc $(i, j) \in A$.

Each task $i \in N$ has an associated demand q_i to be satisfied and a nonnegative revenue r_i. We set the revenue of the depot to zero, $r_0 = 0$. Tasks are to be served within a specific time window $[l_i, u_i]$, where l_i and u_i denote the earliest and latest allowable visit times, respectively. A fleet K consisting of $|K|$ potentially heterogeneous vehicles is initially stationed at the depot vertex 0 whose time window $[l_0, u_0]$ determines the start and end time of the routes, respectively. Each vehicle $k \in K$ has distinct characteristics: travel speed sp_k, autonomy (maximum travel distance between two depot visits) D_k, carrying capacity Q_k, and travel cost per unit of distance traveled o_k. Time t_{ijk} includes traveling time (d_{ij}/sp_k) from vertex $i \in V$ to vertex $j \in V$ and service time $\hat{t}_{jk}$ for accomplishing task j by vehicle $k \in K$. Therefore, $t_{ijk} = d_{ij}/sp_k + \hat{t}_{jk}$. The service time for the depot is $\hat{t}_{0k} = 0$.

We represent vehicle-task compatibility through the travel cost value. The travel cost c_{ijk} of vehicle $k \in K$ compatible with task $j \in N$ is $c_{ijk} = d_{ij} \cdot o_k$, while this cost is set to a very large value M for each vehicle k that is not compatible with task $j \in N$ for all vertex $i \in V$.

The objective of the (static) VRP-FPTW, as defined by López-Sánchez et al. (2023a), is to find a set of routes, one for each vehicle, that maximize the profit of the worst-off vehicle. This profit is calculated as the difference between the cumulative revenue from the served tasks and the total cost of the route. The fleet's mission is to serve each task within their specified time window exactly once by exactly one vehicle while fully meeting their demand. The fleet must start from and return to the depot after completing the service. The constraints include respecting the autonomy and carrying capacity of the vehicles. Specifically, the total travel time and resource demands of the tasks to be served, associated with the length of the route and the load carried, must not exceed the autonomy and load capacity of any vehicle.

For the self-completeness of this work, in the following, we give the Static Restricted Master Problem (SRMP) of the Static VRP-FPTW formulation (11)-(15) in López-Sánchez et al. (2023a) whose aim is to determine the values of binary decision variables λ_k^p equal to 1 if route $p \in \Omega_k$ is assigned to vehicle k, where Ω_k is the set of feasible routes of vehicle k.

$$\max \quad y \tag{1}$$

$$s.t. \quad \sum_{k \in K} \sum_{p \in \Omega_k} a_{jk}^p \lambda_k^p = 1, \forall j \in N, \tag{2}$$

$$\sum_{p \in \Omega_k} w_k^p \lambda_k^p \geq y, \qquad \forall k \in K, \tag{3}$$

$$\sum_{p \in \Omega_k} \lambda_k^p = 1, \qquad \forall k \in K, \tag{4}$$

$$y \in \mathbb{R}, \lambda_k^p \in \{0, 1\}, \quad \forall k \in K, \forall p \in \Omega_k. \tag{5}$$

In this model, parameter $a_{jk}^p = 1$ if the task $j \in N$ is served on route p by vehicle k, and 0 otherwise, while parameter w_k^p is the profit obtained on route p by vehicle k. Constraints (2) are the one-on-one vehicle-task assignment constraints. Fairness constraints (3) fix the minimum profit

for each vehicle to y. Vehicle constraints (4) guarantee the use of all vehicles, while constraints (5) define variable domains. The objective function (1) maximizes profit y of the worst-off vehicle. More details can be found in López-Sánchez et al. (2023a).

The static VRP-FPTW model is designed to accommodate both heterogeneous and homogeneous vehicle fleets. It imposes no restrictions on the structure of the transportation graph, nor on the number of vehicles or tasks. The model provides a feasible solution under the assumption that there is a sufficient number of vehicles to serve all tasks. Thus, there may be instances where no feasible solution exists due to too restrictive constraints. To address this, we recommend the implementation of data pre-processing techniques that are essential for identifying such scenarios where the constraints of vehicle capacity, task demand, and time windows may lead to infeasibility. The clearly infeasible cases that we can detect in preprocessing before the first phase include instances where the total vehicle fleet capacity is insufficient for the overall task demand. Additionally, there are cases where, for example, the maximum autonomy among the vehicles is not sufficient for reaching the farthest task and returning to the depot, or situations where the width of time windows and their sequencing with respect to vehicle speeds are insufficient to meet the required time constraints.

3.2. Dynamic VRP-FPTW formulation

The Dynamic VRP-FPTW (DVRP-FPTW) addresses the limitations of static routing in environments where conditions can change unpredictably. We define a disruptive event e as an unforeseen occurrence or change that impacts the routing solution established beforehand. The disruptions include fluctuations in task demand, unforeseen vehicle breakdowns and unexpected delays, rendering visits to tasks within their designated time windows infeasible as well as new vehicle additions to the fleet, among others. Let $z^e \in [l_0, u_0]$ be the time of such an event.

We assume that the time is synchronized throughout the vehicle fleet. Let K' represent the set of post-disruption operational vehicles, including any possible new vehicle(s) and excluding the ones that are not operational. We define mutually exclusive subsets of tasks N at time z^e: N_1 (already served tasks), N_2 (set of the current tasks of the operational vehicles at the time instant of disruption), N_3 (mandatory tasks, i.e., the tasks to be served and previously assigned to set K' of operational vehicles after disruption), and N_4 (optional tasks, i.e., both new tasks and the tasks that had been assigned to the vehicles that were impacted by a disruptive event and have not been operational afterward). Considering set N_2, current task $s_k \in N_2$ refers to the task that vehicle k is performing at the time instant of disruption z^e or the next task if the vehicle is already en route. Let us define z^e_k as the instant of time, immediately after time z^e, when vehicle k completes task s_k. Thus, vehicles can consistently complete their ongoing tasks or the ones to which they are en route before any potential change of their route occurs. This approach allows us to make dynamic changes based on real-time events avoiding too frequent route fluctuations.

Encompassing these changes, the revised routing problem is modeled on a new complete arc-weighted digraph (V', A') where $V' = \{0\} \cup N_2 \cup N_3 \cup N_4$ and A' is the set of arcs $(i, j) \in V' \times V'$. The notation is summarized in Table 1.

3.2.1. Centralized formulation for the DVRP-FPTW

The DVRP-FPTW is formulated as a mixed-integer linear program. The model includes binary variables x_{ijk} indicating whether the route of vehicle k passes through arc $(i, j) \in A'$, continuous variables v_i representing the task completion time of a vehicle at vertex $i \in V'$ and a continuous variable y associated with the profit of the worst-off vehicle's route. In the following, we propose the (centralized) formulation of the DVRP-FPTW.

$$\max \quad y \tag{6}$$

$$s.t. \quad \sum_{k\in K'} \sum_{i\in V'} x_{ijk} = 1 \qquad \forall j \in N_3 \tag{7}$$

$$\sum_{k\in K'} \sum_{i\in V'} x_{ijk} \le 1 \qquad \forall j \in N_4 \tag{8}$$

$$\sum_{i\in V'} x_{ijk} - \sum_{h\in V'} x_{jhk} = 0 \qquad \forall j \in (N_3 \cup N_4), \forall k \in K' \tag{9}$$

$$\sum_{j\in V'} x_{sjk} = 1 \qquad \forall k \in K', s = s_k \tag{10}$$

$$\sum_{j\in V'} x_{j0k} = 1 \qquad \forall k \in K' \tag{11}$$

$$\sum_{i\in V'} \sum_{j\in V} d_{ij} x_{ijk} \le D'_k \qquad \forall k \in K' \tag{12}$$

$$\sum_{i\in V'} \sum_{j\in V'} q_j x_{ijk} \le Q'_k \qquad \forall k \in K' \tag{13}$$

$$v_i + t_{ijk} \le v_j + M(1 - x_{ijk}) \qquad \forall (i, j) \in A', j \ne 0, \forall k \in K' \tag{14}$$

$$v_{s_k} = z^e_k \qquad \forall k \in K' \tag{15}$$

$$l_i \le v_i \le u_i \qquad \forall i \in V' \tag{16}$$

$$y \le y_k + \sum_{i\in V'} \sum_{j\in V'} (r_i - c_{ijk}) x_{ijk} \quad \forall k \in K' \tag{17}$$

$$y \in \mathbb{R}, x_{ijk} \in \{0, 1\} \qquad \forall (i, j) \in A', \forall k \in K'. \tag{18}$$

The proposed DVRP-FPTW model (6)–(18) aims to incorporate set N_4 of optional tasks into the routes of the operational vehicles that must serve their mandatory tasks in N_3 while striving to maintain or improve each vehicle's profit by including into their routes optional tasks.

The objective function (6) maximizes the profit of the worst-off vehicle. Mandatory task constraints (7) ensure that each mandatory task in N_3 is served once by exactly one vehicle, while optional task constraints (8) ensure that every optional task in N_4 is served at most once. Flow conservation constraints (9) state that if a vehicle serves a task vertex, it should also leave it. Constraints (10) and (11) ensure that each vehicle leaves its actual vertex and returns to the depot, respectively. The adjusted parameters of vehicle k are the remaining autonomy D'_k, the remaining capacity Q'_k, and the profit y_k accumulated at the time of the disruptive event. Constraints (12) and (13) ensure that the total travel time and resource demands of the tasks to be served by each vehicle $k \in K'$ do not exceed its remaining autonomy D'_k nor capacity Q'_k, respectively. Constraints (14) set the arrival time at each vertex and eliminate subtours. The starting time from each vehicle is determined by constraints (15), each vehicle $k \in K'$ begins its updated route after completing the task s_k at the vertex it is currently located at or en route to. Parameter z^e_k represents the completion time of task s_k. Constraints (16) ensure that task time windows are satisfied. We take in consideration the vehicle profits accumulated before the disruptive event and do not reassign already served tasks in set N_1. Fairness constraints (17) keep track of the worst-off vehicle profit y. Each vehicle k's profit y_k accumulated from the tasks served before the disruptive event is added to the profit of its tasks to be served in the newly found route. In this way, we balance the efficiency and fairness of the routes considering the overall profit received by each vehicle in the time horizon $[l_0, u_0]$.

3.2.2. Decomposed formulation for DVRP-FPTW

Next, we present a decomposed mathematical program designed for a multi-agent system, consisting of a coordinator agent and a fleet of vehicles with computation and communication capabilities.

Applying Dantzig–Wolfe decomposition (Lübbecke & Desrosiers, 2005; Pinto et al., 2015), the dynamic centralized model (6)–(18) is reformulated into a Dynamic Restricted Master Problem (DRMP) formulation of the DVRP-FPTW. Each vehicle $k \in K'$ is considered with a set of feasible routes Θ_k, starting at the current location s_k and finishing in the depot 0.

Table 1
Parameters and decision variables of the DVRP-FPTW.

Parameters	Symbol	Description
Shared fleet information $\mathcal{G}$		
Vertices	V'	Set of vertices, $V' = \{0\} \cup N_2 \cup N_3 \cup N_4$.
Arcs	A'	Set of arcs $(i,j) \in V' \times V'$.
Depot	0	Depot vertex, $0 \in V'$.
Serviced tasks	N_1	Set of served tasks.
Current tasks	N_2	Set of current tasks of the operational vehicles at the time instant of disruption.
Mandatory tasks	N_3	Set of pending mandatory tasks' vertices.
Optional tasks	N_4	Set of pending optional tasks' vertices.
Distances	d_{ij}	Travel distance of arc $(i,j) \in A'$.
Task revenue	r_i	Nonnegative revenue obtained by serving task $i \in N_3 \cup N_4$.
Task demand	q_i	Nonnegative demand associated with each task $i \in N_3 \cup N_4$.
Task time window	$[l_i, u_i]$	Time window for serving task $i \in N_3 \cup N_4$.
Vehicles	K'	Updated set of operational vehicles $k \in K'$.
Disruption event time	z^e	Disruption time of event e.
Private information $\mathcal{I}_k$ *for each vehicle* $k \in K'$		
Travel cost	o_k	Travel cost of vehicle k per unit of distance traveled.
Service time	$\hat{t}_{jk}$	Service time of vehicle k required for completing task j.
Travel time	t_{ijk}	Travel time of vehicle k on arc $(i,j) \in A'$ including service time $\hat{t}_{jk}$.
Vehicle autonomy	D'_k	Remaining autonomy of vehicle k in terms of maximum traveling distance.
Vehicle capacity	Q'_k	Remaining capacity of vehicle k.
Vehicle post-disruption location	s_k	Post-disruption vertex location of vehicle k.
Post-disruption start time	z^e_k	Post-disruption start time of vehicle k.
Vehicle profit	y_k	Accumulated profit of vehicle k.
Decision variables		
Route-arc assignment	x_{ijk}	$x_{ijk} = 1$ if vehicle $k \in K$ travels arc $(i,j) \in A'$ and 0 otherwise.
Task completion time	v_i	Time instant of the completion of task i.
Worst-off profit	y	Profit of the worst-off vehicle

We let a^p_{jk} be a binary parameter equal to 1 if route $p \in \Theta_k$ of vehicle k serves task vertex j, and let w^p_k be the profit obtained in route $p \in \Theta_k$ by vehicle $k \in K'$. Moreover, we let δ^p_k be the decision variable that equals 1 if route $p \in \Theta_k$ is selected for vehicle k. The dynamic restricted master problem (DRMP) of the DVRP-FPTW is then:

$$\max \quad y \tag{19}$$

$$s.t. \quad \sum_{k \in K'} \sum_{p \in \Theta_k} a^p_{jk} \delta^p_k = 1, \ \forall j \in N_3, \tag{20}$$

$$\sum_{k \in K'} \sum_{p \in \Theta_k} a^p_{jk} \delta^p_k \leq 1, \ \forall j \in N_4, \tag{21}$$

$$\sum_{p \in \Theta_k} w^p_k \delta^p_k + y_k \geq y, \ \forall k \in K', \tag{22}$$

$$\sum_{p \in \Theta_k} \delta^p_k = 1, \qquad \forall k \in K', \tag{23}$$

$$y \in \mathbb{R}, \delta^p_k \in \{0,1\}, \quad \forall k \in K', \forall p \in \Theta_k, \tag{24}$$

The objective function (19) maximizes the profit y of the worst-off vehicle. Constraints (20) assure serving each mandatory task $j \in N_3$. Constraints (21) assure that all optional tasks $j \in N_4$ are served at most once. Fairness constraints (22) keep track of the minimum profit of the entire fleet K' after disruption. Constraints (23) guarantee that all post-disruption operational vehicles (K') are assigned a route.

