Large Neighborhood Search for rich VRP with multiple pickup and delivery locations

Asvin Goel,Volker Gruhn

Chair of Applied Telematics and e-Business, Department of Computer Science,
University of Leipzig, Klostergasse 3, 04109 Leipzig, Germany
{goel,gruhn}@ebus.informatik.uni-leipzig.de

Abstract

In this paper we consider a rich vehicle routing problem where transportation requests are characterised by multiple pickup and delivery locations. The problem is a combined load acceptance and generalised vehicle routing problem incorporating a diversity of practical complexities. Among those are time window restrictions, a heterogeneous vehicle fleet with different travel times, travel costs and capacity, multi-dimensional capacity constraints, order/vehicle compatibility constraints, and different start and end locations for vehicles. We propose iterative improvement approaches based on Large Neighborhood Search and a relatedness measure for transportation requests with multiple pickup and delivery locations. Our algorithms are characterised by very fast response times and thus, can be used within dynamic routing systems where input data can change at any time.

Keywords: Vehicle Routing Problem, Pickup and Delivery Problem, Large Neighborhood Search

1 Introduction

In this paper we present algorithms for solving a rich VRP with multiple pickup and delivery locations. The problem is a combined load acceptance and generalised vehicle routing problem and incorporates various practical complexities as time window restrictions, a heterogeneous vehicle fleet with different travel times, travel costs and capacity, multi-dimensional capacity constraints, order/vehicle compatibility constraints, and different start and end locations for vehicles. In the problem not all relevant data is known when transportation begins, instead, information can change during the transportation process. Thus, the problem is dynamic and the algorithms must have very fast response times. Otherwise, an optimised solution is likely to be invalid as problem data might have changed before the new solution can be applied. The algorithms we present are characterised by two features: they are capable of handling the practical complexities and they have very fast response times and thus, are suitable for dynamic optimisation.

The paper is organised as follows. First, we verbally describe the problem and discuss related work. Then, we describe Large Neighborhood Search and the specific implementation for our dynamic real-life problem. Eventually, we present computational results for the test cases we generated.

2 Problem Formulation

This work is motivated by a practical problem arising in air-cargo transport. Most of the air-cargo within Europe is transported by so-called *road feeder services (RFS)*, that is the transport is done on roads, see [8]. In the problem considered not all transportation requests are known before load acceptance and planning starts. Instead, transportation requests may become known at any time. In contrast to many other commonly known routing problems, not all transportation requests have to be assigned to a vehicle. Instead, a so-called *make-or-buy* decision is necessary to determine whether a transportation request should be assigned to some vehicle (make) or not (buy).

A transportation request is specified by a nonempty set of pickup, delivery and/or service locations which have to be visited in a particular sequence by the same vehicle, the time windows in which these locations have to be visited, and the revenue gained when the transportation request is served. Furthermore, some characteristics can be specified which constrain the possibility of assigning the transportation requests to certain vehicles due to compatibility constraints and capacity constraints. At each of the locations some shipment(s) with several describing attributes can be loaded or unloaded.

A fleet of heterogeneous vehicles is available to serve the transportation requests. The vehicles can have different capacities, as well as different travel times and travel costs between locations. The vehicles can transport shipments which require some of the capacity the vehicle supplies. Instead of assuming that each vehicle becomes available at a central depot, each vehicle is given a start location where it becomes available at a specific time and with a specific load. Furthermore, the vehicles do not have to return to a central depot and for each vehicle a final location is specified, which has to be reached within a specific time and with a specific load. Each vehicle may have to visit some locations in a particular sequence between leaving its start and reaching its final location. All locations have to be visited within a specific time window. If the vehicle reaches one of these locations before the begin of the time window, it has to wait.

A tour of a vehicle is a journey starting at the vehicles start location and ending at its final location, passing all other locations the vehicle has to visit in the correct sequence, and passing all locations belonging to each transportation request assigned to the vehicle in the correct respective sequence. A tour is *feasible* if and only if for all orders assigned to the tour compatibility constraints hold and at each point in the tour time window and capacity restrictions hold. The objective is to find distinct feasible tours maximising the profit, which is determined by the accumulated revenue of all served transportation requests, reduced by the accumulated costs for operating these tours.

