A Decentralised Approach for the Transportation On Demand Problem

Cyrille Bertelle, Michel Nabaa, Damien Olivier, and Pierrick Tranouez

LITIS, 25 rue Phillipe Lebon, 76058 Le Havre - www.litislab.eu
{Cyrille.Bertelle, Michel.Nabaa, Damien.Olivier,
Pierrick.Tranouez}@univ-lehavre.fr

Summary. Public transport systems are generally organized in a static, a priori way. In such systems, the demand must be adapted to the offer. In this paper, we propose a model based on self-organization in order to dispatch a fleet of vehicles in a purely dynamic Transportation On Demand system (TOD). Our proposal consists in a decentralized approach and a multi-agent system (MAS) to model the environment. This will tackle the problem of vehicles over-concentration or the lack of service in certain areas of the city. We demonstrate that our model addresses these problems by providing vehicle agents, for a given request, to make the final decision thanks to a negotiation process and to calculate overcosts according to an original insertion heuristic.

Key words: transportation on demand, vehicle routing problem, collective intelligence, self-organization.

1 Introduction

Growing environmental concerns are linked to the difficulties of management of urban traffic. They lead to the creation of new solutions improving mobility in agglomeration. Current public transportation systems are determinist and based on frequencies and routes fixed in advance. They are built starting from opportunity studies and are not very adaptive to a request that can change in time: the demand has to adapt to the offer. It is thus advisable to complete the urban transportation services by flexible systems being more adapted to the individual needs. We propose a Transportation On Demand (TOD) system which must adapt to users need in real time. It must allow to generate an important reduction in the traffic and to offer a maximal quality of service to reduce the cost of exploitation. Lastly, it will be the basis for a decision support system, computing vehicle tours in real time, a service which is not offered by the traditional transportation systems. The stake of this article is to study the possibility of the installation of a TOD system to satisfy the

requests of the customers at any moment, by distributing the load inside the fleet of vehicles in order to achieve the goals mentioned previously. This system will adjust dynamically to the customers demand. The scenario of the execution starts with the first customer request which appears randomly in a place of the city. It sends a request indicating his departure point and his destination. The resolution consists in choosing the best located vehicle to satisfy the passengers already on board this vehicle as well as the new request by optimizing its rate of filling with respect to the maximal capacity, its time and cost of travel.

First of all, we will present some previous work of similar or neighbouring problems. Then, we will define the data of our problem. Finally we will present our approach and the preliminary results related to the initial tests to finish by a conclusion and some perspectives.

2 Previous Works

The general problem of the construction of vehicles routes is known under the name of Vehicle Routing Problem (VRP) and represents a combinatorial problem of multi-objective optimization which was the subject of many works and many alternatives in the literature. It belongs to the NP-hard category [2, 10]. In its basic version, the VRP problem (see figure 1) models a well known transportation problem which consists of pickup (and/or collect) products to serve a set of customers using a fleet of vehicles. The resolution consists in determining a set of routes which minimizes objectives as well as possible as the total traveled distance, the number of vehicles used and the sum of customers delays [8].

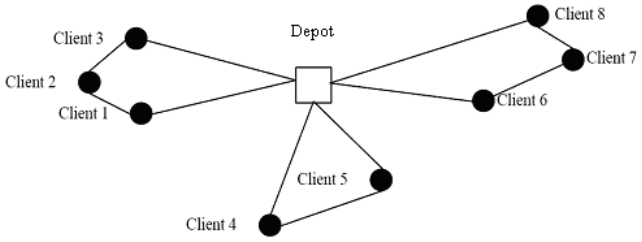

Fig. 1. Vehicle Routing Problem

A complete state of the art of the VRP problems in the static context, and in particular, the dynamic one and their applications is given by [9]. In [13], the Dynamic VRP (DVRP) problem was treated. The resolution consists in dividing time execution into slices. The DVRP problem is considered