The mathematical model (19)–(24) is used by the coordinator agent for the dynamic vehicle-route assignment and the computation of shadow prices (dual variables). The shadow prices are obtained by solving the linear relaxation of the DRMP. They may be interpreted in economic terms as the opportunity cost of resources and can be positive or negative. Specifically, assignment constraints (20) and (21) generate $\pi_j \in \mathbb{R}$ for $j \in N_3$ and $\pi_j \geq 0$ for $j \in N_4$, indicating the marginal cost of including task j in a route. Likewise, fairness constraints (22) and fleet utilization constraints (23) produce shadow prices $\mu_k \leq 0$ and $\alpha_k \in \mathbb{R}$, respectively, for each vehicle $k \in K'$, representing the marginal cost associated with each vehicle. The column generation subproblem associated with the DRMP is an Elementary Shortest Path Problem with Resource Constraints (ESPPRC) (Dror, 1994).

4. DIMASA: Distributed multi-agent system architecture for the DVRP-FPTW

We propose next a new decision-making architecture for the DVRP-FPTW, leveraging a distributed MAS that we name DIMASA. Illustrated in Fig. 1, this architecture includes a Coordinator Agent and multiple Vehicle Agents that mutually collaborate to find the best vehicle routes. The optimization process involves solving both the restricted master problem and dedicated column generation subproblems (pricing problems) through a distributed and asynchronous implementation of the column generation approach, aligning with methodologies in Basso and Ceselli (2017, 2022).

4.1. Coordinator agent

The coordinator agent ensures an equitable distribution of profit and efficiency across the fleet through *systematic optimization of egalitarian social welfare for VRP-FPTW* that we propose in this paper and that extends the model for the multiple Travelling Salesman Problem (mTSP) from López-Sánchez et al. (2022, 2023b). This approach prioritizes vehicle performance optimization in non-decreasing order of vehicle routes' profits.

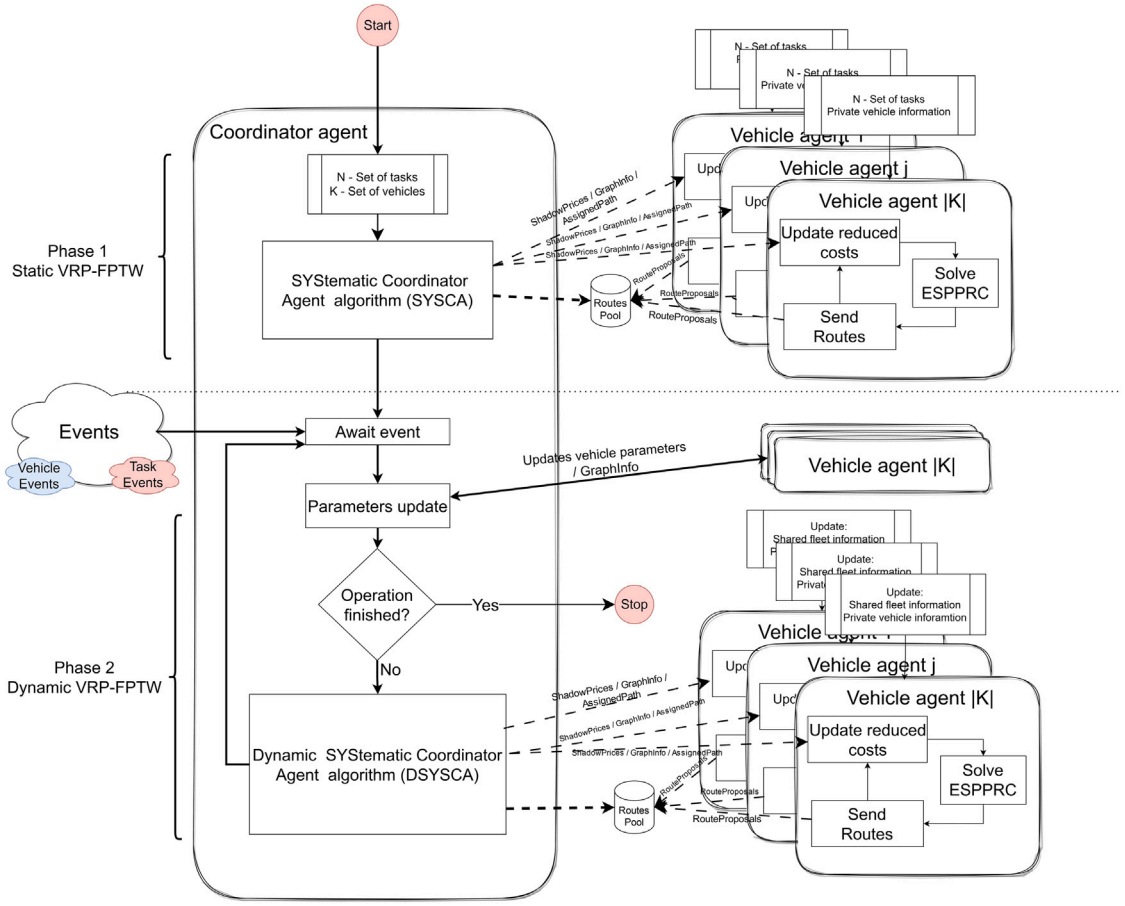


Fig. 1. DIMASA architecture for the Static/Dynamic VRP-FPTW.

The coordinator's decision-making process involves two iterative phases, Fig. 1: before the start of the fleet's operation, routes for the fleet's vehicles that serve pending tasks are found (phase 1); these routes are adapted in response to unexpected events occurring during operation (phase 2). Phase 1 assigns a set of tasks N to the fleet set K based on the static VRP-FPTW model (1)–(5) following the Systematic Coordinator Agent Algorithm – SYSCA (Algorithm 1), while phase 2 considers sets of tasks in N_3 and N_4 for the assignment among available vehicles in set K' by solving the DRMP model (19)–(24), following the Dynamic Systematic Coordinator Agent Algorithm – DSYSCA (Algorithm 3).

DIMASA architecture supports seven types of messages between the coordinator and vehicle agents to manage information exchange: *'GraphInfo'*, *'VehicleEvent'*, *'AssignedRoute'*, *'ShadowPrices'*, *'RouteProposals'*, *'Acknowledged(k, msg)'*, and *'StopRouteFinding'*. Updated shared fleet information from Table 1 is found inside the *'GraphInfo'* messages, required by the vehicles to update their graph information. The *'VehicleEvent'* message informs of a breakdown or addition of a vehicle into the fleet. An *'AssignedRoute'* message sent by the coordinator indicates the route that a vehicle should follow. Also, private shadow prices μ_k and α_k are sent by the coordinator only to the corresponding vehicle $k \in K$ together with (shared) marginal costs π_j for each task $j \in N_3 \cup N_4$ in a *'ShadowPrices'* message. The *'RouteProposals'* message sent from a vehicle agent to the coordinator agent contains a set of pairs $\{(k, p, w_k^p)\}$ with feasible routes with negative reduced costs p and their related profit w_k^p, if any. Ultimately, while the asynchronous nature characterizes the proposed column generation approach, it is essential to note the presence of a synchronization point within the devised solution strategy. This point of convergence is strategically facilitated through the utilization of a specific message, namely, *'StopRouteFinding'*.

Algorithm 1: SYSCA Algorithm

Input: N - Set of tasks, K - Set of vehicles.
Output: For all vehicles $k \in K$: Send routes p, where $(k, p) \in S$.

$K'' = K$;
$N'' = N$;
$S = \emptyset$;
while $K'' \neq \emptyset$ **do**
  $P = VRP - FTPW(N'', K'')$; // (Algorithm 2)
  $(k_{min}, p_{min}) = arg\min_{k \in K'', p \in P}\{w_k^p\}$;
  Send $Assigned\,Route(k_{min}, p_{min})$ to vehicle k_{min} ;
  $K'' = K'' \setminus \{k_{min}\}$;
  $N'' = N'' \setminus Tasks(p_{min})$;
  $S = S \cup \{(k_{min}, p_{min})\}$;
end

4.1.1. Phase 1

In Phase 1, the coordinator agent uses the SYStematic Coordinator Agent (SYSCA) Algorithm (Algorithm 1) to iteratively assign a route to each vehicle in the fleet prioritizing the vehicles with the least profit.

SYSCA Algorithm (Algorithm 1). Initially, the coordinator agent receives the information about the set of tasks N and vehicles K (Input). SYSCA runs over $|K|$ optimization iterations where the auxiliary sets N'' and K'' represent the pending unassigned tasks and vehicles at each iteration, respectively.

At each iteration, the coordinator agent solves the static VRP-FPTW model by employing Algorithm 2 (CA Algorithm from López-Sánchez et al. (2023a)). This is done in collaboration with operational vehicle

Algorithm 2: VRP-FPTW Algorithm

Input: N - set of tasks, K - set of vehicles
Output: $P = \{(k, p, w_k^p), \forall k \in K\}$ - set of assigned routes

1 Send $GraphInfo()$ to all vehicles $k \in K$;
2 Initialize routes;
3 Calculate shadow prices ($\forall j \in N : \pi_j$, $\forall k \in K : \mu_k$ and α_k) in SRMP;
4 Send $ShadowPrices(\pi_j, \forall j \in N; \mu_k, \alpha_k)$ to all vehicles $k \in K$;
5 **repeat**
6 **if** *new 'RouteProposals'*$(\{(k, p, w_k^p)\})$ *received from any* $k \in K$ **then**
7 $\Omega_k = \Omega_k \cup \{(k, p, w_k^p)\}$;
8 Calculate shadow prices ($\forall j \in N : \pi_j$, $\forall k \in K : \mu_k$ and α_k) in SRMP;
9 Send $ShadowPrices(\pi_j, \forall j \in N; \mu_k, \alpha_k)$ to all vehicles $k \in K$;
10 **end**
11 **until** *Termination criteria*;
12 Send $StopRouteFinding()$ to each vehicle $k \in K$;
13 $\{(k, p), \forall k \in K\}$= SRMP($\{\Omega_k, \forall k \in K\}, N$);
14 $P = \{(k, p, w_k^p), \forall k \in K\}$;
15 Return P

agents in K, where each one runs in parallel its own copy of the DRGV Algorithm (Algorithm 5)).

Based on the promising routes received from the vehicle agents, the coordinator agent selects a route for each one of the vehicles aiming at the maximization of the profit of the worst-off vehicle. Then, vehicle k_{min} with the minimum profit of its route p_{min} is selected and the coordinator sends an $AssignedRoute(k_{min}, p_{min})$ message to that vehicle indicating its assigned route. Vehicle k_{min} and tasks ($Tasks(p_{min})$) that are part of its route p_{min}, are removed from sets K'' and N'', respectively. This process is repeated in subsequent iterations, to select the routes for the next worst-off vehicle (one by one) until all $|K|$ vehicles in the fleet are assigned a route.

VRP-FPTW Algorithm (Algorithm 2). For self-sufficiency of this work, we explain the VRP-FPTW solution approach – Algorithm 2 (CA Algorithm from López-Sánchez et al. (2023a)). This algorithm finds a route for each vehicle $k \in K$ by only focusing on the maximization of the profit of the worst-off vehicle.

The input to the algorithm is composed of a set of tasks N and a set of vehicles K while the output is a set of vehicle routes $P = \{(k, p, w_k^p) \mid k \in K\}$ where the profit of the worst-off vehicle is maximized.

