# Adaptive real-world algorithm of solving MDVRPTW (Multi Depots Vehicle Routing Planning with Time Windows) problem

ARCHIL PRANGISHVILI, OTAR SHONIA, IRAKLY RODONAIA
Faculty of Informatics
Georgian Technical University
77 Kostava Str.,Tbilisi
ARTIOM MERABIAN
Faculty of Computer Technologies and Engineering
International Black Sea University
David Agmashenebeli Alley 13km, 2 0131, Tbilisi,
GEORGIA
irodonaia@yahoo.com http://gtu.edu.ge
merabianiartioma@ibsu.edu.ge

*Abstract:-* The adaptive algorithm to solve MDVRPTW problem is proposed in the paper. Realistic real-world situations, such as presence of various congestion types on roads, are carefully considered and accounted for in the algorithm. To overcome the lack of realistic and reliable methods of congestion duration estimation we use the MatSim large-scale agent-based simulation tool. This tool allows users to compose and run complex simulation models that are extremely close to the real-world situations. Our approach implements also autonomic components ensembles concept. Each vehicle is associated with the corresponding autonomic component AC (a virtual machine in datacenter) and exchange on-line information with other vehicles. Besides, ACs can reschedule routes in order to find the acceptable alternative routes that enable vehicles to meet time windows requirements and, at the same time, avoid the congested roads. The adaptive algorithm is able to reschedule and find alternative routed for several vehicle in parallel, which increases the performance of proposed approach.

*Key-Words:* vehicle routing planning, adaptive algorithms, congestion, multi-agent simulation, cloud computing, autonomic component, autonomic ensemble

## 1 Introduction

In the [1] the initial solution of the MDVRPTW (Multi Depots Vehicle Routing Planning with Time Windows) problem was described. The problem was solved by using the Adaptive Large Neighborhood Search (ALNS) framework. However, this is just the first stage of solution. The matter is that in realistic circumstances there are a lot of constraints and limitations, which prevent from the successful and practically acceptable solutions. Due to a growing amount of traffic and a limited capacity of the road network, *traffic congestion* has become a daily phenomenon [1]. Since traffic congestion cause heavy delays, it is very costly for intensive road users such as logistic service providers and distribution firms. In particular, such delays cause large costs for hiring the truck drivers and the use of extra vehicles, and if they are not accounted for in the vehicle route plans they may cause late arrivals at customers or even violations of driving hour's regulations. Therefore, accounting for traffic congestion has a large potential for cost savings. We have developed a modification of the ALNS algorithm (written in the *Jsprit* framework). Namely, our algorithm takes into account a probability of links' congestion, estimation of probability of their release of busy route sections. Our modification of the algorithm can plan routes for any starting and finishing nodes.
To provide the real-time adaptability the proposed approach uses the concept of *autonomic components (AC) and autonomic component ensembles (ACE)*[2]. ACs are entities with dedicated knowledge units and resources that can cooperate while playing different roles. ACs are dynamically organized into ACEs. AC members of an ACE are connected by the interdependency relations defined through predicates (used to specify the targets of communication actions. The functional description of an AC and ACE is shown on Fig.1

The process part of a component is split into an *autonomic manager* controlling execution of a *managed element*[2].] The autonomic manager monitors the state of the component, as well as the

execution context, and identifies relevant changes that may affect the achievement of its goals or the fulfillment of its requirements. It also plans adaptations in order to meet the new functional or non-functional requirements, executes them, and monitors that its goals are achieved, possibly without any interruption**.** A managed element can be seen as an empty "executor" which retrieves from the *knowledge* repository the process implementing a required functionality and bounds it to a process variables, the retrieved process for execution and waits until it terminates.