as a succession of VRP static problems and an ant colony algorithm is used to solve these static problems. When it is about the problem of VRP with pickup and delivery of goods, one speaks about PDP (Pickup & Delivery Problem). The Dynamic PDP problem was studied by [6]. The authors adopted a multi agent approach. The conversation between agents was based on the Contract Net Protocol. The clients demand arrival was calculated with basis on Poisson distribution. A Dial-A-Ride Problem (DARP) is an extension of the PDP in which the transportation of goods is replaced by the transportation of people [4, 7]. Since we talk about people transportation, the DARP focuses better than the PDP on the satisfaction of these people. We can distinguish between a static or dynamic version of the DARP although the difference is not always strict. Indeed, in the static case, some reservations can be canceled at the last minute, which implies a degree of dynamism while in the dynamic case, most reservations are known a priori before planning [1]. The problem is then, in general, treated as a succession of static sub-problems [3]. An application in the urban transport related to the bus on demand was developped in [12]: a customer can give a time window in which he wishes to be served, instead of departure or arrival, but not both at the same time. Some requests are known in advance and other ones can arrive during the execution. The authors adopted a solution based on an insertion heuristic which gave interesting results with short execution time . This problem is known as the Dynamic Dial-A-Ride Problem (DDARP) with several places of pickup and delivery. In [11], the DARP problem was treated online by considering a homogeneous fleet of vehicles with unit capacities, i.e. a vehicle having a passenger on board, cannot serve another one except if it reached the first passenger destination. To our knowledge, the dynamic DARP is rarely studied and does not exist in its purely dynamic version. The problem we deal with in this paper is original, because the requests are dealt with in real time: we dispose of a purely dynamic case. This dynamicity is due to the fact that no reservation is known in advance. Moreover, traditional techniques suppose to have a control central of traffic which knows vehicles positions and their planned routes (the central receives customers requests, calculates new vehicles routes and orders vehicles to service customers). Moreover, these techniques suppose a perfect knowledge by the central of the states of the vehicles (including breaks, breakdowns, communication problems) in real time, which is not realistic and can involve expensive calculations in time. That's why we adopt a decentralized approach (except for the new client request reception) to face this kind of problems; since embarked system is now standardized.

3 Proposed dynamic model

3.1 Problem Description

The model tries to arbitrate between different constraints.

Each user wishes:

- To minimize the waiting time once his demand is accepted,
- To reach his destination, respecting his desired deadline.

Each vehicle tries:

- To maximize its rate of filling by changing an already planned route to service a new request,
- To deal with the evolution of the traffic load and especially unexpected events (accidents, a road becomes blocked up, new roads) and historical events: the system must be adapted to learn about repetitive events to predict similar ones in the future,
- To negotiate with the other vehicles in order to choose the best proposal to serve each new request.

The system tends, as well as possible, to pair users and vehicles by arbitrating and adjusting the previous constraints. The system is not centralized but emerges from the fleet of vehicles.

We propose an agent-oriented approach. The system is composed of the following agents: $Vehicle$, $Interface$ and $Client$. The scenario of the execution is described as follows: a user connects to the system via a given support (service call, Web server.), it is then instantiated by a *Client agent* whose function consists in representing it in the system. The user indicates his departure point and his destination as indicated previously. Thus, the *Client agent* enters in interaction with the $Interface$ *agent* (see figure 2).

The latter broadcasts the request of the user to other $Vehicle$ *agents* located in an environment which is modelled in the following section. Our model is specific by the fact that the requests are not dealt with batch but in "real time".

3.2 Environment modeling

We represent the urban network by a directed dynamic graph $G(t) = (V(t); E(t))$ where $V(t)$ is the set of nodes and $E(t)$ the set of arcs:

- The nodes represent interesting places of the network: crossroads, stations, cinemas, commercial centers ...
- The arcs represent the roads taken by the vehicles,
- The weight on each arc represents the needed time to cross this arc, depending on the current load of the traffic,
- Dynamic aspect relates to the weights of the arcs, which can evolve in time, according to the evolution of the fluidity of circulation. It can be related to the apparition and/or disappearance of arcs.

