Computing and Informatics, Vol. 28, 2009, 277–298

MULTI-AGENT ENVIRONMENT FOR MODELLING AND SOLVING DYNAMIC TRANSPORT PROBLEMS

Jarosław Koźlak

Department of Computer Science, AGH University of Science and Technology
al. Mickiewicza 30, Kraków, Poland
e-mail: kozlak@agh.edu.pl

Jean-Charles Créput, Vincent Hilaire, Abderrafiaa Koukam

Systems and Transport Laboratory
University of Technology of Belfort-Montbeliard
Belfort, France
e-mail: {Jean-Charles.Creput, Vincent.Hilaire, Abder.Koukam}@utbm.fr

Manuscript received 4 August 2005; revised 21 December 2007
Communicated by Baltazár Frankovič

Abstract. The transport requirements in modern society are becoming more and more important. Thus, offered transport services need to be more and more advanced and better designed to meet users demands. Important cost factors of many goods are transport costs. Therefore, a reduction of costs, a better adjustment of strategies to the demand as well as a better planning and scheduling of available resources are important for the transport companies. This paper is aimed at modelling and simulation of transport systems, involving a dynamic Pickup and Delivery problem with Time Windows and capacity constraints (PDPTW). PDPTW is defined by a set of transport requests which should be performed while minimising costs expressed by the number of vehicles, total distance and total travel time. Each request is described by two locations: pickup and delivery, periods of time when the operations of pickup or delivery can be performed and a load to be transported. The nature of this problem, its distribution and the possibility of using a lot of autonomous planning modules, lead us to use a multi-agent approach. Our approach allows the modeling of entities which do not appear in the classical PDPTW such as company organisation, communication among vehicles, interactions between vehicles and company dispatcher or different strategies of requests

acceptation by different vehicles. This paper presents also a software environment and experimentations to validate the proposed approach.

Keywords: Multi-agent simulation, transport planning and scheduling, dynamic pickup and delivery problem with time windows

Mathematics Subject Classification 2000: 68T20, 68U20

1 INTRODUCTION

The transport requirements in modern society are becoming more and more important. The transport services need to be more and more advanced and better designed to meet the needs of users. A significant part of costs of many goods are transport costs. The size of the market of transport services as well as the sales volume and derived profits intensify competition. Therefore, a reduction of costs and better adaptation of strategies to the demand require better planning and scheduling tools for available resources of transport companies. Computer systems can be a useful tool for transport companies. They may support a rapid creation of effective transport plans and schedules or enable simulation research leading to the correct selection of company organization, vehicles and capacities or locations of depots.

This paper is aimed at modeling and simulation of transport system, involving a dynamic Pickup and Delivery problem with Time Windows and capacity constraints (PDPTW). The nature of this problem, its distribution and the possibility of using a lot of autonomous planning modules, lead us to use a multi-agent approach. A multi-agent approach allows to consider aspects which do not appear in classical PDPTW, such as a company organisation, different strategies of requests acceptation by different vehicles or communication among vehicles.

The dynamic PDPTW can be seen as an extension of the standard and static PDPTW. PDPTW is defined as follows: there is a set of transport requests, which should be performed by a fleet of vehicles at the lowest possible cost expressed by the number of vehicles, total travel distance and total travel time. Each request is described by two locations: pickup and delivery, and two time windows, the time window for the pickup operation and the one for the delivery operation. Both the request pickup and delivery places should be visited by the same vehicle in the proper order. The time window is a period of time, when the service may be started. The time window is described by the start time and due time. A vehicle has to arrive at some location before the due time, and must wait if it arrives before the start time, then it performs its services. Each request has a load and each vehicle has a maximum capacity which cannot be exceeded by the total load of goods transported. In the dynamic PDPTW, the difference with the static case is that the request input set now vary dynamically and that the optimisation process has to take place in real-time as new input requests arrive.

The static PDPTW has several practical applications. For example, a version of the problem called Dial-a-Ride problem [10] has applications to the transport of elderly and handicapped people. PDPTW is adequate for minibus companies planning, sealift and airlift, discharge of pesticide, school bus routing and scheduling or shared taxis. Optimisation approaches based on the use of meta-heuristics already exist and have been applied to standard set of benchmarks [24]. But an important feature of a real problem is its dynamic. A lot of transportation applications need to take into account uncertainty of user demands and transport conditions, such as traffic jams or car crashes. In the dynamic case, the problem now becomes considerably harder [3] and needs to consider benchmarking problems with various dynamics [2].

The nature of the problem, which involves distributed entities as vehicles in communication with the company controllers dealing with environment uncertainty and random demands, leads us to use a multi-agent approach. We developed a multi-agent system following the RIO (Roles, Interactions, Organisation) [21] methodology, which allows us to specify the system at the level of role, interaction and organisation, and by using the Contract Net Protocol [33] as an interaction schema. The aim is to exploit the physical distribution among vehicles and a central company in order to distribute computations and solve collectively the optimisation problem. We think that another advantage of the proposed approach is its flexibility, that is its capacity to be easily extended. Furthermore, in contrast to other works on dynamic PDPTW, which model vehicles movng into the Euclidean plan, we represent the transport network as a graph to better reflect the structure of the interconnected streets in a city.

The structure of the paper is as follows: Section 2 contains a research overview. We put the emphasis on heuristic principles to solve the PDPTW and related problems, their architectures and description of multi-agent systems for transport planning and scheduling. In Section 3, main features and advantages of our approach are given and a model of the system is described. Section 4 presents a description of the system architecture and implementation, while Section 5 presents goals and configurations of the performed experiments as well as obtained results. Section 6 concludes and presents plans of future works.

2 RESEARCH OVERVIEW

2.1 Heuristic Method of Plan and Schedule Construction

The PDPTW can be seen as an extension of the Vehicle Routing Problem with Time Windows (VRPTW) which concerns only collections of either pick-ups or deliveries, assuming requests with a single location. Thus, generalizing heuristic principles from VRPTW is a natural way to build new solutions to the PDPTW. In contrast to static vehicle routing problems, that have been well known problems since the seventies [12], fewer works focus on the dynamic aspects of PDPTW. Usually, the algorithms are similar to the ones used for static problems, and it is often assumed

that the incoming of new requests interrupts the optimization process which is then restarted with a new set of requests. An overview of exact and heuristic methods of pickup and delivery problem solving can be found in [10, 13, 27]. Vehicle routing heuristics generally consists of two phases: a construction phase generates an initial schedule and an improvement phase ameliorates the constructed solution. An overview of route construction algorithms may be found in [7, 34, 23, 24, 25, 28]. Improvements in route schedules are generally performed using metaheuristics like evolutionary algorithms [4, 5, 11], tabu search [23, 24, 28], simulated annealing [24], squeaky wheel optimisation [25], or ant-colony systems [18]. One difficulty in the dynamic PDPTW case is the lack of benchmarking testbeds recognized for evaluation. In [22] the author proposes to classify them according to their degree of dynamism. In [26] and [19], the authors describe a method for determining times of requests generation on the basis of Poisson distributions. Here, we will use a similar request generator, but adapted to our transport model based on graphs rather than on Euclidean space.