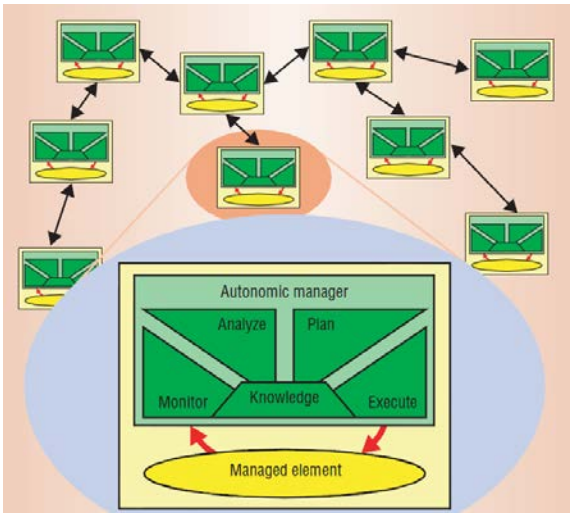


Fig.1 Functional description of a component

The ACs in an ACE may be implemented as *virtual machines (VMs)* in *datacenters (ACE)*. Each AC is associated with the concrete vehicle and comprehensive information of the current location of the vehicle on the route, relevant data on its current state and etc. In our approach the knowledge repository is used to store these data and exchange them with other ACs. Occasionally so called *spatial-temporal* event (that is, a vehicle arrives to a certain service point at a certain time) occurs. The equipment in the car (GPS receivers and GSM telephones (or some similar wireless communications technology)) determines location using the GPS receiver and sends the coordinates and other relevant data to the Web server. The general infrastructure of our approach id shown in the Fig.2

The ***base*** virtual machine $VM_0$ hosts all main structural components of proposed system: JSpirit, MatSim, travel and congestion management database (TCMD), database of simulation results (SRD), web servers for connection with vehicles, GPS, etc. Although $VM_0$ is permanently used and maintained, it is convenient to represent it as a virtual machine because it will intensively interact and exchange data with other virtual machines, each of which represents autonomic component (AC). As it will be shown later, autonomic components are associated with concrete vehicles and constitute an Autonomic Component Ensemble (ACE). The base $VM_0$ executes the initial solution of MDVRPTW problem and generates the initial set of routes RI. The input parameters, such as time windows for each service points, are held at $VM_0$ as well. After generating the initial set of routes, new virtual machines, enumerated from 1 to ***nr*** (where nr is the amount of routes in the initial set RI), are created. The recourses of the datacenter's servers are dynamically allocated to virtual machines.

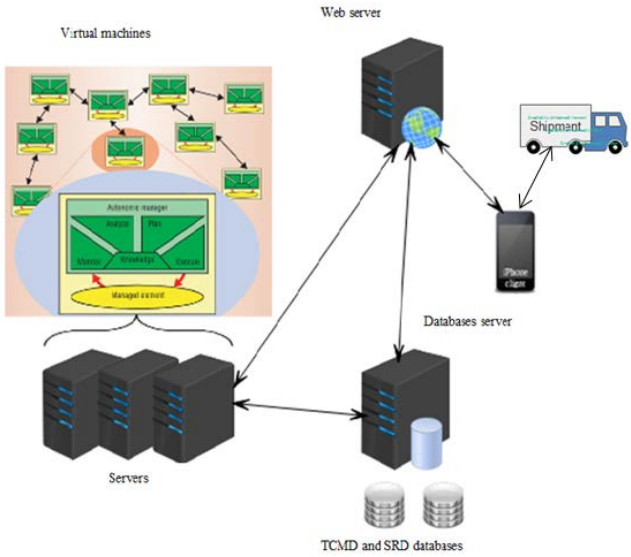


Fig.2 . General infrastructure

As it was said earlier, congestions are the most important critical factors for the successful and practically acceptable solution of the MDVRPTW problem. The congestion duration is defined as the total of detection/reporting, response, and clearance times. Due to the nature of most incident response mechanisms, the longer the incident has not been cleared, the more likely that it will be cleared in the next period. For example, the probability of an incident being cleared in the 15th min, given that it has lasted 14 min, is greater than the probability of it being cleared in the 14th min given that it has lasted 13 min [3, 4]. This is because it is more likely that someone has already reported the congestion and an congestion response team is either on the way or has already responded. Let ***t*** be the time to clear the congestion. Then, we have the increasing hazard (failure) rate (in our case – *clearance* rate) property, e.g., $\lambda(t+1) > \lambda(t)$, where $\lambda(t) = f(t)/(1-F(t))$ is the hazard rate of congestion clearance in duration ***t***, and $f(t)$ and $F(t)$ are the density and cumulative density functions of the clearance duration, respectively. The Weibull distribution with