Customers are associated to a node or vehicles, which are themselves on nodes or arcs. The size of the population of users and vehicles is variable in time, to obtain a day/night simulation mechanism for example. Once a temporal

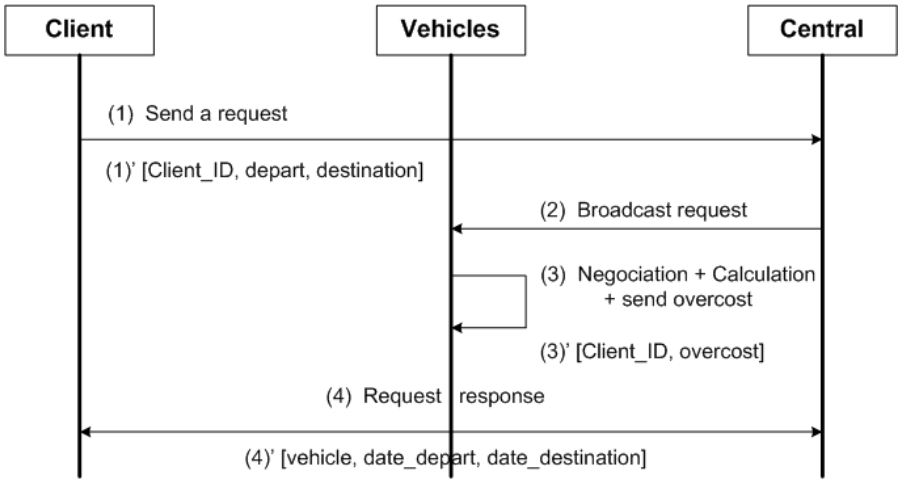

Fig. 2. UML Sequence Diagram showing interactions between the actors of the system

model of the population of users is established, the population of vehicles must be established accordingly, in order to have a satisfying average rate of occupation for vehicles.

3.3 Offers and dynamic choices

Our model is based on two simultaneous phases, an offer phase and a choice phase. We want to establish an agreement between the proposals for a transport and the needs of the customer. This is done according to a mechanism of negotiation. A key element of the system is pairing vehicles and customers. Which vehicle is the best for servicing a given request? Who determines it and how? How a vehicle knows if it has been selected to service a new demand? These questions are not independent. The best vehicle corresponding to each user will be selected; it must minimize the additional effort $\triangle C$ to service the customers. To know this additional effort, a vehicle calculates, on the one hand, the total cost (in time) of its current route, that of the route to discharge the current passengers and charge already planned ones. On the other hand, a vehicle calculates also the cost of the new route to service actual passengers by including the new one. The difference between the two costs is the additional effort or *overcost*.

Each customer request is diffused to all vehicles. When receiving a new request, each vehicle calculates its overcost to service the request and diffuses it to all the fleet. Then, it compares the received answers to its overcost. It orders the received offers and broadcasts the head of list. Finally, the determined winner is the one being the most times ranked first in the received answers. Ideally, one could exempt these last phases: if the diffusion is per-

fect, all the fleet will obtain the correct classification directly. But, because of non perfect diffusion, we proceed to this additional phase after a possible problem (a vehicle crossing a tunnel for example), and this to be sure that all the vehicles agree on the winner vehicle wich will take the passenger.

3.4 The scheduling algorithm

The *vehicle agents* carry out the principal work of planning, and this thanks to the evaluation of the insertion of a travel (source and destination) in such a way to respect the deadlines of the existing passengers in the vehicles. The insertion heuristic is inspired from the ADARTW one [5]. For each vehicle, a scheduling block always starts with the first customer on its way and ends when the last customer reaches its destination. The following figure illustrates the insertion of a customer in a scheduling block of a vehicle having two customers on board ($C1$; $C2$) and going to servicing another client ($C3$) with an already planned route. Each one has a departure point (preceded by a + in figure 3) and a destination (preceded by -). In a block related to a vehicle already containing N clients, corresponds $K = 2N$ stops (2 stops per client) and $(K+1)(K+2)/2$ insertion possibilities when its pickup point must precede its delivery point.

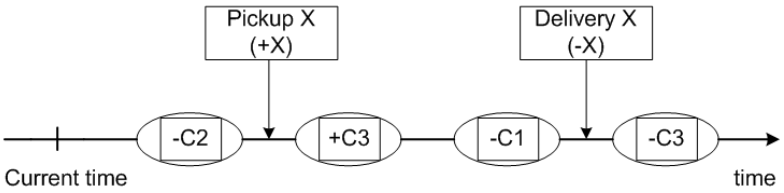

Fig. 3. Insertion heuristic

Complexity depends on the algorithm chosen to find shortest path from a vertex to another - we have considered Djikstra's algorithm ($O(m + \log(n))$ with m edges and n vertices). It depends also on the insertion heuristic which for a vehicle V with a maximal capacity Q_v gives in the worst case a multiplying coefficient : $Q_v(2Q_v - 1)^2$.