2.2 Multi-Agent Systems

The multi-agent approach concerns the development of systems consisting of many autonomous entities which are able to create plans and choose actions to reach their goals [14]. Because of different locations of vehicles, the transport planning and scheduling problem may be considered as a typical example of a problem with a distributed domain, which are very suitable for a multi-agent methodology. The multi-agent approach allows autonomous, goal-driven agents, which represent company or vehicles, to be taken into consideration. Each agent-vehicle manages its route. An agent estimates a request taking into consideration its feasibility, the payment it obtains and the expenses. Thus, in multi-agent approaches, the natural distribution between physical entities (vehicles and company) is exploited in order to achieve distributed computations and optimization.

In the literature, the multi-agent approach to transport problems focuses mostly on complex cargo shipping problems, sometimes taking into consideration transshipments and transport multi-modality as the MARS (Modeling Autonomous CoopeRating Shipping Companies) platform [17] or TeleTruck system [8]. TeleTruck is an extended implementation of the MARS system. While agents of MARS represent homogenous trucks, the TeleTruck approach models the basic physical objects (drivers, trucks, trailers, containers) by basic agents which join together and form holonic agents that act in a corporated way. Some multi-agent systems are specifically targeted for vehicle routing problems [1] and very few on transport-on-demand problems [20]. In the above multi-agent approaches two mechanisms are identified to deal with route optimisation. Route construction is generally performed by instantiating a Contract Net Protocol [33] between agents, whereas route improvement is achieved by the simulated trading procedure [1]. However, few benchmark results exist for staric VRP, VRPTW or PDPTW and no benchmarks are given for

the dynamic case problem. Here, we will instantiate negotiation protocols based on similar structure and dynamic benchmark.

Apart from the approaches based on Contract Net protocol and simulated trading, there are also solutions used which take advantage of the algorithms solving the DCSP (Distributed Constraints Satisfaction Problem) which has been widely researched in the domain of the multi-agent research problem [29]. Among recent results related to the application of multi-agent approaches in solving transport problems, it is worth mentioning the AS/ATN (Living Systems Adaptive Transportation Networks) system [30]. According to its authors, this system was applied in practice by several big international shipping companies to construct their transport schedules, making the application of this system probably the largest commercial use of agent technologies in the world.

3 SYSTEM MODEL

3.1 Specification of System Organization

Our goal is to create a system which makes the simulation of transport company possible. Transport requests performed by a company should suit dynamic PDPTW with capacity constraints. A model of a multi-agent system for transport planning is composed of the following main entities:

- environment: it is a transport network, a graph describing road connections,
- agents: customer agent responsible for the generation of transport requests, agent-company, representing a transport company, and agent-vehicles, representing single vehicles like mini-buses.

Figure 1 shows agent population within its environment. Each agent-vehicle has a representation of the transport network and of its successive positions on roads assuming, that a vehicle uses a geographical information system (GIS) and geo-localizes itself using satellite positioning as GPS (Global Positioning System) system. The agents-vehicle communicate with the agent-company or directly one to each other. The system organization is described using the RIO framework previously defined in [21]. This framework is based on three interrelated concepts: role, interaction and organization. Roles are generic behaviours. These behaviours can interact mutually according to interaction pattern. Such a pattern groups generic behaviours and their interactions into an organization. Organizations are thus descriptions of coordination structures. In this context, an agent is an active communicative entity which plays roles [15].

Figure 2 describes the organizations of our system. There are two organizations, one specifying the Contract Net protocol and one specifying the interactions between clients and pick-up and delivery service providers. The former organization is composed of three roles: *Manager*, *Bidder* and *Contractor*. This organization specifies a Contract Net based negotiation between a Manager which proposes pick-up and delivery requests, some Bidders which are able to realize these requests and

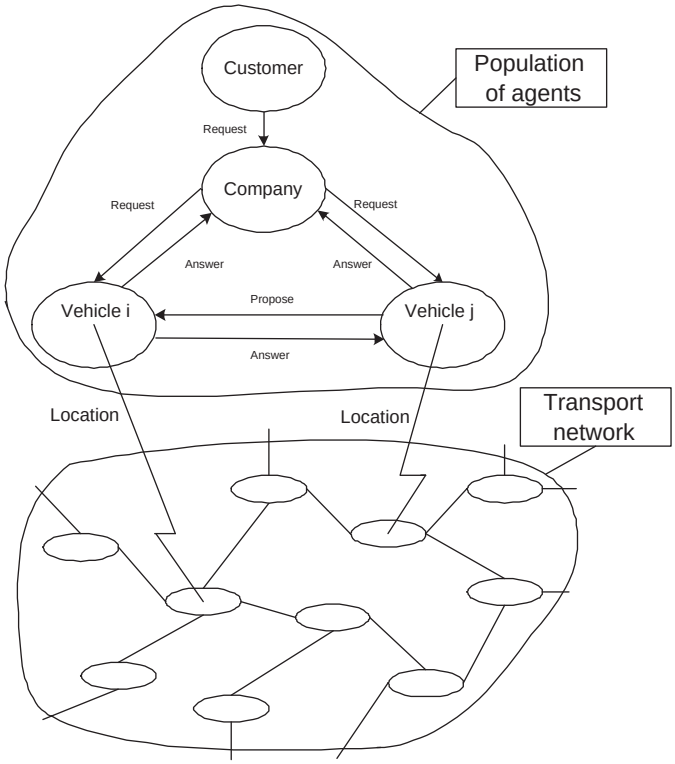


Fig. 1. Multi-agent system model: structure and main interactions

among these Bidders a Contractor who eventually realizes this request. The Manager role is played by the agent-company and the Bidder and Contractor roles are played by agents-vehicles. The latter organization specifies two roles: *Provider* and *Client*. The Client role represents virtual clients that send a request of pick-up and delivery to a virtual Provider that accepts these requests and answers with an offer. The Provider role is played by the agent-company. Agents instantiate organizations (roles and interactions). They always exhibit behaviours defined by the organization's roles. Each time they play one or more roles and they act executing predefined associated protocols. The different protocols coordinate actions in order to favour distributed optimization. Conversely, a role may be instantiated by one or more agents. Furthermore, roles are basic protocol components that are assigned or deleted in a dynamic way, depending on events received and actions performed.

