ORNInA: A Decentralized, Auction-based Multi-agent Coordination in ODT Systems

Alaa Daoud, Flavien Balbo, Paolo Gianessi, Gauthier Picard

▶ **To cite this version:**

HAL Id: hal-03037353
https://hal.science/hal-03037353

Submitted on 23 Feb 2021

ORNInA: A Decentralized, Auction-based Multi-agent Coordination in ODT Systems

Online Rescheduling with Neighborhood exchange, based on Insertion heuristic and Auctions

Alaa Daoud [a,*] Flavien Balbo [a] Paolo Gianessi [b] and Gauthier Picard [c]

[a] *Laboratoire Hubert Curien UMR CNRS 5516, Institut Henri Fayol, Mines Saint Étienne, Saint Étienne, France*
E-mails: alaa.daoud@emse.fr, flavien.balbo@emse.fr

[b] *LIMOS UMR CNRS 6158, Institut Henri Fayol, Mines Saint Étienne, Saint Étienne, France*
E-mail: paolo.gianessi@emse.fr

[c] *ONERA/DTIS, Université de Toulouse, 2 Avenue Edouard Belin, 31055, Toulouse Cedex 4, France*
E-mail: gauthier.picard@onera.fr

Abstract. On-Demand Transport (ODT) systems have attracted increasing attention in recent years. Traditional centralized dispatching can achieve optimal solutions, but NP-Hard complexity makes it unsuitable for online and dynamic problems. Centralized and decentralized heuristics can achieve fast, feasible solution at run-time with no guarantee on the quality. Starting from a feasible not optimal solution, we present in this paper a new solution model (*ORNInA*) consisting of two parallel coordination processes. The first one is a decentralized insertion-heuristic based algorithm to build vehicle schedules in order to solve a particular case of the dynamic Dial-A-Ride-Problem (DARP) as an ODT system, in which vehicles communicate via Vehicle-to-vehicle communication (V2V) and make decentralized decisions. The second coordination scheme is a continuous optimization process namely *Pull-demand* protocol, based on combinatorial auctions, in order to improve the quality of the global solution achieved by decentralized decision at run-time by exchanging resources between vehicles (k-opt). In its simplest implementation, k is set to 1 so that vehicles can exchange only one resource at a time. We evaluate and analyze the promising results of our contributed techniques on synthetic data for taxis operating in Saint-Étienne city, against a classical decentralized greedy approach and a centralized one that uses a classical mixed-integer linear program (MILP) solver.

Keywords: On-demand transport, Coordination, Decentralized optimization, Combinatorial auctions

Introduction

On-Demand Transport (ODT) systems have attracted increasing attention in recent years. The area of application was almost limited to road systems. The ODT system has never been considered for the replacement of public transport services but to extend it. According to [1], the concept of ODT was formulated for the first time in the United States around 1990 as a solution to the growing disaffection of potential users, especially at night.

Within ODT, allocation problems rise as the most studied optimization problems in the literature [2]. They generally can be addressed by linear programming since its relaxation of formulation admits optimal integral solutions. However, in reality, if a cen-

tral dispatcher exists, performing such optimal decisions, it requires vehicles to have continuous access to their portal via the global communications infrastructure such as the Long-Term Evolution (LTE) communication (3G, 4G or 5G), whose data traffic is expensive for the required usage density and the persistent operation, and can cause a critical bottleneck on the portal side.

The computational complexity of ODT allocation problems, which extend the NP-Hard traveling salesman problem (TSP), makes it difficult for the central dispatcher to manage the dynamics of the problem during execution (online requests, variable fleet size, traffic problems, and other environment dynamics). Therefore one may expect that decentralization can cop off with these problems. The insertion heuristics proposed by [3] is a popular method for solving a variety of scheduling and routing problems. It can be used as

---

*Corresponding author. E-mail: alaa.daoud@emse.fr.

a method to quickly find a feasible solution (with no guarantee on solution quality). For instance, in Vehicle Routing Problem (VRP), it creates the solution by repeatedly inserting unscheduled demands in a partially constructed route or as the first demand in a new route. Local search with the k-exchange neighborhoods (k-opt) is one of the most commonly used heuristic methods for problems inherited from TSP. k-opt is a path improvement algorithm, where at each planning phase, k steps from the current plan are replaced by k steps to get a cheaper path [4].

In what follows, we will describe the main characteristics of ODT problem in (Section 1) in order to differentiate it from other modes. In this work, we propose the Autonomous Vehicle Fleet Allocation Problem (AVFAP), which extends the traditional Dial-A-Ride-Problem (DARP) with the usage of a fleet of autonomous vehicles, replacing the central dispatcher with the coordination between each other in a Peer-to-Peer (P2P) manner in order to make decentralized decisions. We more precisely propose a new decentralized heuristic for the AVFAP, named *ORNInA*, benefiting from the fast responsiveness of insertion heuristics and the good results gained by 1-opt optimization.

The paper is organized as follows. In Section 2, we overview the related transport problems in literature, and more precisely, we focus on efforts done to address related ODT problems. Section 3 expounds on the model we consider for the decentralized resource allocation problem for autonomous vehicle fleets with an illustrative scenario showing the main components of the ODT system. Then, the contribution for fast decision is presented in Section 4 and an optimization protocol is detailed in Section 5. Experimental parameters, evaluation, and results are detailed in Section 6. Finally, we conclude the paper in Section 7 with some perspectives.

1. Problem Definition

The concept of On-Demand Transport (ODT) was formulated for the first time around 1990 as a solution to the growing disaffection of potential users of public transport, especially at night [1]. There exist similar terms in the literature describing several transportation systems that could be similar in some characteristics and differ in others; e.g., the term Demand-Responsive-Transport (DRT) refers to the same concept. Moreover, in recent years, the term Dial-A-Ride Problem (DARP) is becoming more common in real-

Fig. 1. A sample AVFAP problem instance, with demand sources (triangles) and taxis (circles) with their respective communication ranges (in blue).

life scenarios, such as the services provided by Uber, Lyft, and other on-demand transport platforms.

For [5], *"On-Demand Transport is a transit mode consisting of passenger vehicles, vans, or small buses that respond to the calls of passengers or their agents to the transit operator, who then sends a vehicle to collect passengers and transport them to their destinations."* Additionally, Demand-Response (DR) operation in a transport system was characterized in [6] as follows:

- Vehicles do not operate on a fixed course or a fixed schedule, except, perhaps, temporarily to satisfy a special need,
- A vehicle can be sent to take several passengers to different collection points before taking them to their respective destinations and can also be stopped on the way to these destinations to catch other passengers,
- Passenger could be served by multi-vehicle trajectory,
- Requests usually present intentions of a trip from source to destination within a certain time window.

Finally, the main features of ODT optimization are defined by [1] to be: firstly, as door-to-door as possible transportation; secondly, minimize waiting times, walking paths, and vehicle changes.

In this context, we propose the Autonomous Vehicle Fleet Allocation Problem (AVFAP), illustrated in Figure 1, as follows:

Definition 1. *AVFAP is an extension of the traditional DARP with the usage of a fleet of autonomous vehicles, replacing the central dispatcher with the coordination between each other in Peer to Peer (P2P) manner in order to make decentralized decisions.*

Since the early 21st century, short- and long-range vehicle communication technologies, including equipment, applications, and systems, have been developed and introduced in the transportation domain to enable the vehicle to everything (V2X) communication. The concept of V2X includes communication between vehicles (V2V), between vehicles and pedestrians (V2P), and between vehicles and infrastructure (V2I). Short-range vehicle communication protocols have been based on the WLAN IEEE 802.11 [7] standards known in the United States as Wireless Access in Vehicle Environments (WAVE) and in Europe as ITS-G5 [8]. Also known as Dedicated Short-Range Communication (DSRC).

V2V communication via DSRC provides low latency, fast network connectivity with a range of communication up to 300 meters.

DSRC has been widely tested in Europe, the United States, and other regions with different road safety and traffic efficiency use-cases. More recently, the direct cellular (C-V2V) using the 3GPP PC5 interface has been introduced as an alternative to IEEE 802.11p for V2V Short Range [9]. In this paper, neither the cases of multi-vehicle trajectories nor the passenger walking parts of the trip are considered, as we define a satisfied request as a request that is picked up from its specified pick-up location during its defined time window and dropped of exactly in its defined drop-off location [10]. Having limited communication ranges, each vehicle is only aware of a subset of demands and communicates with entities (vehicles or demand sources) if they are co-located in each others' range.

