Influence of the Waiting Strategy on the Performance of the Multi-Agent Approach to the DVRPTW

Dariusz Barbucha(✉)

Department of Information Systems, Gdynia Maritime University,
Morska 83, 81-225 Gdynia, Poland
d.barbucha@wpit.am.gdynia.pl

Abstract. A multi-agent approach to the Dynamic Vehicle Routing Problem with Time Windows has been proposed in the paper. The process of solving instances of the problem is performed by a set of software agents. They are responsible for managing the sets of dynamic requests, allocating them to the available vehicles, and optimizing the routes covered by the vehicles in order to satisfy several requests and vehicles constraints. The paper focuses on waiting strategies which aim at deciding whether a vehicle should wait after servicing a request, before heading toward the next customer. The influence of the proposed waiting strategy on the performance of the approach has been investigated via a computational experiment. It confirmed the positive impact of the strategy on the obtained results.

Keywords: Dynamic vehicle routing problem with time windows · Waiting strategy · Multi-agent system

1 Introduction

Dynamic Vehicle Routing Problems class (DVRPs) refers to a group of routing problems that the required information about customers, vehicles, etc. is not given a priori to the decision maker but is revealed concurrently with the process of decision-making. The dynamism can be represented for example by stochastic vehicle speed depending on road's condition or stochastic demand of customers. On the other hand, recent advances in development and possibility of application of different communication and information technologies (global positioning systems, wireless networks, etc.) have allowed transportation companies to benefit from real-time information and to plan vehicles routes in more efficient way.

The most important variants of DVRPs include: classical Dynamic Vehicle Routing Problem (DVRP), where a set of customers' requests has to be served by a set of vehicles in order to minimize the cost of transport, and satisfying several customers and vehicles constraints, Dynamic Vehicle Routing Problem with Time Windows (DVRPTW), where the customers have to be visited during

N.T. Nguyen et al. (Eds.): ICCCI 2016, Part I, LNAI 9875, pp. 464–473, 2016.
DOI: 10.1007/978-3-319-45243-2_43

a specific time interval, Dynamic Pickup and Delivery Problem (DPDP), where goods have to be either picked-up or delivered in specific amounts in each of customer location, and Dynamic Pickup and Delivery Problem wit Time Windows (DPDPTW), a variant of DPDP with time windows. A review of different variants of DVRPs, methods of solving them and examples of practical applications can be found for example in [12].

The paper focuses on Vehicle Routing Problem with Time Windows (VRPTW). The *static* version of it can be formulated as an undirected graph $G = (V, E)$, where $V = \{0, 1, \ldots, N\}$ is a set of nodes and $E = \{(i, j)|i, j \in V\}$ is a set of edges. Node 0 is a central depot with K identical vehicles of capacity W. Each node $i \in V \setminus \{0\}$ denotes a customer characterized by a non-negative demand d_i, and a service time s_i. Moreover, with each customer $i \in V$, a time window $[e_i, l_i]$ wherein the customer has to be supplied, is associated. Here e_i is the earliest possible departure (ready time), and l_i - the latest time the customer's request has to be started to be served. The time window at the depot ($[e_0, l_0]$) is called the scheduling horizon. Each edge $(i, j) \in E$ denotes the path between customers i to j and is described by the cost c_{ij} of travel from i to j by shortest path ($i, j \in V$). It is assumed that $c_{ij} = c_{ji} (i, j \in V)$. It is also often assumed that c_{ij} is equal to travel time t_{ij}.

The goal is to minimize the vehicle fleet size and the total distance needed to pass by vehicles in order to supply all customers (minimization of the fleet size is often considered to be the primary objective of the VRPTW). The following constraints have to be also satisfied: each route starts and ends at the depot, each customer $i \in V \setminus \{0\}$ is serviced exactly once by a single vehicle, the total load on any vehicle associated with a given route does not exceed vehicle capacity, each customer $i \in V$ has to be supplied within the time window $[e_i, l_i]$ associated with it (the vehicle arriving before the lower limit of the time window causes additional waiting time on the route), and each route must start and end within the time window associated with the depot.