The objective function $MinZ = \triangle C$ represents the minimal overcost due to a new client insertion. $\triangle C$ depends on the following variables: additional time to service the new demand, current capacity (number of clients on board) and proximity of the vehicle from the customer. If two or many vehicles give the same value of the overcost for the same customer, the one having the minimal capacity wins this customer. If they have also the same capacity, the winner is the nearest one (in distance).

3.5 Self-organisation

The dynamic and non deterministic aspect of the problem can lead to concentrations of demands in certain zones which are more attractive and may cause a lack of service elsewhere. Indeed, the downtown area, for example, will be a zone of strong attraction at certain hours of the day whereas certain suburban zones become badly serviced. The waiting time of clients in such a zone will then be very important. We thus have chosen to possibly degrade the performances in the attractive zones in spite of having a better service in other areas to avoid any lack of service. Several solutions are possible:

- Injecting some vehicles in the existing fleet but that can violate constraints related to environmental objectives, we could have else a maximum filling of the vehicles,
- A hierarchical centralized resolution which is opposed to the decentralized model we adopt and not very realistic,
- The use of self-organization mechanisms. We chose the last way for its distributed, local and adaptive characteristics.

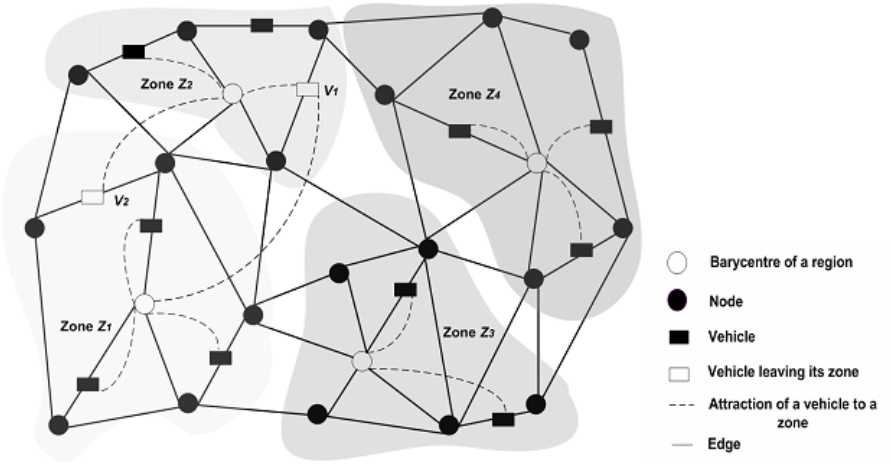

Fig. 4. Attraction of vehicles to zones

Thus, we define zones of attraction within the graph to which the vehicles are attached. These zones will evolve either geographically (the number of node they cover) or according to the number of attached vehicles. These zones have a multi-scale representation by using learning techniques since the vehicles acquire information on the road network state and are able to send the information to the graph.

When a vehicle leaves its zone, it will be penalized in its overcost function which increases while it moves away. This constitutes an exerted attraction

force so we can imagine a spring fixed on the zone center by an extremity (see figure 4) and a vehicle is attached to its other extremity. If this fights against the change of a vehicle zone, it does not prohibit the vehicle from changing its zone. Indeed, the vehicle can gain the bid for a demand coming from another zone. When a vehicle leaves its zone, it can negotiate a change of zone with others. In figure 4, the vehicles $V1$ and $V2$ exchange their zones. This part of the modelisation is under developpement.

4 Simulation

Multi-agent proposed architecture was developed by using the REPAST Simphony (Recursive Porus Agent Toolkit Simulation) multiagent platform written in Java which focuses on social simulation [14]. This platform developed by the Argonne laboratory of the University of Chicago, inherits main functionalities from SWARM platform (into Objective C) and offers several advanced functionalities:

- Built-in 2D, 3D, and geographical information systems (GIS) support and tools,
- Automated connections to enterprise data sources: relational databases, GIS and to external programs for statistical analysis and visualization of model results,
- Provides information about the state of each agent,
- A scheduler which supports concurrent discrete events in a sequential or parallel way.