2. On-Demand Transport and Related Problems

The problem of resource allocation and task assignment across multiple entities is a central concern in both Computer Science and Economics. It has interdisciplinary characteristics that make it relevant for different application areas, including industrial production and planning, routing, traffic management, transport, and logistics. In our scenario, under a set of constraints, assigning demands to vehicles can be seen as an intersection between resource allocation and constraint optimization problems. The classical modeling approach for such problems is Linear Programming (LP), and more specifically, MILP (Mixed Integer LP).

The problem described by the illustrative scenario (in Section 1) extends the Dial-A-Ride Problem (DARP), which belongs to the family of Pick-up and Delivery

problems and can itself be considered as a special variant of the Vehicle Routing Problem with Pick-up and Delivery (VRPPD) with time windows. The problem, which we briefly recall in the following, can be seen as both a scheduling and an allocation problem.

2.1. Deal A Ride Problems

In [11], the Generalized Pick-up and Delivery Problem (GPDP) is defined as the general framework for all Pick-up and Delivery problems. In it, n requests must be served, each associated with a set of origins and destinations to be visited, respectively, to pick-up or deliver goods or people. A fleet of m heterogeneous vehicles, each with its own capacity, is used to serve requests. A request is unsplittable (must be served entirely by one same vehicle) and all origins must be visited before any destination, without transshipment in intermediate locations. Each vehicle has specific start and end locations and can serve more than one request, provided that all along its service route its load is non-negative and within its capacity. The well-known Vehicle Routing Problem (VRP), which is known to be NP-hard [12], is the particular case of the GPDP with the same capacity for all the vehicles and only one origin or destination per request, the depot, which also acts as the start/end point of all routes. Another particular case of GPDP, which also generalizes VRP, is what in [13] is named the Vehicle Routing Problem with Pick-up and Delivery (VRPPD), in which each request has one origin and one destination and the vehicles are heterogeneous but all based at a depot. The authors also call it the n-commodities PDP to distinguish it from the single-commodity PDP, in which one type of product is collected or delivered to each node, and the two-commodity PDP that implies two products and every node can act as a pick-up point for one and delivery point for the other: the delivery of beverages, in which vehicles deliver full bottles and collect empty ones, is cited as a common real-life case.

In the same work, the VRPPD with time windows (VRPPDTW) is tackled, motivated by the fact that most practical VRPPD applications enforce restrictions on time slots in which a vehicle can visit a site. VRPPDTW is defined on a directed graph $G = (N, A)$ with $2n + 2$, i.e. as many nodes as twice the request plus the two nodes needed to be able to represent outgoing and incoming trips of the depot at which vehicles are based. Vehicles are heterogeneous and must complete their routes within a given time horizon, and service at a node must begin within a time window as-

sociated with it. VRPPDTW is NP-hard as it generalizes the TSP with Time Windows and Precedence Constraints (TSP-TWPC), which is known to be NP-hard (see e.g. [14]). The 3-index Mixed-Integer Linear Programming (MILP) formulation proposed in [13] allows to consider a variant in which vehicles have different start- and endpoint, and which is not easier than the basic VRPPDTW version.

As stated by the authors of [13], DARP is actually a version of the basic VRPPD in which the objective mostly revolves around Quality of Service (QoS), i.e. minimizing user inconvenience, which plays a central role [6] (though more specific DARPs allow for one passanger only and thus also restrict the capacity of vehicles to 1). In addition, vehicle capacity is normally limited in DARP, while it is often redundant in VRPPD applications (particularly those related to the collection and delivery of letters and small parcels [15]). The work [13] then reviews the different cases of DARP in the literature, like e.g. single- and multi-vehicle cases, or the different functions to model passenger inconvenience (possibly giving rise to quadratic or convex functions) and how its minimization is dealt with. Quality of Service can replace, or being evaluated along with, some cost related terms, as well as DARP-specific constraints, as e.g. maximum waiting time or maximum excess in user ride times.

In [16], the notion of time window is integrated in the basic definition of DARP, as we will consider in the following of the present work. The authors of [16] review some of the most common real-world aspect that are dealt with in literature, especially for what concerns the diversity of vehicles, which can differ not only in capacity but also for what concerns e.g. speed, travel cost, equipment, number of person that can be transported, transportation mode depending on the passengers, possibility of accompanying persons. As pointed out in [15], the main consideration in some of such a diversity of problems is to tackle the strategic problem of determining a size and composition of the fleet that will satisfy the entire demand, while in others the goal is to maximize the number of applications that can be offered with a fleet of fixed dimensions. A compromise is to serve part of the demand with a basic fleet of vehicles and to use additional vehicles (e.g. regular taxis) if necessary. The authors of [16] also propose a wide review of exact methods for static DARP variants, as well as and heuristic approaches for static, dynamic and stochastic variants that are found in the literature.

Along with static Dial-a-ride services, in which all requests for transport are known in advance, there also exist dynamic services, in which requests are progressively revealed throughout the day, or existing requests ar cancelled, and vehicle routes are adjusted in real time accordingly, trying to do so without causing too much trouble for other passengers. In the literature, dynamic variants of DARP can be found, although in practice pure dynamic DARPs rarely exist because a subset of requests is often known in advance, according to [15], who reports some examples of dynamic DARP. [17] developed an insertion algorithm, RE-BUS, based on the ADARTW procedure of [18] for a real-life problem involving services to elderly and disabled people in Copenhagen. Requests arrive dynamically along a time horizon and are inserted in existing routes considering the difficulty of insertion. The algorithm is reportedly capable of a good quality-time compromise solutions on a 300-customer, 24-vehicle instance. [19] uses three different objectives, namely the number of serviced requests, the perceived QoS level by users and the traveled distance. The insertion of new requests is based on a clustering phase and a routing phase. The routing algorithm applies branch-and-bound to a set of requests that according to time windows are closer to occur. The authors report that they have performed experiments in two cities in northern Italy. Finally, in [20], an off-line/on-line two-phase strategy is used for inserting of a new request into an existing route, the objective being minimum user dissatisfaction.

To conclude this review of Dial-A-Ride Problems, we cite some more recent works. [21] defines an Adaptive Large Neighborhood Search (ALNS) to find a set of minimum-cost routes on a DARP variant with time windows and additional constraints on and maximum user ride times. [22] proposes a distributed algorithm to solve large scale DARP instances: tests on a set of 24 different scenarios with up to 16,000 requests or 32,000 locations in the city of San Francisco prove it effective. [23] tackles a dynamic DARP found in a mobility service operated by a private company, in which service requests are either in advance or in real-time and get an immediate answer about being accepted or rejected. The main goal is to maximize the number of accepted requests. Three main online reinsertion heuristics (HDR, GH, IGH) based on different neighborhoods are proposed. [24] presents the e-ADARP, a DARP variant which considers the use of electric autonomous vehicles, thus introducing battery management, charging stations, recharge times. The goal of the problem is to minimize a weighted objective function of vehicles total travel time and excess ride-time of

the users. The authors propose a 3-index and a 2-index MILP formulations, along with a branch-and-cut algorithm with new valid inequalities. [25] addresses the dial-a-ride problem (DARP) using private vehicles and alternative nodes (DARP-PV-AN), in which to achieve greater flexibility the on-demand transportation service can be done either by a public fleet or by clients that use their private vehicles. Several pickup/delivery nodes for the transportation requests are considered to address the resulting privacy concerns. A compact MILP model and an Evolutionary Local Search (ELS) algorithm are proposed.

2.2. Variations of Dial-a-Ride Problem

A common example in many countries is door-to-door transportation for the elderly or disabled persons (i.e. Paratransit [26, 27]). [28] presents an integrated version of the DARP (IDARP), Some part of each journey can be carried out by a fixed route public transport service, without synchronizing time tables, which assumes the services are frequent. In many of the practical cases described by IDARP it is useful and reasonable to allow a request to end its journey on a transfer node near its drop-off node without taking another vehicle for the last leg (part) of the journey. Likewise, there may be cases where it is reasonable, from reducing costs and user-inconvenience points of view, to start the journey on a transfer node rather than on the original pick-up locations. This has lead the authors of [29] to propose the integrated dial-a-ride problem with timetables (IDARP-TT), in which, under some circumstances, requests may start or end the journey from/to transfer nodes sufficiently close to the pick-up or drop-off nodes without involving the vehicles in the first or last part of the journey. Each request has a given origin, destination, and demand for a set of resources, such as regular seats, wheelchair spaces, and baggage. A request can be served by a single vehicle or transferred between a demand-responsive vehicle and fixed routing system. A heterogeneous fleet of vehicles with different speeds, operating costs, and capacities is located in a depot and used to serve requests. The goal is to find vehicle routes that minimize the cost of service on demand and the cost of using the fixed routing system.

