CIRRELT

Centre interuniversitaire de recherche
sur les réseaux d'entreprise, la logistique et le transport

Interuniversity Research Centre
on Enterprise Networks, Logistics and Transportation

A Dynamic Vehicle Routing Problem with Multiple Delivery Routes

Nabila Azi
Michel Gendreau
Jean-Yves Potvin

October 2010

CIRRELT-2010-44

Bureaux de Montréal :
Université de Montréal
C.P. 6128, succ. Centre-ville
Montréal (Québec)
Canada H3C 3J7
Téléphone : 514 343-7575
Télécopie : 514 343-7121

Bureaux de Québec :
Université Laval
2325, de la Terrasse, bureau 2642
Québec (Québec)
Canada G1V 0A6
Téléphone : 418 656-2073
Télécopie : 418 656-2624

www.cirrelt.ca

A Dynamic Vehicle Routing Problem with Multiple Delivery Routes

Nabila Azi[1,2], Michel Gendreau[1,3], Jean-Yves Potvin[1,2,*]

[1] Interuniversity Research Centre on Enterprise Networks, Logistics and Transportation (CIRRELT)

[2] Department of Computer Science and Operations Research, Université de Montréal, P.O. Box 6128, Station Centre-ville, Montréal, Canada H3C 3J7

[3] Department of Mathematics and Industrial Engineering, École Polytechnique de Montréal, P.O. Box 6079, Station Centre-ville, Montréal, Canada H3C 3A7

Abstract. This paper considers a vehicle routing problem where each vehicle performs delivery operations over multiple routes during its workday and where new customer requests occur dynamically. The proposed methodology for addressing the problem is based on an adaptive large neighborhood search heuristic, previously developed for the static version of the problem. In the dynamic case, multiple possible scenarios for the occurrence of future requests are considered to decide about the opportunity to include a new request into the current solution. It is worth noting that the real-time decision is about the acceptance of the new request, not about its service which can only take place in some future routes (a delivery route being closed as soon as a vehicle departs from the depot). In the computational results, a comparison is provided with a myopic approach which does not consider scenarios of future requests.

Keywords. Dynamic vehicle routing, multiple routes, scenarios, acceptance rule, adaptive large neighborhood search.

Acknowledgements. Financial support for this work was provided by the Natural Sciences and Engineering Council of Canada (NSERC). This support is gratefully acknowledged.

---

* Corresponding author: Jean-Yves.Potvin@cirrelt.ca

Dépôt légal – Bibliothèque et Archives nationales du Québec,
Bibliothèque et Archives Canada, 2010

1 Introduction

A relatively recent development in the field of vehicle routing relates to the study of dynamic variants, where information about the problem is revealed as the current routes are executed by the vehicles. In general, the dynamic aspect comes from the occurrence of new customer requests, although a few papers address other types of events, like dynamic travel times (see, for example, [6, 11]).

In this paper, a previous algorithm developed for a vehicle routing problem with multiple delivery routes is applied in a dynamic setting where customer requests occur dynamically and must be responded to in real-time. It is worth noting that the most stringent real-time decision is about accepting or not a new request, not about finding the best possible way to integrate it into the current solution, since the request can only be included in some future route (a delivery route being closed as soon as the corresponding vehicle departs from the depot). This is illustrated in Figure 1 for a single vehicle's workday. When the new customer request is received, the vehicle is currently moving between customers i and j in route 1. Given that the vehicle has already departed from the depot (black square) to execute route 1, this route is closed. Only planned routes 2 and 3, which contain customers that have been assigned to the vehicle but will only be served after the return of the vehicle to the depot, can accept the new request. This problem is inspired from e-grocery applications where perishable goods are delivered to customers, thus leading to multiple short vehicle routes where the last customer in each route must be served within a given time limit from the route start time.

The rule for accepting a new request is based on multiple possible scenarios for the occurrence in time and space of future requests. Given that a mix of true and expected requests are found in the solution associated with each scenario, the adverse effects of a myopic decision rule are alleviated. In the literature, future requests are either considered in an implicit way (see, for example, the double horizon approach [9]), or explicitly using different approaches for exploiting information about requests to come, including relocation and waiting strategies [3, 4, 7, 8, 10]. However, these methods are used to guide the integration of new requests into the current solution, not to accept or reject new requests based on some profit measure.