3.2 System Entities and Their Distribution

Environment. The transport network is represented by a directed graph $TN(N, E)$, where N is a set of nodes and E is a set of arcs. Nodes represent

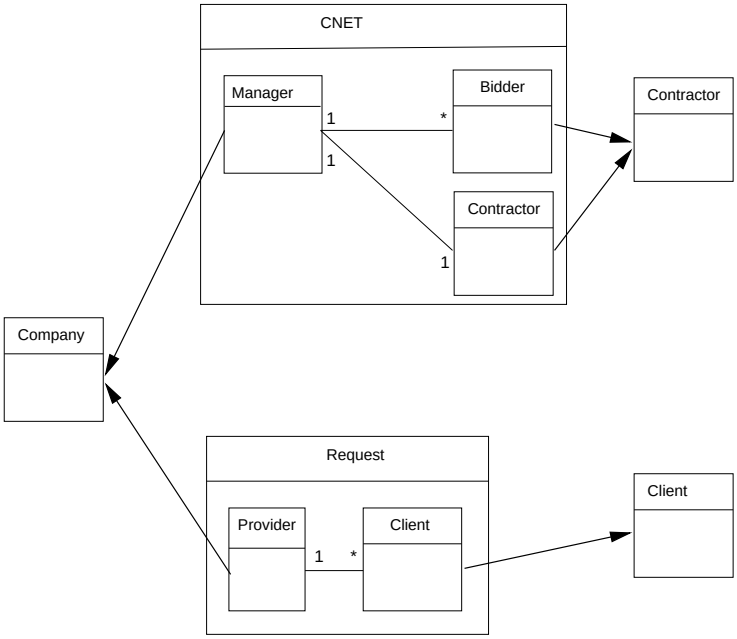


Fig. 2. System organization

the locations. They may be either a start or destination point of the vehicle route or a travel intermediate node. With each node of the transport network $V_i \in N$, a pair (x_i, y_i), where x_i, and y_i are coordinates on the map, a numerical value (w_i), describing the probability that this node will appear in the transport request as pickup or delivery point and a list of requests waiting for pickup from this node (pl_i) are associated. Each arc E_i has a time period tp_i, which informs how much time the vehicle needs to traverse it. The time period tp_i expresses also the main travel cost carried out by a vehicle.

The following types of agents exist in the system: agent-customer, agent-vehicle and agent-company.

Agent-customer. The agent-customer is responsible for the generation of random events. The methods of events generation is inspired by [19] and then the agent-customer sends request events to agent-company. The agent-customer $ACust$ is described by a list of periods of requests generations (prg_i), each period is characterized by: coefficient of Poisson distribution describing the frequency of request generation (γ), probability that time window of pickup starts just after request arriving (β) and an expected size of time windows (δ).

Agent-company. Agent-company is responsible for request reception from agent-customer and after it schedules requests to agent-vehicles. Agent-company AC_i is represented as a tuple $(g, c, gains, RL)$, where:

g – incomes;
c – costs;
$gains$ – (gains = incomes-costs);
RL – a list of received requests with information concerning the state of their realization (received, rejected, scheduled to agent, pickup performed, delivery performed).

Agent-vehicle. Agent-vehicle represents a vehicle. It moves among nodes in the transport network and possesses plans of its routes and schedules of requests realisation. To obtain a request, it sends offers to the agent-company. Agent-vehicle AV_i is defined as a tuple $(loc_i, g_i, c_i, cap_i, pass_i, LRN_i, R_i, O_i, LO_i)$, where:

loc_i – a current location, expressed by a node, a direction and a percentage of traversed arc;
g_i – current incomes;
c_i – current costs;
cap_i – maximal capacity allowed;
$pass_i$ – current number of passengers;
LRN_i – list of request nodes in the schedule;
R_i – list of nodes forming the route;
O_i – list of accepted orders,
LO_i – list of embarked orders (after pickup and before delivery).

3.3 System Dynamics

Vehicles communicate, with the company or with other vehicles, following protocols and defined organization roles. Thus, they are equipped with message communication modules. Each vehicle has its proper decision module of acceptance and estimation of request. Vehicles negotiate, knowing only local information about their own situation. They are able to determine their localization and compute shortest distances on road network. They memorized their own route or potential routes. While following their plans, vehicles can open or close a company negotiation, open an optimizing round with other vehicles or do nothing else. Roles are activated within an event driven framework. A clock generator serves as a based time to generate random events, which in turn start agent activities.

The originality of our approach, among the others, is based on the application of multi-agent methodology and specification of agent roles, which makes a presentation of system functions and its future development easier. The main organization scheme is the Contract Net Protocol based interactions between agents-vehicles and agent-company. After reception of a randomly generated transport request, the agent-company sends this information to its agents-vehicles. Agent-vehicles respond with their estimation of expenses that they will bear while realizing the request. To do so they consider the optimum possible insertion points for pickup and delivery. In

our case, evaluation of expenses concerns an extra time to travel, since a reduction of travel time is the single objective of PDPTW once the number of vehicles is fixed. If a request realization is impossible for a vehicle, it informs the company about it. Furthermore, the company makes a selection of the vehicle which will perform the request and will become a contractor of the allocated request.

There are two other situations, when request allocation takes place. The first one takes place when a new request arrives, which cannot be served. In this case, vehicles try to find in their schedules non-served requests which cause the highest distance increase and then remove them from the schedules to make a realisation of a new request possible. If such a request is found and there is another vehicle which may serve it, then the "bed" request is re-scheduled to this other agent and a new request is inserted in its place. The second situation is when an agent tries to get rid of a request which highly increases a distance of its route and move it to another agent. If such an agent is found then the request is moved from the schedule of the first agent and inserted into the schedule of the second one (Figure 3).

```
OfferRequesToOtherVehicles(r) {vehicle tries to reschedule request r}
calculateCost(O) {travel cost of the route serving set of requests O}
{Main}
δ_max = 0;
for all r in O_i − LO_i do
   δ_r = calculateCost(O_i) − calculateCost(O_i − r)
   if δ_r > δ_max then
      r_max = r; δ_max = δ_r
   end if
end for
if δ_max > threshold then
   OfferRequestToOtherVehicles(r_max)
end if
```

Fig. 3. Request rescheduling

All these request allocation operations are performed using Contract Net Protocol scheme, only the roles played by agents change. A manager may be, beside an agent-company agent, also an agent-vehicle which tries to get rid of the request. If there is no agent being able to perform the request, it remains not scheduled and not realized.

When a new request arrives, the decision module tries to generate a now route by adding (construction) or exchanging (optimisation) the new request to the current route. While doing so, the quality of the new route is rated. The rating is sent to the agent-company. If the modification of the route is confirmed, then the current route is updated. The payment for realization is calculated on the basis of the

shortest path in the graph between the pickup and delivery nodes and the number of transported persons.