2.3. ODT Solution Methods and Approaches

Traditional approaches for ODT consider a centralized dispatcher architecture like in [30, 31] or a decentralized Multiagent System (MAS) to reduce problem complexity with a central coordinator like in [32, 33]. There exist several approaches for decentralized decisions and self-coordination, like [34] which introduce a multi-agent bid-based real-time scheduling solution in fully decentralized settings. Here each vehicle represented by an agent can negotiate via radio channels with flexible decision criteria. A pattern recognition algorithm is used to predict the most likely locations for the next demand using agent-based data mining to recommend movements to these locations.

Investigating the applicability of genetic programming (GP) for developing decentralized MAS that solve dynamic DARP, [35] presents a method to automatically generate a MAS that can solve the DARP for a specific set of scenarios. GP is used to generate a heuristic that is effective in solving the DARP compared to centralized solutions. The best result achieved with this approach is by planning only one demand in advance by vehicle, which maximizes the agent's local interests (greedy) and produces a feasible solution very fast.

An agent-based model based on simulated events for the real taxi market was proposed by [36], where supply and demand matching depends on event-based interactions. According to their conclusion, one of the main limitations is the assumption of uniform distribution of demand in the service area. To address the uncertainty caused by the dynamic nature of online demands, with ignoring the time required to execute a planning algorithm, [37] proposed using a deterministic rolling horizon solution approach, in which plans are drawn up using all known information in a planning horizon, that is "rolled" forward to include more available information.

Based on vehicle coordination via message passing, and extending the generic model for Online Localized Resource Allocation (OLRA) [38], using P2P communication, in our previous work [39] we proposed the Online Localized with Communication Constraint Resource Allocation (OLC^{2}RA) for concurrently solving the allocation problem over a fleet of autonomous taxis, in which a vehicle decide its next destination (scheduling only one demand in advance). On the contrary, ALMA decentralized heuristic proposed by [2] is wholly decoupled and does not require direct communication between the participants. They demonstrate an upper bound of the speed of convergence which is polynomial to the desired quantity of resources and competing agents per resource; in the realistic case where the mentioned quantities are limited whatever the total number of agents/resources, the convergence

time remains constant as the total size of the problem increases. However, in [2] conflict detection still requires communication with other vehicles, resources, or a central entity to enable resources to share information about their status, such as the blackboard coordination mechanism.

So far and to the best of our knowledge, efforts done in ODT planning domain provide either optimized solutions ignoring the execution time, or provide very fast, just feasible solution to respond quickly to dynamic online demands. In this work, as the optimal solution for this dynamic problem is not achievable within reasonable computational time, our proposal combines the benefits of these approaches in one heuristic that provide a fast auction-based response (fast feasible solution), and this solution is gradually improved at run time with demand-exchange rescheduling.

3. AVFAP Model

The section expounds on the problem addressed in the remainder of the paper. The Autonomous Vehicle Fleet Allocation Problem (AVFAP) extends the traditional DARP by considering a fleet of autonomous vehicles that coordinate without a central dispatcher. The vehicles make decentralized decisions based on information exchanged via Peer-to-Peer (P2P) communication as illustrated in Figure 1.

The city map graph, shown in Figure 2, consists of nodes representing geographic locations and edges representing the road connections between these locations. A fleet of autonomous vehicles is distributed throughout the city. Each vehicle has a set of properties whose values are constant (capacity, cost, and average speed) or variable (location, schedule) because they are time-dependent. Passengers emit demands from different locations (which we will call sources later). Each one takes the form of a request that defines: the pick-up and delivery locations associated with the desired service time window. We define a solution as a schedule for each vehicle that meets the demands by satisfying their constraints, minimizing passenger waiting time, and minimizing vehicle travel costs.

The vehicles communicate by broadcast via an ad-hoc Vehicle-to-vehicle communication (V2V) network, where the communication range is limited. Each vehicle that receives new information broadcasts it again, and the vehicles are thus connected by transitivity within their communication range. In the rest of
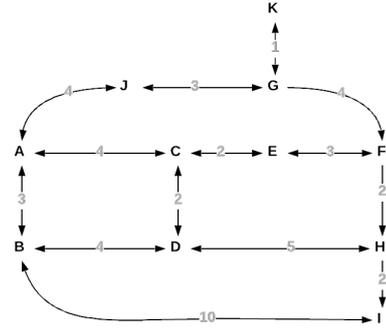

Fig. 2. A sample AVFAP city infrastructure

this article, we use the term "connected set" to refer to a set of items (vehicles and sources) that can communicate with each other directly or by transitive message passing.

In this context, a vehicle is not aware of requests outside its connected set. The vehicle's belief evolves during the execution time, as the vehicle moves around, receives new requests from request sources, meets other vehicles, and exchanges messages with them. By using a low latency, fast connectivity communication mean like DSRC, we can ignore the time loss caused by the communication delay within connected sets

In the same direction of the work done in [39], the AVFAP model is defined as:

$$AVFAP := < M, V, D, T >$$
$$M := < G, w >$$
$$V := \{v_1, v_2, \ldots, v_n\}$$
$$D := \{d_1, d_2, \ldots, d_m\}$$
$$T := \{t_0, t_1, \ldots, t_{end}\}$$

where M defines the urban infrastructure map (locations, roads, and distances); the offer is represented by a V fleet of n autonomous vehicles; D defines a dynamic set of passenger demands that occur at the time of execution, and T defines the time horizon within which vehicles must respond to passenger demands. We define time T as a discrete set of *ticks*.

3.1. Urban Infrastructure Map

The urban infrastructure map is defined by a weighted directed graph M, as shown in Figure 2.

$$G := < N, E >$$
$$w := \{w_{e_1}, w_{e_2}, \ldots, w_{|E|}\}$$

G is a directed connected graph where N is the set of nodes, E is the set of edges between nodes. The valuation function w associates each edge e with the value w_e based on a measure of temporal distance (for example, average driving time in minutes), which will be used to calculate the operational costs of vehicle trips.

3.2. Vehicle Properties

Each $v \in V$ is defined by

$$v :=< capa, cpd, range >$$

and has a set of dynamic properties

$$v\_location : V \times T \to N \cup E$$

$$free\_seats : V \times T \to \mathbb{N}^+$$

Where *capa* defines the total number of passenger seats in the vehicle, *cpd* defines the vehicle's cost per unit of distance, and *range* defines the communication range within which the vehicle can communicate with other entities. At any given time $t \in T$ a vehicle $v \in V$ knows its current situation: *v_location* defines the location of v that could be located in a node or moving through an arc; and *free_seats* defines the number of free seats available in v at time t.

3.3. Demand Properties

A demand $d \in D$ is defined as a request

$$d :=< required, tw, pick\_up, drop\_off >$$

where *required* is the number of seats required; *tw* defines a time interval $[t_{min}, t_{max}]$ in which the pickup event is considered acceptable; *pick_up* and *drop_off* are the origin and destination of the request, respectively. As for vehicles, at time $t \in T$, we will consider that a request $d \in D$ can also be communicated using the V2V network. The request could be issued by the customer or the infrastructure (roadside unit).

4. Auction-based Insertion Heuristics in ORNinA

This section presents our new insertion heuristic, where vehicles coordinate through auctions. The main objective of vehicle agents is to provide feasible schedules that maximize their utility by minimizing the global operational cost of serving the maximum number of requests.

Given a vehicle v having a potential demand set D_v, providing a schedule for v that satisfies all the requests $d \in D_v$ means solving a TSP to produce the best ordering of requests in the schedule. Considering the dynamic aspect of our model, we use an insertion heuristic like the one described in [3] to adapt local vehicle schedules continuously. The result of this algorithm is a set of requests; each of them is associated with the potential time at which a vehicle will be at the pickup location. The insertion heuristic is proven to be efficient in finding feasible schedules very fast [40], but since it handles the scheduling of requests one by one, its performance is relative to the order of these demands.

In our model, the agents handle the request based on their priority order. An agent uses a priority function to assign priority values to the requests he knows. Given a vehicle v, the priority function returns for a demand d a value that is inversely proportional to the cost $costDemand(d, v)$ of inserting the demand in the schedule of v: $\frac{1}{costDemand(d,v)}$. Thus, the lower the cost, the higher the priority. The cost function returns the distance of d's pick-up location from v's location in its simplest form. Thus agents assign the highest priority to the request with the nearest pick-up location, wrt their current location.

Each agent determines his schedule to maximize the value of the quality of his solution. Since several vehicles may be interested in the same request, we need a coordination mechanism to resolve these conflicts. We will use an auction-based mechanism for this purpose, which is one of the effective and proven ways to solve such problems [41].

4.1. Auction Theory

Since ancient times, auctions have been used to answer the most fundamental questions in the Economy: who should get the goods and at what price? Therefore, auction theory is one of the most significant and widely studied topics in Economics. Some types of auctions are well known, such as the ascending-bid auction or the first-price auction used in many public markets.