The remainder of the paper is as follow. In Section 2 the problem is defined. Section 3 then briefly describes the algorithm previously developed for the static version of the problem. The dynamic environment, as well

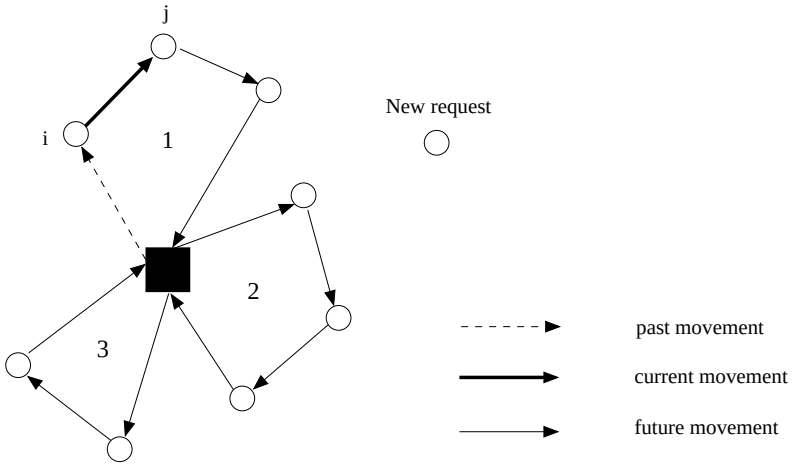

Figure 1: Current vehicle's workday

as the proposed methodology for deriving a non myopic acceptance rule, are found in Section 4. Finally, computational results in Section 5 report a comparison between the non myopic and myopic decision rules (i.e., with and without scenarios).

2 Problem definition

The static version of the problem is defined on a directed graph $G = (V, A)$ with $V = \{0, 1, 2, ..., n\}$ the vertex set and A the arc set. Vertex 0 is the depot while the remaining vertices are customers. With each customer $i \in V \backslash \{0\}$ is associated a gain or revenue g_i, a service or dwell time s_i and a time window $[a_i, b_i]$, where a_i and b_i are the earliest and latest time, respectively, to start the service (with $a_0 = 0$ and $b_0 = \infty$). Thus, a vehicle has to wait if it arrives at customer i before a_i. With each arc $(i, j) \in A$ is associated a distance d_{ij} and a travel time t_{ij}. We also have a set $K = \{1, 2, ..., m\}$ of vehicles to deliver goods from the depot to customers. The duration of each route is limited by forcing the last customer to be served within t_{max} time units of the route start time. This restriction leads to short routes that must be combined and sequenced to form vehicle workdays. Also, a setup time for loading the vehicle, noted σ_r, is associated with each route r in the solution. This setup time is proportional to the sum of service times over all

customers in the route. The objective is to maximize the total profit, which is the total gain associated with served customers minus the total traveled distance. A mathematical description of this problem can be found in [1].

In the dynamic version of the problem, the customers are not known in advance but must be responded to as the routes for serving previously assigned customers are executed by the vehicles. Once a customer request is received, an answer must be provided in real-time about the ability of the fleet to accommodate or not the request. On the other hand, the real-time aspect is much less stringent in the case of the actual service to the customer. As a pure delivery problem is dealt with, a route is fixed as soon as the corresponding vehicle departs from the depot to serve it. Accordingly, a new customer request can only be inserted in routes that will be executed later during a vehicle's workday.

3 Problem-solving methodology

An Adaptive Large Neighborhood Search (ALNS) [12] has been previously developed in [2] for solving the static version of the problem. First, an initial solution is constructed with an insertion heuristic. Then, a local search heuristic based on a large neighborhood is applied to improve this solution. This is explained in the following

3.1 Insertion heuristic

An insertion heuristic is used to construct an initial solution or to insert new customer requests in the current solution (in the dynamic setting). Every customer is inserted at its best feasible insertion place over every route in every workday, including an empty workday if one is still available. In this work, the best insertion place corresponds to the smallest detour in distance, where the detour is $d_{ji} + d_{il} - d_{jl}$ for the insertion of customer i between customers j and l. If there is no feasible insertion place for customer i, then each route is considered in turn and split into two subroutes, with an additional copy of the depot between the two subroutes. Once the original route is split, the insertion of the customer can take place in any of the two new routes. Each route is split in every possible way (i.e., at every customer location along the route) to find the best insertion place. If there is still no feasible insertion place, then customer i is left aside.