Agent-vehicle performs five basic activities which have influence on its state: moving (*move*), request acceptation or rejection (*accept*), request loading (*pickup*), request unloading (*delivery*) or request rescheduling – as a result of negotiations performed to optimise a solution by exchanging the request another other vehicle (*reschedule*). Assume that there are two state configurations of AV_i, one at time t, before an action is performed, and the second at time t', when action is finished:

1. $AV_i(t) = (loc_i, g_i, c_i, cap_i, pass_i, LRN_i, R_i, O_i, LO_i)$
2. $AV_i(t') = (loc'_i, g'_i, c'_i, cap'_i, pass'_i, LRN'_i, R'_i, O'_i, LO'_i)$

The actions modify the following parameters of agent state:

- move: $(loc_i, c_i) \rightarrow (loc'_i, c'_i)$;
- accept: $(LRN_i, R_i) \rightarrow (LRN'_i, R'_i)$;
- pickup: $(pass_i, LO_i) \rightarrow (pass'_i, LO'_i)$;
- delivery: $(g_i, pass_i, LO_i) \rightarrow (g'_i, pass'_i, LO'_i)$;
- reschedule: $(LRN_i, R_i) \rightarrow (LRN'_i, R'_i)$;

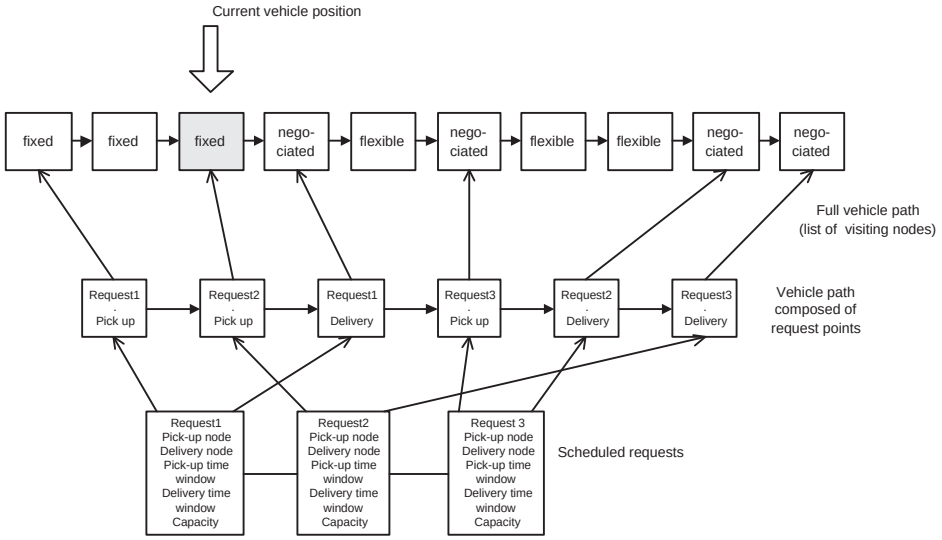


Fig. 4. Vehicle schedule representation

Agents-vehicles plan their routes so as to contain all nodes of accepted transport requests. Each agent-vehicle uses two types of route representation (Figure 4): a list of request nodes and a list of nodes. The list of request nodes is composed of pickup

and delivery points for each accepted request ordered according to vehicle time of the visit. Thus, each request is represented in the list by two points. Additionally, if the vehicle agent currently is not on a node associated with the request, this current node also belongs to the list of request nodes. If an agent is currently moving between nodes, the future visited node is taken into consideration. It is assumed that an agent is not able to change its destination node during moving between nodes. On the basis of request nodes, the full vehicle route composed of all traversed nodes is constructed. The point of current vehicle position as well as the status of the vehicle route points are marked (Figure 4). It is possible to distinguish points which either have to be visited (*fixed*), or may be omitted but only if another vehicle will commit itself to visit it (*negotiated*) or may be omitted without necessity of additional agreements (*flexible*).

4 ARCHITECTURE OF THE SIMULATOR

4.1 Main Software Components

In this section, the architecture of the system is described. It is an event-driven simulation, written in Java and having benefit of MadKit platform [16]. As presented in Figure 5, the system consists of the *Simulation module*, *GUI module*, *Configuration module* and *Statistics module*. The *simulation module* is composed of the multi-agent system, the clock responsible for the management of simulation time and the event queue. The UML diagram (in Figure 6) shows the main simulation classes. The *GUI module*, which manages the main screen shown in Figure 7, is responsible for interactions with the user, setting the configuration parameters, controlling visualisation of simulation progress as well as presentation of the results.

The *Configuration module* provides a functionality to define parameters like the number of vehicles, their features, initial and final positions, duration of simulation, parameters describing transport requests, as well as access the configuration file and a map file describing the transport network structure. The *Statistic module* gathers data about a simulation process and builds statistics.

4.2 Dynamics of Implemented Agents

The activities performed by agents are consequences of received events from the event queue. An agent can perform operations on the event queue. It can create an event and insert it into the event queue. Agent-creator of an event may remove this event from the event queue. The other operations are execution of an event or reaction on effects of event, if the agent is on the list of the event recipients.

Each event is a sub-class of a generic event, which has the following attributes: type of event (*type*), the time at which the event is performed (*time*), agent which inserts the event into the queue (*creator*), agent which performs the event (*executor*), and agents which are directly influenced by the execution of the event (*recipients*).

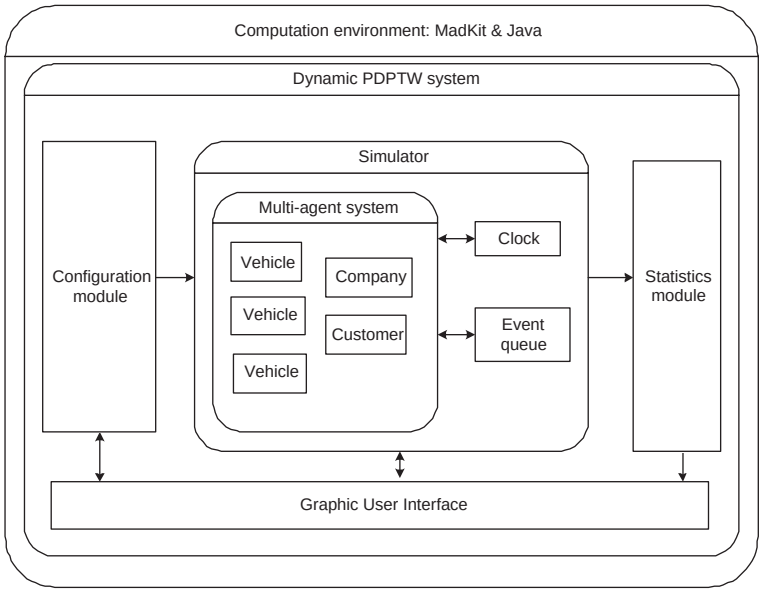


Fig. 5. System architecture

Based upon the generic event, the several types of events which have additional specific attributes may be constructed: *Start Moving*, *End Moving*, *Request Generation*, *Request Arrival*, *Pick-up Start*, *Pick-up End*, *Delivery Start*, *Delivery End*, *Scheduling* and *Check for Messages*, *Start Moving* and *End Moving* are performed when an agent-vehicle starts to move between the nodes and when it arrives at destination node. When the *Request Generation* is executed, a random *Request Arrival* event which attributes are set as described in Section 4.4 and a new *Request Generation* event is inserted into the event queue. When a *Request Arrival* event is performed, the agent-company obtains a new transport request. Then, it initiates the scheduling process to allocate the request to an agent-vehicle. *Pick-up Start*, *Pick-up End*, *Delivery Start* and *Delivery End* launch or finish the execution of loading and unloading operations. Invoking *Scheduling* event an agent-company asks agents-vehicles if they are able to realize the transport request and which costs they will bear for its realization. During *Check for Messages* event an agent checks if it has received a new message, thus it performs actions in response.