Auctions are distinguished not only by the terms and conditions of the auction but also by the environment in which they take place. These auctions can be studied in a wide range of characteristics of the auction environment, including the number of participants, the number of items traded, the parties' preferences, and the form of private information that participants have about their preferences.

A common aspect of auction forms is that they receive information, in the form of *bids*, from potential participants about their willingness to pay the price, and the result –i.e., who wins what and how much has to be paid– is determined exclusively based on the information received. This implies that auctions are universal in the sense that they can be used for many scenarios to determine winners [42], especially in resource allocation [43, 44].

4.2. Bid Criterion

In our model, we use a *first-price auction form* [42] to answer the question "which vehicle will consume the request and at what cost?". When a vehicle v becomes aware of a request d, it ranks it in its queue according to the priority it has assigned to it. At the time t, v selects the first request d_s in the queue, generates a set of alternatives, each of which is a potential schedule resulting from the insertion of d_s in a feasible step (that does not violate any of d_s's constraints) of the v's current schedule. Every alternative is associated with a *cost*, which is the marginal operational cost of adding this request to the schedule. The choice with the best *cost* is considered to broadcast an offer

$$Bid_v^d(t_{start}, cost)$$

with t_{start} the time of *pick_up* for d_s.

In this paper, we consider the operational cost of trips as a linear relation to the routes' lengths, so that for a vehicle whose $cpd = 1$, the cost of a trip is simply the total length of its route (sum of the distances, as shown in Figure 2). We consider the value of $cpd = 1$ for every vehicle in the following examples. Therefore, the marginal cost of insertion is the difference in path length between the initial path and the new path.

4.3. Winner Determination

Given a set of bids B in a combinatorial auction on a set of resources R, the winner determination problem (WDP) is defined as finding an allocation of items to
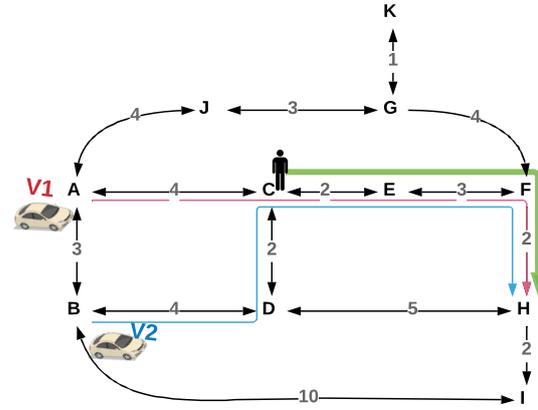

Fig. 3. V_1 wins the auction to serve d_1 from C to H

bidders that maximizes the auctioneer's revenue [45]. The WDP is known as a NP-Hard problem because each subset of bids would have to be checked for feasibility and the revenue that this subset of bids provides. In its simple form, when the auction is about a single, indivisible resource $\nabla \in R$, the question becomes, "to whom ∇ should be allocated?", which can be expressed as a linear relation:

$$winner : R \rightarrow B \cup \phi$$

Seeking decentralization, we considered a binary version of this relation corresponding to our problem

$$win : V \times D \rightarrow \{0, 1\}$$

where the bidders are the vehicles and the resource items are the requests. We assume all agents to be truthful collaborating agents so that every vehicle $v \in V$ that is aware of a request d is responsible for determining the value of $win(v, d)$. The default value for $win(v, d)$ is 0. After their announcement, the bids remain available for a specific time period t_{expire}. Until t_{expire} the vehicle listen to other vehicles' bids on d, if it receive a better offer, it assigns the value 0 to $win(v, d)$ which means it determines itself as a looser and withdraw from the auction. On t_{expire} if the bid cost of v is less than any other bid $Bid_{v'}^d$ received at $t_{Bid_v^d} + t_{expire}$ to serve a request d, v considers itself the winner of the auction (i.e. assigns 1 to $win(v, d)$), and updates its schedule with the new bid path. This mechanism implies that on t_{expire} of each auction d is allocated to at most one vehicle (in the connected set).
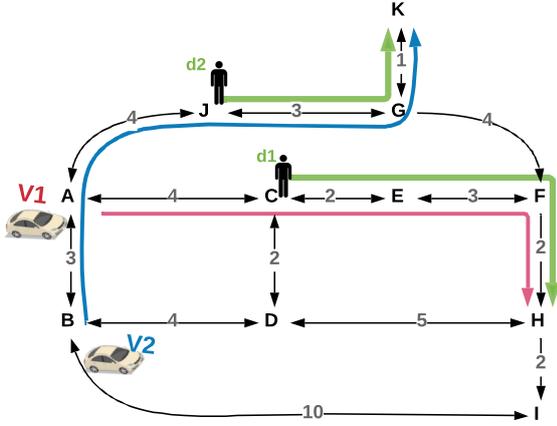
Fig. 4. With no demand exchange, V_2 wins d_2 and V_1 keeps d_1

Fig. 5. Global improvement when V_1 abandons d_1 to serve d_2

Example 1. *The simple scenario in Figure 3 shows two vehicles V_1 and V_2 located in A and B, with empty schedules at the beginning. At time t_1, the first request is announced $d_1 :< 1, (t_{10}, t_{20}), C, H >$. Both vehicles now know d_1. V_1 can serve it by following the path $A \rightarrow C \rightarrow E \rightarrow F \rightarrow H$, so V_1 places the offer $Bid_{V_1}^{d_1}(t_{10}, 11)$. V_2 can serve it via the path $B \rightarrow D \rightarrow C \rightarrow E \rightarrow F \rightarrow H$, so issues the offer $Bid_{V_2}^{d_1}(t_{10}, 13)$. V_1 is considered a winner and adds d_1 to its schedule, so that the overall operational cost of the fleet is now 11.*

4.4. Limitations in Dynamic Settings

The problem at hand is considered as dynamic in both spatial and temporal dimensions. On the one hand in the temporal dimension, the requests are not known in advance but announced in a stochastic manner at run-time. On the other hand, the vehicle's knowledge is also limited to the information received and shared in its neighborhood through its connected set. This implies the dynamics in spatial dimension as the vehicle location affects the context of the problem. This dynamics may affect the quality of upcoming planning decisions dramatically and reduces benefits of the already taken long-term planning decisions [35] similar to the situation shown in the following example.

Example 2. *Figure 4 shows a situation where the use of the bid-based insertion heuristic is very reactive but does not guarantee good scheduling. At the moment t_2 when the new request $d_2 :< 1, (t_{15}, t_{40}), J, K >$ arrives, both vehicles are aware of it and place their possible bids. In the absence of any exchange capacity, V_1 still*
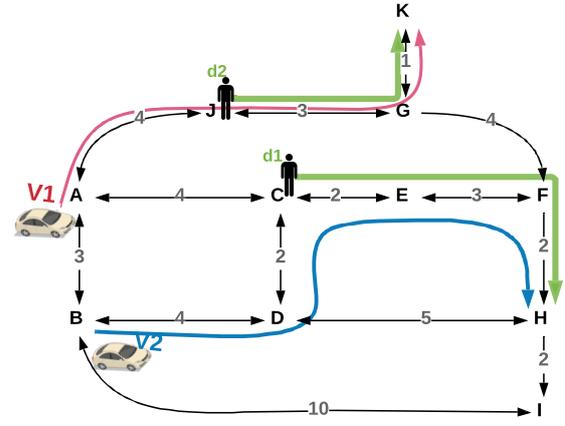
has d_1 in its schedule (with an initial cost of 11), so the best offer it can place is to serve both requests with a marginal cost of 14. While V_2 places the winning bid $Bid_{V_2}^{d_2}(t_{15}, 11)$, it adds d_2 to its schedule, and the overall cost becomes 22. Note that in this case, serving d_2 with V_1 and letting V_2 take care of d_1 (as shown in Figure 5) results in an overall gain of global operational cost which becomes 21, but this solution is never realized because d_1 is already scheduled on V_1.

To be able to make effective bids for new requests or improve the solution, we also propose that the vehicles exchange their planned requests, as illustrated in Example 2.

5. Online Solution Improvement in ONRInA

In the following, we propose a local optimization protocol to improve the quality of the solution for cases like Figure 4.

This protocol is based mainly on the *k-exchange* neighborhoods (*k-opt*), a path improvement algorithm, where at each planning phase, *k* steps from the current plan are replaced by *k* steps to get a cheaper path [4]. In its simplest settings, *k* is set to 1, so the *k-opt* becomes 1-opt, which means a vehicle can exchange at most one demand at a time. In this work, we seek the implementation of the simple auction mechanism 1-opt to avoid dealing with the NP-complete winner determination of the *k-opt* combinatorial auctions.