3.2 Large neighborhood search

For improving the solution, a large neighborhood structure is obtained through the use of solution destruction and reconstruction operators. These operators exploit the hierarchical nature of the problem by working either at the customer, route (made of customers) or workday (made of routes) level. The adaptive feature comes from a weight associated with each operator. This weight is modified depending if the corresponding operator is successful or not in finding improved solutions. Clearly, an operator is more likely to be selected and applied to the current solution if its corresponding weight is larger.

A generic description of this problem-solving methodology is shown in pseudo-code in Algorithm 1, where $s*$ is the best known solution and where the acceptance criterion is probabilistic and based on simulated annealing ideas. For more details, the reader is referred to [2].

Algorithm 1 ALNS

1. construct a feasible solution s;
2. $s* \leftarrow s$;
3. initialize weights;
4. **while** the stopping criterion is not met **do**
  4.1 for L = workday, route, customer **do**
    4.1.1 for I iterations **do**
      a. probabilistically select a destruction operator at level L and a reconstruction operator based on their current weights;
      b. apply the destruction and reconstruction operators to s to obtain s';
      c. **if** s' satisfies the acceptance criterion **then**
        $s \leftarrow s'$;
        if s' is better than $s*$ **then** $s* \leftarrow s'$;
    4.1.2 adjust weights;
5. return $s*$.

4 Dynamic environment

The algorithm developed for the static version of the problem was integrated into the new dynamic environment. This is described in the following, start-

ing with the acceptance rule for new requests.

4.1 Acceptance rule

A set S of s-solutions, where each s-solution includes a possible scenario for the occurrence of future requests, is used to evaluate the profitability of any new incoming request. The scenarios are based on some probabilistic knowledge that could be obtained, for example, through historical data. It is worth noting that the use of scenarios is also reported in [3]. However, only one scenario is associated with a solution and this scenario is used to guide the insertion of new requests into the current solution, not to decide about their acceptance. Acceptance rules with metrics based on true and expected requests can be found in [5], where the authors assume that all potential customer locations are known in advance and that a service request probability is associated with each location.

Here, it is assumed that the customer requests are received according to independent time-space Poisson processes. Basically, the service area is a grid made of Z squared zones and the horizon is divided into T time periods. The request arrival intensity within each zone z at time period t is λ_{zt} with $\sum_{z=1}^{Z} \lambda_{zt} = \lambda_t$, $t = 1, 2, ..., T$. Within each zone, the request is located uniformly randomly. Finally, the gain follows a normal law with average $4 \times \max_{i \in V \setminus \{0\}} d_{0i}$ and standard deviation $2 \times \max_{i \in V \setminus \{0\}} d_{0i}$, where d_{0i} is the distance between the depot and customer i. The definitions of the average and standard deviation imply that it is always profitable to serve a customer, since the gain exceeds the additional traveled distance needed to serve the customer (note that non profitable customers would always be rejected).

At the start, a s-solution is produced with the expected requests of each scenario using our ALNS (see Section 3). Each one of these s-solutions is a blueprint for evaluating the opportunity value of new incoming requests. When the discrete simulation starts and time unfolds, new requests are received. If a new request cannot be incorporated into the true solution (made only of true requests) due to the time constraints, it is automatically rejected. Otherwise, the opportunity value of the new request is calculated as the sum of the differences over all scenarios of the s-solution quality with and without the insertion of the new request. If δ_i^s denotes this difference for request i and s-solution s, then the opportunity value of i is $\sum_{s \in S} \delta_i^s$. If this value is positive then the new request is accepted, otherwise it is rejected. When accepted, the new request is finally inserted in the true solution and

in every s-solution. This chain of events is summarized in the pseudo-code below.

> If there is no feasible insertion place for the new request i in the true solution ts then reject i
>
> else
>
> 1. consider the insertion of the new request i in every s-solution;
> 2. calculate $\Delta = \sum_{s \in S} \delta_i^s$;
> 3. if $\Delta > 0$ then accept i and insert it in ts and in every s-solution else reject i.

