Multidimentional Self-Organization for Online Time-Constrained Vehicle Routing Problems

Besma Zeddini[1], Mahdi Zargayouna[2]

[1] University of Le Havre, LMAH Laboratory
25 rue Philippe Lebon
76063 Le Havre Cedex, France
[2] INRETS Institute, Gretia laboratory
Building "Descartes II", 2 rue de la Butte Verte
93166 Noisy le Grand Cedex, France
zeddini@ens.univ-evry.fr, zargayouna@inrets.fr

Abstract. Vehicle Routing problems are highly complex problems for which different Artificial Intelligence techniques have been used. In this paper, we propose an agent-oriented self-organization model for the dynamic version of the problem with time windows. Our proposal is based on a space-time representation of the agents' *Action Zones*, which is able to maintain a good distribution of the vehicles on the environment. This distribution answers the objective of the dynamic problem, since it allows the agents to take their decisions while anticipating future changes in the system's parameters.

1 Introduction

Several operational distribution problems, such as the deliveries of goods to stores, the routing of school buses, the distribution of newspapers and mail etc. are instantiations of NP-Hard theoretical problems called the Vehicle Routing Problems (VRP). In its original version, a VRP is a multi-vehicle Traveling Salesman Problem: there exists a certain number of nodes to be visited once by a limited number of vehicles. The objective is to find a set of vehicles' routes that minimizes the total distance traveled. Besides their practical usefulness, the VRP and its extensions are challenging optimization problems with an academic stimulating issues. One of the most widely studied variant of the problem is the time (and capacity) constrained version: the Vehicle Routing Problem with Time Windows (VRPTW henceforth), in which the requests to be handled are not simply nodes, but customers. For each customer, the following information are informed: the concerned node, two temporal bounds between which he desires to be visited and a quantity (number of goods to receive, number of persons to transport, etc.). Each vehicle has a limited capacity, which should not be exceeded by the quantities that it transports. The addition of time windows increases the complexity of the problem, since it narrows the space of valid solutions. The VRPTW can be formally stated as follows.

Let $G = (V, E)$ be a graph with node set $V = N \cup 0$ and edge set $E = (ij)|i \in V, j \in V, i \neq j$, $N = 1, 2..., n$ is the customer set with node 0 is the depot. With each node $i \in V$ is associated a customer demand $q_i(q_0 = 0)$, a service time $s_i(s_0 = 0)$, and a hard service-time window $[e_i, l_i]$ i.e. a vehicle must be at i before l_i but can be at i before e_i and must wait until the service starts. For every edge $(i, j) \in A$, a distance $d_{ij} \geq 0$ and a travel time $t_{ij} \geq 0$ are given. Moreover, the fleet of vehicles is homogeneous and every vehicle is initially located and end its route at a central depot. Each customer demand is assumed to be less than the vehicle capacity Cap. The objective is to find an optimal set of routes (with the minimal cost) such that:

1. All routes start and end at the depot;
2. each customer in N is visited exactly once within its time window;
3. the total of customer demands for each route cannot exceed the vehicle capacity Cap.

The performance criteria are in general (following this order):

1. The number of vehicles used,
2. the total distance traveled,
3. the total waiting time.

Since the problem is NP-hard, exact approaches are only of theoretical interest, and heuristics are performed in order to find good solutions, not necessarily optimal, within reasonable computational times. The VRP and the VRPTW can be divided into two sets [20]: static problems and dynamic problems. The distinction between these two categories relies traditionally on the knowledge (static problem) or the ignorance (dynamic problem) before the start of the solving process of all the customers that have to be visited. The operational problems are rarely fully static and we can reasonably say that today a static system cannot meet the mobility needs of the users. Indeed, operational vehicle routing problems are rarely fully static. In operational settings, and even if the whole number of customers to be served is known, there is still some elements that makes the problem dynamic. These elements include breakdowns, delays, noshows, etc. It is thus always useful to consider a problem that is not fully static.

We rely on the multiagent paradigm for solving the dynamic VRPTW. Multiagent Systems (MAS) are a paradigm having as objective to canalize the distributed artificial intelligence in a simple conceptual approach in the apprehension of complex problems [1]. An agent is a software system, that is situated in some environment and that is able to apply autonomous actions to satisfy its goals [29], and a MAS is a network of loosely coupled agents, which interact to solve problems that overpass the capacities or the knowledges of each one [27]. A multi-agent modeling of the dynamic VRPTW is relevant for the following reasons. First, since it's a hard problem, choosing a design allowing for the distribution of computation can be a solution to propose short answer times to customers requests. Second, with the technological developments, it is reasonable to consider vehicles with onboard calculation capabilities. In this context,

the problem is, *de facto*, distributed and necessitates an adapted modeling to take profit of the onboard equipments of the vehicles. Finally, the consideration of a multi-agent point of view allows to envision new measures, new heuristics, not envisaged by centralized approaches.

In this paper, we propose a distributed version of the "insertion heuristic". Insertion heuristics is a method which consists in inserting the customers following their revealing order in the routes of the vehicles. The vehicle chosen to insert the considered customer is the one that has to make the minimal detour to visit him. Several multi-agent works in the literature have been proposed to distribute insertion heuristics, but very few propose new measures of the insertion cost of a customer in the route of a vehicle, as an alternative to the traditional measure of its incurred detour. In the present work, we do propose such a new measure, based on a space time representation of the *Vehicle* agents' action zones. The objective is to allow the MAS to self-adapt exhibiting an equilibrated distribution of his Vehicle agents, and to decrease this way the number of vehicles mobilized to serve the customers.

The remainder of this paper is structured as follows. In section 2, we discuss previous proposals for the dynamic VRPTW w.r.t our approach. Section 3 presents the architecture of the MAS that we propose. In the section 4, we detail the space-time representation of the Action Zones of the vehicles and its use as a measure for the insertion decision of the customers. We report our experimental results in section 5 and the extensions envisioned for the model in section 6, before to conclude.

2 Related Work

As we said in the introduction, exact approaches cannot meet operational settings, and upon the relatively small set of benchmarking problems of [26] - 56 problems of 100 Euclidean customers[3] each -, only 45 have a known optimal solution up until today [23]. However, interested readers by optimization approaches can refer to, e.g. [18] for a survey.

In fact, most of the proposed solution methods are heuristic or metaheuristic methods, which provide good results in reasonable times, and which have presented good results with benchmark problems. For instance, large-neighborhood local search [2, 24], iterative local search [17, 16], multi-start local search [21], simulated annealing [4], evolutive strategies [22, 13] and ant colonies [9]. These approaches present the best performances with static problems (where the set of transport requests is known *a priori*). For an extensive survey of the literature for the VRPTW approaches, the reader is invited to refer to, e.g. [12, 5].