The *dynamic* version of the VRPTW considered in the paper assumes that customers' requests are not known in advance but they are revealed dynamically and unpredictably during the execution of already arrived requests. Let the planning horizon starts at time 0 and ends at time T. Let $t_i \in [0, T]$ $(i = 1, \ldots, N)$ denotes the time when the i-th customer request is submitted. Following the *degree of dynamism* measure $dod = N_d/N$ [9] (N_d - number of dynamic requests, N - number of all requests), the problem considered in the paper is fully dynamic ($dod = 1$).

The main contribution of the paper is to propose an approach based on a multi-agent paradigm to the DVRPTW with a waiting strategy implemented within the approach. The paper aims at investigation of the influence of the waiting strategy on the performance of the proposed approach. The approach presented in the paper extends the multi-agent environment for solving DVRP and DVRPTW proposed by author in [1,2].

The rest of the paper is divided on four sections. Section 2 includes a review of different waiting strategies implemented in frameworks proposed by other

authors to solve different variants of VRP. Section 3 presents the multi-agent approach with the waiting strategy to the DVRPTW. Goal, assumptions and results of the computational experiment are presented in Sect. 4. Finally, Sect. 5 includes main conclusions and directions of the future research.

2 Waiting Strategies

Different strategies have been implemented within the approaches dedicated to solve Vehicle Routing Problems in order to improve their performance. The most known ones refer to *request buffering* and *vehicle waiting*. The *request buffering* strategy (proposed by Pureza and Laporte [13] and by Mitrovic-Minic et al. [11] for dynamic Pickup and Delivery Problems with Time Windows) aims at postponing a request assignment decision by storing some requests in a buffer. It means that allocation of each new request to one of the available vehicles is not performed immediately whenever the new request arrives. After arriving, the request is stored in the buffer, and it is considered to allocate at later stages. It is expected that by buffering the requests, better routing decisions are more likely to be achieved due to the larger number of accumulated requests available [13]. The *waiting strategy* [10] aims at deciding whether a vehicle should wait after servicing a request, before driving toward the next customer. This strategy is particularly important in problems with time windows, where time lags may appear between requests. It is expected that using information about the likely location of new customers, better decisions may be taken. Besides the waiting after servicing a customer, a vehicle can be also relocated (positioned) to a strategic position where probability of occurrence of new customers is higher.

The waiting strategies have been implemented in different frameworks for the DVRP [6], DVRPTW [4,5,8], and DPDPTW [10,13], where authors looked at the potential benefit of applying these strategies. Branke et al. [6] considered a standard DVRP, where one additional customer arrives at a beforehand unknown location when the vehicles are already under way. Their objective was to maximize the probability that the additional customer can be integrated into one of the fixed tours without violating time constraints. This was achieved by allowing the vehicles wait at suitable locations during their tours in order to maximize the probability that a new customer, appearing anywhere in the service region, can be integrated into one of the tours. The authors proposed several waiting strategies and an evolutionary algorithm to optimize the waiting strategy. Empirical comparison of the strategies allowed them to conclude that a proper waiting strategy can greatly increase the probability of being able to service the additional customer, at the same time reducing the average detour to serve that customer.

Another approach to increase the probability of servicing future unknown customers has been proposed by Mitrovic-Minic and Laporte [10]. They examined whether waiting strategies can reduce the total detour or the number of required vehicles for DPDPTW. They adapted a tabu search procedure proposed by Gendreau et al. [7] and suggested four waiting strategies: drive-first, wait-first,

dynamic waiting, and advanced dynamic waiting. The results of computational experiment allowed them to discover benefits of applying these strategies within their approach and to point the last strategy as the most efficient.

Bent and Hentenryck [4] considered online Stochastic Multiple Vehicle Routing Problem with Time Windows in which requests arrive dynamically and the goal is to maximize the number of serviced customers. Contrary to many other algorithms which only move vehicles to known customers, they investigated waiting and relocation strategies in which vehicles may wait at their current location or being relocated to arbitrary sites. The decisions (to wait and/or to relocate) did not exploit any problem-specific features but rather were obtained by including choices in the online algorithm that are necessarily sub-optimal in an offline setting. Experimental results showed that waiting and relocation strategies may dramatically improve customer service, especially for problems that are highly dynamic and contain many late requests.