As time unfolds, a mix of true and expected customer requests are found in each s-solution. Since the true requests have been incorporated into each s-solution with the insertion heuristic described in Section 3.1, these solutions are reoptimized with ALNS after the occurrence of I new requests, to maintain s-solutions of sufficiently good quality (in our computational results, $I = 10$).

4.2 Dynamic environment

There are two types of events that ask for a reaction: the occurrence of a new request and the departure of a vehicle from the depot. This is described in the following pseudo-code.

> if "event" then
>
> 1. remove obsolete portion of every solution;
> 2. if event is "occurrence of a new request" then
>   - 2.1 apply the acceptance rule;
>   - 2.2 apply ALNS to the true solution ts;
> 3. if event is "vehicle departure" then
>   - 3.1 fix the vehicle's current route in ts;
>   - 3.2 update every s-solution;
>   - 3.3 apply ALNS to the true solution ts.

It is worth noting that ALNS is run on the planned portion of solution ts in steps 2.2 and 3.3 (excluding the current route, which is fixed). Removing the obsolete part of every solution in step 1 consists in removing customer requests that have been served since the occurrence of the previous event. In the case of a vehicle departure, the update in step 3.3 is aimed at maintaining the consistency of every s-solution with the true solution ts. That is, the current route should be the same, as well as the set of true requests to be served in the planned routes (although they are not necessarily served in the same order). Consider the example in Figure 2 where the sequences of customers in the true solution ts and in some s-solution are illustrated (for simplicity, expected customers in the s-solution are not shown). When the currrent route of the s-solution is replaced by the current route of the true solution ts, customer 2 is duplicated, while customer 3 is missing. To maintain consistency, the following repair actions are taken:

1. In each case of duplication, the duplicate customer is simply removed from the planned routes.
2. In each case of omission, the customer is reintroduced into the planned routes using the insertion heuristic of Section 3.1.

After the repair, the s-solution is said to be consistent with the true solution.

5 Computational results

A simulator was developed to test the proposed acceptance rule in different operating scenarios and compare it to a myopic approach where each new request is accepted as long as it is feasible. In the following, the characteristics of the simulator are first described. Then, a comparison is provided between the myopic and non myopic decision rules with an increasing number of requests over the same fixed horizon. Finally, the impact of the number of scenarios on solution quality is presented.

5.1 Simulator

The service area is a 5km $\times$ 5km square divided into into a grid of $Z = 2 \times 2 = 4$ zones. Within that square, the depot is located at coordinates $(2.0\text{km}, 2.5\text{km})$. The time horizon is divided into $T = 4$ periods of one hour