First, the coordinator sends a $GraphInfo()$ message containing tasks in N to all vehicles $k \in K$ (line 1). The routes are initialized on line 2 by creating an artificial solution for the SRMP model (1)–(5), where a single artificial vehicle serves all the tasks obtaining a large (infinitely) negative profit. Thanks to this artificial solution, the linear relaxation of the model can be carried out. Next, the coordinator calculates shadow prices by solving the linear relaxation of the SRMP (line 3), and sends a *'ShadowPrices'* message to each vehicle agent with its dedicated private information, thus maintaining privacy in the system (line 4).

Whenever the coordinator receives any vehicle's new routes with negative reduced costs, it adds them into the set $\Omega = \{\Omega_k, \forall k \in K\}$ of columns for the SRMP (line 7) and updates and sends shadow prices to all vehicles $k \in K$ (lines 8 and 9). This asynchronous and iterative process is repeated until the Termination criteria are met (line 11).

Choosing appropriate termination criteria is a crucial algorithmic decision in column generation, as there is no formula that works well for all instances (Basso & Ceselli, 2022; Bennett et al., 2000). Termination criteria based on the number of iterations are not reliable in this context. Since the messages exchanged among the vehicles and the coordinator agent are asynchronous, they can arrive at different iterations, so termination criteria based on the computation time or the number of columns are more suitable. The criteria we implement in our solution approach include the receipt of an empty route proposal by all vehicles $k \in K$ (implying that vehicles cannot find any new routes with negative reduced costs), or when a given time limit is reached.

Based on the features and the analysis of our problem (López-Sánchez et al., 2023a), we have fixed the time limit for the coordinator agent to 10 min for the static case and 1 min for the dynamic case, and for the vehicle agent in 1 min and 10 s, respectively. However, in all the experiments performed, our algorithm finished before the time limit was reached, which shows its effectiveness.

There may exist vehicles that have not finished the computation process for finding their routes with negative reduced cost within this time limit. Thus, the coordinator agent sends to all vehicles a *'StopRouteFinding'* message (line 12) to ensure that they are informed of the termination of Algorithm 2. Finally, the coordinator agent solves the SRMP (line 13), and the algorithm returns a set of routes $P = \{(k, p, w_k^p)\}$ (line 15). At termination, every vehicle $k \in K$ is assigned route p. However, if the original artificial solution with the negative profit is found in the final assignment, the algorithm has not found a solution.

Thanks to the systematic optimization of egalitarian social welfare, the values of the worst-off profit y in one iteration may surpass the profit from the previous iteration(s). By adopting this iterative method, we ensure both equity and efficiency for all vehicles. The advantage of computing the systematic egalitarian social welfare in this way is that the solution of the SRMP takes advantage of the previously computed routes to produce new shadow prices and reiterate the process.

4.1.2. Phase 2

In phase 2, the coordinator agent assumes an idle state awaiting a disruptive event denoted as e within the operational timeframe $z^e \in [l_0, u_0]$.

Upon a disruptive event, the coordinator triggers the Dynamic Systematic Coordinator Agent Algorithm (DSYSCA) as delineated in Algorithm 3.

DSYSCA Algorithm (Algorithm 3). After classifying event e, Dynamic SYStematic Coordinator Agent (DSYSCA) Algorithm updates the set of available vehicles K', mandatory tasks N_3, and optional tasks N_4 as follows: a new task n and a new vehicle k are added to subset N_3 and K', respectively (lines 2 and 4); a broken down vehicle k is removed from the fleet K' (line 8) and set $N^k \subset p_k$ of its not yet served tasks are added to optional tasks in N_4 (line 9).

Next, similar to the SYSCA algorithm (Algorithm 1), the coordinator agent instantiates auxiliary sets N_3', N_4', S', and K'' representing the pending mandatory and optional unassigned tasks, the solution set, and operational vehicles in each iteration, respectively (lines 11–13). The algorithm finds the routes for the vehicles following the systematic egalitarian social welfare optimization (lines 14–22). It performs $|K'|$ iterations where in each one, a route maximizing the profit of the worst-off vehicle k_{min} is found considering tasks from N_3 and N_4. In each iteration, the coordinator solves the DVRP-FPTW (Algorithm 4), line 15.

DVRP-FPTW Algorithm (Algorithm 4).

This algorithm follows the idea of the VRP-FPTW Algorithm (Algorithm 2) but solves the dynamic restricted master problem (DRMP) (19)–(24) to find shadow prices (lines 3 and 8). The coordinator initializes an empty set, denoted as Θ, to store feasible routes with negative reduced costs received from each operational vehicle post-disruption.

Whenever the coordinator receives any new routes with negative reduced costs, it adds them to the set $\Theta = \{\Theta_k, \forall k \in K'\}$ of columns for the DRMP (line 7) and calculates and sends shadow prices to all

Algorithm 3: DSYSCA Algorithm

Input: K' - set of vehicles, N_3 - set of pending mandatory tasks, N_4 - set of pending optional tasks, e - disruptive event
Output: For all vehicles $k \in K'$: Send new routes p, where$(k,p) \in S'$.

1 **if** *e: New_task {n}* **then**
2 $N_3 \leftarrow N_3 \cup \{n\}$;
3 **end**
4 **if** *e: New_vehicle {k}* **then**
5 $K' \leftarrow K' \cup \{k\}$;
6 **end**
7 **if** *e: Vehicle_breakdown {k}* **then**
8 $K' \leftarrow K' \setminus \{k\}$;
9 $N_4 \leftarrow N_4 \cup N^k$;
10 **end**
11 $N_3' = N_3,\ N_4' = N_4$;
12 $S' = \emptyset$;
13 $K'' = K'$;
14 **while** $K'' \neq \emptyset$ **do**
15 $P' = DVRP - FPTW(N_3', N_4', K'')$; // (Algorithm 4)
16 $(k_{min}, p_{min}) = \arg\min_{k \in K'', p \in P'} \{w_k^p\}$;
17 Send $Assigned\,Route(k_{min}, p_{min})$ to vehicle k_{min} ;
18 $K'' = K'' \setminus \{k_{min}\}$;
19 $N_3' = N_3' \setminus Tasks(p_{min})$;
20 $N_4' = N_4' \setminus Tasks(p_{min})$;
21 $S' = S' \cup \{(k_{min}, p_{min})\}$
22 **end**

Algorithm 4: DVRP-FPTW Algorithm

Input: N_3' - Set of mandatory tasks, N_4' - Set of optional tasks, K' - Set of operational vehicles
Output: $P' = \{(k, p, w_k^p), \forall k \in K\}$ - set of assigned routes

1 Send $GraphInfo(N_3', N_4')$ to all vehicles $k \in K'$;
2 $\Theta = \emptyset$;
3 Calculate shadow prices ($\forall j \in N_3' \cup N_4' : \pi_j$; $\forall k \in K' : \mu_k$ and α_k) ;
4 Send $ShadowPrices(\pi_j, \mu_k, \alpha_k)$ to all vehicles $k \in K'$);
5 **repeat**
6 **if** *new 'RouteProposals'*$(\{(k,p,w_k^p)\})$ *received from any* $k \in K'$ **then**
7 $\Theta_k = \Theta_k \cup \{(k, p, w_k^p)\}$;
8 Calculate shadow prices ($\forall j \in N_3' \cup N_4' : \pi_j$; $\forall k \in K' : \mu_k, \alpha_k$) ;
9 Send $ShadowPrices(\pi_j, \forall j \in N_3' \cup N_4'; \mu_k, \alpha_k)$ to all vehicles $k \in K'$) ;
10 **end**
11 **until** *Termination criteria*;
12 Send $StopRouteFinding()$ to each vehicle $k \in K'$;
13 $\{(k,p), \forall k \in K'\}$= DRMP$(\{\Theta_k, \forall k \in K'\}, N)$;
14 $P' = \{(k, p, w_k^p), \forall k \in K'\}$;
15 **return** P'

vehicles $k \in K'$ (lines 8 and 9). This asynchronous and iterative process is repeated until the Termination criteria are met (line 11).

The *'StopRouteFinding'* message (line 12) is sent to all vehicles $k \in K'$ to inform them of the termination of the algorithm.

Solution $P' = \{(k, p, w_k^p), \forall k \in K\}$ found by solving the DRMP contains an assigned route p for each vehicle $k \in K'$ and its associated profit w_k^p.

Algorithm 5: DRGV Algorithm followed by each vehicle $k \in K'$

Input: $\mathcal{G}$ - shared fleet information, $\mathcal{I}_k$ - private vehicle information
Output: $RouteProposals(\{(k, p, w_k^p)\})$ sent to coordinator agent

1 $InitializeESPRC(\mathcal{G}, \mathcal{I}_k)$;
2 $terminate \leftarrow$ False ;
3 **while** !$terminate$ **do**
4 $msg \leftarrow WaitMessage()$;
5 **Send** $Acknowledged(k, msg)$;
6 **if** *msg == GraphInfo()* **then**
7 $UpdateGraphInfo()$;
8 **if** *msg == StopRouteFinding()* **then**
9 $StopESPPRC()$;
10 **if** *msg == AssignedRoute(k,p)* **then**
11 **do in parallel**
12 $RunRoute(p)$
13 **if** *RouteFinished(p)* **then**
14 Send $RouteFinished(k, p)$ to coordinator agent
15 $terminate \leftarrow$ True
16 **goto line 4**
17 **if** *msg==ShadowPrices*$(\pi_j, \forall j \in N; \mu_k, \alpha_k)$ **then**
18 $C_G \leftarrow UpdateReducedCosts(\pi_j, \forall j \in N; \mu_k, \alpha_k)$;
19 $\{p\} \leftarrow SolveESPPRC(\mathcal{G}, \mathcal{I}_k, C_G)$;
20 **Send** $RouteProposals(\{(k, p, w_k^p)\})$;

After an event is processed and new routes are obtained, the coordinator returns to the idle state awaiting new events. Once all the pending tasks have been served, the condition *Operation finished?* in Fig. 1 returns "Yes" to indicate that the operation is over.

4.2. Vehicle agent

To generate new feasible routes with negative reduced cost (new columns in the SRMP in phase 1 and the DRMP in phase 2) in response to updated shadow prices received from the coordinator, each vehicle agent $k \in K$ maintains a message pool with the coordinator, wherein messages are retained until the vehicle agent reads and responds to them. The decision making of each vehicle is determined by the *Distributed Route Generation by Vehicle* (DRGV) Algorithm (Algorithm 5) that sends messages to the coordinator.

DRGV Algorithm (Algorithm 5). The algorithm initializes by receiving from the coordinator agent the shared fleet information $\mathcal{G}$ (Table 1). Each vehicle k keeps its own private information $\mathcal{I}_k$ in memory.

The algorithm is iterative. At each iteration, a vehicle reads and processes the messages from the coordinator (lines 4-20). If it is the *'GraphInfo'* message, it updates its own graph (line 7). If it receives a *'StopRouteFinding()'* message (line 8), it signals the termination of the ESPPRC algorithm by $StopESPPRC()$. Upon receipt of the *'AssignedRoute(k,p)'* message (line 10), by $RunRoute(p)$, the vehicle starts serving the tasks in its route and, in parallel, awaits messages for any route changes (line 17); if no messages are received before the route is finished, the vehicle sends $RouteFinished(k, p)$ message to the coordinator (line 13) and the algorithm terminates.

Each time a vehicle receives a *'ShadowPrices'* message with updated values μ_k, α_k, π_i for all $i \in N$ (line 17), it updates its reduced costs in $C_G = \{\hat{c}_{ijk} \mid (i,j) \in A\}$ (line 18). These updates are made as follows:

- $\hat{c}_{ijk} = (\pi_j + (r_i - c_{ijk}) \cdot \mu_k)$, for all arcs $(i,j) \in V \times N$, representing the revised reduced cost for arcs $(i,j) \in V \times N$ considering revenue r_i and cost c_{ijk} for vehicle k.
- $\hat{c}_{i0k} = (r_i - c_{i0k}) \cdot \mu_k + \alpha_k$, indicating the new reduced cost for all arcs $(i,0)$, where $i \in V$ returning to depot 0.

In the $SolveESPPRC(\mathcal{G}, \mathcal{I}_k, C_G)$ function (line 19), each vehicle agent resolves its own column generation subproblem, the Elementary Shortest Path Problem with Resource Constraints (ESPPRC), which is an NP-Hard problem (Dror, 1994).

The ESPPRC seeks the best route with negative reduced cost and without violating any constraints for each vehicle $k \in K'$. The route starts from current task s_k to depot 0. Because the ESPPRC is computationally expensive, we use two different algorithms to find good routes with negative reduced costs: 1) a labeling heuristic algorithm based on the SPPRC (Shortest Path Problem with Resource Constraints) (Irnich & Villeneuve, 2006), which avoids cycles and compares routes with a simple dominance rule based on the total reduced cost; 2) an exact method that follows the ESPPRC rules as proposed by Feillet et al. (2004), which makes sure that we do not miss any promising routes with negative reduced cost. Given the complexity of the ESPPRC, we have imposed a time limit within which the algorithm is expected to return a set of best solutions or an empty route if none is found. Subsequently, in the *'RouteProposals'*$(\{(k, p, w_k^p)\})$ message (line 20), the vehicle agent transmits to the coordinator a set of its best feasible routes $\{p\}$ along with their associated profits w_k^p, thus preserving vehicle privacy.