As mentioned previously, the calculation of an overcost related to a new request, is done by each vehicle. At a time step, if a vehicle receives a request, it collects the other vehicles answers (overcost of the other vehicles) and compares their overcost to its own, broadcasts a winner message (if he is the winner), vehicles and the concerned customer before going to service him, as described in 3.3 and detailed in activity diagram (see figure 5).

5 Preliminary results

We have implemented our model on a graph with 50 nodes, 7200 time steps, with 4-passenger-seat vehicles. The customers appear at random places and hours, and give random destinations. In the table below, the optimal cost column indicates minimal total time to service all the customers from their departure until their destination nodes. The cost column represents the real time in which we serviced all customers. The client optimal itinerary column indicates the itinerary time average of the served clients. Variation per client is the percentage of difference between the two costs. Filling indicates the interval in which the capacity has oscillated.

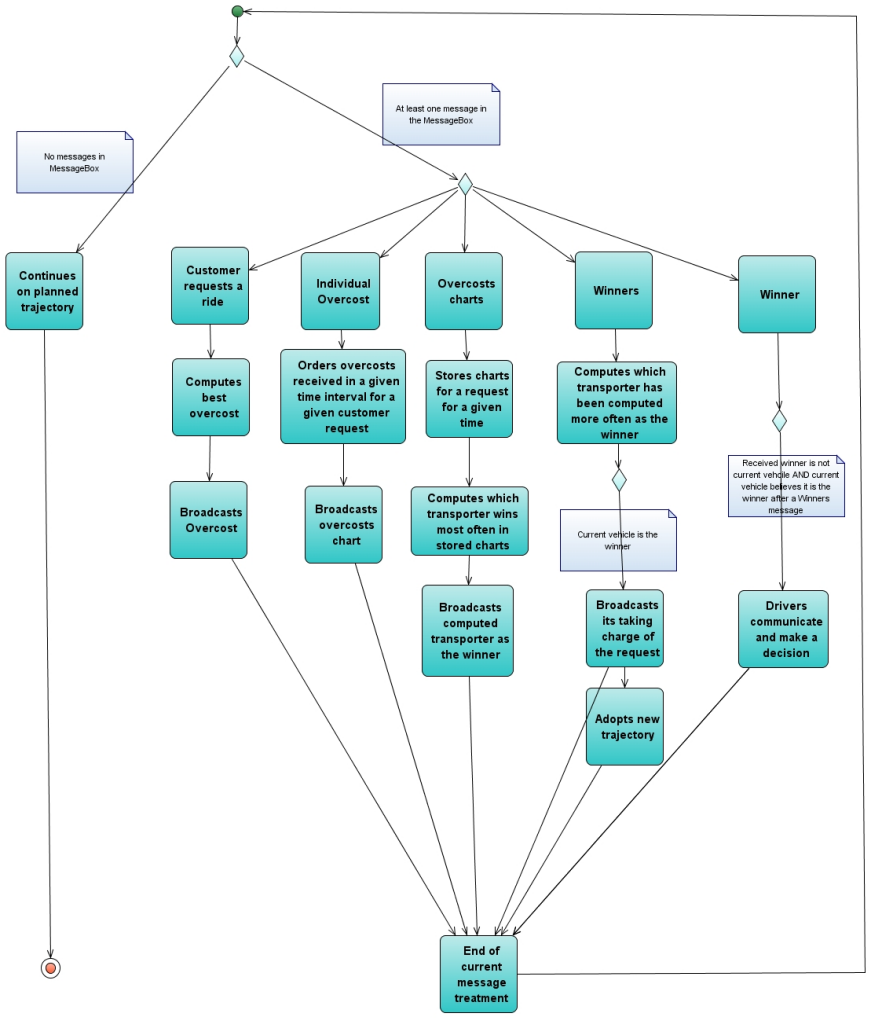

Fig. 5. Activity diagram for a vehicle

Vehicle number	Clients number	Served Clients	Optimal cost	Cost	Client optimal itinerary	Variation per client	Served Clients %	Rate of filling
4	227	227	4409	4878	19.42	10.6%	100%	$0 \rightarrow 4$
4	47	47	7324	7922	155	8.2%	100%	$0 \rightarrow 4$
8	357	315	26745	29381	85	9.8%	88%	$0 \rightarrow 4$
8	200	142	16066	17243	113.14	7.31%	71%	$0 \rightarrow 4$