Generally speaking, most of the works dealing with the dynamic VRPTW are more or less direct adaptations of static methods. For instance, the large-neighborhood local search is adapted to a dynamic context in [10]. In [15], the authors propose to adapt the genetic algorithms to deal with the dynamic VRPTW.

[3] Euclidean customers have cartesian coordinates, and the distance and the le travel times between each pair of customers are calculated following the Euclidean metric.

The proposed algorithm starts by creating a population of initial solutions and tries continually to improve their quality. When a new customer reveals, he is inserted in all current solutions in the position minimizing the additional cost. Upon the static methods, insertion heuristics are the most widely adapted in a dynamic environment (e.g. [8, 14, 6]). Insertion heuristics are, in their original version, greedy algorithms, in the sense that the decision to insert a given customer in the route of a vehicle is irrevocable. They are also combined with meta-heuristics to improve the quality of the solutions. In [33], the authors propose an approach for the dynamic VRP, in which a central solver made of reactors manage the events coming up in the network. When a customer reveals, he is inserted in the route of a vehicle as for insertion heuristics. After each insertion, an optimization procedure is launched trying to reduce the number of used vehicles and the total traveled distance. The procedure is repeated until the current solution doesn't get better anymore. The customers are handled sequentially following a decreasing priority order, which is function of their respective distance and the decreasing order of their opening time windows.

The advantage of using insertion heuristics is that they are intuitive and fast. However, when they are applied in a dynamic context, their solving process is said to be myopic. Indeed, the system doesn't know which customers will appear once it has assigned the known customers to the vehicles. And even if we could have an optimal assignment and scheduling of the known customers, a new coming customer could make the old assignment sub-optimal, which would - in the worst case - necessitate a whole recomputation of all the routes.

Most of the multiagent approaches for the dynamic VRPTW are grounded, at least partially, on insertion heuristics. In [28] and in [19], the authors propose a multiagent architecture to solve a VRP and a multi-depot VRP for the first and a dial-a-ride problem for the second. The principle is the same: distribute an insertion heuristic, followed by a post-optimization step. In [28], the customers are handled sequentially, broadcasted to all the vehicles, which in turn propose insertion offers and the best proposal is retained by the customer. In the second step, the vehicles exchange customers to improve their solutions, each vehicle knowing the other agents of the system. Since vehicles are running in parallel, the authors envision to apply different heuristics for each vehicle, without changing the architecture. In-Time [19] is a system composed of *Customer* agents and *Vehicle* agents. The *Customer* agent announces himself and all the *Vehicle* agents calculate his insertion cost in their routes. Again, the *Customer* agent selects the cheapest offer. The authors propose a distributed local search method to improve the solutions. Indeed, they allow a customer to ask stochastically to cancel his current assignment and to de reannounce himself to the system, with the objective of having a better deal with another vehicle. MARS [7] models a cooperative scheduling in a maritime shipping company in the form of a multiagent system. The solution to the global scheduling problem emerges from the local decisions. The system uses an extension of the Contract Net Protocol (CNP) [25] and shows that it can be used for having good initial solutions to complex problems of tasks assignment. The MAS profits from an *a*

priori structuring of the agents, since each vehicle is associated with a particular society and can handle the only customers of this society.

From a protocol and an architecture point of view, our system sticks with the systems we have just described, since we propose a distributed version of insertion heuristics. However, in these proposals, none have focused on the re-definition of the insertion cost of a customer. The traditional insertion cost of a customer in the route of a vehicle is based on the incurred detour of the vehicle. We propose a new insertion cost measure, focused on the space-time coverage of the vehicles, which aims at counterbalancing the myopia of the traditional measures, by privileging an insertion process that is future-centered.

3 Multiagent System for the Dynamic VRPTW

Our system is composed of a dynamic set of agents which interact to solve the dynamic VRPTW. A solution consists of a series of vehicles routes, each route consists of a sequence of customers with their associated visit time. We define three categories of agents. *Customer* agents, which represent users of the system (persons or goods), *Vehicle* agents, which represent vehicles in the MAS and *Interface* agents which represent an access point to the system (Web server, GUI, simulator, etc.). When a user logs in the MAS, the data he provides are verified (existing node, valid time windows, etc.) and, if the data are correct, a *Customer* agent representing him and described by the data he provided is created.

In [32], we have designed, implemented and compared three possible architectures to model the dynamic VRPTW: a centralized architecture, a decentralized architecture and a hybrid architecture. We present them in the following paragraphs.

3.1 Centralized Architecture

In this architecture, all the requests are handled by the same "agent". He has all the required information about each vehicle and each customer: the occupancy rate of vehicles, their current positions and the traffic conditions in real time. Having all these information, he assigns to each customer the most appropriate vehicle to serve them, i.e the one having the minimal overcost related to the customer insertion. Figure 1 illustrates this architecture, in which, besides the three agents described above, we add a *Planner* agent which represent the decision-making center, he has in charge the routes computation and vehicles notification of his decision.

The scenario that we have proposed to study is the following: At a given moment, a user interacts with the *Interface* agent, which creates a *Customer* agent to represent him in the system. Once created, the *Customer* agent sends his request to the *Planner* agent which tries to insert him in each vehicle's route, and retains the one with the minimal additional cost. If there is no *Vehicle* agent which can insert the customer, a new vehicle is created. Finally, the *Planner*

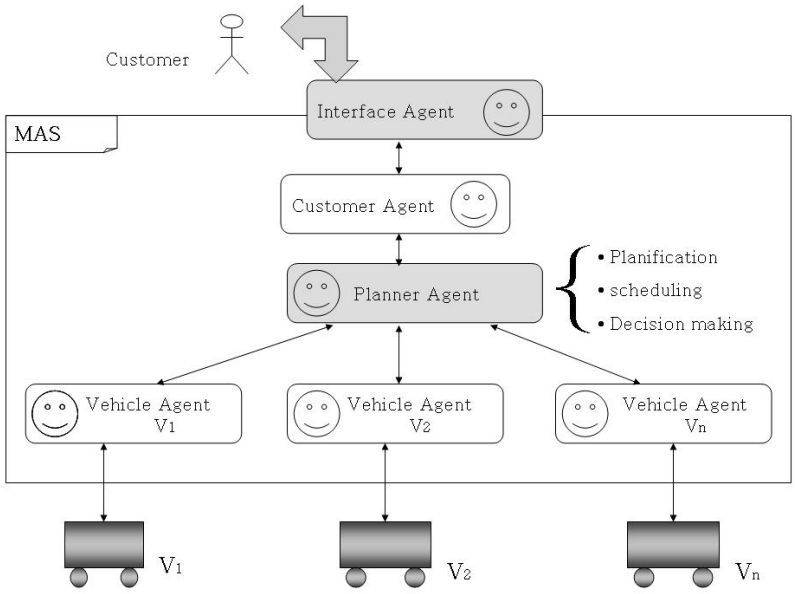

Fig. 1. Centralized Architecture

agent sends the current route to each vehicle and informs the *Customer* agent of his vehicle and his visit times. The *Vehicle* agents don't execute any operation, they merely receive their current route and update their information.