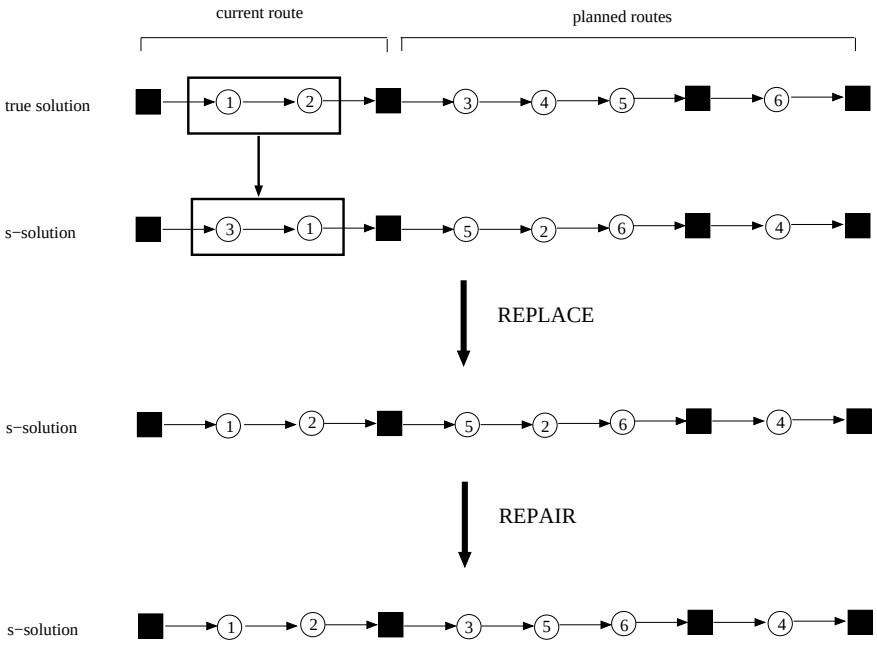

Figure 2: Maintaining the consistency of every s-solution

each. All vehicles move at a constant average speed of 30 km per hour. The service or dwell time is equal to 5 minutes at each customer location and the t_{max} value is set to 40 minutes. As previously mentioned, the new requests are received according to independent time-space Poisson processes where λ_{zt} is the request arrival intensity in zone z at time period t. Within each zone, the request is located uniformly randomly. In the computational results, the intensity is the same in each zone and in each time period, for the first three time periods. In the last period, no request is generated because it is too late to serve them the same day. The time windows of the new customer requests correspond to a full time period of one hour and are equally distributed among the time periods that follow the current one (which is the time period associated with the occurrence of the new request). Finally, the gain follows a normal law with average $4 \times \max_{i \in V \setminus \{0\}} d_{0i}$ and standard deviation $2 \times \max_{i \in V \setminus \{0\}} d_{0i}$, where d_{0i} is the distance between the depot and customer i.

5.2 Comparison between myopic and non myopic approaches

In this section, we compare the performance of the myopic and non myopic acceptance rules with an increasing number of customer requests. This increase is obtained by varying the intensity parameter in the Poisson law to obtain 24, 36 and 48 customer requests per hour, on average (for a total of 72, 108 and 144 customers, respectively). Tables 1 and 2 show the results for the myopic approach and the non myopic approach, respectively, with 80 scenarios and using fleets of 3 and 5 vehicles. Each number is an average taken over 10 different instances.

As we can see, the non myopic approach provides a significant improvement over the myopic one. For a fleet of 3 vehicles, the profit increase in percentage varies between 7.5% and 16.8%. With 5 vehicles, the improvement is somewhat smaller but still stands between 4.8% and 9.8%. In addition to the profit increase, the solutions obtained with the non myopic approach systematically visit a larger number of customers. For both methods, an increase in the number of served customers and profit is observed when the number of vehicles increases from 3 to 5, for the same number of customers. Each vehicle also performs a smaller number of routes during its workday.

# vehicles	# cust.	# served cust.	% served cust.	# routes per workday	# cust. per route	Profit	CPU (s)
3	72.2	44.5	61.8	2.8	5.5	447.9	0.6
	108.4	55.4	51.2	3.3	6.0	564.7	1.7
	144.2	56.1	39.0	3.3	5.8	579.0	2.7
5	72.2	51.6	71.8	1.7	6.3	534.3	1.2
	108.4	69.4	64.1	2.3	6.2	710.8	3.8
	144.2	82.3	57.2	3.0	5.6	855.3	7.6

Table 1: Simulation of 4 hours with the myopic approach, fleets of 3 and 5 vehicles and an increasing number of customers

Through additional experiments, we also observed that the gap between the two approaches vanishes as the average number of customers per vehicle per hour becomes very high (30 customers per hour per vehicle) or very low (2 customers per hour per vehicle). In the former case, the overwhelming complexity of the problem does not allow any method to take the edge. In the latter case, the simplicity of the problem has a similar impact. Between these two values, the non myopic approach is systematically better than the myopic one, although the gap can vary a lot, even for the same vehicle

# vehicles	# cust.	# served cust.	% served cust.	# routes per workday	# cust. per route	Profit	CPU (s)
3	72.2	48.6	67.8	2.1	8.1	508.9	204.7
	108.4	57.5	53.1	2.1	9.2	606.9	272.8
	144.2	62.6	43.6	2.4	8.6	676.3	471.1
5	72.2	55.6	77.3	1.5	7.8	587.0	305.6
	108.4	70.2	64.9	1.6	8.9	745.0	733.4
	144.2	83.1	57.9	1.8	9.2	901.6	963.7

Table 2: Simulation of 4 hours with the non myopic approach, fleets of 3 and 5 vehicles and an increasing number of customers

load. For example, by considering instances with 24 requests per hour and 2 vehicles, 36 requests per hour and 3 vehicles and 48 requests per hour and 4 vehicles, for the same average load of 12 customers per vehicle per hour, profit increases of 17.0%, 9.4% and 6.0%, respectively, have been obtained.