The approach proposed by Ichoua et al. [8] focused on DVRPTW. It exploited probabilistic knowledge about future events to better manage the fleet of vehicles and provide good coverage of the territory. They extended a parallel tabu search heuristic developed by Gendreau et al. [7] by allowing a vehicle to wait in its current zone instead of driving to the next planned destination located in another zone. The waiting strategy determined the interval of time a vehicle should wait at the current position, however, waiting was only allowed if the probability of a future request reached a particular threshold. Computational tests performed by authors showed that the inclusion of the waiting strategy improves the tabu search performance.

Pureza and Laporte [13] proposed a constructive-deconstructive heuristic for the DPDP with time windows and random travel times, where two kinds of strategies (request buffering and vehicle waiting) have been implemented within it. The waiting strategy based on fastest paths (WE_FP) took advantage of the fact that it was possible to reach a location earlier by using an indirect path instead of the direct path. Since the basic approach has already used a waiting strategy that allowed to arrive at the beginning of the time window of the locations, the extra time provided by faster paths was used to anticipate arrivals (when applicable), or to extend waiting after service. Comparisons of the quality of solutions obtained by an implementation of this strategy to an approach without it confirmed the advantages of this strategy both in terms of lost requests and number of vehicles.

Branchini et al. [5] proposed an adaptive granular local search heuristic for DVRP, where the strategies of vehicle waiting, positioning a vehicle in a region where customers are likely to appear, and diverting a vehicle away from its current destination have been integrated. They implemented the wait-first strategy proposed by Mitrovic-Minic and Laporte [10]. Good performance of the proposed approach has been confirmed by computational experiment performed by them on test problems derived from real-life Brazilian transportation companies.

3 Agent-Based Approach to DVRPTW

The proposed approach uses a multi-agent platform proposed by the author for simulating and solving the dynamic VRP/VRPTW [1,2]. The architecture of the platform is based on Java Agent Development Framework (JADE) [3], where several autonomous agents are defined:

- `GlobalManager` - an agent which initializes and destroys all agents,
- `RequestGenerator` - an agent which is responsible for generating (or reading from a file) new customers' requests,
- `RequestManager` - an agent which manages the list of customers' requests and coordinates the activity of other agents,
- `Vehicle` agents - represent vehicles which are responsible for serving the customers' requests.

The steps of the approach are presented in Algorithm 1. Because of the fact that all requests arrive while the process of solving the problem is ongoing, the most important activities refer to handling the newly arrived requests. Hence, the Algorithm 1 emphasizes this part of the process.

Algorithm 1. Main steps of the algorithm MAS-DVRPTW

1: The system initializes the `GlobalManager` agent, which next initializes other agents: `RequestsGenerator`, `RequestsManager`, and `Vehicle`
2: `RequestsManager` is waiting for events: $event = getEvent()$
3: **while** ($event$! = `endOfRequests`) **do**
4: `RequestGenerator` agent generates (or reads) a new request and sends it to the `RequestManager` (`newRequest` event)
5: if `RequestManager` receives a message with a new request then it creates/updates the routing plan R_s by (re-)solving the problem P_{DVRPTW} taking into account the requests do not assigned to any `Vehicle` agents (s is a new solution of P_{DVRPTW})
6: On the other hand, if `Vehicle` agents reach the locations of their current customers, they inform the `RequestManager` about their readiness for serving next requests (`vehicleStopAtLocation` event)
7: According to the current routing plan, the `RequestManager` allocates next requests to the `Vehicle` agents using *Centralized Dispatching Strategy* combined with *Waiting Strategy*.
8: `RequestsManager` is waiting for next events: $event = getEvent()$
9: **end while**
10: All vehicles finishes their routes and drive back to the depot

The process of simulating and solving the DVRPTW starts with initialization of all variables (states of requests, vehicles, etc.) and creation of the above mentioned agents. When all agents report their readiness to act, the system is waiting for next events. Although several events are generated by the agents and many messages are sent between them, the most important ones refer to

requests (announcing arriving a new request, end of requests, etc.), and to vehicles (messages reporting about their states, for example reaching a customer, waiting before servicing at a given location, etc.).