The centralized approach poses several problems. Indeed, sequential treatment of the customer requests slow down the system response time, which goes against the requirement of fast response to dynamic customers. Moreover, the failure of the *Planner* agent leads to a blackout at the global level. Nevertheless, the centralized architecture has the advantage of minimizing communications and information updates by the agents.

3.2 Decentralized Architecture

The decentralized architecture is illustrated in Figure 2. In this architecture, there is no bottleneck for routes computation. Each *Vehicle* agent tries to insert the new customer in his route, informs the other vehicles of the customer's insertion cost, and the vehicle with the minimal additional cost informs the customer and inserts him in his route. At each appearance of a new customer, the *Customer* agent broadcasts his request to all the vehicles in the system. *Vehicle* agents exchange their overcosts via messages. Each *Vehicle* agent compares his own cost with other agents' costs, and stops bidding if the cost that is being offered to him is better than his. Finally, the winner agent (the *Vehicle* agent with the minimal insertion cost) communicates with the *Customer* agent and both

(the *Vehicle agent* and *Customer agent*) update their information. This architecture offers the advantage of a distributed processing and to be fault-tolerant. However, the communication costs explode with this architecture: the number of messages exchanged between *Vehicle* agents is of quadratic complexity.

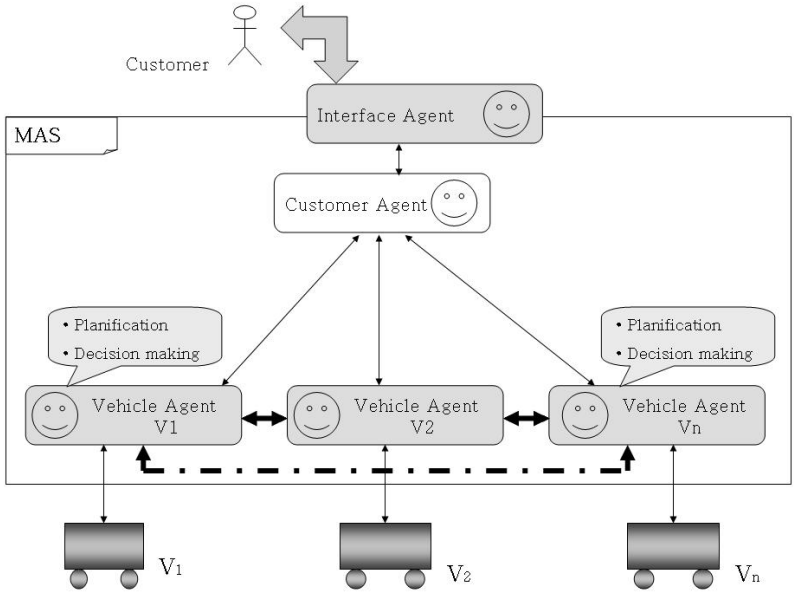

Fig. 2. Decentralized Architecture

3.3 Hybrid Architecture

The hybrid architecture (cf. Figure 3) is a compromise between the centralized and the decentralized approach. A new agent *Dispatcher* is inserted between the *Customer* and *Vehicle* agents and he has the role of dispatching the customer's request, collecting bids from the *Vehicle* agents and choosing the one offering the minimal cost. The process describes a CNP (*Contract Net Protocol*) [25] where, in each occurrence of a *Customer* agent, the *Dispatcher* agent receives a set of proposals and selects those with a minimal cost.

In [31], we have implemented these three architectures, and executed them on a network of four computers. The results show that the hybrid architecture exhibits the best results in terms of the system's answer time to online customers. The decentralized architecture comes in the second position, taking advantage of the computation distribution but suffering from its big consumption of bandwidth. The centralized architecture comes in the last position, since its gain

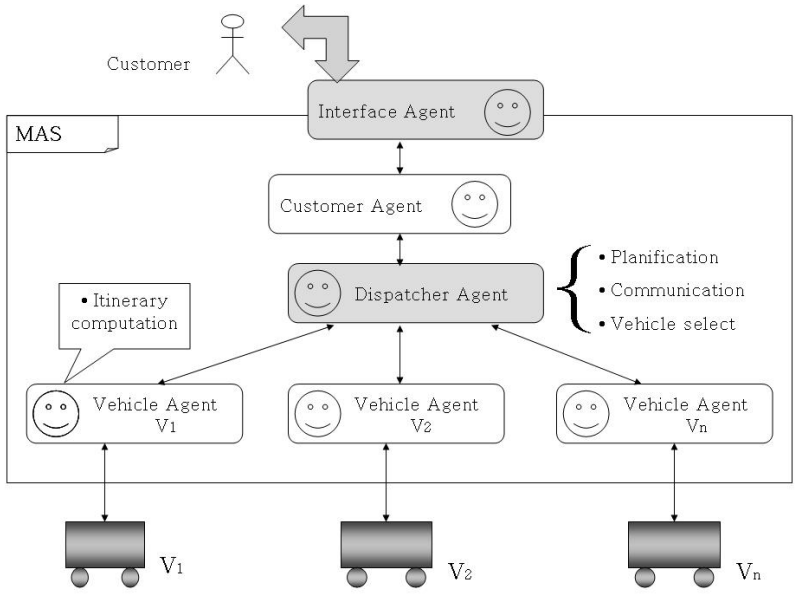

Fig. 3. Hybrid Architecture

in terms of number of exchanged messages does not counterbalance its sequentialization of the processes. For these reasons, we maintain in the remainder of this paper the hybrid architecture as a reference, and we implement the other versions of the MAS (detailed in the rest of this paper) with this architecture.