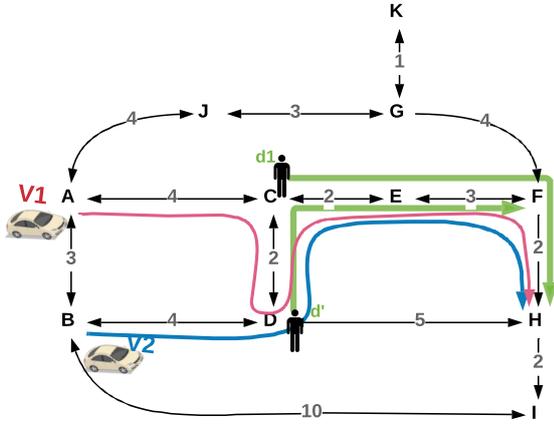**Example 3.** *Let's consider another case shown in Figure 6, where V_1 has d_1 in its schedule, V_2 has*

Fig. 6. While it can serve both demands, V_2 can only bid for one demand at a time

empty schedule, and a new demand is announced $d' :< (t_{10}, t_{20}), D, F >$. V_1 offers two alternative bids $Bid_{V_1}^{d_1, d'}((t_{10}, t_{12}), +4)$, and $Bid_{V_1}^{d'}(t_{10}, +15)$ (taking into account abandoning d_1), while V_2 can only offer $Bid_{V_2}^{d'}(t_{10}, +13)$.

Obviously $Bid_{V_1}^{d_1, d'}$ is the winner and V_1 takes the charge of both demands with global cost $+15$, while the optimal solution in this case is that V_2 who takes both demands which make the global cost only $+13$, but using the auctions with abandon strategies lets V_2 in this case either bid for d' alone when it is announced, or bid for d_1 alone in response of the abandon suggestion by V_1. So in this case, the optimal solution is not achievable with the aforementioned abandon strategies.

5.1. Pull-demand Optimization Bids

Similar to the *rolling horizon* strategy of [37], we propose an optimization protocol to improve our heuristics. In the *rolling horizon* strategy, all vehicle schedules are considered temporary and available to be scheduled by any vehicle, unless they are considered as *committed requests* by particular events; for example, *v* has started serving (moving towards) *d*, whose remaining time to serve it is below the horizon threshold.

The application of this strategy requires that all vehicle schedules are in shared memory, so that when a vehicle v_i offers to serve a request *d*, it knows if it is scheduled by another vehicle v_j, and therefore if it should send its offer cost to v_j. Then v_j will calculate

---

Protocol 1 Pull-demand Optimization Protocol

Step 1 A set of new demands enters the system based on announcement time order.

Step 2 Each new request is distributed to the connected set to which the sources belong. Each agent in this set can select his potential requests from requests that include new requests, scheduled and unscheduled requests that have not yet reached their scheduled departure time.

Step 3 The agents enter the auction to serve their potential demands in similar auction criteria (i.e. for each agent in the connected set, the potential requests are chosen and ordered in a list of candidates based on the same priority function, and the same cost function) of the initial phase.

Step 4 Each agent searches among its scheduled requests for the one to satisfy the next tick; this request is called d_{next}. If d_{next} exists, the agent broadcasts a "*clear_demand*" message to inform other agents that it handles d_{next}. Each receiver deletes it from their potential and known sets of requests. In addition, each agent deletes any other requests that reach their time window upper-bound because staying any longer available for rescheduling would violate their time constraints.

Step 5 The scheduled and unscheduled requests that still have time remain announced by their sources (*Step 2*). This allows better planning in the next tick if new requests are announced or some new agents join the connected set.

---

its estimation to the global gain (or loss) in operating cost by abandoning *d* compared to the cost proposed by v_i. If there is a gain, it agrees to abandon *d* and then v_i updates its schedule with *d*, otherwise the bid is rejected.

In our protocol, we do not use the concept of *committed request*, but a vehicle can only bid on requests that it can satisfy, so requests that are rescheduled or that do not have enough time to be rescheduled are automatically ignored by the agent. Another difference here is that we don't have a shared memory. Agents exchange information about the context of the environment and about requests through information messages. Moreover, in [37], optimization is performed periodically at a predefined frequency, while the protocol we propose is executed in parallel with the auction-based insertion strategy to have a fast rescheduling for continuous requests. Based on shared information of the current

context, the optimization protocol is executed between connected sets of vehicles when any change in this set context is detected (the set of vehicles in the connected set is changed or at least one of them is newly aware of some requests which are already scheduled by others). Protocol 1 details this strategy.

5.2. Discussion

Given the decentralized context, the insertion heuristic is very efficient in terms of response time. The temporal complexity of the basic insertion heuristic for the Vehicle Routing Problem (VRP) is of $\mathcal{O}(n^3)$[40]. This type of heuristic is often used to solve DARPs, where new incoming requests have to be continuously processed in real-time and integrated into the evolving schedules of vehicles. The usage of k-opt may add local improvements to the insertion heuristic solution based on the current context. However, the Pull-demand protocol can significantly improve the quality of the solution, as illustrated in the following example.

Example 4. *Let's consider another case illustrated in Figure 6, where V_1 has d_1 in its schedule, V_2 has an empty schedule, and a new request d' :$<(t_{10}, t_{20}), D, F >$ is announced. The bids of the vehicles are thus $Bid_{V_1}^{d'}(t_{12}, +4)$, and $Bid_{V_2}^{d'}(t_{10}, +13)$. It is obvious that $Bid_{V_1}^{d_1,d'}$ is the winner, and V_1 will handle the two requests with an overall cost of 15. With the Pull-demand protocol, d_1 and d' enters in the set of candidate requests for V_1 and V_2, so that the vehicles can make combinatorial offers: $Bid_{V_1}^{d_1,d}((t_{10}, t_{12}), 0)$ and $Bid_{V_2}^{d',d_1}(-2)$. The cost of V_2 is 13 and the gain of V_1 is 15). V_1 has nothing to change in its schedule. V_2 wins and the solution is improved with an additional optimization round. Let's look at the applied protocol, step by step.*

Step 1: d' enters the system at t_2 and both vehicles are aware of this, V_1 wins the auction with an overall cost of 15

Step 2: d_1 and d' are now in the set of requests known by both vehicles, V_2 calculates the costs to serve d_1 alone (13), d' alone (9) and both requests together making 13. It then selects the two requests as its potential requests. V_1 has no potential request because it already has the two requests in its schedule.

Step 3: V_2 places a bid $Pull\_Bid_{V_2}^{d',d_1}((t_{10}, t_{12}), 13)$. For V_1 the cost to serve both requests is 15 so it accepts $Pull\_Bid_{V_2}^{d',d_1}$ because it causes a gain of 2.

Step 4: None of the requests reach their scheduled service time or the upper-bound of their time window.

Step 5: All known requests remain announced and available for the next potential improvement.

6. Experimental Evaluation

In this section, we experimentally evaluate the performance of our contributed *ORNInA* approach, using synthetic data and Open Street Map information.

6.1. Experimental Setup

The city map of Saint-Étienne was chosen for the simulation. The structure of the graph $G =< N, E >$ including nodes, edges and a set of sources of the $S \subset N$ request is extracted from OpenStreetMap (OSM[1]) and post-processed by *Plateforme Territoire*[2]. In all the experiments, we set the number of sources $|S| = 40$, having a set $E_S \subset E$ of edges, such that $|E_S| = 71$ connecting the sources, each edge has a number of points which varies according to its length and the information extracted from OSM. The distance between two consecutive points is 40 meters. We used a discrete-time transport simulator available in the *Plateforme Territoire* to evaluate the proposed strategy and analyze it in terms of quality of service and gain.

The Java-based ORNInA and IBM ILOG CPLEX Optimizer Version 12.9.0 have been executed on an octa-core Intel® Core™ i7-8650U CPU @ 1.90GHz, with 32GB DDR4 RAM.

A fleet V of n vehicles is distributed randomly through S at the beginning of execution. Each vehicle $v \in V$ moves from one point to another on the same edge during each simulation cycle. In our test, we consider that vehicles communicate via DSRC with a realistic communication range of 250m so that a vehicle can send/receive messages to/from other entities located in its range. Each vehicle v can adopt two distinct travel behaviors: either *marauding* for requests or *going to* a destination:

- going_to defines the state of a vehicle when it has a specific destination, i.e. a request to serve. The vehicle is either *going_to* pickup location if the request is not yet picked up, or *going_to* delivery location otherwise.