increasing hazard rate is chosen to model the congestion clearance duration [3]. We assume that its starting time ($t^0_{cong}$), current status (i.e. cleared/not cleared), expected duration ($\mu$), and standard deviation ($\sigma$) are available through the travel and congestion management database (TCMD). However, this kind of real-world data is rarely available, especially at the beginning phase of VRPTWMD planning. In the case when these data do not yet exist for the current spatial-temporal unit, the responsibility of the corresponding AC of the ACE is to determine and register these parameters for the current vehicle and update the local copy of the TCMD. This action is performed in the following manner. If after arriving to the service point (with observed congestion in its outgoing arc) either of 3 above-said congestion types is observed, the decision whether to wait or reschedule and select an alternative acceptable route is to be made by using the algorithm described in details later in the paper text. If the decision to wait has been made, then the actual congestion-induce delay time is available (this is equal to the difference between time when the vehicle continued the movement on the arc and the starting time of congestion occurrence $t^0_{cong}$; this time $t^0_{cong}$ can be find out by interview with responsible persons (incident response team or road police) or got from available GSM data). This value is stored to the local copy of the TCMD. If the decision not to wait and reschedule has been made, then two possible solutions may be considered. The first one is to retrieve the relevant information from other ACs that corresponds to vehicles visited the service point of interest at subsequent time intervals. Of course, only visits associated with the congestion event under consideration (*current* spatial-temporal event) must be accounted for. This action is executed by the AM of the current AC by generating the SCEL query statement **qru** with the input arguments: current service point, current route, current vehicle, current time unit, congestion state). The **qru** is addressed to all relevant ACs that meet conditions[5]. The output of the statement **qru** is defined by the group-oriented communication predicate:

$$\Omega(\mathbf{X})=(\mathbf{X.Vehicle}_{ID} = \mathbf{List(Vehicles(Routes} \ni \mathbf{SP}_i) \wedge \mathbf{X.Arc}(i,k) \wedge \mathbf{X.Time\ (Time<CurrentTime))} \quad (1)$$

In other words, this predicate yields IDs of all relevant vehicles (*if any*). Additionally, congestion states at arrival times of all vehicles met input arguments' conditions, arrival times of all relevant vehicles can be obtained. We assume that these attributes are provided by the interface of each component and obtain dynamically updated values from corresponding probes (sensors) as a result of constant monitoring (sensing) of the ACE environment.

If at the arrival time of any of these vehicles to the service point (whose outgoing arc was earlier in congestion state) the arc is free of congestion, then the congested-induced time (see below in the text) is computed by the current AC and the corresponding record is stored to the local copy of the TCMD. The second solution of the problem (in the case when no vehicles visited the service point of interest) is to use results of simulation of the ***MatSim*** (*Multi-Agent Transport Simulation*) – a highly productive and efficient large-scale multi-agent simulation framework implemented in Java [6]. These results can be found in the special database of simulation results (SRD). The MatSim allows users to compose and run complex simulation models that are extremely close to the real-world situations. In particular, this language allows to enter various input parameters, obtain process simulation outputs and a great lot of parameters and distributions functions of simulated processes. Namely, we have simulated a large number of situations with various amounts of vehicles, roads, routes, service points, depots, loads, speeds, etc. The main types of possible congestions and incidents (cases, which are most characteristic for the actual conditions of Georgia - there are totally 18 cases (both recurrent and non-recurrent congestions)) – have been simulated. As a result, numerous characteristics, such as possible congestion durations (and their various distributions and parameters), congestion clearance times, road non-congested capacities, road congested capacities, vehicles' arrival rates, etc. for all roads and service points of Georgia, have been obtained and stored to the special simulation results database (SRD). These data can serve as a good approximation of real-world parameters and be used for preliminary estimation of processes of interest. It is to be pointed out that the Matsim simulation study of roads and service points in modes, which are maximally close to real conditions and comprehensively cover all possible states of the transport system, is the first and indispensable stage of the successful MDVRPTW problem solution. Careful consideration and identification of all the factors and parameters affecting the actual behavior of the transport system, inclusion them in the simulation models, statistically justified processing of simulation outputs and storing them to the special database of simulation results (SRD) is the

key to successful real-world solution of MDVRPTW problem with maximum consideration of the actual conditions. It is also to be pointed out that the MatSim simulation study is periodical and is invoked whenever the current situations lack the reliable data from the TCMD database and, therefore, running of simulation to obtain some approximated to the real changed circumstances information is necessary. In the later stages, when the database is already filled with real data, the MatSim's function should be minimized (to save computing power).