Finally, Table 3 illustrates the impact of the number of scenarios on the performance of the non myopic approach, based on simulations with 3 vehicles and 36 requests per hour on average. An improvement is observed up to about 80 scenarios (with a corresponding increase in computation times). Beyond that point, the performance of the method reaches a plateau.

# scenarios	# cust.	# served cust.	% served cust.	# routes per workday	# cust. per route	Profit	CPU (s)
10	108.4	55.8	51.5	2.1	9.0	587.0	36.7
20	108.4	57.0	52.7	2.2	8.9	596.7	75.6
40	108.4	57.2	52.9	2.0	9.4	600.0	137.8
80	108.4	57.5	53.1	2.1	9.2	606.9	272.8
120	108.4	57.7	53.3	2.2	9.0	607.9	433.3
150	108.4	57.6	53.2	2.1	9.2	607.0	500.4

Table 3: Simulation of 4 hours with 3 vehicles, 36 requests per hour on average and an increasing number of scenarios

6 Conclusion

This paper shows the benefits of accounting for future customer requests when deciding about the acceptance or rejection of a new request in a dy-

namic setting where each vehicle executes multiple routes during its workday. Each decision is based on the planned routes to be executed later (thus, excluding the current route) due to the delivery nature of the problem. The use of multiple possible scenarios for the occurrence of new requests is also shown to be beneficial by providing solutions with an increased profit.

Acknowledgements. Financial support for this work was provided by the Canadian Natural Sciences and Engineering Research Council (NSERC). This support is gratefully acknowledged.

References

[1] Azi N., Gendreau M., Potvin J.-Y., "An Exact Algorithm for a Vehicle Routing Problem with Time Windows and Multiple Use of Vehicles", *European Journal of Operational Research 202*, 756-763, 2010.

[2] Azi N., Gendreau M., Potvin J.-Y., "An Adaptive Large Neighborhood Search for a Vehicle Routing Problem with Multiple Trips", Technical Report CIRRELT-2010-08, Montreal, submitted to *Computers & Operations Research*, 2010.

[3] Bent R. and Van Hentenryck P., "Scenario-based Planning for Partially Dynamic Vehicle Routing with Stochastic Customers", *Operations Research 52*, 977-987, 2004.

[4] Branke J., Middendorf M., Noeth G. and Dessouky M., "Waiting Strategies for Dynamic Vehicle Routing", *Transportation Science 39*, 298-312, 2005.

[5] Campbell A.M. and Savelsbergh M., "Decision Support for Consumer Direct Grocery Initiatives", *Transportation Science 39*, 313-327, 2005.

[6] Fleischmann B., Gnutzmann S. and Sandvoss E., "Dynamic Vehicle Routing based on Online Traffic Information", *Transportation Science 38*, 420-433, 2004.

[7] Hvattum L.M., Lokketangen A. and Laporte G., "Solving a Dynamic and Stochastic Vehicle Routing Problem with a Sample Scenario Hedging Heuristic", *Transportation Science 40*, 421-438, 2006.

[8] Ichoua S., Gendreau M., Potvin J.-Y., "Exploiting Knowledge about Future Demands for Real-Time Vehicle Dispatching", *Transportation Science 40*, 211-225, 2006.

[9] Mitrović M., Krishnamurti R. and Laporte G., "Double-Horizon Based Heuristics for the Dynamic Pickup and Delivery Problem with Time Windows", *Transportation Science Part B 38*, 669-685, 2004.

[10] Mitrović M. and Laporte G., "Waiting Strategies for the Dynamic Pickup and Delivery Problem with Time Windows", *Transportation Research Part B 38*, 635-655, 2004.

[11] Potvin J.-Y., Ying X. and Benyahia I., "Vehicle Routing and Scheduling with Dynamic Travel Times", *Computers & Operations Research 33*, 1129-1137, 2006.

[12] Ropke S. and Pisinger D., "An Adaptive Large Neighborhood Search Heuristic for the Pickup and Delivery Problem with Time Windows", *Transportation Science 40*, 455-472, 2006.