---

[1]https://www.openstreetmap.org/
[2]https://territoire.emse.fr/

- `marauding` defines the state of a vehicle when it does not have a request to serve, i.e., at the beginning of the simulation, each vehicle marauds until it decides to serve a request. Once a passenger is dropped off, the vehicle reverts to `marauding`. In this state, the vehicle randomly moves through its neighborhood to find requests to serve.

At each simulation cycle, 0, 1 or 2 requests are generated in a uniform random manner. For each request, the origin and destination points are randomly and uniformly generated from all sources. The time window for the requests is generated using two constant parameters l and u for the lower and upper limits as follows $[tw_{min}, tw_{max}]$ is initialized with two uniform random values where :

$$tw_{min} < tw_{max}$$

$$tw_{min} \geqslant t_{actual} + l$$

$$tw_{max} \leqslant tw_{min} + u$$

The evaluation criteria for these simulations are mainly the number of requests satisfied as a measure of Quality of Service (QoS), the simulated profit of the solution as a measure of Quality of Business (QoB). The profit is calculated in terms of the difference between the simulated travel price and the cost.

$$profit = total\_income - total\_cost$$

where

$$total\_income = \sum_{d \in D_s} P + p * distance(d)$$

$D_s \subseteq D$ is the set of all satisfied requests, P is a fixed price (service fee) per request, p is a pricing factor per unit of distance travelled, $distance(d)$ is the total travel distance for a request d and

$$total\_cost = \sum_{v \in V} cpd(v) * total\_distance(v)$$

where $total\_distance(v)$ is the total traveled distances by v including `marauding` and `going_to` distances.

In real-life scenarios, the pricing criteria is more complex and depend on many other factors, here we provide an approximate pricing model based only on distance. In our tests, we consider the vehicles identical in terms of $cpd(\cdot)$ travel cost and p pricing factor. We set $P = 1.5$, $p = 2$ and $cpd(v) = 1$ for all v, so that the vehicle will gain 1.5 money-unit per satisfied request in addition to 2 money-unit per traveled distance-unit with passenger on-board, while it looses 1 money-unit per traveled distance-unit. A successfully traveled distance with passenger on-board leads to a net gain of 1 money unit.

Maximizing the income requires maximizing the number of satisfied requests and the riding distance, while to reduce the costs, the vehicles need to reduce the empty driving slots in their trip schedules. In this notion, we can define the allocation process's objective as maximizing the QoB function *profit*.

To assess the feasibility of the ORNInA heuristic, we compare it to two other approaches:

GREEDY: a decentralized approach based on planning a single request in advance. This approach has been mentioned by [35] as the best strategies for dynamic settings, following genetic algorithm selection.

CPLEX: building vehicle schedules dynamically by a central dispatcher that uses CPLEX solver to solve the allocation problems as MILP which maximize the objective function *profit*. The central dispatcher calculates a solution in response to a query from a connected set.

The two decentralized approaches use the same request selection strategy (request priority function that is derived from the objective function) during each scenario. Being optimal, the CPLEX approach is reported as an upper-bound for the solution quality, regardless of its computational time and practical applicability.

6.2. Experimental Results

The results presented in the Figures 7 and 8 concern a scenario in which vehicles select the cheapest request considering the $costDemand(d, v)$ is the marginal cost mentioned in Section 4; the same applies to the greedy algorithm since the vehicle schedule can only contain one demand. We have executed several instances of problems that vary with the size of the fleet n. Each instance of these tests is executed 10 times with different probability seeds. First, we evaluate with a fixed fleet size $n = 20$ to track the quality of the solution over time, then with a variable fleet size $n \in [3..35]$ over 300 cycles for each scenario. The global quality of solutions obtained by the CPLEX solver depends on two factors:
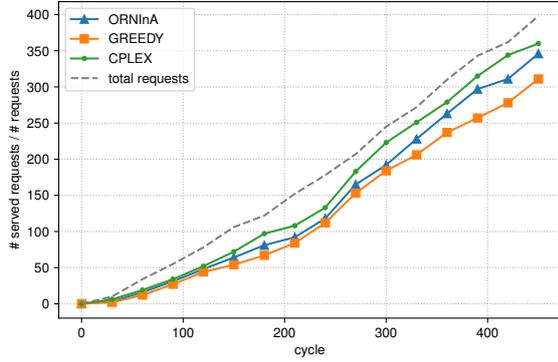
Fig. 7. Quality of service for a fleet of 20 vehicles

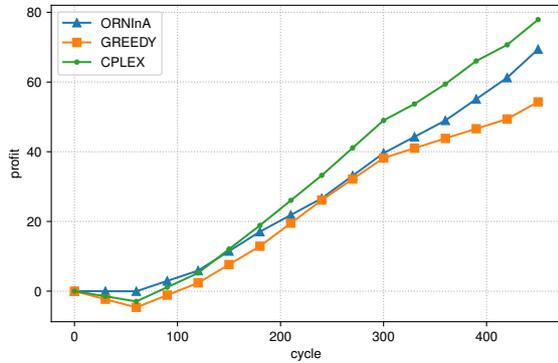Fig. 9. Quality of Service with increasing fleet size

Fig. 8. Quality of business for a fleet of 20 vehicles

- the amount and accuracy of information about the context of the vesicles and the requests,
- the frequency of schedule update queries.

To achieve a global solution that is optimal (or very close to optimal) in this dynamic settings, the solver must be queried at each simulation cycle for each connected set to consider the new announced requests. This is not feasible in practice because of the computational complexity of the problem but still provides offline bounds on the solution. This strategy is coined CPLEX_R in the figures. In our experiments, we examined, in addition, another strategy in which the CPLEX solver is queried each time a connected set member changes in order to obtain an optimal solution for the sub-problem defined by the context of the connected set. This solution is still valid as an optimal decision until the context is changed. This strategy is coined CPLEX_CS in the figures. The results obtained by the first strategy (CPLEX_R) represent an upper-bound for the objective function (QoB) as the CPLEX solver is queried to calculate the optimal solution in the highest possible frequency (i.e., every time a new request
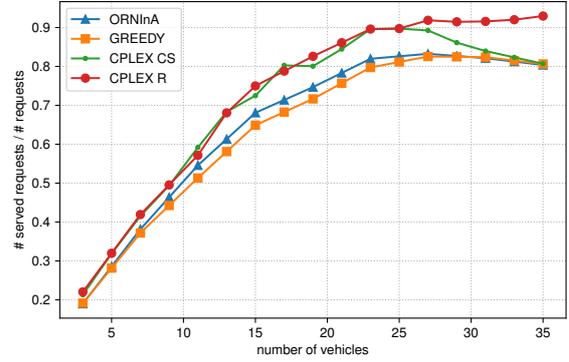
is announced). Usually, the vehicles move and change their connected sets frequently, which means even the second strategy (CPLEX_CS), can keep the schedules mostly up-to-date. However, on the one hand in some scenarios with few vehicles in a wild map, vehicles may be far away from each other, so they do not belong to any connected set, and their knowledge about requests is limited. On the other hand, with more vehicles, the connectivity between vehicles increases, leading to some point where all vehicles are stably connected, e.g., all vehicles belong to a unique connected set that does not change. In such a case, the CPLEX solver is never queried with the CPLEX_CS for a while until a vehicle leaves the connected set. These two cases explain why in the medium fleet sizes, the results from CPLEX_CS are close to the upper-bound obtained by CPLEX_R, as the connected sets change more frequently and so the rescheduling. Note that it is impossible to achieve a 100% QoS rate in these scenarios since there will always be requests that will be generated until the last cycle, assuming that the scenario execution continues to serve them, while the execution stops at the last cycle, leaving them unserved.

With a low number of vehicles, the connected sets are small and, as a consequence, the amount of shared information is reduced so that there are no quality discrepancies between the compared approaches. With larger fleet sizes, more information is shared in the connected sets. Additionally, vehicles switch from one connected set to another more frequently. Although QoS values of our algorithm are close to those obtained by the greedy approach, they remain slightly ahead of them, and to achieve the same values of QoS with no coordination, more vehicles in the fleet are required, as shown in Figure 9.