In addition to the above cases it is necessary to take into consideration one particular case. When a vehicle arrives to a service point whose outgoing arc is congested, it generates the query **qru**. This query is addressed to all vehicle executing other planned routes and having to visit in subsequent time intervals the service point indicated in the **qru**'s input arguments (and, additionally, being travelled on the arc incoming the service point under consideration). This query is received by all relevant ACs of vehicles. In this case the decision for these vehicles (which met conditions of the **qru**'s input parameters) whether to continue travelling on the remaining part of the arc or reschedule and select some alternative route is made by using the algorithm described below in the paper.

Hence, we can estimate the parameters of the Weibull distribution $\varphi$ of the congestion clearance duration using TCMD or SRD databases. The congestion-induced delay function $\Theta(\cdot)$ is based on the congestion duration $\varphi$, road non-congested capacity denoted with $c$ (vehicle per hour, or *vph* in short), road capacity during the congestion denoted with $\rho(vph)$, and arrival rate of vehicles to the congested arc denoted with $q(vph)$. Given these parameters for an congestion started at $t^0_{cong}$, the vehicle arriving to the congested arc at time t experiences the following expected congestion-induces delay[3]:

$$E(\Theta(\cdot) = (\frac{c-\rho}{c})(D_{12} - D_1 P_3) + P_2 d_m,$$

$$where\ D_{12} = \int_{D_1}^{D_2} x\varphi(x)dx, \quad D_1 = ((c-\rho)/(c-q))/(t - t^0_{cong}),$$

$$D_2 = (q/\rho)(t - t^0_{cong}),\ d_m = ((q-\rho)/\rho)/(t - t^0_{cong}),\ P_1 = \int_0^{D_2} x\varphi(x)dx,$$

$$P_2 = \int_{D_2}^{\infty} x\varphi(x)dx, \quad P_3 = 1 - (P_1 + P_2) \qquad (2)$$

We assume again that parameters $c, \rho, q$ (or their approximations) and distribution function $\varphi(t)$ are available at any current moment and at any current location of the vehicle ***i*** by querying the TCMD or SRD database in response of requirement of the corresponding $AC_i$

## 2 Adaptive rescheduling algorithm

Recall that the algorithm ALNS of finding the initial solution generates the set of routes RI; the amount of initial routes is ***nr***. Besides, the main priority of the proposed approach is non-violence of time windows for each service point of each generated routes. These time windows are defined before execution of the algorithm and entered as parameters to the matrix TW[nr,nsp], where ***nsp*** is the total amount of service points (nodes of the graph). In each row ***i*** an element ***j*** is equal to the given time window for the service point ***j*** only if this service point belongs to the route $r \in$ RI, otherwise it is 0.

Suppose that the vehicle $V_i$ arrives to the service point $SP_j$. Suppose also that the vehicle $V_i$ travels along the route $RI_r$ which was initially generated by the algorithm ALNS. The time of arrival is assumed equal to $t^r_{ij}$. The communication equipment on the board of $V_i$ sends the message to the autonomic component $AC_i$ associated with the vehicle $V_i$. The message delivers the above information to the $AC_i$, the information is stored to the knowledge database of the $AC_i$ and to the TCMD. Upon arrival the driver of $V_i$ finds out (by direct observation or by interviewing a congestion response team or road policy) the current situation: congestion status (congested or not) $Cong_{jk}$ of the next segment of the route $RI_r$ (the arc $A_{jk}$ connecting the current service points $SP_j$ and the next service points $SP_k$, both SPs belong to the route $RI_r$), the type of congestion (if any), starting time of the congestion occurrence $tcong^0_{jr}$, (i.e. the congestion that occurred on the arc $A_{jk}$ during execution of the route $RI_r$), the expected time $tcong^{clear}_{jr}$ of congestion clearance. We assume that the latter time can be clarified during the interview with the staff (congestion response team or road police) and with a certain probability; if this probability (in the staff's opinion) exceeds 0.5 (practically the most reliable and acceptable threshold), that time is taken as the base for further application in the algorithm. If the staff cannot determine that time even approximately or its confidence is below 0.5, the decision on generating an alternative route is made. The autonomic component $AC_i$ receives the above described information and retrieves also the additional information (the congestion duration $\varphi$, road non-congested capacity $c$, road capacity during

the congestion $\rho$, and arrival rate of vehicles to the congested arc $q$) from the TCMD or SRD (see the explanation of details above). The $AC_i$ computes the expected congestion-induces delay by using formulas (2) and determines the following important costs:

- $CDW_{jr}$ – cost to pay the driver that waits at the service point $SP_j$ and executes the route $\boldsymbol{r}$ (based on the computed expected congestion-induces delay)
- $CTW_{kr}$ –cost (penalty) of violation the time windows assigned to the $SP_k$ when executing the route $\boldsymbol{r}$ ( $\boldsymbol{k}$ is the end node of the arc)

If $CDW_{jr} \leq CTW_{kr}$ then the decision to stay at the $SP_j$ is made. Otherwise the decision of rescheduling and selecting an alternative route is made (the destination point of the route is the same as the destination point of the "old" route $\boldsymbol{r}$). Rescheduling is executed by the autonomic manager AM of the $AC_i$. Before executing the algorithm all already traversed service points are excluded, however, arcs, which income in and outgo from the excluded service point, are concatenated. It is necessary also to point out that by the moment of rescheduling (generating an alternative route) the complete picture of congestion states of all arcs at the current moment must be available. It can be obtained from congestion map received from GSM (as a rule , this information is available in real-time scale and is updated dynamically). Congested arcs must be marked and cannot be used when planning alternative routes. For this reason the congestion matrix MCong[m,n] (where m=1÷ NSP, n=1÷ NSP) is created and periodically updated. The values of elements of MCong are as follows: MCong[j,k]=-1 if the $A_{jk}$ does not exist, MCong[j,k]=0 if the arc $A_{jk}$ is not congested, MCong[j,k]=1 if the arc $A_{jk}$ is congested). Moreover, the partial rescheduling can be executed for several service points in parallel, because corresponding ACs of the ensemble ACE can run the rescheduling algorithms simultaneously and independently of each other. This is one of the advantages of the proposed approach.

After rescheduling it may turn out that there is no alternative route (the algorithm ALNS failed to find the alternative route for the current state of the system and given time windows); in this case the vehicle $V_i$ must simply wait at the $SP_j$.

Regardless of whether the vehicle continued along the "new" or "old" route's segment , or it waits at the service point $SP_j$ , we assume that service of the $SP_j$ is completed. The actual service time is registered. If this time is equal or less of the required time window for the $SP_j$, then we increment the amount of successfully served service points: NSSP++, otherwise we increment the amount of overdue (delayed) serviced service point: OSSP++. At the end of planning horizon (as a rule, starting time of planning is 8:00 of each day, ending time is 24:00) the number of successfully and overdue service points is displayed and stored to the TCMD ( for statistical processing and performance evaluation).

Now consider another option. As it was mentioned, the vehicle $V_i$ upon arrival at the $SP_j$ sends the message to the autonomic component $AC_i$ associated with the vehicle $V_i$. Then the $AC_i$ by implementing the query statement **qry** propagates information to all relevant ACs, defined by the group-oriented communication predicate (1). The arrival time of those vehicles are less than the arrival time $t_{ij}^{r}$. Hence, at the moment of receiving the propagated message vehicles are somewhere on the corresponding arcs. The decision whether to continue travelling on the remaining part of the arc or reschedule and select some alternative route is made similarly to the algorithm described above. The only difference is that the costs of travelling along the remaining part of the arcs are computed (instead of computing the cost of travelling along the whole arc $A_{jk,}$ connected the current service points $SP_j$ and the next service points $SP_k$).