In the previous description, we use the additional cost related to the customer insertion to make a decision to include the new customer in a vehicle's route. This cost is function of the detour made by the vehicle to integrate the new customer. As an alternative to this traditional measure, we propose in the following section "the variation of *Vehicle* agents' action zones as a cost for customers' insertion".

4 Self-Organization Model

In the heuristics and multiagent methods of the literature, the hierarchical objective of minimizing the number of mobilized vehicles is considered in priority w.r.t the distance traveled by all the vehicles. The vast majority of the literature heuristics are as a consequence based on a two-phase approach: the minimization of the number of vehicles, followed by the minimization of the traveled distance [23]. The self-organization model that we propose in this section has the objective of minimizing the number of used vehicles, while keeping the use of a "pure" insertion heuristics, i.e. without any further improvements.

To this end, our model allows *Vehicle* agents to cover a maximal space-time zone of the transportation network, avoiding this way the mobilization of a new

vehicle if a new customer appears in an uncovered zone. A space-time pair $\langle i, t \rangle$ - with i a node and t a time - is said to be "covered" by a *Vehicle* agent v if v can be in i at t. In the context of the dynamic VRPTW, maximizing the space-time coverage of *Vehicle* agents results in giving the maximum chance to satisfy the demand of a future (unknown) customer. The logic of this measure is different from the traditional measures', which focus on the increase of the traveled distance, neglecting the impact of the current insertion decision on future insertion possibilities.

Following the description of the previous section, the *Dispatcher* agent chooses between several *Vehicle* agents the one with the minimal proposed insertion cost. The systems that are based on this heuristic use generally the measure of Solomon [26] as an insertion cost. This measure consists in inserting the customer which has the minimal impact on the general cost of the vehicle (which is generally function of the vehicle's incurred detour). This measure is simple and the most intuitive but has a serious drawback, since the insertion of the current customer might result in making the insertion of a great number of future customers infeasible, with the current number of vehicles. Its problem is that it generates vehicles' plans that are very constrained in time and space, i.e. plans that offer a few possibilities of insertion between each pair of adjacent planned customers. As a consequence, the appearance of new customers risks to oblige the system to create a new vehicle to serve them. Through the modeling of *Vehicle* agents' Action Zones, we propose a new way to compute the customer's insertion cost in the route of a vehicle, and a new choice criterion between vehicles for the insertion of a given customer. We propose a computation which objective is to choose, provided a newcomer customer, the *Vehicle* agent "which decrease in the probability to participate in future insertions is minimal". We use that variation of *Vehicle* agents' Action Zone as an insertion cost for the insertion of a given customer in its route.

4.1 Environment Modeling

The space-time Action Zone of a *Vehicle* agent is composed of a subset of the network nodes, together with the times that are associated to them. We model the MAS environment in the form of a space-time network, inferred from the network graph. Each node of the graph becomes a pair $\langle space, time \rangle$, which represents the "state" of the node in a discrete time period. The space-time network is composed of several subgraphs, where each subgraph is a copy of the static graph, and corresponds to the state of the graph in a certain period of time (cf. Figure 4). We index the nodes of a subgraph as follows: $\langle 0, t \rangle, \ldots, \langle N, t \rangle$, with $t \in \{1, ..., h\}$, with $0, \ldots, N$ are the nodes of the network and h the number of considered discrete periods. The total number of nodes in the space-time network is equal to $h \times N$. The edges linking the nodes of a subgraph are those of the static graph, and the costs are the travel times as described in the introduction (t_{ij}).

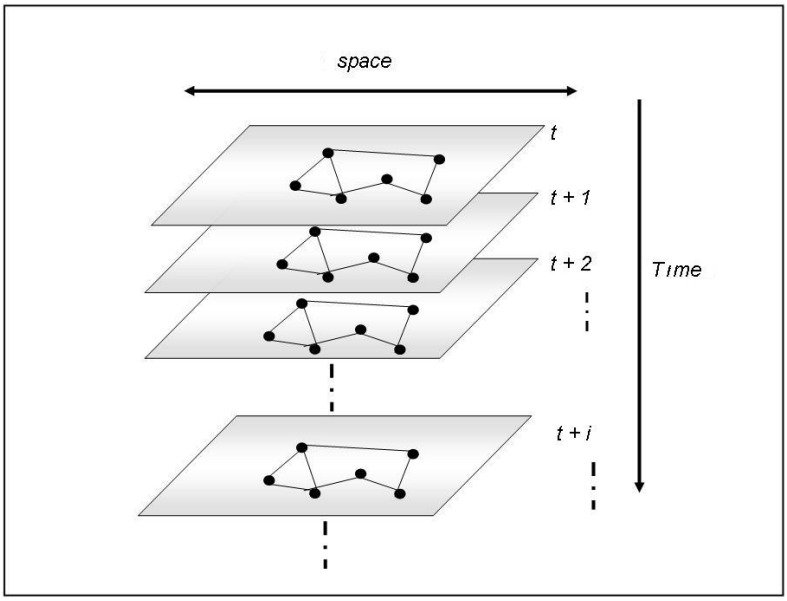

Fig. 4. Space-Time Network

4.2 Intuition of the Action Zones

Consider a *Vehicle* agent v that has an empty route. In order for this agent to be able to insert a new customer c - described by: n a node, $[e, l]$ a time window, s a service time, and q a quantity - l has to be big enough to allow v to be in n without violating his time constraints. More precisely, the current time t, plus the travel time between the depot and n has to be less or equal to l (cf. Figure 5). Starting from this observation, we define the Action Zone of a *Vehicle* agent as the potential customers that satisfy this constraint. To do so, we define the Action Zone of a *Vehicle* agent as the set of pairs $\langle n, t \rangle$ of the space-time network that remain valid given his current route (n can be visited by the vehicle at t). The Action Zone of a *Vehicle* agent with an empty route is illustrated by the triangular shadow in the Figure 6 (it is actually a conic shadow in a three-dimensional space).

When a *Vehicle* agent inserts a customer in his route, his Action Zone is recomputed, since some $\langle node, time \rangle$ pairs become not valid because of his insertion. In the Figure 7, a new customer is inserted in the route of the vehicle. The Action Zone of the *Vehicle* agent after inserting the customer is represented by the interior of the contour of the bold lines, which represent the space-time nodes which remain accessible after the insertion of the customer (the computation of the new Action Zone is explained later).