Table 1. Some results of simulation

We notice that the first results are very encouraging. The variation between real cost and optimal cost is not important if we take into account the time to park an individual vehicle in the real life case and the cheaper price proposed to customers in collective transportation. We remark that, for a given number of vehicles and a given period of simulation (7200 steps in the above tableau), our model gives better results with a limited number of clients having long itineraries than with a big number of clients having short itineraries. It is because of the supplementary time due to satisfy all clients which is more important in the case of a big number of clients. The simulation must be improved: probabilistic model for the population of customers, better statistics for filling ... The self-organization mechanism is under development.

6 Conclusion and perspectives

In this paper, we presented a Transportation On Demand system which is purely dynamic, in an environment in perpetual change. We have adopted a decentralized approach based on the optimization and negotiation between vehicles. To face the lack of service in certain zones or the over-concentration of vehicles in certain other zones, we have proposed a self-organizational model which can adapt to the environmental changes. The obtained results are encouraging and the phase of self-organization is under development. We will continue our work by the complete validation of the proposed model.

References

1. Attanasio A, Cordeau J.F, Ghiani G, Laporte G (2004) Parallel Tabu search heuristics for the dynamic multi-vehicle dial-a-ride problem. Parallel Comput. 30:377–387
2. Christofides N, Mingozzi A, Toth P (1979) The vehicle routing problem. In: Combinatorial Optimization. volume 11, pages 315338. John Wiley,
3. Cordeau J.F, Laporte G, Potvin J.Y, Martin W.P (2004) Transportation on Demand. Handbooks in Operations Research and Management Science.
4. Hauptmeier D, Krumke S. O, Rambau J (2000) The online dial-a-ride problem under reasonable load. In proceedings of the 4th italian conference on algorithms and complexity, volume 1767. Springer.
5. Jaw J.J, Odoni A.R, Psaraftis H. N, Wilson N.H. M (1986) A heuristic algorithm for the multi-vehicle many-to-many advance request dial-a-ride problem with time windows. Pergamon Journals Ltd., vol. 20B, no. 3, pages 243-257.
6. Koźlak J, Crput J.C, Hilaire V, Koukam A (2006) Multiagent approach to dynamic pick-up and delivery problem with uncertain knowledge about future transport demands. Fundam. Inf., vol. 71, no. 1, pages 27-36.
7. Krumke S.O, De Paepe W.E, Poensgen D, Lipmann M, Marchetti-Spaccamela A, Stougie L (2006) On minimizing the maximum flow time in the online dial-a-ride problem, volume 3879. Springer Berlin Heidelberg.

8. Laporte G, Gendreau M, Potvin J.Y, Semet F (1999) Classical and Modern Heuristics for the Vehicle Routing Problem. International Transactions in Operational Research, vol. 7, pages 285-300.
9. Larsen A (2000) The dynamic vehicle routing problem. PhD thesis, Technical University of Denmark.
10. Lenstra J, Rinnooy Kan A.H.G (1981) Complexity of the vehicle routing and scheduling problems. In: Networks, volume 11, pages 221-228. Springer.
11. Lipmann M, Lu X, De Paepe W.E, Sitters R.A, Stougie L (2004) On-Line Diala-Ride Problems Under a Restricted Information Model. In: Algorithmica, volume 40, pages 319-329. Springer.
12. Madsen O.B.G, Ravn H.F, Rygaard J.M (1995) A heuristic algorithm for the a diala-ride problem with time windows, multiple capacities, and multiple objectives. Annals of Operations Research, vol. 60, pages 193-208.
13. Montemanni R, Gambardella L, Rizzoli A, Donati A (2003) A new algorithm for a dynamic vehicle routing problem based on ant colony system. In Second International Workshop on Freight Transportation and Logistics.
14. North M.J, Howe T.R, Collier N.T, Vos R.J (2005) The Repast Simphony Runtime System. Proceedings of the Agent 2005 Conference on Generative Social Processes, Models, and Mechanisms, Argonne National Laboratory, Argonne, IL USA .