4.3 Agents

Relations among various types of agents are presented in Figure 8. Agents are a specialization of a *Generic Agent* which offers basic features, like name, access to the clock, representing time flow during simulation process, handling of event queue

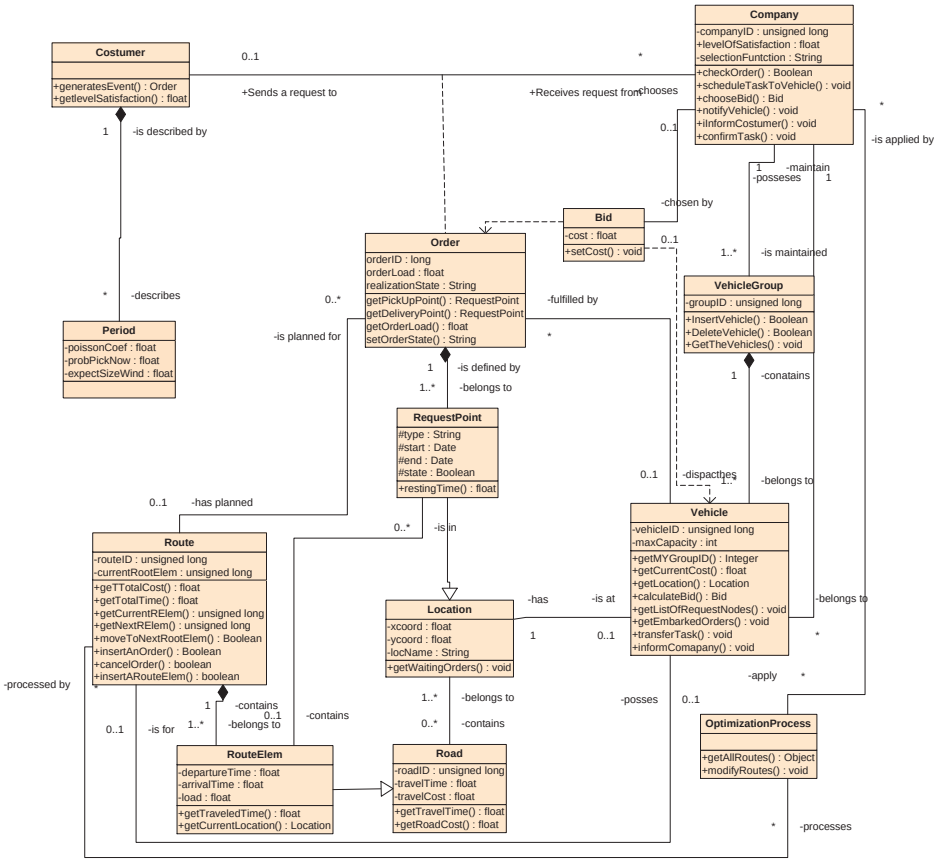


Fig. 6. Main simulator classes

as well as message sending and receiving modules. The decision module depends on application refinement.

Agent-company and *agent-customer* have a simple decision module which simply contains the automaton responsible for the negotiation protocol. *Vehicle-agent* is more complex than other types of agents, especially considering the different sets of actions which this agent is able to perform. The sub-components of an agent-vehicle decision module are presented in Figure 10. An agent-vehicle manages its current route following travel steps, from node to nodem in the transport network. When a new request arrives, it executes the required protocol. Particularly, it generates a new route by construction and optimisation heuristics and evaluates the gains of the new route to be returned to the agent company.

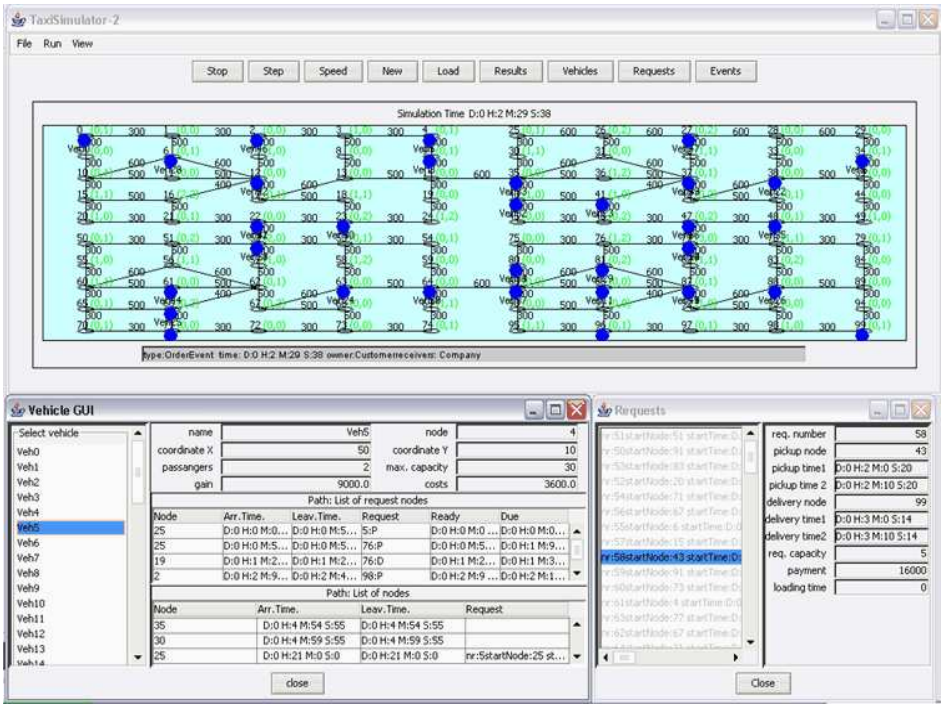


Fig. 7. Simulator in MadKit environment

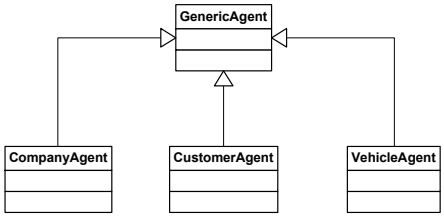


Fig. 8. Agents generalisation/specialisation tree

4.4 Transport Requests Generator

Request generation is performed by the agent-customer. The time of arrival of request at a transport company is calculated on the basis of a Poisson distribution defining a period of minutes after which a new request arrives. The method of defining time windows is inspired by the one described in [19]. For the time window of pickup, the ready time is selected by using a uniform probability distribution within the period (time of request generation + a constand delay, last possible time of pickup realization). The last possible time of pickup realisation is a time needed