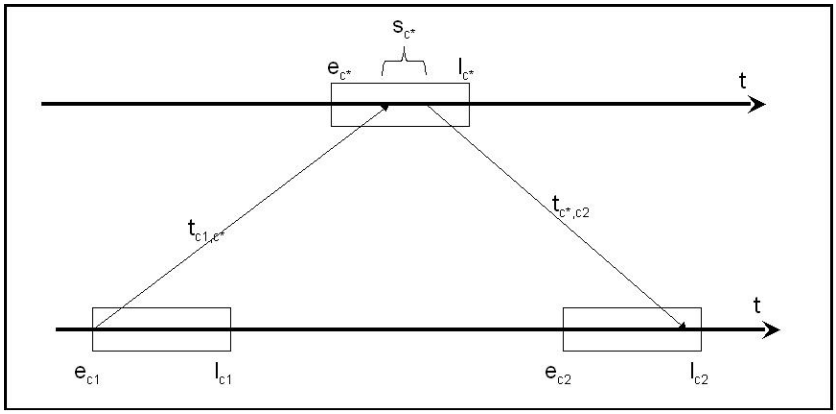

Fig. 5. Feasible insertion

The associated cost to an offer from a *Vehicle* agent v for the insertion of a *Customer* agent c corresponds to the hypothetical decrease of the Action Zone of v following the insertion of c in his route.

The idea is that the chosen *Vehicle* for the insertion of a customer is the one that looses the minimal chance to be candidate for the insertion of future customers. Thus, the criterion that is maximized by the society of *Vehicle* agents is the sum of their Action Zones, i.e. the capacity that the MAS has to react to the appearance of *Customer* agents, without mobilizing new vehicles.

To illustrate the Action Zones and their dynamics, we present the version of the measure that is related to an Euclidean problem, i.e. where travel times are computed following the Euclidean metric. The following paragraphs detail the measure as well as its dynamics.

4.3 The Computation of Action Zones

In the Euclidean case, the transportation network is a plane, and the travel times between two points i (described by (x_i, y_i)) and j (described by (x_j, y_j)) is equal to

$$\sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}$$

Therefore, if a vehicle is in i at the moment t, he cannot be in j earlier than $t_i + \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}$.

We can compute at any time, from the current position of a vehicle, the set of triples (x, y, t) where he can be in the future. Indeed, considering a plane with an X-axis in $[x_{min}, x_{max}]$ and a Y-axis in $[y_{min}, y_{max}]$, the set of space-time positions is the set of points in the cube delimited by $[x_{min}, x_{max}]$,$[y_{min}, y_{max}]$ and $[e_0, l_0]$ (recall that e_0 and l_0 are the scheduling horizon and are the minimal and maximal values for the time windows). Consider a vehicle in the depot

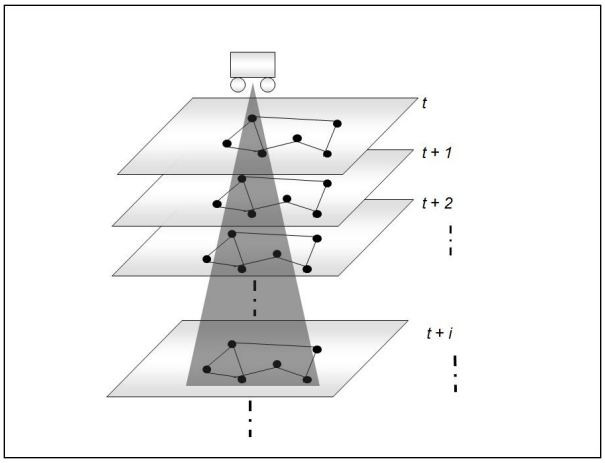

Fig. 6. Initial Space-Time Action Zone

(x_0, y_0) at t_0. The set of points (x, y, t) that are accessible by this vehicle are described by the following inequality:

$$\sqrt{(x - x_0)^2 + (y - y_0)^2} \leq (t - t_0)$$

The (x, y, t) satisfying this inequality are those that are positioned inside the cone $\mathcal{C}$ of vertex (x_0, y_0, t_0) and with the equation $\sqrt{(x - x_0)^2 + (y - y_0)^2} = (t - t_0)$ (c.f Figure 8). This cone represents the Action Zone of a *Vehicle* agent, with an empty route, in the Euclidean case. It represents all the possible space-time positions that this *Vehicle* agent is able to have in the future.

We use the Action Zone of the *Vehicle* agents when a *Customer* agent has to choose between several *Vehicle* agents for his insertion. We have to be able to compare the Action Zones of different *Vehicle* agents. To do so, we propose to quantify it, by computing the volume of the cone $\mathcal{C}$ representing the future possible positions of the vehicle:

$$Volume(\mathcal{C}) = \frac{1}{3} \times \pi \times (l_0 - e_0)^3$$

This is the quantification of the initial Action Zone of any new *Vehicle* agent joining the MAS. When a new *Customer* agent appears, a *Vehicle* agent computes his new Action Zone, the cost that he proposes to the *Dispatcher* agent is the difference between his old Action Zone and his new one. The new Action Zone computation is detailed in the following paragraph.

4.4 Dynamics of the Action Zones

Consider a customer c_2 (of coordinates (x_2, y_2) and with a time window $[e_2, l_2]$) that joins the system, and suppose that v is temporarily the only available

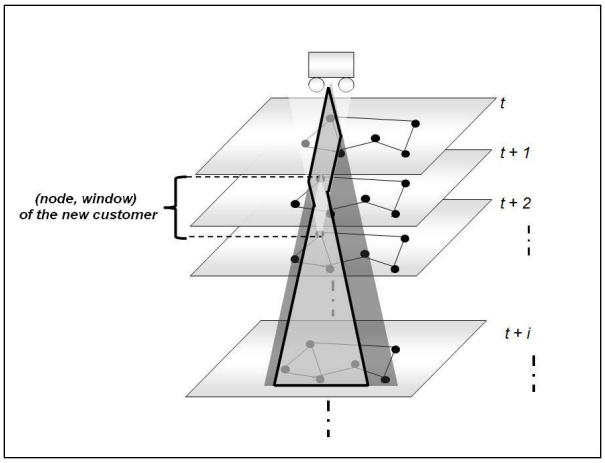

Fig. 7. Action Zone after the Insertion of a Customer

Vehicle agent of the system and has an empty route. The agent v has to deduce his new space-time action zone, i.e. the space-time nodes that he can still reach without violating the time constraints of c_2. The new action zone answers the following questions: "if v had to be in (x_2, y_2) at l_2, where would he have been before? And if he had to be there at e_2 where would he be after $e_2 + s_2$?". The triples (x, y, t) where the *Vehicle* agent can be before visiting c_2 are described by the inequality $[a]$, and the triples (x, y, t) where he can be after visiting c_2 are describe by the inequality $[b]$.

$$\sqrt{(x - x_2)^2 + (y - y_2)^2} \leq (l_2 - (t)) \quad [a]$$

$$\sqrt{(x - x_2)^2 + (y - y_2)^2} \leq (t - (e_2 + s_2)) \quad [b]$$

The new Action Zone is illustrated by the Figure 9: the new measure consists in the intersection of the initial cone $\mathcal{C}$ with the union of the two new cones described by the inequalities $[a]$ and $[b]$ (denoted respectively by $\mathcal{C}_1$ and $\mathcal{C}_2$). The new measure of the Action Zone is equal to the volume of the intersection of $\mathcal{C}$ with the union of $\mathcal{C}_1$ and $\mathcal{C}_2$. The complete computation of the volume of the intersection of these two cones is reported in the Appendix A of [30].