During the whole process of simulating and solving the DVRPTW, the new requests arrive to the system. All received requests are collected and maintained by the `RequestManager` agent, and next they are allocated to the `Vehicle` agents, which have just announced their readiness to act. The process of allocating the requests to the available `Vehicle` agents is performed according to the Centralized Dispatching Strategy (CDS) defined in [2].

In particular, when the `newRequest` event (sent by the `RequestGenerator`) is received by the `RequestManager` agent, it creates or updates the current global routing plan $R_s = [R_s^1, R_s^2, \ldots, R_s^K]$ by resolving the problem P_{DVRPTW} including the requests which have not been assigned to any `Vehicle` agents (s is a new solution of P_{DVRPTW}). A procedure of *cheapest insertion* of this request to the existing routes is performed. Moreover, the `RequestManager` performs a few local improvements of the current solution taking into account the requests which have not been yet assigned to the vehicles [2].

On the other hand, at any iteration, let $v(i)$ be a `Vehicle` agent, let $R^{v(i)} = [r_1^{v(i)}, r_2^{v(i)}, \ldots, r_k^{v(i)}]$ be a current route assigned to the vehicle $v(i)$ $(i = 1, \ldots, K)$ (it contains the customers already visited by vehicle), and $r_k^{v(i)}$ is a customer (location) the vehicle $v(i)$ is currently driving to. After reaching a location of the customer r_k^i and finishing its service, the `Vehicle` agent $v(i)$ sends the message `vehicleStopAtLocation` to the `RequestManager` informing it about readiness for serving next requests. According to the current global routing plan, the `RequestManager` agent sends the next request to the `Vehicle` agent $v(i)$. The request is inserted on position $k+1$ of the route of $v(i)$ agent.

The algorithm stops when the set of requests has been exhausted (message `endOfRequests`). Then all vehicles finish their mission and return to the depot.

Fundamental version of the proposed approach assumes that each `Vehicle` agent, after finishing its partial route at some location, immediately leaves it and starts driving to the next planned customer, according to the global routing plan. Following the constraint of the VRPTW that each customer has to be supplied within the time window associated with it, the vehicle arriving before the lower limit of the time window has to wait before servicing a customer at this location. In this case, the waiting time w_i of the vehicle *before* servicing i-th request can be calculated as $w_i = (e_i - t_i)$, where t_i is time of arrival the vehicle to the customer i.

Following the observations provided by a few authors and included in Sect. 2, it has been also decided to extend the above approach by implementing the strategy which aim at deciding whether a vehicle should wait also *after* servicing a request, before heading toward the next customer. The proposed strategy, called *waiting strategy*, is inspired by Mitrovic-Minic and Laporte (wait-first) strategy applied by them to the DPDPTW [10]. Its general assumption is that vehicle should leave current location at the latest possible departure time.

4 Computational Experiment

A computational experiment has been carried out to evaluate the influence of the waiting strategy on the performance of the proposed multi-agent approach to the DVRPTW. The performance of the approach has been measured by the number of vehicles needed to serve all requests in the predefined time and the total distance needed to pass by these vehicles.

The approach was tested on 56 classical VRPTW instances of Solomon [14] (available at [15]) including 100 customers, each, which have been transformed into the dynamic version through revealing all requests dynamically. The whole set of instances is divided into six groups (R1, R2, C1, C2, RC1, RC2) including customers with randomly generated coordinates (R1, R2), clustered coordinates (C1, C2) or both (RC1, RC2). Additionally, instances belonging to R1, C1, and RC1 have a short scheduling horizon, whereas the instances from R2, C2 and RC2 have a long scheduling horizon.

It has been assumed that requests may arrive with various frequencies, and in the experiment arrivals of the dynamic requests have been generated using the Poisson distribution with λ parameter denoting the mean number of requests occurring in the unit of time (1 h in the experiment). For the purpose of experiment it has been assumed that: λ is equal to 5, 10, 15, and 30, all requests have to be served, and the vehicle speed is set to 60 Km/h.