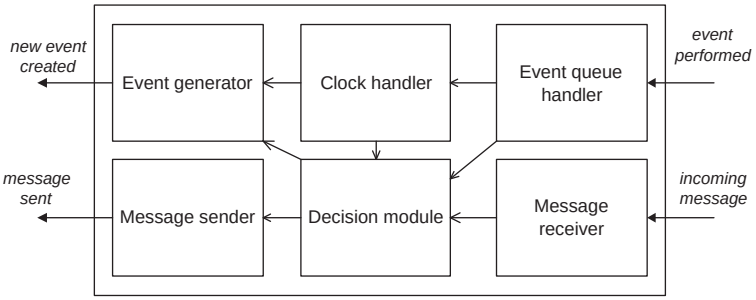


Fig. 9. Generic agent architecture

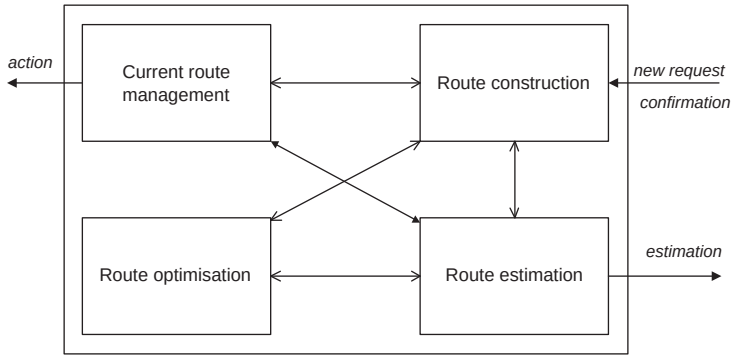


Fig. 10. Agent-vehicle decision module

for a vehicle to perform the delivery operation and return to a terminal location given for this vehicle before the simulation end. Depending on experiments, the due time is selected using uniform distribution within the period (ready time, last possible time of pickup realisation) or the size of time window is set as a constant. In the case of the time window of delivery, ready time is selected using a uniform probability distribution on the interval (ready time of pickup + time of trip between pickup and delivery points, last possible time of delivery operation, leaving a time necessary to return to terminal). Due time is calculated by using a uniform distribution on the interval (staring time of delivery time window, last possible time of delivery realisation) or the size of the time windows may be set as a constant. The last possible time of delivery realisation is the delay for a vehicle needed to return to terminal point on time.

The pickup node is randomly generated taking into consideration the weights w_i attributed to nodes. The probability of being selected is proportional to the weight of the given node. The delivery node is chosen similarly, but it is assumed that the delivery node has to be different from the pickup node. The quantity of transported people is a random integer value from the interval $(1, \ldots, \text{maxpers})$, calculated using

the uniform distribution. The payment for realization is calculated on the basis of the shortest path in the graph between the pickup and delivery nodes and the number of transported persons.

5 EXPERIMENTS

5.1 Configuration of Experiments

The goal of the performed experiments is to choose the optimum company features, i.e. a number of vehicles or a capacity of vehicles, for the given topology of transport network, as well as the quantity and features of requests. In particular, we will examine configurations with different time window sizes denoted by δ and request frequencies expressed by γ (see description of agent-customer in Section 3.2). In the experiments performed, the transport network as presented in Figure 11 was used. The numbers above the arcs represent the length of travelled distances in seconds. Nodes marked as a black ones are the nodes with 5 times higher probability of being a pickup or delivery node (described by w_i weight) than other nodes. The numbers above the nodes are node identifiers. The network consists of 100 nodes and 272 arcs. The arcs represent the travel time between the nodes of which the average is about 7 minutes.

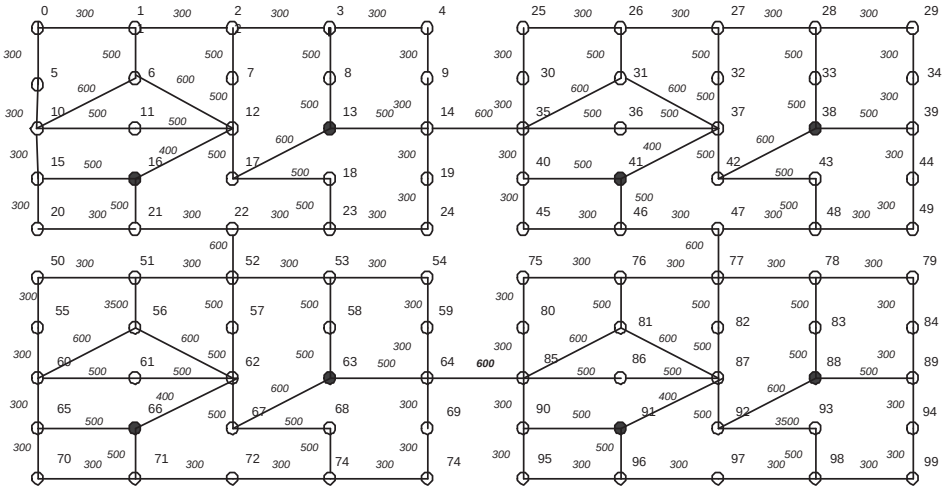


Fig. 11. Network topology

The number of vehicles, their capacities as well as their time windows sizes were changed from experiment to experiment. The total simulation time was divided into 5 intervals (4 simulation hours each) with different requests arrival frequencies describing the period in minutes in which new request arrives. Their Poisson distribution parameters which describe request frequency are equal to 0.75, 1.10, 0.25,

0.40 and 0.10. A sixth supplementary period of 1 hour gives a time for the last requests to be completed. The average value of generated requests is equal to 457. Two sizes of time windows are used in the experiments. The short time windows are equal to 10 minutes each. For long time windows, ending times of windows are calculated using uniform distribution, as described in 4.4.

5.2 Optimal Configuration of Transport Company and Negotiation Process

Figure 12 presents the percentages of performed requests in relation to the quantity of vehicles used and their capacities. Looking at Figure 12 we can see that the increase of request realization is relatively small. For this configuration of requests and transport network, the vehicles rarely have a total load higher than 20 because they are performing other requests previously submitted and do not have time to pickup more requests to travel fully laden. In Figure 13 the results obtained for short and long time windows are shown. The negotiation among vehicles takes place, if it is necessary. In this experiments vehicles capacities are equal to 50.