The messages *'GraphInfo'*, *'AssignedRoute'*, *'StopRouteFinding'*, and *'Acknowledged'* are crucial for the synchronization between the coordinator and the vehicle agents. These messages serve as synchronization points between the asynchronous algorithms. The communication system must ensure that these messages have been received timely and orderly before proceeding. If the coordinator does not receive an *'Acknowledged(k,msg)'* from a vehicle, it is considered broken. The coordinator agent generates a *'Vehicle_breakdown{k}'* event, adding its pending tasks to the set of optional tasks N_4.

5. Use case and functional experiments

In this Section, we first demonstrate the functioning of the proposed DIMASA architecture for DVRP-FPTW through a use case example and then discuss its computational complexity and demonstrate its scalability through extensive functional experiments.

5.1. Use case example

Consider the scenario illustrated in Fig. 2. Suppose we have a set of 10 tasks that must be served. The fleet is composed of 3 homogeneous vehicles initially positioned at depot $\{0\}$, as shown in Fig. 2(a). In phase 1 of the DIMASA architecture, an initial routing plan is found. This plan assigns each vehicle to a specific set of tasks, resulting in the following:

- Vehicle 1: Route $[0, 4, 6, 10, 8, 0]$ with a profit of 97.31.
- Vehicle 2: Route $[0, 5, 9, 3, 1, 0]$ with a profit of 165.78.
- Vehicle 3: Route $[0, 2, 7, 0]$ with a profit of 106.35.

Vehicle 1 starts its route, but at time $z^e = 20$, while it is serving task 6, it breaks down and does not send the $Acknowledged(1, msg)$ message after receiving the $GraphInfo()$ message from the coordinator agent, Fig. 2(b). The coordinator agent generates a $Vehicle\_breakdown\{1\}$ event. This event triggers adding vehicle 1's pending tasks $[8, 10]$ into the set of optional tasks N_4. The coordinator removes vehicle 1 from the set of functioning vehicles K'.

In this scenario, the task sets are $N_1 = [4, 6, 5, 9, 2, 7]$, $N_2 = [9, 7]$, $N_3 = [3, 1]$ and $N_4 = [8, 10]$, and the private information of the vehicles is:

- Vehicle 1: $s_1 = 6, v_6 = 21.12, y_1 = 50.00, D_1' = 75.32, Q_1' = 37.0$.
- Vehicle 2: $s_2 = 9, z_2^e = v_9 = 24.04, y_2 = 118.97, D_2' = 90.87, Q_2' = 41.0$.
- Vehicle 3: $s_3 = 7, z_3^e = v_7 = 22.04, y_3 = 121.22, D_3' = 77.95, Q_3' = 32.0$.

The coordinator agent solves the DRMP (19)–(24) considering the previous routes of the remaining operational vehicles $K' = \{2, 3\}$, Fig. 2(c), aiming at finding appropriate route modifications. Phase 2 of our DIMASA architecture obtains the solution illustrated in Fig. 2(d):

- Vehicle 2 had no tasks added, maintaining its pending route $[9, 3, 1, 0]$ with an initially planned profit of 165.78.
- Vehicle 3 was added tasks 10 and 8 into its new route $[7, 10, 8, 0]$ with an updated profit of 167.33.

As may be seen from this example, the proposed DIMASA architecture enables us to effectively manage unexpected events, such as a vehicle breakdown.

The proposed solution approach is designed to solve a class of problems that are known as NP-hard, which means that there is no efficient way to find the optimal solution in general, nor is there a termination criterion that ensures finding a solution. However, in practice, the DIMASA architecture works very well and can find solutions in a relatively short amount of time, which is shown next in the functional experiments.

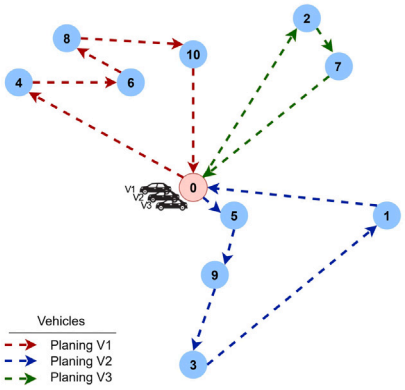
5.2. Functional experiments

In this Section, we evaluate the performance of the proposed DIMASA architecture. In phase 1, the proposed SYSCA and VRP-FPTW algorithm are implemented in the coordinator agent that interacts with the DRGV algorithm implemented in each of the vehicle agents. In phase 2, the coordinator's DSYSCA and the DVRP-FPTW algorithm are in interaction with the DRGV algorithm implemented in each of the vehicle agents. We compare the results of the proposed distributed multi-agent solution approach for the proposed DVRP-PFPTW problem with those derived from the centralized monoblock model (Centr.), (6)—(18). The results of the centralized model are obtained using the Gurobi 10.0.2 solver with *gurobipy* Python library with a time limit of one hour. We run the simulation experiments on an Intel Xeon Gold 6226R[1] virtualized cluster with 16 CPUs, 32 GB of RAM and a clock frequency of 2.9 GHz.

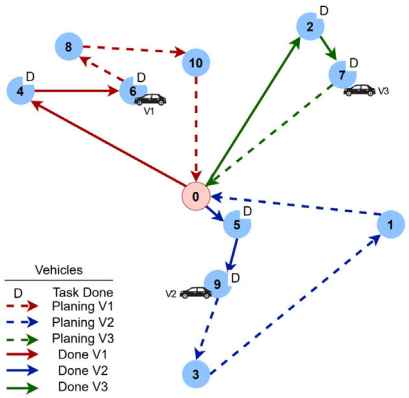
Although the centralized model offers quality of solution guarantees, its suitability diminishes in inherently distributed vehicle fleet environments due to high computational complexity, as shown by our simulation results. This is why we evaluate our proposed approach also by comparing its results with those obtained from a heuristic that we propose in this paper and name Systematic Cheapest Insertion (SCI), Algorithm 6.

The SCI Algorithm inserts optional tasks into pre-determined routes of the operational vehicles as long as their profits improve, starting from the least profitable one. Initially, the algorithm sorts the set of operational vehicles K' into the set $sorted\_K'$, arranging them in non-decreasing order based on their obtained profits (line 2). Then, for each vehicle in the ordered set $sorted\_K'$, the algorithm iteratively attempts to insert each optional unassigned task $n \in N_4^{ua}$ while keeping track of the task n_{max} with the best profit w_{max} (lines 3–23). Function $insert(v, p, n)$ attempts insertion of task n among any two vertices within the route p of vehicle k, while satisfying the time window, autonomy, and capacity constraints and returns the route p_{new} with the highest profit w_{new}, if any (line 9). If at least one insertion is feasible, the function returns the best updated route p_{new} and the maximum associated profit w_{new}; otherwise, it returns $(\emptyset, 0)$, meaning the insertion is not feasible. Then, if the profit of the found path w_{new} is higher than the best profit found so far w_{max}, the best path p_{max} and the inserted task n_{max} are updated (lines 10–13). When all unassigned tasks have been tested for vehicle k, if its route p_k has been updated with task n_{max} (lines 16–19), task n_{max} is removed from the set of unassigned tasks

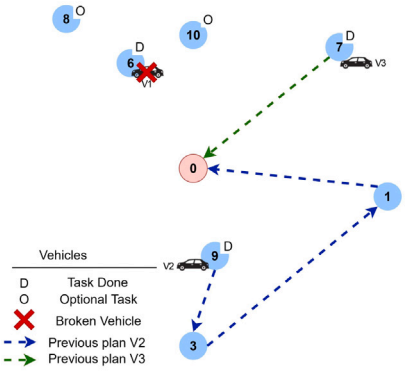
[1] https://www.intel.es/content/www/es/es/products/sku/199347/intel-xeon-gold-6226r-processor-22m-cache-2-90-ghz/specifications.html



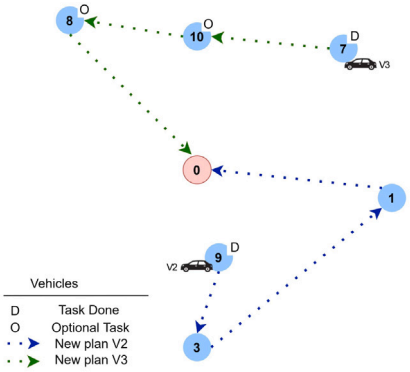
(a) Initial planning.



(b) Tasks done until time $z^e = 20$ of vehicle 1 breakdown event



(c) Vehicle 1 breaks down.



(d) Reassignment of the tasks of vehicle 1 to other vehicles.

Fig. 2. Example of the execution of the dynamic approach when vehicle 1 breaks down in the middle of its route.

N_4^{ua} (line 17). Route p_{max} with the highest nonnegative profit margin w_{max} in respect to the previously assigned route p_k becomes the new updated route (line 18), and vehicle k is reinserted into the ordered set $sorted\_K'$ based on its updated profit (line 19). If the route has not been updated (line 21), the current route is retained, and route data P' is updated with the current route p_{max} with associated profit w_{max} for vehicle k. The algorithm terminates when $sorted\_K'$ is empty, i.e. when all operational vehicles have been considered and returns the set of new updated routes for each. This implies that there are no more unassigned optional tasks in N_4^{ua} that can improve the profits of operational vehicles if inserted in their routes.

Contrary to the DIMASA architecture, the proposed SCI algorithm does not allow the exchange of obligatory tasks between vehicles to obtain better solutions and its efficiency depends on the order of the tasks in the unassigned optional task set N_4^{ua}.

5.2.1. Benchmark instances for the static VRP-FPTW

The proposed DVRP-PFPTW is a new problem that was not studied previously. Therefore, there are no relevant state-of-the-art benchmarks. Consequently, in this Section, we provide the strategy for creating them as well as a set of publicly accessible[2] benchmark instances together with a set of relevant Key Performance Indicators for the DVRP-FPTW problem.

Inspired by the generation of benchmark instances (Li et al., 2009; Solomon, 1987; Soriano et al., 2023; Uchoa et al., 2017), we consider a 2-dimensional Euclidean space $[0, 100] \times [0, 100]$, with a single depot located in the center of this space, i.e., at $(50, 50)$ and multiple task vertices distributed uniformly randomly over the same. The distance matrix $\mathbf{d} = \{d_{ij} | (i, j) \in A\}$ is created based on the Euclidean distance between each pair of vertices in the given space. For simplicity, we assume that the cost and travel times matrix for each vehicle is equal to the distance matrix. The parameters related to tasks include their location coordinates (x, y), demand, revenue and time windows. Tasks demand is uniformly randomly distributed between 1 and 10 and task revenue between 10 and 100. The vehicle parameters include the number of vehicles, the cost, velocity, autonomy, and capacity.

Feasibility and binding constraints. In the creation of benchmark instances, we ensure that the autonomy, capacity, and timing constraints are binding by specifically adjusting the values of time windows $[l_i, u_i]$ for each task i as well as autonomy D_k and capacity Q_k for each vehicle k. Defining these values is still an open challenge for the capacitated VRP (Uchoa et al., 2017). This is essential since if the problem is too constrained, it might not be feasible, while if it is too loose, the constraints that distinguish this problem from the capacitated VRP will not be considered. Thus, the parameters of the proposed instances are determined in such a manner that they strike a balance between being sufficiently restrictive to enforce the binding nature of associated constraints and yet guaranteeing the presence of a feasible solution.

[2] https://github.com/aitorls/DVRP-PFPTW-instances.git

Algorithm 6: Systematic Cheapest Insertion (SCI) Algorithm

Input: N_4 - Set of optional tasks, K' - Set of operational vehicles, Previous route data - $P = \{(k, p_k, w_k^p) \mid k \in K'\}$.
Output: New route data - $P' = \{(k, p_{max}, w_{max}) \mid k \in K'\}$.

Initialization: $P' \leftarrow \emptyset$; $N_4^{ua} \leftarrow N_4$;
$sorted\_K' \leftarrow sort(K')$;
while $|sorted\_K'| > 0$ **do**
  $k \leftarrow sorted\_K'.pop(0)$;
  $p_{max} \leftarrow p_k$;
  $w_{max} \leftarrow w_k^p$;
  $n_{max} \leftarrow None$;
  for *each* $n \in N_4^{ua}$ **do**
    $(p_{new}, w_{new}) \leftarrow insert(k, p, n)$;
    if $p_{new} \neq \emptyset$ && $w_{new} > w_{max}$ **then**
      $w_{max} \leftarrow w_{new}$;
      $p_{max} \leftarrow p_{new}$;
      $n_{max} \leftarrow n$;
    end
  end
  if $n_{max} != None$ **then**
    $N_4^{ua} \leftarrow N_4^{ua} \setminus \{n_{max}\}$;
    $p_k, w_k^p \leftarrow p_{max}, w_{max}$;
    $sorted\_K' \leftarrow sort(sorted\_K' \cup \{k\})$;
  else
    $P' \leftarrow P' \cup \{(k, p_{max}, w_{max})\}$;
  end
end
return P'

The procedure we use is as follows. First, the region of interest is partitioned into $|K|$ sections, where $|K|$ is the number of vehicles. The partition is made similarly to a cake-cutting approach, starting from the depot in the center and adding $|K|$ equidistant rays where the angle between the rays is $360/|K|$ and the depot belongs to every created partition, as seen in Fig. 3. Then, we compute the optimal route for serving all the task vertices from the depot 0 by a single vehicle in each of these sections by solving the traveling salesman problem (TSP). Note that these routes only take into account the distances, but not the vehicle autonomy and capacity nor the task time windows. The information related to each of the $|K|$ found TSP routes leaving from and returning to the depot 0 includes a sequence of task vertices, the time needed to serve each of them, the total distance traveled, and the route's accumulated task demand. The route with the maximum accumulated demand among all routes defines the capacity, and similarly, the route with the maximum distance among all routes defines autonomy for each vehicle. To set the time windows for each task, we first determine the center of its time window as the arrival time on the previously found TSP route. Next, we define the time window limits for each task vertex in the same order as the one along each found TSP route. Specifically, the first service time l_i for each task $i \in \{1, \ldots, |V|-1\}$ is set as the center time of the previous served vertex and u_i is set as the center time of the next vertex. In the case of the first task vertex leaving from the depot, l_i is set to the departure time from the depot, and in the case of the last task vertex i in the route before the depot, u_i is set to the arrival time at the depot. We set the first service time l_0 of the depot to 0 while its last service time u_0 is the maximum of the last service times of any task plus the time to return from this task to the depot $u_0 = \max_{i \in N}\{u_i + t_{i0k}\}$, where vehicle k is the one that serves task i.