In order to discover the influence of the proposed waiting strategy on the obtained results, two cases have been tested and compared in the experiment:

- NO-WAIT - it assumes that each `Vehicle` agent does not consider waiting at its current location after servicing a request. It means that the `Vehicle` agent after finishing its partially route at some destination, it immediately leaves this location and starts driving to the next customer assigned to it,
- WAIT - the case where the waiting strategy has been applied. It means that each `Vehicle` agent after finishing its partially route at some destination, it waits a period of time and then starts driving to the next planned customer. This strategy assumes that the vehicle leaves its current location at the latest possible departure time.

For each case, each instance was repeatedly solved five times and mean results from these runs were recorded. All simulations have been carried out on PC Intel Core i5-2540M CPU 2.60 GHz with 8 GB RAM running under MS Windows 7.

The experiment results for both cases (NO-WAIT and WAIT) are presented in Table 1. The table includes the following columns: the name of the instance set, frequencies of arrivals of new requests (λ), and the results (the average number of vehicles used, the average distance traveled by all vehicles, and difference in % between results obtained for both cases).

Analysis of the results presented in the table provides several interesting conclusions. The first observation which stems from the dynamization of the problem is that the results for dynamic instances of the VRPTW are worse that the results obtained for their static counterparts (best known solutions identified by heuristics averaged for each group of instances the reader can find

Table 1. Results obtained by the proposed multi-agent approach (cases: NO-WAIT, WAIT)

		NO-WAIT		WAIT		Difference	
Instance	λ	#Vehicles	Distance	#Vehicles	Distance	#Vehicles	Distance
R1	5	13.58	2114.09	14.12	2092.95	-4 %	1 %
	10	13.54	1736.71	14.08	1719.34	-4 %	1 %
	15	13.32	1469.36	13.72	1425.28	-3 %	3 %
	30	13.09	1261.63	13.35	1236.40	-2 %	2 %
C1	5	10.97	1532.50	11.08	1517.18	-1 %	1 %
	10	10.55	1272.35	10.97	1246.90	-4 %	2 %
	15	10.00	870.94	10.50	853.52	-5 %	2 %
	30	10.00	853.50	10.10	836.43	-1 %	2 %
RC1	5	12.42	2336.13	13.17	2312.77	-6 %	1 %
	10	13.40	1810.14	13.53	1773.94	-1 %	2 %
	15	13.00	1516.53	13.13	1486.20	-1 %	2 %
	30	13.00	1489.81	13.13	1445.12	-1 %	3 %
R2	5	3.87	1827.66	3.99	1809.38	-3 %	1 %
	10	3.50	1191.66	3.71	1179.74	-6 %	1 %
	15	3.48	1171.94	3.55	1148.50	-2 %	2 %
	30	3.81	982.37	3.89	962.72	-2 %	2 %
C2	5	3.00	1103.04	3.09	1092.01	-3 %	1 %
	10	3.00	899.72	3.27	881.73	-9 %	2 %
	15	3.00	718.83	3.09	697.27	-3 %	3 %
	30	3.00	627.72	3.06	621.44	-2 %	1 %
RC2	5	4.21	2137.96	4.63	2116.58	-10 %	1 %
	10	4.21	1800.48	4.55	1782.48	-8 %	1 %
	15	4.04	1276.20	4.12	1250.68	-2 %	2 %
	30	4.24	1149.08	4.37	1137.59	-3 %	1 %

for example in [15] or [2]). Deterioration of the results often depends on the frequency of request arrivals (λ). General observation is that low ratio of request arrivals implies that the results are worse in comparison with the case where a lot of customers arrive at early stage of computation.

By comparison of the results obtained by the approach for both cases, one can conclude that waiting strategy (WAIT case) allows the algorithm to build shorter routes when compare to the NO-WAIT case for all tested instances. By allowing the vehicles to wait at their early locations, it is expected that more requests are being known at the time they leave their current locations. Moreover, the greatest reduction of the routes length has been observed for cases when new

requests arrive often ($\lambda = 15, 30$) than in cases when $\lambda = 5, 10$. The observed reduction is up to 3 %.

On the other hand, the number of required vehicles, in case of WAIT strategy, is greater than the number of vehicles required by the system without implementation of the waiting strategy (NO-WAIT case). The observed increase ranges from 1 % to 10 %. The greatest increase is observed for cases when new requests arrive rather slowly ($\lambda = 5, 10$).