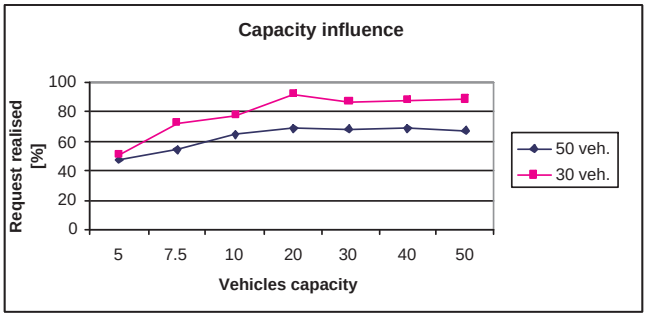


Fig. 12. Request realization vs. vehicles quantity and capacity for a problem with short time windows

In the case of long time windows, the percentage of the performed requests is significantly increased when the number of vehicles increases. For short time windows, the percentage of the requests performed by the same number of vehicles and with the same capacities is clearly lower than for the long time windows, because of much stronger time constraints during route construction. Similarly, as in the case of a long time window, increase of the degree of requests realisation when applying two times higher maximal capacities of vehicles in comparison to the smaller ones is relatively small (Figure 12).

Experiments with negotiations have shown a slight increase of the degree of requests performed (Figure 13), but it was associated with a high increase of computation time. Thanks to the negotiation process, also an average cost of request

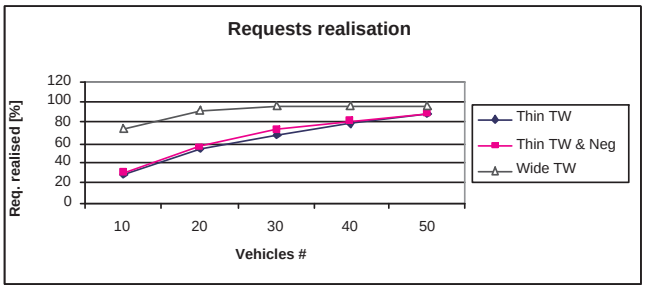


Fig. 13. Request realization vs vehicles quantity

realisation decreases slightly, because the vehicles get rid of the realisation of requests which cause the highest costs (Figure 14).

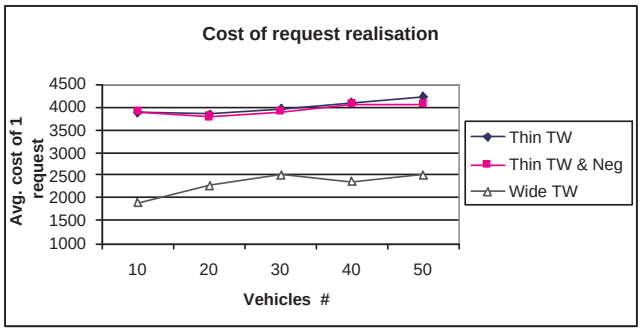


Fig. 14. Cost of request realisation

6 CONCLUSIONS

In this paper we have presented a simulation environment to solve the dynamic PDPTW by means of the multi-agent paradigm.

Both the analysis and design of the system have been detailed. In this environment, several experiments were carried out and their results were presented. So far, only few papers concerning dynamic PDPTW have been published, like [19]. In comparison to other research concerning PDPTW problems, our system uses the transport network as a graph to make the best representation of the network of interconnected streets in a city possible. Comparison with the results obtained by MARS system is difficult, because only the results of benchmarks for static VRPTW were presented and our system deals with dynamic PDPTW.

We developed a system which simulates company organization with vehicles communicating with the central depot. Following multi-agents development me-

thodology as RIO framework and using the Contract Net Protocols to distribute request among vehicles, we claim that the proposed approach's main advantages are its flexibility and modularity to build systems to transport planning and scheduling. It becomes possible to add emergencies or unexpected events, modify configurations, or change features of particular vehicles. In the same way, the system simulates transport organization and trading policies to perform distributed computations and optimization directly on the multi-agent structure, thus using natural distribution between physical entities. The multi-agent environment will have to be enriched by an introduction of several cooperating transport companies, allowing transport requests to be tackled by different sources, as autonomous vehicles and not only by the company. Other improvements are in the optimization algorithm. For example, one is able to equip vehicle agents with a cache of recently used requests in order to limit Dijkstra executions for finding shortest distances. The next step is the distribution on different processors, to follow natural decomposition between vehicles to go beyond real case company simulations.

REFERENCES

[1] Bachem, A.—Hochstattler, W.—Malich, M.: Simulated Trading a New Parallel Approach for Solving Vehicle Routing Problem. Proceedings of the International Conference Parallel Computing: Trends and Application, 1994.

[2] Bent, R. W.—van Hentenryck, P.: Scenario-Based Planning for Partially Dynamic Vehicle Routing with Stochastic Customers. Operations Research, Vol. 52, 2004, No. 6, pp. 977–987.

[3] Bertsimas, D.—Simchi-Levi, D.: A New Generation of Vehicle Routing Research: Robust Algorithms, Addressing Uncertainty. Operations Research, Vol. 44, 1996, pp. 286–304.

[4] Braysy, O.: Genetic Algorithms for the Vehicle Routing Problem with Time Windows. Arpakanuus 1, 2001, Special Issue on Bioinformatics and Genetic Algorithms.

[5] Braysy, O.—Gendreau, M.: Genetic Algorithms for the Vehicle Routing Problem with Time Windows. SINTEF Applied Mathematics Report, Oslo, Norway, 2001.

[6] Braysy, O.—Gendreau, M.: Metaheuristics for the Vehicle Routing Problem with Time Windows. SINTEF Applied Mathematics Report, Oslo, Norway, 2001.

[7] Braysy, O. –Gendreau, M.: Route Construction and Local Search Heuristics for the Vehicle Routing Problem with Time Windows. SINTEF Applied Mathematics Report, Oslo, Norway, 2001.

[8] Burckert, H.-J.—Fischer, K.—Vierke, G.: TeleTruck: A Holonic Fleet Management System. Available on `http://citeseer.ist.psu.edu/76560.html`, 2000.

[9] Burckert, H.—Fischer, K.—Vierke, G.: Holonic Transport Scehduling with TeleTruck. Applied Artificial Intelligence, Vol. 14, 2000, pp. 697–725.

[10] Cordeau, J.-F.—Laporte, G.: The Dial-A-Ride Problem: Variants, Modeling Issues and Algorithms. Les Cahiers du GERAD, 2002.

[11] Créput, J. C.—Koukam, A.—Kozlak, J.—Lukasik, J.: An Evolutionary Approach To Pickup-And Delivery Problem With Time Windows. Lecture Notes in Computer Science, Vol. 3038, 2004, pp. 1135–1142.

[12] Christofides, N.—Mingozzi, A.—Toth, P.: The Vehicle Routing Problem. In Christofides N. et al. (eds): Combinatorial Optimization, Wiley, pp. 315–338, 1979.

[13] Desaulniers, G.—Desrosiers, J.—Erdmann, A.—Solomon, M. M.—Soumis, F.: The VRP with Pickup and Delivery. Les Cahiers du GERAD, 2000.

[14] Ferber, J.: Les Systèmes Multi-Agents. Vers une intelligence collective, InterEditions, 1995.

[15] Ferber, J.—Gutknecht, O.: A Meta-Model for the Analysis and Design of Organizations in Multi-Agent Systems. In: Y. Demazeau, E. Durfee, N. R. Jennings (Eds.), Proceedings of ICMAS '98, 1998.