In this way, we make sure that the problem has at least one feasible solution and that the autonomy, capacity, and timing constraints are binding. We give an example in Fig. 3, with a randomly generated graph in 2-dimensional space in Fig. 3(a), the resulting partitioning of the space in Fig. 3(b), and the optimal TSP solutions for such partitioned space in Fig. 3(c). Moreover, for the same instance, in Fig. 4, we give the optimal solution of (1)-(10) in López-Sánchez et al. (2023a), found with the Gurobi solver, Fig. 4(a). Additionally, in Fig. 4(b), we provide the solution for that instance found by the proposed DIMASA Phase 1 approach.

The presented benchmark specification is used for the creation of the instances that are named SFPTW_N_K_X, where S stands for static, FPTW is the proposed kind of the VRP problem, N is the number of task vertices (25, 50, 75 and 100), K is the number of vehicles (5, 10, 15 and 20). X ranges from 1 to 10 and represents the instance number. The number of vehicles is chosen to maintain the $|N|/|K|$ ratio (the average number of tasks served by a vehicle). The proposed instances are publicly accessible on GitHub.[3]

5.2.2. Benchmark instances for the DVRP-FPTW

In the benchmark creation for the dynamic VRP-FPTW problem, we use the same structure as in the previous instances but extend it with dynamic events that happen throughout the route. The two possible events that we consider in the experiments are the appearance of new tasks and vehicle breakdowns that can occur at any time $t \leq u_0$. When a vehicle breaks down, we assume that it will not be repaired in the considered time horizon. New tasks are defined by their coordinates (x, y), demand q, revenue r, and time windows $[l, u]$:

$$New\_task :< name, t, (x, y), q, r, [l, u] >,$$

while the vehicles that break down with :

$$Vehicle\_breakdown :< k, t > .$$

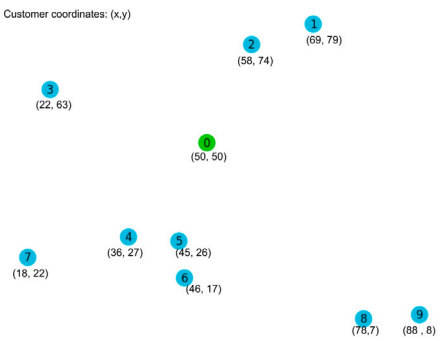
Usually, the addition of new tasks occurs in batches, as in real-world scenarios where tasks often accumulate before being introduced into the system. In this paper, we only conduct experiments on the breakdown of a functional vehicle, because, in this case, the associated tasks' vertices are removed from the mandatory tasks set and added to the optional set. The benchmark instances for the DVRP-FPTW are named DFPTW_N_K_X, where N, K, and X represent the number of tasks, vehicles, and problem instances, respectively. These instances are similar to the static case, but they also include a list of predetermined events that take place during the execution of the simulation.

As presented in Section 3.2, the proposed DVRP-FPTW model considers mandatory tasks that are part of the taken route, and optional tasks that might be considered for inclusion in the route if and only if they bring benefit to a vehicle. Thus, if a new task appears close to the maximum return time to the depot u_0, the remaining vehicles may not have enough time to accommodate them. Similarly, if the capacity and/or autonomy of a vehicle is too limited, they may not be able to serve this task. To show the effectiveness of our Dynamic VRP-FPTW algorithm, we allocate the events of vehicle breakdown at the beginning of the execution plan. This allows for flexibility in the reassignment of tasks to vehicles.

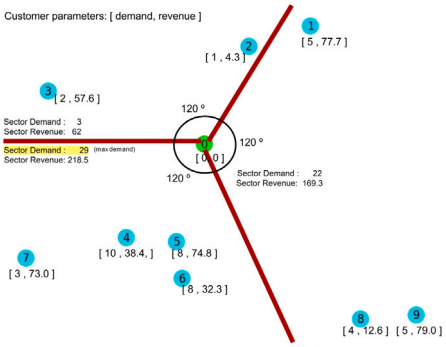
Specifically, we differentiate between scenarios where the best-off (highest profit) vehicle, the worst-off vehicle, and a randomly chosen vehicle suffers a breakdown. By examining these three different cases, we evaluate the performance and adaptability of our algorithm in different profit-based scenarios.

To validate that the proposed approach is able to reassign the optional tasks while not worsening the vehicles' profits, within the time and autonomy constraints, we compare our real-time dynamic algorithm with the exact solution, considering a one-hour computation time limit. The objective of this comparison is to evaluate the impact of having more time to reassign vehicles, where an hour of waiting to recalculate the route is not applicable in dynamic environments. This allows us to evaluate the effectiveness of our algorithm's approach to

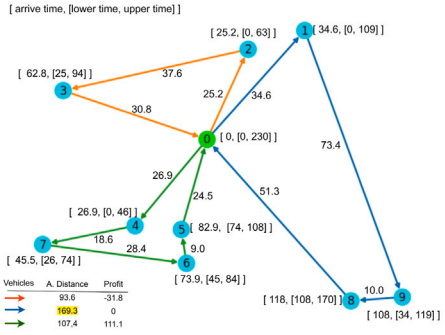
[3] https://github.com/aitorls/DVRP-PFPTW-instances



(a) A random graph in 2D space. (Tasks in blue, depot in red). Coordinates (x,y).

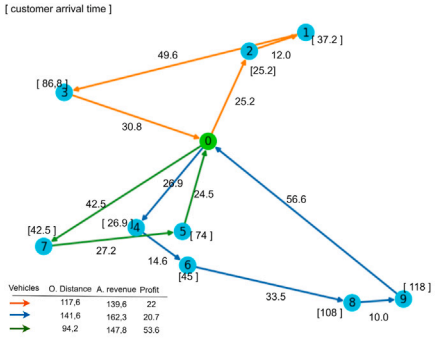


(b) Partition of the graph depends on the vehicles.

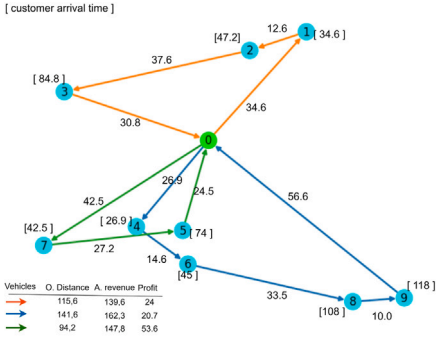


(c) Optimal TSP solution in the partitioned space.

Fig. 3. Example of generation of the parameters for the VRP-FPTW with 9 consumer vertices and 3 vehicles.



(a) Optimal solution given by (1)-(10) in López-Sánchez et al. (2023a).



(b) Optimal Phase 1 solution.

Fig. 4. Examples of optimal solutions for a given instance with the DIMASA Phase 1 .

react to unpredicted events, thus, achieving a dynamically adaptable real-time solution. Additionally, we also compare our approach by increasing the revenue of the optional tasks (extra revenue), to 10000 unit each. Thanks to these experiments, we incentivize vehicles to perform optional tasks when autonomy and time window constraints allow, thereby obtaining an upper limit on the number of optional tasks that can be visited.

5.2.3. Experimental results

Next, we compare the performance of the two phases of the DIMASA architecture with the benchmark centralized models.

DIMASA phase 1 results. The results obtained for the static VRP-FPTW benchmark instances are presented in Tables 2 and 3. In these tables, we report the results achieved by the two models (Phase 1 and Centr.) within a one-hour time limit. For each instance and method, we provide the instance name (Name), the worst-off vehicle profit among all the vehicles (Worst-off), the computation time in seconds (Time) or a star (*) if the one-hour limit is reached, the best upper bound for the worst vehicle's profit (UB) which is the value of the linear relaxation of the models, and the overall profit collected for all the vehicles (Overall). Optimal solutions, where the worst profit and the upper bound value are equal, are highlighted in bold. In Table 2, we observe that for instances with 25 tasks and 5 vehicles, the exact centralized model (Centr.) achieves the optimal solution for all 10 instances with an average computation time of 3.3 s, while our proposed Phase 1 approach achieves the optimal solution in 9 out of 10 instances with an average computation time of 0.8 s. On the instances with 50 tasks and 10 vehicles, the centralized model only obtains the optimal solution in 3 instances (SFPTW_50_10_[0,7,8]) with an average computation time of 1814 s. On the other hand, the DIMASA Phase 1 obtains the optimal

Table 2
Static FPTW benchmark results.

Name	Approach	Worst-off	UB	Overall	Time
SFPTW_25_5_0	Centr.	**78.38**	78.38	525.16	6.54
	Phase 1	**78.38**	–	534.43	0.42
SFPTW_25_5_1	Centr.	**120.69**	120.69	662.09	15.38
	Phase 1	118.42	–	718.72	0.84
SFPTW_25_5_2	Centr.	**51.61**	51.61	672.36	0.83
	Phase 1	**51.61**	–	677.58	0.60
SFPTW_25_5_3	Centr.	**74.68**	74.68	649.80	6.61
	Phase 1	**74.68**	–	633.42	1.18
SFPTW_25_5_4	Centr.	**97.80**	97.80	718.65	0.58
	Phase 1	**97.80**	–	713.80	0.49
SFPTW_25_5_5	Centr.	**17.21**	17.21	533.99	0.40
	Phase 1	**17.21**	–	550.16	0.86
SFPTW_25_5_6	Centr.	**97.05**	97.05	651.54	0.89
	Phase 1	**97.05**	–	651.57	1.68
SFPTW_25_5_7	Centr.	**58.45**	58.45	487.85	0.51
	Phase 1	**58.45**	–	466.95	0.65
SFPTW_25_5_8	Centr.	**91.02**	91.02	829.23	0.36
	Phase 1	**91.02**	–	829.23	0.76
SFPTW_25_5_9	Centr.	**103.52**	103.52	838.70	0.93
	Phase 1	**103.52**	–	840.28	0.70
SFPTW_50_10_0	Centr.	**67.60**	67.60	1273.24	2006.28
	Phase 1	**67.60**	–	1287.87	8.48
SFPTW_50_10_1	Centr.	151.26	161.54	1574.17	*
	Phase 1	137.50	–	1574.64	8.57
SFPTW_50_10_2	Centr.	141.98	147.38	1488.08	*
	Phase 1	127.05	–	1564.56	9.59
SFPTW_50_10_3	Centr.	94.24	129.82	1232.44	*
	Phase 1	55.60	–	1260.26	12.90
SFPTW_50_10_4	Centr.	142.29	175.51	1678.82	*
	Phase 1	133.54	–	1661.76	53.82
SFPTW_50_10_5	Centr.	129.63	147.05	1386.70	*
	Phase 1	111.65	–	1308.03	13.28
SFPTW_50_10_6	Centr.	112.04	166.62	1500.76	*
	Phase 1	83.08	–	1577.97	22.09
SFPTW_50_10_7	Centr.	**24.12**	24.12	1406.72	64.25
	Phase 1	**24.12**	–	1415.14	7.62
SFPTW_50_10_8	Centr.	**53.02**	53.02	1384.81	3372.55
	Phase 1	**53.02**	–	1319.68	8.24
SFPTW_50_10_9	Centr.	141.55	149.51	1577.79	*
	Phase 1	111.39	–	1595.87	12.89

solution for the same 3 instances, but in an average time of 8 s. For the remaining instances with 50 tasks, the centralized model yields an average of 14.2% higher profit for the worst-off vehicle compared to our DIMASA Phase 1 solution. However, DIMASA finds the solution with an average computation time of 15.748 s, while the average computation time of the centralized model is 3064 s. For 3 out of 10 instances with 75 tasks and 15 vehicles, the centralized model (Centr.) does not get any solution in one hour of computation time, Table 3. This is the case for instances SFPTW_75_15_1_[0,2,6]. Finally, when we increase the number of tasks to 100, the centralized model does not find any solution for these instances, we skip the rows where no solution is found. In these cases, it can only provide us with the best bound obtained with the solver. The DIMASA Phase 1, on the other hand, always finds a solution in less than 2 min for instances with 75 tasks with an average computation time of 55 s, and less than 15 min for instances with 100 tasks, with an average computation time of 5.85 min. For the instances where the solution of the centralized algorithm is unknown, we use the best-bound value and determine that the theoretical gap between the obtained DIMASA Phase 1 solution and the value of the linear relaxation is within 30%. This is a theoretical bound that in practice might be significantly lower since, in general, the optimum value is not known. To further improve the performance, the DIMASA Phase 1 solution may be introduced into a branch-and-price algorithm to find the optimal solution. However, the average gap between the DIMASA Phase 1 and the centralized method for the instances for which the centralized solution did find a solution is less than 15%.