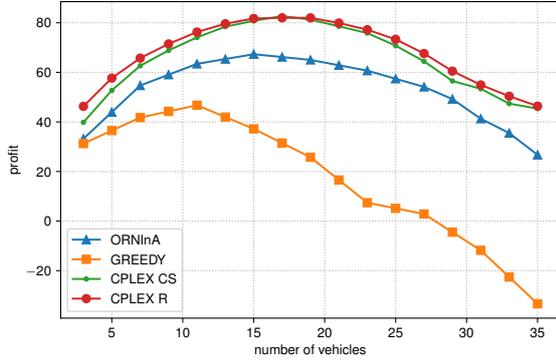Starting from small fleet of 3 vehicles, the ORNInA heuristic allows obtaining solutions with the same QoS

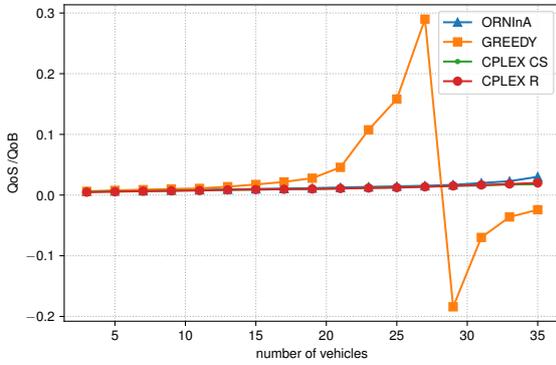Fig. 10. Quality of Business with increasing fleet size

Fig. 11. Quality of Business vs. Quality of Solution

Table 1
Exchanged messages per simulation cycle

Approach	Count	Avg size (Bytes)	Max size (Bytes)
GREEDY	$O(n)$	88	140
CPLEX	$O(2n)$	163	2500
ORNInA	$O(n * m)$	112	140

its n_{QoB} growth threshold. In general, n_{QoB} is less than n_{QoS}, and we can see a trade-off between improving QoS or QoB.

According to the Figure 10, we see that the greedy fleet has $n_{QoB}^{\text{GREEDY}} = 12$ with a *QoS* of only about 60% compared to $n_{QoB}^{\text{ORNInA}} = 16$ with a *QoS* of about 70%, while the highest value of *QoB* in these experiments is about 80% with $n_{QoB}^{\text{CPLEX}} = 18$. Note that $n_{QoB}^{\text{GREEDY}} < n_{QoB}^{\text{ORNInA}}$ and even if it decreases afterwards, the value of n_{QoB}^{ORNInA} remains higher than n_{QoB}^{GREEDY}. This gives a broader range of options for combining the enhancements of the provided solution (QoS and QoB). Figure 11 particularly demonstrates how GREEDY has an abysmal performance when looking at both QoS and QoB (and their ratio), while the other allocation mechanisms remain in the same group, completing both qualities, but at different levels.

In the three compared approaches, every simulation cycle, agents in a connected set of size *n* that are aware of *m* requests exchange info messages. In addition, agents in ORNInA approach exchange bid and pull messages, while in the centralized CPLEX approach, agents send queries and receive schedules in answer from the solver. Table 1 shows statistics about the exchanged messages by the three approaches in these experiments. In GREEDY, agents exchange the minimal number and size of messages that contain the essential information about known requests and the contexts of connected sets. Each agent may receive up to one message from each other member of its connected set.

The size of a query message to the CPLEX solver depends on the sub-problem instance size. In our experiments, the largest sub-problem instance was 10 vehicles and 22 requests; this instance's query message reached the size of 2.5 kB. In ORNInA, the bid and pull messages and their answers are of small size, but the message count is higher.

The above results show that in all cases, and in the condition of updating schedules frequently, scheduling several requests in advance (like ORNInA and CPLEX) gives better results than scheduling a single request with a GREEDY approach.

In practice, the dynamic aspect of the problems requires frequent and real-time computation of sched-

and QoB values as those of the GREEDY algorithm and the CPLEX solver. More requests can be satisfied by increasing the size of the fleet, which increases the QoS and QoB values.

We find that the increase in these values for the ORNInA heuristic is more significant than that of the greedy approach. This can be explained by the fact that demand planning improves the system profit in the foreseeable future, as it reduces the process of roaming without a specific destination until a new demand is chosen, which is the behavior of the vehicles in the greedy algorithm.

The value of QoS continues to grow as the fleet size increases until it reaches a n_{QoS} threshold where the addition of new vehicles becomes unnecessary, as all requests received in the system now or in the future (except those received in the last moments of execution) can be served with the current fleet size. The same is true for QoB, but the difference here is that the vehicles added will result in additional operational expenses, resulting in a loss of profit value after reaching

ules, eliminating the CPLEX central dispatcher approach because of the NP-Hard complexity of the problem. However, ORNInA provides a good combination of quality and feasibility of the solution with its reduced computation time and no global information sharing.

7. Conclusion

In this paper, we proposed ORNInA, a decentralized coordination mechanism for the exchange of requests, based on an insertion heuristic and auctions, to allocate requests to vehicles in the context of dynamic on-demand transport with V2V communication. We model this allocation problem with connectivity constraints as the AVFAP, which extends traditional DARP.

We show through examples that the request exchange protocol can be a promising improvement in the quality of the solutions. To assess the feasibility of the proposed protocol, we evaluated our technique's results on synthetic data for taxis operating in the city of Saint-Étienne and showed that it outperforms a classical greedy approach. Moreover, compared to classical centralized MILP-based solvers (computing the optimal allocation either at each simulation step or at each connected set update), ORNInA provides very good quality results on both QoB and QoS while following fully decentralized decision making.

In future work, we plan to have more evaluation on the efficiency, performance, robustness, and optimality of this heuristic with respect to different approaches, like distributed constraint optimization, by simulating different parameters on information distribution, decision criteria, and different levels of problem dynamics. Finally, to assess the scalability of such decentralized approaches, we plan to simulate larger cities and fleets, based on real on-demand transport dataset, like the NYC-TLC open dataset[3] or an adaptation of the Li and Lim benchmark[4] to AVFAP.

---

[3]https://www1.nyc.gov/site/tlc/about/tlc-trip-record-data.page
[4]https://www.sintef.no/projectweb/top/pdptw/li-lim-benchmark/

References

[1] C. Bellini, G. Dellepiane and C. Quaglierini, The demand responsive transport services: Italian approach, *Transaction on the Built Environment, WIT Press , www.witpress.com, ISSN 1743-3509* **64** (2003), 10. www.witpress.com.

[2] P. Danassis, A. Filos-Ratsikas and B. Faltings, Anytime Heuristic for Weighted Matching Through Altruism-Inspired Behavior, in: *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence*, Macao, China, 2019, pp. 215–222. ISBN 978-0-9992411-4-1. https://www.ijcai.org/proceedings/2019/31.

[3] M.M. Solomon, Algorithms for the Vehicle Routing and Scheduling Problems with Time Window Constraints, *Operations Research* **35**(2) (1987), 254–265. http://www.jstor.org/stable/170697.

[4] K. Helsgaun, General k-opt submoves for the Lin–Kernighan TSP heuristic, *Mathematical Programming Computation* **1**(2) (2009), 119–163.

[5] KFH-Group, U. Associates, M.M. Consulting, C. Systematics, T.C.R. Program, U.S.F.T. Administration and T.D. Corporation, *Guidebook for measuring, assessing, and improving performance of demand-response transportation*, Vol. 124, Transportation Research Board, Washington, DC, 2008.

[6] N. Ronald, R. Thompson and S. Winter, Simulating demand-responsive transportation: a review of agent-based approaches, *Transport Reviews* **35**(4) (2015), 404–421.

[7] T.W.G. for WLAN Standards, IEEE 802.11, The Working Group Setting the Standards for Wireless LANs. https://www.ieee802.org/11/.

[8] E.T.S.I. (ETSI), Intelligent Transport Systems (ITS);Access layer specification for Intelligent Transport Systems operating in the 5 GHz frequency band, Technical Report, 2012.

[9] CE Delft: Arno SCHROTEN, Anouk van GRINSVEN, Eric TOL, Louis LEESTEMAKER and TNO: Peter-Paul SCHACKMANN, Diana VONK-NOORDEGRAAF, Jaco van MEIJEREN, Sytze KALISVAART, Research for TRAN Committee: The impact of emerging technologies on the transport system, Technical Report.

[10] K.C. Dey, A. Rayamajhi, M. Chowdhury, P. Bhavsar and J. Martin, Vehicle-to-vehicle (V2V) and vehicle-to-infrastructure (V2I) communication in a heterogeneous wireless network – Performance evaluation, *Transportation Research Part C: Emerging Technologies* **68** (2016), 168–184. http://www.sciencedirect.com/science/article/pii/S0968090X16300018.

[11] M.W.P. Savelsbergh and M. Sol, The General Pickup and Delivery Problem, *Transportation Science* **29**(1) (1995), 17–29. http://pubsonline.informs.org/doi/abs/10.1287/trsc.29.1.17.

[12] J.-F. Cordeau, G. Laporte, M.W.P. Savelsbergh and D. Vigo, Chapter 6 Vehicle Routing, in: *Handbooks in Operations Research and Management Science*, Vol. 14, Elsevier, 2007, pp. 367–428. ISBN 978-0-444-51346-5. https://linkinghub.elsevier.com/retrieve/pii/S0927050706140062.