[16] Ferber, J.—Gutknecht, O.—Michel, F.: MadKit Development Guide Version 3.1. 2003, Available on `http://www.madkit.org`.

[17] Fischer, K.—Muller, J. P.—Pischel, M.: Cooperative Transportation Scheduling: An Application Domain for DAI. Applied Artificial Intelligence, Vol. 10, 1996, pp. 1–33.

[18] Gambardella, L. M.—Taillard, E.—Agazzi, G.: MACS-VRPTW: A Multiple Ant Colony System for Vehicle Routing Problems with Time Windows. Technical Report IDSIA, 1999.

[19] Gendreau, A.–Guertin, F.—Potvin, J. Y.—Seguin, R.: Neighborhood Search Heuristics for a Dynamic Vehicle Dispatching Problem With Pick-Ups and Deliveries. Technical Report CRT-98-10, Université de Montreal, 1998.

[20] Gruer, P.—Hilaire, V.—Kozlak, J.—Koukam, A.: A Multi-Agent Approach to Modelling and Simulation of Transport on Demand Problem. In: J. Soldek, L. Drobiazgiewicz (Eds.): Artificial Intelligence and Security in Computing Systems, The Kluwer International Series In Engineering And Computer Science 752, 2003.

[21] Hilaire, V.: Vers Une Approche de Spécification. De Prototypage et de Vérification de Systèmes Multi-Agents. Ph. D. thesis, UTBM, 2000.

[22] Larsen, A.: The Dynamic Vehicle Routing Problem. Ph. D. thesis, Technical University of Denmark, Lyngby, Denmark, 2000.

[23] Lau, H. C.—Liang, Z.: Pickup and Delivery with Time Windows: Algorithms and Test Case Generation. Proceeedings of 13th IEEE International Conference on Tools with Artificial Intelligence (ICTAI '01), Dallas, USA, 2001.

[24] Li, H.—Lim, A.: A Metaheuristic for the Pickup and Delivery Problem with Time Windows. In: Proceedings of 13th IEEE International Conference on Tools with Artificial Intelligence (ICTAI '01), Dallas, USA, 2001.

[25] Lim, H.—Lim, A.—Rodrigues, B.: Solving the Pick Up and Delivery Problem Using "Squeaky Wheel" Optimization with Local Search. Proceedings of American Conference on Information Systems, AMCIS 2002, Dallas, USA, 2002.

[26] Madsen, O. B. G.—Ravn, H. F.—Rygaard, J. M.: A Heuristic Algorithm for a Dial-A-Ride Problem With Time Windows, Multiple Capacities, and Multiple Objectives. Annals of Operations Research, Vol. 60, 1995, pp. 193–208.

[27] Mitrowic-Minic, S.: Pickup and Delivery Problem with Time Windows: A Survey. Technical Report, SFU CMPT TR, 1998-12, Available on ftp://fas.sfu.ca/pub/cs/techreports/1998.

[28] Nanry, W. P.—Barnes, J. W.: Solving the Pickup and Delivery Problem With Time Windows Using Reactive Tabu Search. Transportation Research Part B 34, 2000, Elsevier Science Ltd., pp. 107–121.

[29] Neagu, N.–Dorer, K.—Calisti, M.: Solving Distributed Delivery Problems with Agent-Based Technologies and Constraint Satisfaction Techniques. Dist. Plan and Schedule Management, 2006 AAAI Spring Symp., The AAAI Press, USA.

[30] Neagu, N.—Dorer, K.—Greenwood, D.—Calisti, M.: LS/ATN: Reporting on a Successful Agent-Based Solution for Transport Logistics Optimization. Proceedings of IEEE 2006 Workshop on Distributed Intelligent Systems (WDIS), June 15–16, 2006, Prague, Czech Republic.

[31] Potvin, J. Y.—Bengio, S.: The Vehicle Routing Problem With Time Windows — Part II: Genetic Search. INFORMS Journal on Computing, Vol. 8, 1996, pp. 165–172.

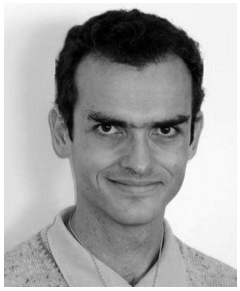
[32] Sandholm, T.: An Implementation of the Contract Net Protocol Based on Marginal Cost Calculations. In: Proceedings of AAAI-93, pp. 256–262, 1995.

[33] Smith, R. G.: The Contract Net Protocol: High-Level Communication and Control in a Distributed Problem Solver. IEEE Transactions on Computer, Vol. C-29, December, 1980, pp. 1104–1113.

[34] Solomon, M.: Algorithms for the Vehicle Routing and Scheduling Problems With Time Window Constraints. Operations Research, Vol. 35, 1987, pp. 254–265.

Jarosław Koźlak works as an Assistant Professor at the Department of Computer Science of the AGH University of Science and Technology in Krakow, Poland. He received his Ph. D. degree in computer science from the Institut National Polytechnique de Grenoble (INPG), France and AGH University of Science and Technology, Kraków, Poland in 2000. His research interests include multi-agent systems, transportation modelling and optimisation, integration of knowledge expressed using ontology description languages and social network analysis. He is an author or co-author of more than 70 scientific publications.

Jean-Charles Créput received his Ph. D. in computer science from University of Paris 6, France, in 1997. He defended his Habilitation in November 2008 at the University of Bourgogne, France. He is currently an Associate Professor in computer sciences and engineering at the University of Technology of Belfort-Montbeliard, France, and performs research activity at the Systems and Transportation (SeT) Laboratory. His current research interests include evolutionary algorithms, neural networks and multi-agent systems applied to telecommunications and intelligent transportation services.



Vincent Hilaire works as an Associate Professor, he gets his Ph. D. in computer science and his position in the University of Technology of Belfort Montbéliard in 2000. He got his Habilitation á Diriger les Recherches (HDR) degree from University of Franche Comté in 2008. The main concerns of the Ph. D. was formal specification and methodologies for engineering of Multi-Agent Systems (MAS). Since then his research is focused on MAS organizational theories and holonic systems, languages for formal specification prototyping and proofs of MAS, agent architectures and agent mediated knowledge management.

Abderrafiaa Koukam received the Ph. D. in computer science from University of Nancy I (France) in 1990, where he served as an Assistant Professor from 1986 to 1989 and researcher in Centre de Recherche en Informatique de Nancy (CRIN) from 1985 to 1990. In 1999, he received the authorisation to direct the research in computer science from University of Bourgogne (France). Presently, he is Professor of computer science at Université de Technologie de Belfort-Montbeliard (UTBM). He is the Director of Systems and Transportation Laboratory and heads research activities on modeling and analysis of complex systems, including software and knowledge engineering, multi-agent systems and optimization. He is the author or co-author of over 100 scientific publications.