In summary, the computation time of the proposed DIMASA Phase 1 solution approach is significantly better than the one of the centralized model, especially as the number of agents and/or tasks increases. Even more, the fleet on average receives a larger profit from the DIMASA Phase 1 than from the centralized and exact VRP-FPTW model in Gurobi solver. Obtaining the exact solution with the centralized model is intractable. For 75 tasks and 5 vehicles, it does not find any solution in 1 h for 3 out of 10 tested instances, while for 100 tasks and 20 vehicles, it does not find a solution for any instance. The proposed DIMASA Phase 1 approach, on the other hand, obtains solutions for all tested instances within less than 15 min with an average gap with respect to the exact solution of the centralized model of less than 15%, when it is identified.

DIMASA phase 2 results. In Tables 4, 5, and 6, we report the results for the Dynamic VRP-FPTW problem with 100 tasks and 20, 30, and 40 vehicles, respectively. For each experiment, we compare the SCI solution, the Phase 2 solution, the centralized dynamic solution, and the solution with more revenues in the optional tasks. We also report the information of the dynamic instance, such as the static instance name SFPTW_N_K_X where N, K and X are the number of tasks, vehicles, and instance number, the broken vehicle (B. vehicle), the broken time (B. time), the finish expected time of the vehicle (exp. finish), the last time to return to the depot for all the vehicles (u_0), the current value of the worst-off vehicle (Worst-off), the number of tasks of the broken vehicle that it planned to serve (Plan) and, lastly, the number of tasks left without serving due to vehicle breakdown (Remain). The DIMASA Phase 2 solution columns give the objective function value, that is profit of the worst-off vehicle (Worst-off), the number of tasks that can be reassigned to another vehicle (Ressign), and the time to obtain the solution, expressed in seconds (Time). The SCI columns

Table 3
Static FPTW benchmark results.

Name	Approach	Worst-off	UB	Overall	Time
SFPTW_75_15_0	Centr.	–	–	–	*
	Phase 1	102.98	169.89	2313.25	56.77
SFPTW_75_15_1	Centr.	174.17	–	2760.73	*
	Phase 1	162.77	203.91	2759.00	64.21
SFPTW_75_15_2	Centr.	–	–	–	*
	Phase 1	124.38	178.36	2398.53	47.34
SFPTW_75_15_3	Centr.	116.58	–	2083.51	*
	Phase 1	96.85	161.16	2089.09	72.39
SFPTW_75_15_4	Centr.	139.05	175.43	2339.88	*
	Phase 1	123.82	–	2432.46	31.08
SFPTW_75_15_5	Centr.	139.74	–	2434.89	*
	Phase 1	125.71	178.10	2375.72	51.61
SFPTW_75_15_6	Centr.	–	–	–	*
	Phase 1	106.91	183.88	2398.75	73.47
SFPTW_75_15_7	Centr.	66.24	–	2164.58	*
	Phase 1	62.00	170.01	2276.58	37.68
SFPTW_75_15_8	Centr.	137.24	–	2277.05	*
	Phase 1	124.06	169.61	2304.81	59.13
SFPTW_75_15_9	Centr.	145.35	199.14	2597.43	*
	Phase 1	145.90	–	2740.77	54.23
SFPTW_100_20_0	Phase 1	154.53	191.65	3456.50	233.09
SFPTW_100_20_1	Phase 1	158.34	218.35	3859.57	449.89
SFPTW_100_20_2	Phase 1	150.07	200.28	3503.05	609.41
SFPTW_100_20_3	Phase 1	125.13	182.87	3188.60	811.43
SFPTW_100_20_4	Phase 1	120.19	163.05	2961.78	204.06
SFPTW_100_20_5	Phase 1	80.48	206.47	*	485.43
SFPTW_100_20_6	Phase 1	123.20	179.18	3327.81	101.37
SFPTW_100_20_7	Phase 1	132.61	200.05	3443.45	209.58
SFPTW_100_20_8	Phase 1	138.43	195.29	3376.28	152.66
SFPTW_100_20_9	Phase 1	148.77	193.61	3419.45	248.20

report the same values for the solutions obtained by the Systematic Cheapest Insertion. The DIMASA Phase 2 (extra revenue) columns show that this approach increases the revenues of the optional tasks. The worst-off column reports the actual reward value that the worst vehicle would get. Finally, the last set of columns is the statistics related to the resolution of the centralized formulation (6)–(18) with Gurobi. The three columns are associated with the same statistics as in DIMASA Phase 2.

The results show that the Phase 2 approach is very effective in handling dynamic vehicle breakdown scenarios when the broken vehicle is the worst-off in terms of profit. In average the Phase 2 approach can find a solution in less than 18.55 s on average, and the profit of the worst-off vehicle is only 3.2% lower than the centralized solution in one hour. Moreover, the Phase 2 approach can reassign 54% of the optional tasks that could be reassigned, compared with 44% for the centralized solution, due to the time limit.

Analyzing the experiments for different numbers of vehicles, we can see that the average difference in profits weighted with respect to the centralized approach does not show a clear tendency as the number of vehicles increases, varying from 4.1% with 20 vehicles to 1% with 30 vehicles and 4.7% with 40 vehicles. Similarly, the differences in the percentage of optional tasks reassigned do not seem to vary significantly for different numbers of vehicles (41% with 20 vehicles, 72% with 30 vehicles and 57% with 40 vehicles). The differences are probably more due to the specific difficulties to reassign optional tasks. In terms of computation time, one cannot notice a significant difference.

Comparing the results with those obtained by the SCI heuristic, the Phase 2 algorithm achieves better results in the worst-off profit and reassignment of the remaining tasks for all the instances. An interesting observation is that in some examples where no new tasks were reassigned by CI and Phase 2, such as SFPTW_100_20_1 with broken vehicle 9, the worst-off vehicle value obtained by SCI remains the same 158.34 and those in Phase 2 manage to improve it by swapping tasks between the remaining vehicles up to 162. Thus, the model can effectively reorganize the tasks, even though optional tasks cannot be included.

The results obtained by artificially increasing the rewards of the optional tasks are very illustrative. In cases where the worst-off vehicle is the same in both the Phase 2 and Phase 2 (extra revenues) columns, this indicates that the algorithm has not been able to accommodate optional customers in the worst-off vehicle's route likely due to constraints like time windows, autonomy, or capacity. Conversely, if the worst-off value decreases, it means that the vehicle has prioritized optional tasks over its mandatory ones, leaving them among the other vehicles, potentially resulting in negative profits for extreme cases. There is only one case in the experiments where the profit of the worst-off improves (SFPTW_8 with broken vehicle 6), this is the case where the worst-off vehicle includes optional tasks without giving up the mandatory tasks. On the other hand, when we look at the average value of reassigned tasks, it is more significant. With more revenue in the optional tasks, the algorithm can reassign 62.9% of them. This also implies that the other 37.1% of optional tasks may not be reassigned due to the constraints of autonomy, time or capacity. Thus, compared with the original algorithm without extra revenue, the Phase 2 approach can reassign 54% of the optional tasks, which means that it can reassign 85.5% of the possible unassigned optional tasks (54/62.9 = 0.855).

The main advantage of the Phase 2 approach is that it can quickly adapt to the dynamic vehicle breakdown situation and generate a feasible solution in real time. This is a significant advantage over the centralized approaches that have to start the planning from scratch. The Phase 2 approach can make timely decisions and reduce the impact of unforeseen events on the overall performance of the fleet.

6. Conclusions and future work

Focusing on open challenges in agriculture cooperative fleet routing, this paper proposed a centralized and decomposed mathematical model with the related DIMASA distributed solution approach for the new DVRP-FPTW problem.

Aiming at serving tasks in their required time windows and respecting their demands and vehicle autonomy, the proposed problem requires adapting to unforeseen events dynamically while systematically optimizing the egalitarian social welfare of the fleet. Our approach

Table 4
Comparison of the results between the DIMASA Phase 2 approach and the exact centralized algorithm over the static instances SFPTW_100_20_X with 100 tasks and 20 vehicles where X is the number of instance.

Dynamic Instance								SCI			Phase 2			Phase 2 (extra revenues)			Centralized dynamic solution		
X	B. vehicle	B. time	Exp. time	u_0	Worst-off	Plan	Remain	Worst-off	Reassign	Time	Worst-off	Reassign	Time	Worst-off	Reassign	Time	Worst-off	Reassign	Time
0	vehicle_2	49.00	117.01	220	154.52	4	3	156.99	1	0.74	159.66	2	5.15	126.7	2	4.94	159.66	2	80.44
0	vehicle_12	28.18	117.80	220	154.52	5	4	154.53	1	0.8	154.53	1	26.99	24.45	2	16.96	163.27	3	*
0	vehicle_15	39.85	130.46	220	154.52	7	6	156.99	2	0.86	159.66	3	12.77	−19.95	3	12.26	160.43	2	*
1	vehicle_8	19.92	118.14	210	158.34	5	4	162.22	4	0.74	177.95	4	47.76	67.68	4	48.74	192.04	4	*
1	vehicle_9	43.48	115.64	210	158.34	7	5	158.34	0	0.86	162.17	0	7.58	162.17	0	7.09	163.67	0	79.82
1	vehicle_5	34.41	132.73	210	158.34	6	5	158.34	2	0.76	167.05	2	13.78	95.59	2	17.30	192.04	4	*
2	vehicle_17	35.18	118.93	220	150.07	4	2	150.27	0	0.8	150.27	0	11.12	88.1	1	11.21	167.55	1	109.07
2	vehicle_5	24.08	97.46	220	150.07	7	6	150.07	4	0.87	166.70	6	38.77	53.85	6	38.60	175.47	5	*
2	vehicle_15	14.04	108.16	220	150.07	7	6	150.07	1	0.75	154.82	2	68.99	−14.98	3	66.53	180.21	2	*
3	vehicle_9	49.07	127.87	210	125.13	4	2	135.75	0	0.77	137.35	0	14.04	72.58	1	3.01	146.50	1	41.18
3	vehicle_8	32.28	136.54	210	125.13	7	5	137.35	3	0.87	145.83	3	13.56	−8.93	4	13.94	148.24	3	1178.10
3	vehicle_11	27.46	111.65	210	125.13	5	4	125.13	2	0.73	147.07	3	19.56	75.39	3	21.20	148.24	3	*
4	vehicle_19	31.76	127.81	210	120.19	5	4	121.74	2	0.84	121.74	3	39.16	79.68	4	37.97	127.30	1	*
4	vehicle_12	46.33	103.82	210	120.19	7	5	120.19	2	0.71	120.19	2	12.28	120.19	4	14.58	121.74	1	38.76
4	vehicle_13	55.00	136.36	210	120.19	5	4	120.19	1	0.69	121.74	2	3.92	111.74	2	4.35	121.74	1	57.66
5	vehicle_8	59.00	124.44	210	80.48	3	2	116.67	1	0.71	116.67	1	1.74	116.67	1	1.59	116.67	1	4.43
5	vehicle_18	27.76	113.93	210	80.48	8	6	80.48	1	0.84	80.48	1	18.28	80.48	1	18.75	80.48	1	81.48
5	vehicle_17	60.34	115.25	210	80.48	4	2	80.48	2	0.82	80.48	2	1.66	62.13	2	1.51	80.48	1	4.07
6	vehicle_3	60.11	123.80	200	123.20	5	4	127.5	2	0.65	127.50	2	2.14	78.22	2	2.17	127.50	2	7.97
6	vehicle_13	24.18	129.06	200	123.20	6	4	123.2	0	0.84	123.99	1	32.74	4.4	1	29.00	142.45	1	960.04
6	vehicle_9	42.06	133.34	200	123.20	7	6	123.2	2	0.75	123.20	2	10.70	52.49	3	11.28	123.20	2	109.51
7	vehicle_1	3.61	109.39	210	132.61	3	2	143.3	0	0.81	151.09	1	49.70	128.99	1	49.78	166.94	1	*
7	vehicle_12	34.80	138.33	210	132.61	6	4	132.61	0	0.87	149.77	0	22.06	135.45	0	23.26	156.51	0	*
7	vehicle_8	61.39	129.99	210	132.61	6	3	143.3	0	0.68	143.42	0	1.60	143.42	0	1.44	143.42	0	4.98
8	vehicle_7	46.59	127.08	220	140.92	6	3	142.25	2	0.84	144.61	2	5.43	0.01	3	6.03	146.32	2	41.46
8	vehicle_15	41.83	145.40	220	140.92	8	5	140.92	0	0.75	142.04	0	12.73	140.92	1	13.21	143.55	0	66.41
8	vehicle_16	38.28	105.75	220	140.92	5	4	140.92	1	0.8	142.04	1	17.13	57.67	2	16.07	148.79	1	656.04
9	vehicle_4	45.89	104.28	220	148.77	4	3	152.25	2	0.85	154.20	2	4.71	101.47	2	4.15	154.20	1	490.71
9	vehicle_12	16.12	98.04	220	148.77	5	4	148.77	0	0.73	149.68	1	45.49	149.66	1	47.47	152.39	0	*
9	vehicle_17	64.00	111.75	220	148.77	3	2	155.59	1	0.73	158.27	1	1.14	158.27	1	1.14	159.00	1	3.74
Mean		38.5	120.34	213	133.42	5.46	3.96	136.98	1.3	0.78	141.13	1.66	18.73	81.48	2.06	18.18	147	1.56	1460

Table 5
Comparison results between the Phase 2 approach and the exact centralized algorithm over the static instances SFPTW_100_30_X with 100 tasks and 30 vehicles, where X is the number of instance.