The cost of the insertion of a customer in the route of a vehicle is equal to the measure associated with the old Action Zone of the vehicle minus the measure of the new Action Zone, after the insertion of the customer. The quantity measured represents the space-time positions that the vehicle cannot have anymore, if he had to insert this customer in his route. The retained *Vehicle* agent to visit a given customer is the one for which the insertion of the customer causes less loss in his space-time Action Zone. This corresponds to choosing the vehicle that looses the minimal possibilities to be candidate for future customers.

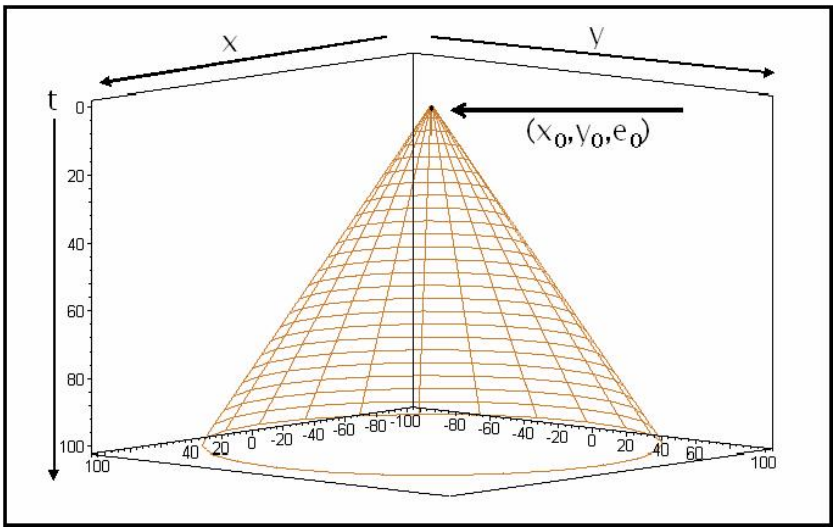

Fig. 8. Initial Action Zone

5 Results

There exists a validation problem of the different approaches for solving the Vehicle Routing Problems. Indeed, the different heuristics, to be comparable, have to be executed with the same data. This problem has been partially solved thanks to the work of Marius M. Solomon [26], who has created a set of different problems for the VRPTW. It is now admitted that these problems are challenging and diverse enough to compare with enough confidence the different proposed methods. A proof for that claim is that there is no unique heuristic that provides the best results for each one of these problems at the same time. In Solomon's benchmarks, six different sets of problems have been defined: C1, C2, R1, R2, RC1 and RC2. The customers are geographically uniformly distributed in the problems of type R, clustered in the problems of type C, and a mix of customers uniformly distributed and clustered is used in the problems of type RC. The problems of type 1 have narrow time windows (very few customers can coexist in the same vehicle's route) and the problems of type 2 have wide time windows. Finally, a constant service time is associated with each customer, which is equal to 10 in the problems of type R and RC, and to 90 in the problems of type C.

The validation problem remains for the dynamic case, since there is no benchmark for the dynamic problem. In our case, where the dynamicity of the problem comes from our ignorance of future customers, we could have several configurations. Indeed, we can have more or less customers known in advance. We can also define different cadencies for customers appearence. In the literature, the authors have made different choices. Some of them start from Solomon problems

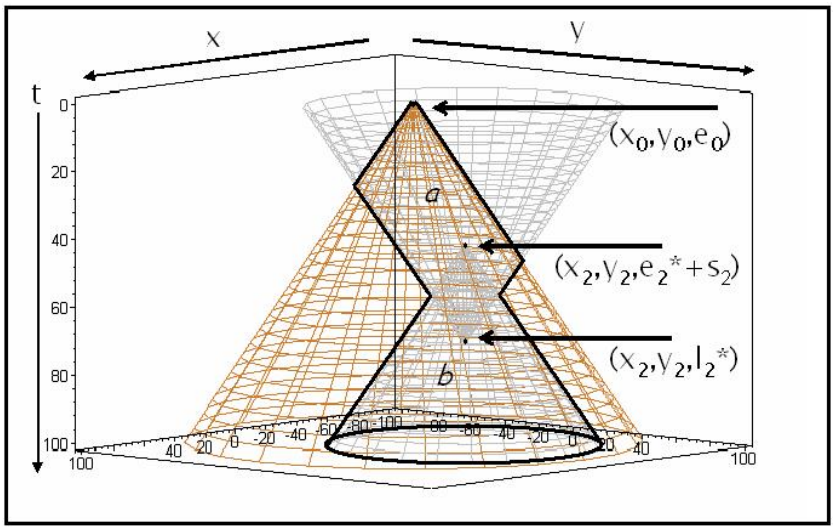

Fig. 9. Space-Time Action Zone after the insertion of c_2

and introduce the dynamicity by hiding a part of the customers and by diffusing them to the system one by one (e.g. [3]). Others consider new problems created from real data, while the dynamicity parameters are those of the real problem (e.g. [6]). The problem is that, if the data of the problem, together with the systems parameters, are not rigourously the same with the system one wants to compare his proposal with, the results can simply not be compared. And since all the systems parameters are generally not published, it becomes really hard to compare a new proposal with the existing ones.