Pseudo-code of proposed adaptive algorithm:

**input**: nsp: number of service points; RTW[nsp]: required time windows array;
PH[2]: planning horizon (starting and ending times);
nv: amount of available vehicles; np: amount of given characteristics of vehicles;
VP[nv,np]: vehicles characteristics array;
TCMD: travel and congestion management database ;
SRD: MatSim simulation results database;
**Output**: nr: amount of generated routes; RI[r]: set of generated routes;
NSSP: amount of successfully served service points;
OSSP: amount of overdue (delayed) serviced service points

**1**. **while** CurrentTime < PH[2] **do**:
**2.** spatial-temporal event occurs (a vehicle arrives to a service point at a certain time)
**3.** t← current time
**4.** i← number of a vehicle arrived to some service point at time t

**5.** j← number of service point SP at which the vehicle $V_i$ arrived at time t
**6.** r← number of route along which the vehicle $V_i$ travels at time t
**7.** k← number of SP, connected to $SP_j$ in route r
**8.** MCong[j,k] ← congestion status of arc $A_{jk}$
**9.** TW[r,j] ← required time window for the service point **j** in the route **r**
**10.** $CT_{jr}$ ← type of congestion;
**11.** $tcong_{jr}^{0}$ ← starting time of congestion occurrence
**12.** $tcong_{jr}^{clear}$ ← expected time of congestion clearance
**13.** $\varphi$← congestion duration; $c$← road non-congested capacity; $\rho$← road capacity during the congestion; $q$ ← arrival rate of vehicles to the congested arc
**14.** Autonomic component $AC_i$ receives a message from vehicle $V_i$
**15.** $CDW_{jr}$← cost to pay the driver that waits at the service point $SP_j$ and executes the route ***r*** (see formulas 2)
**16.** $CTW_{kr}$← cost (penalty) of violation the time windows assigned to the $SP_k$ when executing the route ***r*** (see formulas 2)
**17.** **if** $CDW_{jr} \leq CTW_{kr}$ **then**
**18.** decision: wait at $SP_j$
**19.** **else**
decision: reschedule and generate new route ***r**** (by using the modified ALNS algorithm)
**21.** **end if**
**22.** **if *r** is null then**
**23.** decision: wait at $SP_j$
**24.** **end if**
**25.** NSSP++ or OSSP++
**23.** store obtained results to TCMD
**24. end while**
**25.** display obtained results

## 3. Conclusions

In our paper we described the adaptive algorithm to solve MDVRPTW problem. The algorithm is aimed to account for realistic real-world situation, such as presence of various congestion types. The congestions are the most important critical factors for the successful and practically acceptable solution of the MDVRPTW problem. Since the realistic estimation of congestion duration is rather difficult and non-standard problem, we use the MatSim large-scale agent-based simulation tool which allows users to compose and run complex simulation models that are extremely close to the real-world situations. In particular, this language allows to enter various input parameters, obtain process simulation outputs and a great lot of parameters and distributions functions of simulated processes. Another distinctive feature of our approach is the use of autonomic components ensembles. Each vehicle is associated with the corresponding autonomic component AC (implemented as a virtual machine in datacenter) and exchange on-line information with other vehicles. This allows a vehicle to notify other vehicles about expected and actual congestion. Besides, ACs can reschedule routes in order to find the acceptable alternative routes that enable vehicles to meet time windows requirements and, at the same time, avoid the congested roads. It is necessary to point out that the algorithm of adaptation is able to reschedule and find alternative routed for several vehicle in parallel. The latter significantly increases the performance of proposed approach.

*REFERENCES:*

[1]. I.Rodonaia, A. Merabian. Real-world applications of the vehicle routing problem in Georgia. Proceedings of the International Black Sea University, Journal Technical Science & Technologies (JTST), November, 2016
[2]. A.Prangishvili, O.Shonia, I.Rodonaia, M. Mousa. Formal verification in autonomic-component ensembles, WSEAS / NAUN International Conferences, Salerno, Italy, 2014
[3]. Ali R. Guner, Alper Murat, Ratna Babu Chinnam. Dynamic routing under recurrent and non-recurrent congestion using real-time ITS information, Computers & Operations Research 39 (2012) 358–373
[4]. S. Kritzinger, K.F. Doerner, F. Ttricoire, R.F. Hartl. Adaptive search techniques for problems in vehicle routing,Part 1: a survey. Yugoslav Journal of Operations Research 25 (2015), Number 1, 3–31
[5]. Rocco De Nicola, Michele Loreti, Rosario Pugliese, Francesco Tiezzi. "SCEL- a Language for Autonomic Computing". ASCENS project , Technical report, January 2013
[6]. MATSim (2016) Multi-Agent Transportation Simulation, webpage, http://www.matsim.org