Dynamic Instance								SCI			Phase 2			Phase 2 (extra revenues)			Centralized dynamic solution		
X	B. vehicle	B. time	Exp. time	u_0	Worst-off	Plan	Remain	Worst-off	Reassign	Time	Worst-off	Reassign	Time	Worst-off	Reassign	Time	Worst-off	Reassign	Time
0	vehicle_25	39.00	77.28	220	71.44	2	1	73.65	1	1.11	76.24	1	20.11	18.39	1	25.02	78.06	1	*
0	vehicle_21	17.46	136.32	220	71.44	7	6	73.65	3	1.14	80.47	5	51.31	−86.17	6	58.97	83.17	3	*
0	vehicle_26	12.21	32.17	220	71.44	3	2	71.44	2	1.2	83.99	2	71.08	6.91	2	58.40	71.44	0	*
1	vehicle_23	40.20	120.06	210	80.94	4	3	83.91	1	1.06	89.70	1	13.30	−11.16	1	12.24	89.70	1	1554.43
1	vehicle_17	45.61	98.97	210	80.94	4	2	80.94	2	1.06	88.31	2	4.80	88.31	2	4.80	89.70	2	704.76
1	vehicle_29	61.72	133.81	210	80.94	3	2	80.94	1	0.99	80.94	1	1.29	76.53	1	1.28	80.94	1	2.34
2	vehicle_25	16.03	44.43	220	79.56	2	1	80.72	1	1.16	81.76	1	62.27	31.63	1	63.70	86.89	1	*
2	vehicle_17	48.47	148.80	220	79.56	5	3	80.85	1	0.77	81.65	2	4.94	51.86	3	4.67	81.65	1	305.75
2	vehicle_7	52.25	106.00	220	79.56	4	2	80.85	2	0.86	85.94	2	3.15	−28.56	2	3.22	85.94	2	50.82
3	vehicle_15	26.57	101.54	200	70.43	3	2	74.21	1	1.18	75.37	2	30.69	−31.48	2	32.04	75.63	2	*
3	vehicle_30	37.06	123.41	200	70.43	8	5	70.43	2	1.0	74.94	3	8.34	31.74	3	8.55	74.94	2	210.32
3	vehicle_27	60.81	129.94	200	70.43	3	2	70.43	1	0.61	74.94	1	0.93	74.94	1	0.92	74.94	1	2.08
4	vehicle_10	50.00	119.14	200	59.57	3	2	60.16	2	1.0	64.07	2	4.86	−5.42	2	4.98	67.85	2	736.85
4	vehicle_26	52.54	108.03	200	59.57	4	2	59.57	2	1.03	59.57	2	3.30	59.57	2	3.37	60.16	0	36.22
4	vehicle_4	31.14	76.21	200	59.57	4	3	59.57	1	1.13	59.57	1	27.21	59.57	2	23.98	0.00	0	*
5	vehicle_20	52.40	137.43	210	65.57	3	2	87.47	0	0.98	87.54	0	2.72	87.54	0	2.74	87.54	0	33.97
5	vehicle_2	55.00	114.70	210	65.57	4	2	65.57	2	0.76	65.57	2	1.73	−30.72	2	1.73	65.57	2	4.99
5	vehicle_21	30.00	75.07	210	65.57	3	2	65.57	2	1.15	65.57	2	37.21	40.24	2	19.53	65.57	2	767.18
6	vehicle_3	51.24	112.99	200	62.01	5	3	66.46	1	0.93	66.46	1	2.45	38.5	2	2.34	66.46	0	7.98
6	vehicle_17	52.55	128.67	200	62.01	5	4	66.46	2	0.88	62.01	2	2.01	−14.46	3	2.01	66.46	1	6.30
6	vehicle_13	22.47	122.23	200	62.01	3	2	62.01	2	1.2	70.90	2	31.76	−14.18	2	30.70	74.51	2	*
7	vehicle_9	60.99	127.08	210	79.92	5	3	82.41	1	0.78	82.41	3	1.3	42.23	3	1.23	82.41	2	3.01
7	vehicle_23	45.18	98.99	210	79.92	4	2	79.92	2	1.12	79.92	2	6.19	22.33	2	5.98	82.57	0	*
7	vehicle_15	17.20	40.85	210	79.92	2	1	79.92	1	1.18	80.26	1	58.59	−43.81	1	59.91	0.00	0	*
8	vehicle_6	33.24	97.44	220	64.56	2	1	64.82	0	1.12	70.23	1	23.35	70.36	1	26.83	75.56	1	*
8	vehicle_23	41.05	102.29	220	64.56	5	4	64.82	2	1.12	71.61	2	16.62	−53.64	2	16.91	72.58	2	*
8	vehicle_21	47.43	102.98	220	64.56	2	1	64.56	1	1.05	64.56	1	5.58	45.53	1	5.27	67.59	1	595.65
9	vehicle_4	57.14	132.74	210	70.26	4	3	71.3	1	0.91	71.30	1	1.56	9.34	1	1.85	71.30	1	4.40
9	vehicle_7	20.62	124.38	210	70.26	3	2	70.26	2	1.12	73.59	2	47.75	−36.37	2	40.40	74.77	2	*
9	vehicle_2	37.07	87.63	210	70.26	4	2	70.26	1	1.18	70.26	2	15.69	70.26	2	15.81	70.26	1	311.25
Mean		40.48	105.38	210	70.42	3.76	2.4	72.10	blue1.46	1.02	74.65	1.73	18.74	18.99	1.90	17.98	75.45	1.23	1625.3

Table 6
Comparison results between the Phase 2 approach and the exact centralized algorithm over the static instances SFPTW_100_40_X with 100 tasks and 40 vehicles, where X is the number of instance.

Dynamic Instance								SCI			Phase 2			Phase 2 (extra revenues)			Centralized dynamic solution		
X	B. vehicle	B. time	Exp. time	u_0	Worst-off	Plan	Remain	Worst-off	Reassign	Time	Worst-off	Reassign	Time	Worst-off	Reassign	Time	Worst-off	Reassign	Time
0	vehicle_16	42.76	109.52	220	39.48	2	1	42.29	0	1.33	42.29	0	9.33	42.29	0	9.52	42.29	0	*
0	vehicle_30	39.85	111.92	220	39.48	5	4	39.48	1	1.26	41.72	1	11.68	−5.39	2	10.61	37.65	0	*
0	vehicle_24	3.00	37.72	220	39.48	2	1	39.48	0	1.57	43.58	0	64.51	43.58	0	65.07	41.72	0	*
1	vehicle_27	85.00	99.42	210	46.16	1	0	53.35	0	0.33	53.35	0	0.22	53.35	0	0.22	53.35	0	0.17
1	vehicle_1	47.16	93.59	210	46.16	4	2	46.16	2	1.11	49.89	2	5.94	30.26	2	6.02	49.89	2	501.70
1	vehicle_37	61.00	89.10	210	46.16	2	1	47.92	1	0.92	47.92	1	1.13	47.92	1	1.14	47.92	1	2.55
2	vehicle_35	21.02	42.05	220	40.95	1	0	42.17	0	1.51	43.00	0	43.80	43.0	0	44.07	40.80	0	*
2	vehicle_10	8.25	51.40	220	40.95	3	2	42.17	2	1.56	43.00	2	77.38	−29.19	2	70.62	0.00	0	*
2	vehicle_5	65.00	145.56	220	40.95	3	2	42.17	0	0.66	42.17	0	0.70	−9.0	1	0.70	42.17	0	1.43
3	vehicle_1	31.78	91.69	200	37.04	2	1	37.8	1	1.43	37.80	1	16.61	26.11	1	15.98	0.00	0	*
3	vehicle_7	46.44	125.68	200	37.04	5	3	38.29	1	1.17	38.29	1	5.41	−51.34	1	5.56	38.71	1	448.38
3	vehicle_9	32.70	102.01	200	37.04	3	2	37.04	1	1.34	38.29	1	15.59	4.31	1	13.47	38.71	1	*
4	vehicle_11	17.72	36.11	200	22.88	2	1	23.99	0	1.52	27.56	0	27.91	−8.97	1	34.90	0.00	0	*
4	vehicle_5	55.00	59.47	200	22.88	1	0	23.99	0	1.0	23.99	0	1.56	23.99	0	1.54	23.99	0	5.00
4	vehicle_2	52.00	115.53	200	22.88	3	2	24.75	2	0.92	24.75	2	2.55	24.75	2	2.34	24.75	2	52.67
5	vehicle_40	33.62	86.26	210	46.13	2	1	49.65	1	1.36	46.70	1	15.34	42.45	1	14.61	49.74	1	*
5	vehicle_5	39.20	129.69	210	46.13	3	2	46.13	1	1.28	46.13	2	7.25	46.13	2	7.05	48.83	2	1946.09
5	vehicle_25	17.46	105.70	210	46.13	3	2	46.7	2	1.48	47.15	2	42.28	−6.52	2	42.15	51.00	2	2758.92
6	vehicle_36	38.47	106.35	190	30.64	3	2	31.07	0	1.3	31.08	1	10.50	−49.6	1	9.83	40.03	1	*
6	vehicle_5	61.31	125.50	190	30.64	5	3	31.08	1	0.76	31.08	1	0.67	31.08	2	0.67	31.08	1	1.07
6	vehicle_35	26.17	88.02	190	30.64	2	1	30.65	1	1.39	31.08	1	19.09	27.27	1	26.77	40.03	1	*
7	vehicle_9	57.01	136.91	210	36.09	3	2	38.59	0	0.84	38.59	0	1.05	38.59	0	1.04	42.47	0	3.04
7	vehicle_29	55.07	120.69	210	36.09	3	2	36.09	1	0.88	36.09	1	1.37	36.09	1	1.35	36.09	1	3.91
7	vehicle_36	43.38	99.69	210	36.09	2	1	36.09	1	1.16	36.09	1	7.38	30.74	1	6.92	36.09	1	1319.40
8	vehicle_13	22.83	45.65	220	35.35	1	0	36.85	0	1.53	38.61	0	27.40	38.61	0	27.50	0.00	0	*
8	vehicle_5	37.58	101.00	220	35.35	4	3	37.34	1	1.15	37.34	1	10.44	−13.31	1	10.30	45.07	2	357.91
8	vehicle_28	10.44	66.24	220	35.35	2	1	35.35	1	1.6	37.34	1	42.04	36.85	1	43.84	0.00	0	*
9	vehicle_37	29.21	109.70	200	39.48	2	1	41.02	1	1.33	41.02	1	18.98	5.46	1	19.74	44.96	1	*
9	vehicle_29	27.02	85.91	200	39.48	3	2	39.49	2	1.43	41.02	2	29.64	−8.85	2	26.59	43.61	2	*
9	vehicle_20	27.00	40.00	200	39.48	1	0	39.49	0	1.35	41.02	0	28.07	41.02	0	27.80	43.61	0	*
Mean		37.81	91.93	208	37.42	2.6	1.5	38.55	0.8	1.21	39.48	0.86	18.19	19.66	0.99	18.26	40.34	0.73	2177.71

focuses on iterative maximization of the profit of the least profitable vehicle in a non-decreasing order of vehicle profitability across the fleet.

The proposed DVRP-FPTW problem and the DIMASA solution approach are particularly relevant to real-world agriculture cooperative fleets composed of tractors and agrirobots, focusing on both fairness and efficiency among individually rational vehicle owners with limited funds, each one owning a single vehicle and farmers owning multiple crop fields. In this context, each vehicle disposes of private information that it should not share with others and the dynamic route generation time is relatively short.

While preserving the vehicles' privacy, the DIMASA architecture, composed of vehicle agents and a coordinator agent that works in two phases, efficiently and effectively solves the static and dynamic VRP-FPTW, as demonstrated through the performed simulation experiments. The solution is achieved through distributed decision-making for column generation between vehicle agents and the coordinator agent that solves the restricted master problem by using the SYSCA algorithm in (static) phase 1 and the DSYSCA algorithm in (dynamic) phase 2. Each vehicle agent, on the other hand, uses the proposed DRGV algorithm to iteratively and dynamically find its good paths (columns) in response to the updated shadow prices of the coordinator. Our dynamic solution approach extends the static VRP-FPTW to handle real-time contingencies, such as changes in tasks or vehicle breakdowns.