We have made the choice not to have any known customer before the system starts. Besides, we don't possess any real data that could inform us about the cadency of customers appearance. We choose to use Solomon benchmarks, while following the modification proposed by [11] to make the problem dynamic. Indeed, Gendreau *et al.* [11] propose to modify Solomon benchmarks to create dynamic problems. To this end, let $[0, T]$ the simulation time. All the date related to the time (time windows, service times and travel times) are multiplied by $\frac{T}{l_0 - e_0}$, with $[e_0, l_0]$ the scheduling horizon of the problem. The authors divide the customers set in two subsets, the first defines the customers known in advance, while the second defines dynamic customers. We don't have this distinction since all our customers are dynamic. For all our customers, an occurrence time is associated, defining the moment when the customer is known by the system. Given a customer i, its occurrence time is generated randomly between $[0, \overline{e}_i]$, with:

$$\overline{e}_i = e_i \times \frac{T}{l_0 - e_0}$$

We have implemented two MAS with an identic behavior, following the hybrid architecture. The only difference concerns the measure used by *Vehicle* agents to compute the insertion cost of a customer. For the first implemented MAS, it relies on the Solomon measure (noted Δ Distance) and on the Action Zones measure for the second (noted Δ Action Zone). We have chosen to run our experiments with the problems of class R and C, of type 1, which are the instances that are very constrained in time (narrow time windows). Recall that the primary objective is to minimize the total distance traveled by the vehicles mobilized by the system. Table 1 reports the results with the files of class R1 where we consider successively 25, 50 and 100 customers, while Table 2 reports the results with files from class C1 with 25, 50, 100 and 200 customers. The results show, with the two classes of problems, that the use of our measure mobilizes less vehicles in average than the traditional measure, whatever the number of considered customers ($6.3 < 6.4; 10.6 < 10.7; 18.8 < 19.1$). These results are verified with all the considered problem files, but only one (where the traditional measure behaves better than ours, relatively to the number of used vehicles), the file 5 with 100 customers of class C1. These results validate the intuition of the measure that consists of maximizing the future insertion possibilities for a *Vehicle* agent.

However, since our measure focuses exclusively on the insertion possibilities, the total distance traveled by the vehicles is superior to the distance induced by the traditional measure. We believe that a compromise between the two measures, e.g. a weighted sum of the increase of the distance and the Action Zones loss would be able to produce better results. The fact remains that our results dominate those of the traditional heuristic, given the hierarchical objective of minimizing the number of used vehicles in priority.

Δ Distance		
	Average number of vehicles	Average Distance
25	6.4	637.1
50	10.7	1203.7
100	19.1	1968.4
Δ Action Zones		
	Average number of vehicles	Average Distance
25	6.3	679.3
50	10.6	1286.7
100	18.8	2149.7

Table 1. Experimental Results for class R1 with 25, 50 and 100 customers

Δ Distance		
	Average number of vehicles	Average Distance
25	3.4	316.6
50	6	671.2
100	12.1	1601.3
200	21.6	6315.5
Δ Action Zones		
	Average number of vehicles	Average Distance
25	3.3	347.9
50	5.9	731.5
100	11.9	1774.4
200	21.4	6979.8

Table 2. Experimental Results for class C1 with 25, 50, 100 and 200 customers

6 Extensions

With the mechanism described in this paper, the optimized criterion by *Vehicle* agents is the size of their own Action Zones. Thus, the measure of Action Zones that we propose allows the MAS to maximize the sum of the space-time Action Zones of the *Vehicle* agents of the system. However, a more interesting measure would be to maximize the union of these action zones, instead of their sum. More precisely, the fact that a vehicle looses space-time nodes that he is the only one to cover, should be more costly than a vehicle loosing nodes that others do cover. To this end, to each node of the space-time network, we associate a list of the vehicles that are covering it. With each creation of a new *Vehicle* agent, the set of space-time nodes that are part of his Action Zone is calculated as described previously. The vehicle then proceeds with the notification of these nodes that they are part of his zone. In turn, each node maintains a list of the vehicles covering it, and when it's notified by a vehicle, it updates this list. Similarly, when the Action Zones of a *Vehicle* agent looses a node, the latter is also notified, and its vehicles' list updated.

Now, when a *Vehicle* agent has to calculate the cost of inserting a customer, he starts by calculating the space-time nodes that he would loose as described all along this paper. After that, he interrogates each of these nodes about "the price he has to pay" for not covering them. This price is inversely proportional to the size of the vehicles' list of each node. More precisely, the price for each node is equal to

$$\frac{1}{card(v_{\langle n,t\rangle})}$$

with $v_{\langle n,t\rangle}$ denoting the *Vehicle* agents that cover the space-time node $\langle n,t\rangle$. The cost proposed to the customer is the sum of these "prices" provided by the space-time nodes od the *Vehicle* agent. This way, the MAS environment being the only one knowing the Action Zones of all the agents, it associates more or

less penalty to the decisions of not covering the network over time. The criterion henceforth optimized by the society of agents is not the minimization of the Action Zones sum anymore, but their union.

In addition, the model we have proposed in this paper is easily extensible to a stochastic problem. Stochastic VRP have data about the history of transport demands, and can have more or less accurate predictions about future demands. Our model offers the advantage of being able to integrate these predictions in the solving process in a natural way. Indeed, to integrate these prediction, we follow the same process as described in the two preceding paragraphs. The only difference is that the price a *Vehicle* agent has to pay for loosing a space-time node would become

$$\frac{\tau_{\langle n,t\rangle}}{card(v_{\langle n,t\rangle})}$$

with $\tau_{\langle n,t\rangle}$ a parameter providing the prediction of demands on the node n at moment t. We orient this way the fleet of vehicles towards the highly dense zones, without changing the model nor the interaction protocol.

7 Conclusion

In this paper, we have proposed an agent-oriented self-organization model for the dynamic VRPTW based on the agents' action zones, based on a MAS designed following a hybrid architecture. The action zones of the *Vehicle* agents reflect their space-time coverage of the environment. We use the variation of these Action Zones as a new metric between *Vehicle* agent to reduce the myopic behavior of traditional metrics. By optimizing the space-time coverage of the environment by the *Vehicle* agents, our model allows the MAS to self-adapt by exhibiting an equilibrated space-time distribution of the vehicles, and to lessen this way the number of vehicles mobilized to serve the customers.

References

1. J. A. Adams. Multiagent Systems: A modern approach to distributed artificial intelligence. *AI Magazine*, 22(2):105–108, 2001.
2. R. Bent and P. Hentenryck. A two-stage hybrid local search for the vehicle routing problem with time windows. *Transportation Science*, 38(4):515530, 2004.
3. R. W. Bent and P. V. Hentenryck. Scenario-based planning for partially dynamic vehicle routing problems with stochastic customers. *Operations Research*, 52(6):977–987, 2004.
4. Z. J. Czech and P. Czarnas. A parallel simulated annealing for the vehicle routing problem with time windows. In *Proceedings of the 10th Euromicro Workshop on Parallel, Distributed and Network-based Processing*, pages 376–383, Canary Islands (Spain), 2002.
5. G. Desaulniers, J. Desrosiers, M. Solomon, F. Soumis, and J.-F. Cordeau. The VRP with time windows. In D. Vigo and P. Toth, editors, *The Vehicle Routing Problem*, SIAM Monographs on Discrete Mathematics and Applications, pages 157–193. SIAM, 2002.