[13] J.-F. Cordeau, G. Laporte, J.-Y. Potvin and M.W.P. Savelsbergh, Chapter 7 Transportation on Demand, in: *Handbooks in Operations Research and Management Science*, C. Barnhart and G. Laporte, eds, Transportation, Vol. 14, Elsevier, 2007, pp. 429–466. http://www.sciencedirect.com/science/article/pii/S0927050706140074.

[14] A. Mingozzi, L. Bianco and S. Ricciardelli, Dynamic Programming Strategies for the Traveling Salesman Problem with Time Window and Precedence Constraints, *Operations Research* **45**(3) (1997), 365–377.

[15] J.-F. Cordeau and G. Laporte, The dial-a-ride problem: models and algorithms, *Annals of Operations Research* **153**(1) (2007), 29–46. http://link.springer.com/10.1007/s10479-007-0170-8.

[16] S.N. Parragh, K.F. Doerner and R.F. Hartl, Demand responsive transportation, *Wiley Encyclopedia of Operations Research and Management Science* (2010).

[17] O.B. Madsen, H.F. Ravn and J.M. Rygaard, A heuristic algorithm for a dial-a-ride problem with time windows, multiple capacities, and multiple objectives, *Annals of Operations Research* **60**(1) (1995), 193–208.

[18] J.J. Jaw, A.R. Odoni, H.N. Psaraftis and N.H. Wilson, A heuristic algorithm for the multi-vehicle advance request dial-a-ride problem with time windows, *Transportation Research Part B: Methodological* **20**(3) (1986), 243–257.

[19] A. Colorni and G. Righini, Modeling and optimizing dynamic dial-a-ride problems, *International transactions in operational research* **8**(2) (2001), 155–166.

[20] L. Coslovich, R. Pesenti and W. Ukovich, A two-phase insertion technique of unexpected customers for a dynamic dial-a-ride problem, *European Journal of Operational Research* **175**(3) (2006), 1605–1615.

[21] T. Gschwind and M. Drexl, Adaptive Large Neighborhood Search with a Constant-Time Feasibility Test for the Dial-a-Ride Problem, *Transportation Science* **53**(2) (480-491), 2019.

[22] S. Muelas, A. LaTorre and J.-M. Peña, A distributed VNS algorithm for optimizing dial-a-ride problems in large-scale scenarios, *Transportation Research Part C: Emerging Technologies* **54** (2015), 110–130.

[23] S. Vallee, A. Oulamara and W.R. Cherif-Khettaf, New online reinsertion approaches for a dynamic Dial-a-Ride Problem, *Journal of Computational Science* **47** (2020).

[24] C. Bongiovanni, M. Kaspi and N. Geroliminis, The electric autonomous dial-a-ride problem, *Transportation Research Part B: Methodological* **122** (2019), 436–456.

[25] D. Brevet, C. Duhamel, M. Iori and P. Lacomme, A dial-a-ride problem using private vehicles and alternative nodes, *Journal on Vehicle Routing Algorithms* (2019), 89–107.

[26] Canadian Urban Transit Association, Specialized Transit Eligibility Certification Programs, Overview of Canadian and U.S. Experience, 2013. http://www.bv.transports.gouv.qc.ca/mono/1136638.pdf.

[27] D. Cannon and D. A. Martin, Shaping National Disability Policy: Transportation Access and Social Security Reforms, 2004. https://oac.cdlib.org/view?docId=hb2j49n5h3&query=&brand=oac4.

[28] C.H. Häll, H. Andersson, J.T. Lundgren and P. Värbrand, The integrated dial-a-ride problem, *Public Transport* **1**(1) (2009), 39–54.

[29] M. Posada, H. Andersson and C.H. Häll, The integrated dial-a-ride problem with timetabled fixed route service, *Public Transport* **9**(1–2) (2017), 217–241. http://link.springer.com/10.1007/s12469-016-0128-9.

[30] M. Egan and M. Jakob, Market Mechanism Design for Profitable On-Demand Transport Services, *arXiv:1501.01582 [cs]* (2015), arXiv: 1501.01582. http://arxiv.org/abs/1501.01582.

[31] W. Shen and C. Lopes, Managing Autonomous Mobility on Demand Systems for Better Passenger Experience, *arXiv:1507.02563 [cs]* **9387** (2015), 20–35, arXiv: 1507.02563. http://arxiv.org/abs/1507.02563.

[32] M. El Falou, M. Itmi, S. El Falou and A. Cardon, On demand transport system's approach as a multi-agent planning problem, in: *2014 International Conference on Advanced Logistics and Transport (ICALT)*, IEEE, 2014, pp. 53–58.

[33] J.M.S. Grau, M.A.E. Romeu, E. Mitsakis and I. Stamos, Agent based modeling for simulation of taxi services, *Journal of Traffic and Logistics Engineering* **1**(2) (2013), 159–163.

[34] A. Glaschenko, A. Ivaschenko, G. Rzevski and P. Skobelev, Multi-Agent Real Time Scheduling System for Taxi Companies, *AAMAS* (2009), 8.

[35] R.R.S. van Lon, T. Holvoet, G. Vanden Berghe, T. Wenseleers and J. Branke, Evolutionary synthesis of multi-agent systems for dynamic dial-a-ride problems, in: *Proceedings of the fourteenth international conference on Genetic and evolutionary computation conference companion - GECCO Companion '12*, ACM Press, Philadelphia, Pennsylvania, USA, 2012, p. 331. ISBN 978-1-4503-1178-6. doi:10.1145/2330784.2330832. http://dl.acm.org/citation.cfm?doid=2330784.2330832.

[36] J.M.S. Grau and M.A.E. Romeu, Agent Based Modelling for Simulating Taxi Services, *Procedia Computer Science* **52** (2015), 902–907. doi:10.1016/j.procs.2015.05.162. https://linkinghub.elsevier.com/retrieve/pii/S187705091500962X.

[37] N.A.H. Agatz, A.L. Erera, M.W.P. Savelsbergh and X. Wang, Dynamic ride-sharing: A simulation study in metro Atlanta, *Transportation Research Part B: Methodological* **45**(9) (2011), 1450–1464, Select Papers from the 19th ISTTT. doi:https://doi.org/10.1016/j.trb.2011.05.017. http://www.sciencedirect.com/science/article/pii/S0191261511000671.

[38] M. Zargayouna, F. Balbo and K. Ndiaye, Generic model for resource allocation in transportation. Application to urban parking management, *Transportation Research Part C: Emerging Technologies* **71** (2016), 538–554. doi:https://doi.org/10.1016/j.trc.2016.09.002. http://www.sciencedirect.com/science/article/pii/S0968090X16301589.

[39] G. Picard, F. Balbo and O. Boissier, Approches multiagents pour l'allocation de courses à une flotte de taxis autonomes, in: *RIA2018*, Revue d'intelligence artificielle, 2018, pp. 223–247. doi:10.3166/RIA.32.223-247.

[40] A.M. Campbell and M. Savelsbergh, Efficient insertion heuristics for vehicle routing and scheduling problems, *Transportation science* **38**(3) (2004), 369–378.

[41] P. Cramton, Y. Shoham and R. Steinberg, An overview of combinatorial auctions, *ACM SIGecom Exchanges* **7**(1) (2007), 3–14. doi:10.1145/1345037.1345039. http://portal.acm.org/citation.cfm?doid=1345037.1345039.

[42] V. Krishna, Introduction, in: *Auction Theory*, V. Krishna, ed., Academic Press, San Diego, 2003, pp. 1–10. ISBN 978-0-12-426297-3. doi:https://doi.org/10.1016/B978-012426297-3.50027-8. http://www.sciencedirect.com/science/article/pii/B9780124262973500278.

[43] Y. Chevaleyre, P. E. Dunne, U. Endriss, J. Lang, M. Lemaître, N. Maudet, J. A. Padget, S. Phelps, J. A. Rodr and P. Sousa, Issues in Multiagent Resource Allocation, *Informatica* **30** (2006), 37.

[44] Y. Shoham, *Multiagent Systems: Algorithmic, Game-Theoretic, and Logical Foundations*, Cambridge University Press, 2009, p. 532.

[45] D. Lehmann, R. Müller and T. Sandholm, The Winner Determination Problem, in: *Combinatorial Auctions*, P. Cramton, Y. Shoham and R. Steinberg, eds, The MIT Press, 2005, pp. 297–318. ISBN 978-0-262-03342-8. doi:10.7551/mitpress/9780262033428.003.0013. http://mitpress.universitypressscholarship.com/view/10.7551/mitpress/9780262033428.001.0001/upso-9780262033428-chapter-13.