We tested our approach in simulation experiments on a newly proposed set of benchmark instances that we created for the static and dynamic VRP-FPTW problem and showed that the DIMASA architecture solutions are generally of high quality and are more flexible, robust, fast, and fair than the solutions obtained through the proposed centralized model solved through Gurobi solver. Moreover, the distributed DIMASA approach is ideal for cooperative vehicle-sharing scenarios, protecting intrinsically decentralized and private vehicle information while avoiding reliance on a unique decision maker. However, the solution quality is heavily dependent on the quality of communication between the coordinator and vehicle agents, performed by synchronous and asynchronous message exchange. This can be implemented through message queues in each agent, with the additional advantage that the system will become more resistant to communication channel breakdowns or other contingencies in message exchange (e.g., message loss or out-of-order delivery).

We showed through simulation that the proposed DIMASA approach is computationally efficient, scalable, and suited for large cooperative fleets, responding in close-to real time to unpredicted contingencies on the terrain.

Future work. In future work, we will consider real-world agriculture cooperatives and their data, where we will analyze the influence of the parameters and variables on the solution's efficiency and the computation time.

The requirements for communication and addressing technical issues in the real world are beyond the scope of this paper and will be addressed in future work. Furthermore, we plan to create models for systematic optimization of the egalitarian social welfare considering ownership of multiple vehicles by a single owner or multiple owners for a single vehicle.

A central coordinator agent is a bottleneck for the scalability and robustness of the proposed solution but guarantees the quality of the solution. A multi-agent architecture where vehicle agents coordinate in a completely decentralized manner while maintaining quality of solution guarantees remains an open issue.

We also consider to extend our model and architecture to accommodate open peer-to-peer fleets, allowing vehicles to dynamically join or leave the system as needed. In this context, it is reasonable to assume that strategic vehicle owners may deceive information shared regarding their vehicle agents to maximize their profits. Considering the potentially extensive size of these peer-to-peer fleets, we recognize the importance of studying scalable, strategy-proof vehicle routing algorithms within this framework.

Finally, our forthcoming plans involve refining the proposed DVRP-FPTW model and the DIMASA architecture by integrating vehicle implements (tools). This expansion aims to address the need to integrate multi-functional reconfigurable agrirobots that effortlessly switch between detachable implements like plows, harvesters, and more during operations. Together with the proposed DIMASA architecture, this innovative solution will mark a pivotal advancement, and significantly increase the overall efficiency and adaptability of the agricultural autonomous vehicle fleet routing.

CRediT authorship contribution statement

Aitor López Sánchez: Methodology, Data curation, Software, Investigation, Formal analysis, Visualization, Writing – original draft. **Marin Lujak:** Project administration, Resources, Conceptualization, Methodology, Investigation, Formal analysis, Funding acquisition, Supervision, Validation, Writing – review & editing. **Frédéric Semet:** Methodology, Supervision, Validation, Writing – review & editing. **Holger Billhardt:** Methodology, Supervision, Validation, Writing – review & editing.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be made available on request.

Acknowledgments

This work was supported by the VAE grant: TED2021-131295B-C33 funded by MCIN/AEI/10.13039/501100011033 and by the European Union's "NextGenerationEU/PRTR", by grant COSASS: PID2021-123673OB-C32 funded by MCIN/ AEI/ 10.13039/501100011033 and by "ERDF A way of making Europe", by the AGROBOTS Project funded by the Community of Madrid, Spain and by the AGROBOTIX Project funded by the University Rey Juan Carlos.

References

Agra, A., Christiansen, M., Figueiredo, R., Hvattum, L. M., Poss, M., & Requejo, C. (2013). The robust vehicle routing problem with time windows. *Computers & Operations Research*, *40*(3), 856–866.

Aksen, D., & Aras, N. (2006). Customer selection and profit maximization in vehicle routing problems. In *Operations research proceedings 2005* (pp. 37–42). Springer.

Alipour, M. M., Emami, H., & Abdolhosseinzadeh, M. (2022). A MAS approach for vehicle routing problem. *Neural Computing and Applications*, *34*, 4387–4411.

Archetti, C., Speranza, M. G., & Vigo, D. (2014). Chapter 10: Vehicle routing problems with profits. In *Vehicle routing: Problems, methods, and applications, second edition* (pp. 273–297). SIAM.

Barbucha, D. (2016). An improved agent-based approach to the dynamic vehicle routing problem. In *Intelligent decision technologies 2016: Proceedings of the 8th KES international conference on intelligent decision technologies (KES-IDT 2016)–Part I* (pp. 361–370). Springer.

Barbucha, D. (2019). VNS-based multi-agent approach to the dynamic vehicle routing problem. In *Computational collective intelligence: 11th international conference* (pp. 556–565). Springer.

Basso, S., & Ceselli, A. (2017). Asynchronous column generation. In *2017 proceedings of the ninteenth workshop on algorithm engineering and experiments* (pp. 197–206). SIAM.

Basso, S., & Ceselli, A. (2022). Distributed asynchronous column generation. *Computers & Operations Research*, *146*(105894), 1–16.

Bennett, K. P., Demiriz, A., & Shawe-Taylor, J. (2000). A column generation algorithm for boosting. In *Proceedings of the seventeenth international conference on machine learning iCML'00* (pp. 65–72).

Bertsimas, D., Brown, D. B., & Caramanis, C. (2011). Theory and applications of robust optimization. *SIAM Review*, *53*(3), 464–501.

Bono, G., Dibangoye, J. S., Simonin, O., Matignon, L., & Pereyron, F. (2020). Solving multi-agent routing problems using deep attention mechanisms. *IEEE Transactions on Intelligent Transportation Systems*, *22*(12), 7804–7813.

Dror, M. (1994). Note on the complexity of the shortest path models for column generation in VRPTW. *Operations Research*, *42*(5), 977–978.

Duan, J., He, Z., & Yen, G. G. (2021). Robust multiobjective optimization for vehicle routing problem with time windows. *IEEE Transactions on Cybernetics*, *52*(8), 8300–8314.

Feillet, D., Dejax, P., Gendreau, M., & Gueguen, C. (2004). An exact algorithm for the elementary shortest path problem with resource constraints: Application to some vehicle routing problems. *Networks: An International Journal*, *44*(3), 216–229.

Fonseca-Galindo, J. C., de Castro Surita, G., Neto, J. M., de Castro, C. L., & Lemos, A. P. (2022). A multi-agent system for solving the dynamic capacitated vehicle routing problem with stochastic customers using trajectory data mining. *Expert Systems with Applications*, *195*(116602), 1–14.

Gansterer, M., & Hartl, R. F. (2018). Collaborative vehicle routing: a survey. *European Journal of Operational Research*, *268*(1), 1–12.

Gansterer, M., & Hartl, R. F. (2020). Shared resources in collaborative vehicle routing. *TOP*, *28*, 1–20.

Gendreau, M., Laporte, G., & Semet, F. (1997). The covering tour problem. *Operations Research*, *45*(4), 568–576.

Guajardo, M., & Rönnqvist, M. (2016). A review on cost allocation methods in collaborative transportation. *International Transactions in Operational Research*, *23*(3), 371–392.

Guo, Y., Cheng, J., & Ji, J. (2016). Robust dynamic vehicle routing optimization with time windows. In *Advances in swarm intelligence: 7th international conference* (pp. 28–36). Springer.

Irnich, S., & Villeneuve, D. (2006). The shortest-path problem with resource constraints and k-cycle elimination for k ≥ 3. *INFORMS Journal on Computing*, *18*(3), 391–406.

Lee, D., & Ahn, J. (2017). Vehicle routing problem with vector profits (VRPVP) with max-min criterion. (pp. 1–33). arXiv preprint arXiv:1710.10550.

Li, J.-Q., Mirchandani, P. B., & Borenstein, D. (2009). Real-time vehicle rerouting problems with time windows. *European Journal of Operational Research*, *194*(3), 711–727.

Liu, S., & Papageorgiou, L. G. (2018). Fair profit distribution in multi-echelon supply chains via transfer prices. *Omega*, *80*, 77–94.

López-Sánchez, A., Lujak, M., Billhardt, H., & Semet, F. (2023a). Vehicle routing problem with fair profits and time windows (VRP-FPTW). In *2023 IEEE international conference on systems, man, and cybernetics* (pp. 1530–1536). IEEE.

López-Sánchez, A., Lujak, M., Semet, F., & Billhardt, H. (2022). On balancing fairness and efficiency in routing of cooperative vehicle fleets. In *CEUR proceedings, 12th int. workshop on ATT 2022 co-located with IJCAI-ECAI 2022, vol. 3173* (pp. 1–14).

López-Sánchez, A., Lujak, M., Semet, F., & Billhardt, H. (2023b). How to achieve fair and efficient cooperative vehicle routing? *AI Communications*, *Pre-press*(Pre-press), 1–23.

Los, J., Schulte, F., Gansterer, M., Hartl, R. F., Spaan, M. T., & Negenborn, R. R. (2022). Large-scale collaborative vehicle routing. *Annals of Operations Research*, 1–33.

Los, J., Schulte, F., Spaan, M. T., & Negenborn, R. R. (2022). An auction-based multi-agent system for the pickup and delivery problem with autonomous vehicles and alternative locations. In *Dynamics in logistics: proceedings of the 8th international conference LDIC 2022* (pp. 244–260). Springer.

Lübbecke, M. E., & Desrosiers, J. (2005). Selected topics in column generation. *Operations Research*, *53*(6), 1007–1023.

Lujak, M., Giordani, S., Omicini, A., & Ossowski, S. (2020). Decentralizing coordination in open vehicle fleets for scalable and dynamic task allocation. *Complexity*, *2020*, 1–21.

Lujak, M., Sklar, E., & Semet, F. (2021). Agriculture fleet vehicle routing: A decentralised and dynamic problem. *AI Communications*, *34*(1), 55–71.

Mancini, S., Gansterer, M., & Hartl, R. F. (2021). The collaborative consistent vehicle routing problem with workload balance. *European Journal of Operational Research*, *293*(3), 955–965.

Monroy-Licht, M., Amaya, C. A., Langevin, A., & Rousseau, L.-M. (2017). The rescheduling arc routing problem. *International Transactions in Operational Research*, *24*(6), 1325–1346.

Pillac, V., Gendreau, M., Guéret, C., & Medaglia, A. L. (2013). A review of dynamic vehicle routing problems. *European Journal of Operational Research*, *225*(1), 1–11.

Pinto, T., Alves, C., Mansi, R., & Valério de Carvalho, J. (2015). Solving the multiscenario max-min knapsack problem exactly with column generation and branch-and-bound. *Mathematical Problems in Engineering*, *2015*, 1–11.

Psaraftis, H. N., Wen, M., & Kontovas, C. A. (2016). Dynamic vehicle routing problems: Three decades and counting. *Networks*, *67*(1), 3–31.

Qi, M., Xiong, W., Zhou, Q., & Hua, S. (2018). Robust periodic vehicle routing problem with service time uncertainty. In *2018 IEEE international conference on industrial engineering and engineering management* (pp. 1431–1435). IEEE.

Rodríguez-Martín, I., Salazar-González, J.-J., & Yaman, H. (2019). The periodic vehicle routing problem with driver consistency. *European Journal of Operational Research*, *273*(2), 575–584.

Seyyedhasani, H., & Dvorak, J. S. (2018). Dynamic rerouting of a fleet of vehicles in agricultural operations through a dynamic multiple depot vehicle routing problem representation. *Biosystems Engineering*, *171*, 63–77.

Solomon, M. M. (1987). Algorithms for the vehicle routing and scheduling problems with time window constraints. *Operations Research*, *35*, 254–265.

Soriano, A., Gansterer, M., & Hartl, R. F. (2023). The multi-depot vehicle routing problem with profit fairness. *International Journal of Production Economics*, *255*(108669), 1–14.

Stavropoulou, F., Repoussis, P. P., & Tarantilis, C. D. (2019). The vehicle routing problem with profits and consistency constraints. *European Journal of Operational Research*, *274*(1), 340–356.

Steever, Z., Karwan, M., & Murray, C. (2019). Dynamic courier routing for a food delivery service. *Computers & Operations Research*, *107*, 173–188.

Uchoa, E., Pecin, D., Pessoa, A., Poggi, M., Vidal, T., & Subramanian, A. (2017). New benchmark instances for the capacitated vehicle routing problem. *European Journal of Operational Research*, *257*(3), 845–858.

Ulmer, M. W., Goodson, J. C., Mattfeld, D. C., & Hennig, M. (2019). Offline–online approximate dynamic programming for dynamic vehicle routing with stochastic requests. *Transportation Science*, *53*(1), 185–202.

Ulmer, M. W., Heilig, L., & Voß, S. (2017). On the value and challenge of real-time information in dynamic dispatching of service vehicles. *Business & Information Systems Engineering*, *59*, 161–171.

Ulmer, M. W., & Streng, S. (2019). Same-day delivery with pickup stations and autonomous vehicles. *Computers & Operations Research*, *108*, 1–19.