6. M. Diana. The importance of information flows temporal attributes for the efficient scheduling of dynamic demand responsive transport services. *Journal of advanced Transportation*, 40(1):23–46, 2006.
7. K. Fischer, J. Muller, M. Pischel, and D. Schier. A model for cooperative transportation scheduling. In V. R. Lesser and L. Gasser, editors, *Proceedings of the First International Conference on Multiagent Systems (ICMAS'95)*, pages 109–116, Menlo park, CA (USA), 1995. AAAI Press / MIT Press.
8. L. Fu and S. Teply. On-line and off-line routing and scheduling of dial-a-ride paratransit vehicles. In *Computer-Aided Civil and Infrastructure Engineering*, volume 14, pages 309–319. Blackwell Publishers, Oxford (UK), 1999.
9. L. M. Gambardella, E. D. Taillard, and G. Agazzi. MACS-VRPTW: A multiple ant colony system for vehicle routing problems with time windows. In M. D. D. Corne and F. Glover, editors, *New Ideas in Optimization*, pages 63–76, McGraw-Hill (London), 1999.
10. M. Gendreau, F. Guertin, J.-Y. Potvin, and R. Sguin. Neighborhood search heuristics for a dynamic vehicle dispatching problem with pick-ups and deliveries. *Transportation Research Part C*, 14:157–174, 2006.
11. M. Gendreau, F. Guertin, J.-Y. Potvin, and E. D. Taillard. Parallel tabu search for real-time vehicle routing and dispatching. *Transportation Science*, 33(4):381–390, 1999.
12. B. Golden, S. Raghavan, and E. Wasil. *The vehicle routing problem, latest advances and new challenges*, volume 43 of *Operations research/computer science interfaces*. Springer Verlag, 2008.
13. J. Homberger and H. Gehring. Two-phase hybrid metaheuristic for the vehicle routing problem with time windows. *European Journal of Operational Research*, 162:220238, 2005.
14. M. E. Horn. Fleet scheduling and dispatching for demand-responsive passenger services. *Transportation Research C*, 10(1):35–63, 2002.
15. H. Housroum, T. Hsu, R. Dupas, and G. Goncalves. A hybrid ga approach for solving the dynamic vehicle routing problem with time windows. In I. C. Society, editor, *Proceedings of the IEEE Conference on Information and Communication Technologies: from Theory to Applications*, pages 3347–3352, Damascus, Syria, 2006.
16. T. Ibaraki, S. Imahori, K. Nonobe, K. Sobue, T. Uno, and M. Yagiura. An iterated local search algorithm for the vehicle routing problem with convex time penalty functions. *Discrete Applied Mathematics*, 156(11):20502069, 2008.
17. T. Ibaraki, M. Kubo, T. Masuda, T. Uno, and M. Yagiura. Effective local search algorithms for the vehicle routing problem with general time window constraints. *Transportation Science*, 39(2):206232, 2005.
18. M. Jepsen, B. Petersen, S. Spoorendonk, and D. Pisinger. Subset-row inequalities applied to the vehicle-routing problem with time windows. *Operations Research*, 56(2):497–511, 2008.
19. R. Kohout and K. Erol. In-Time agent-based vehicle routing with a stochastic improvement heuristic. In *Proceedings of the sixteenth national conference on Artificial intelligence and the eleventh Innovative applications of artificial intelligence (AAAI'99/IAAI'99)*, pages 864–869, Menlo Park, CA (USA), 1999. AAAI Press.
20. A. Larsen. *The Dynamic Vehicle Routing Problem*. PhD thesis, University of Denmark, 2000.
21. A. Lim and X. Zhang. A two-stage heuristic with ejection pools and generalized ejection chains for the vehicle routing problem with time windows. *INFORMS Journal on Computing*, 19(3):44357, 2007.

22. D. Mester and O. Brysy. Active guided evolution strategies for large scale vehicle routing problems with time windows. *Computers & Operations Research*, 32(6):15931614, 2005.
23. Y. Nagata, O. Bräysy, and W. Dullaert. A penalty-based edge assembly memetic algorithm for the vehicle routing problem with time windows. *Computers & Operations Research*, xx(x), July 2009.
24. D. Pisinger and S. Ropke. A general heuristic for vehicle routing problems. *Computers & Operations Research*, 34(8):24032435, 2007.
25. R. G. Smith. The contract net protocol: High-level communication and control in a distributed problem solver. *IEEE Trans. on Comp.*, C-29(12):1104–1113, December 1980.
26. M. Solomon. Algorithms for the vehicle routing and scheduling with time window constraints. *Operations Research*, 15:254–265, 1987.
27. K. P. Sycara. Multiagent Systems. *AI Magazine*, 19(2):79–92, 1998.
28. S. R. Thangiah, O. Shmygelska, and W. Mennell. An agent architecture for vehicle routing problems. In *Proceedings of the 2001 ACM symposium on Applied computing (SAC '01)*, pages 517–521, New York, NY (USA), 2001. ACM Press.
29. M. Wooldridge and N. R. Jennings. Intelligent agents: Theory and practice. *Knowledge Engineering Review*, 10(2):115–152, 1995.
30. M. Zargayouna. *Modèle et langage de coordination pour les systèmes multi-agents ouverts. Application au problème du transport à la demande.* Phd Thesis, University of Paris-Dauphine, Paris (France), 2007. In french.
31. B. Zeddini. *Modèles d'auto-organisation multi-agent pour le transport à la demande.* Phd Thesis, Le Havre University, Le Havre (France), 2009. In french, to appear.
32. B. Zeddini, A. Yassine, M. Temani, and K. Ghédira. Collective intelligence for demand-responsive transportation systems: a self organization model. In *Proceedings of the 8th international conference on New technologies in distributed systems (NOTERE'08)*, pages 1–8, New York, NY (USA), 2008. ACM Press.
33. K. Q. Zhu and K.-L. Ong. A reactive method for real time dynamic vehicle routing problem. In *12th IEEE International Conference on Tools with Artificial Intelligence (ICTAI'00)*, pages 176–179, Vancouver (Canada), 2000. IEEE Computer Society.