5 Conclusions

The multi-agent approach to the Dynamic Vehicle Routing Problem with Time Windows has been proposed in the paper. The architecture of the system includes the set of software agents with different roles and abilities. In order to increase the efficiency of the process of solving the DVRPTW, the waiting strategy has been also implemented in the system. It aims at deciding whether a vehicle should wait (or not) after servicing a request, before driving toward the next customer. The computational experiment which has been carried out on several benchmark instances allowed one to investigate the impact of waiting strategy on the performance of the system. It confirmed outperformance of the version with waiting strategy implemented within the system over the version without waiting strategy in terms of the total distance, but not necessarily in terms of the number of vehicles.

Future research will aim at an implementation of the proposed waiting strategy to other problems (for example DPDPTW), implementation of different variants of waiting strategies, and integration of waiting strategy with other reported efficient strategies (for example request buffering [11,13]).

References

1. Barbucha, D., Jędrzejowicz, P.: Agent-based approach to the dynamic vehicle routing problem. In: Demazeau, Y., Pavón, J., Corchado, J.M., Bajo, J. (eds.) 7th International Conference on Practical Applications of Agents and Multi-Agent Systems (PAAMS 2009). AISC, vol. 55, pp. 169–178. Springer, Heidelberg (2009)
2. Barbucha, D.: A multi-agent approach to the dynamic vehicle routing problem with time windows. In: Bădică, C., Nguyen, N.T., Brezovan, M. (eds.) ICCCI 2013. LNCS, vol. 8083, pp. 467–476. Springer, Heidelberg (2013)
3. Bellifemine, F., Caire, G., Greenwood, D.: Developing Multi-Agent Systems with JADE. John Wiley & Sons, Chichester (2007)
4. Bent, R., Van Hentenryck, P.: Waiting and relocation strategies in online stochastic vehicle routing. In: Veloso, M. (ed.) Proceedings of the 20th International Joint Conference on Artificial Intelligence (IJCAI-07), pp. 1816-1821 (2007)
5. Branchini, R.M., Armentano, A.V., Lokketangen, A.: Adaptive granular local search heuristic for a dynamic vehicle routing problem. Comput. Oper. Res. **36**(11), 2955–2968 (2009)
6. Branke, J., Middendorf, M., Noeth, G., Dessouky, M.: Waiting strategies for dynamic vehicle routing. Transp. Sci. **39**(3), 298–312 (2005)

7. Gendreau, M., Guertin, F., Potvin, J.-Y., Taillard, E.: Parallel tabu search for real-time vehicle routing and dispatching. Transp. Sci. **33**(4), 381–390 (1999)
8. Ichoua, S., Gendreau, M., Potvin, J.-Y.: Exploiting knowledge about future demands for real-time vehicle dispatching. Transp. Sci. **40**(2), 211–225 (2006)
9. Larsen, A.: The dynamic vehicle routing problem. Ph.d. thesis, Institute of Mathematical Modelling, Technical University of Denmark (2001)
10. Mitrovic-Minic, S., Laporte, G.: Waiting strategies for the dynamic pickup and delivery problem with time windows. Transp. Res. Part B **38**, 635–655 (2004)
11. Mitrovic-Minic, S., Krishnamurti, R., Laporte, G.: Double-horizon based heuristics for the dynamic pickup and delivery problem with time windows. Transp. Res. Part B **38**, 669–685 (2004)
12. Pillac, V., Gendreau, M., Guret, C., Medaglia, A.L.: A review of dynamic vehicle routing problems. Eur. J. Oper. Res. **225**, 1–11 (2013)
13. Pureza, V., Laporte, G.: Waiting and buffering strategies for the dynamic pickup and delivery problem with time windows. INFOR **46**(3), 165–175 (2008)
14. Solomon, M.: Algorithms for the vehicle routing and scheduling problems with time window constraints. Oper. Res. **35**, 254–265 (1987)
15. Solomon, M.: VRPTW Benchmark problems. http://w.cba.neu.edu/msolomon/problems.htm