3 Related work

The dynamic real-life problem discussed in this paper is a generalisation of the vehicle routing problem (VRP) and the pickup and delivery problem (PDP), see [3], and [11] and secondary literature given there. The most widely studied extensions of the VRP are the VRP with time windows, see [2], and the capacitated VRP, see [10]. In many cases it is assumed that load is accepted before planning begins and tours are generated assuming that all accepted transportation requests must be served. Work regarding load acceptance issues for the travelling salesman problem (TSP) has been surveyed by [4], but only few attempts have been made to tackle extensions of this problem. VRP with multiple pickup and delivery locations have been studied by [16], [17] and [7].

A comprehensive discussion of dynamic vehicle routing can be found in [14] and [15]. Dynamic real-life problems often require rich models, in most of the literature on dynamic routing problems however, some simplifying assumptions are made. For example, in the dynamic full-truckload pickup and delivery problem, which recently has received increasing attention, see [5], [19], and [13], each vehicle can only carry one transportation request at a time and cannot load further shipments until all currently loaded shipments are unloaded. The only work known to the authors regarding rich VRP in a dynamic context is presented by [17]. A column generation approach is used to solve the general pickup and delivery problem (GPDP), but this approach cannot be used to solve highly dynamic problems were response times of only a few seconds are required.

4 Large Neighborhood Search

In this paper we present algorithms based on Large Neighborhood Search (LNS). LNS has been introduced for the VRP with time windows by [18] and can be interpreted as a special case of Variable Neighborhood

Initialisation: Find an initial solution s; choose a stopping condition

Repeat the following until the stopping condition is met:

(1) Choose $k \in \{1, \ldots, |\mathcal{O}_s|\}$

(2) *Shaking:* Generate a point $s' \in N_k(s)$

(3) *Local search:* Apply an insertion method with s' as initial solution; denote with s^* the so obtained new solution

(4) *Move or not:* If the new solution s^* is better than s, move there $(s \leftarrow s^*)$

Figure 1: Large Neighborhood Search

Search (VNS), see [12] and [6]. VNS is a meta-heuristic especially suited for problems, where local search methods are very likely to get trapped in locally optimal solutions of poor quality.

Denote with $N_k(s)$ the k^{th} neighborhood of a solution s and with $|\mathcal{O}_s|$ the number of transportation requests which are assigned to the tour of some vehicle. The neighborhood $N_k(s)$ is determined by removing k transportation requests from their tours in the solution s. The LNS algorithm can be outlined as illustrated in figure 1. The algorithm starts with an initial solution s which can be obtained by any tour construction method. In each iteration k transportation requests are removed from the tours they are currently assigned to. A new solution s^* is then generated by inserting unscheduled transportation requests. The new solution is accepted as the next current solution if the objective value is improved. If no stopping condition is met, the algorithm continues with the next iteration. The number of removals can be adjusted in the next iteration. The choice of the next neighborhood is, differently as proposed in [6], completely undetermined. The probability that a better solution can be found is strongly connected to the choice of the neighborhood $N_k(s)$. If k is too small, the solutions which can be found by an LNS move will be very similar to the current solution. If k is too large, the insertion method may need too much time and the new solution found may not be much better than a solution generated from scratch. As the problem considered in this paper is dynamic, data may have changed between two iterations. In this case there may not be the need to change the neighborhood structure to increase the probability that a better solution can be found. Furthermore, it is not clear how to change k effectively. [9] have shown that LNS is well suited for the VRP with several additional constraints. We will see that LNS can also be used to solve the dynamic real-life problem considered in this paper. The goal of removing transportation requests in step 2 of the LNS method is to generate an auspicious interim solution such that the insertion method can find a new solution with better quality. Transportation requests can be

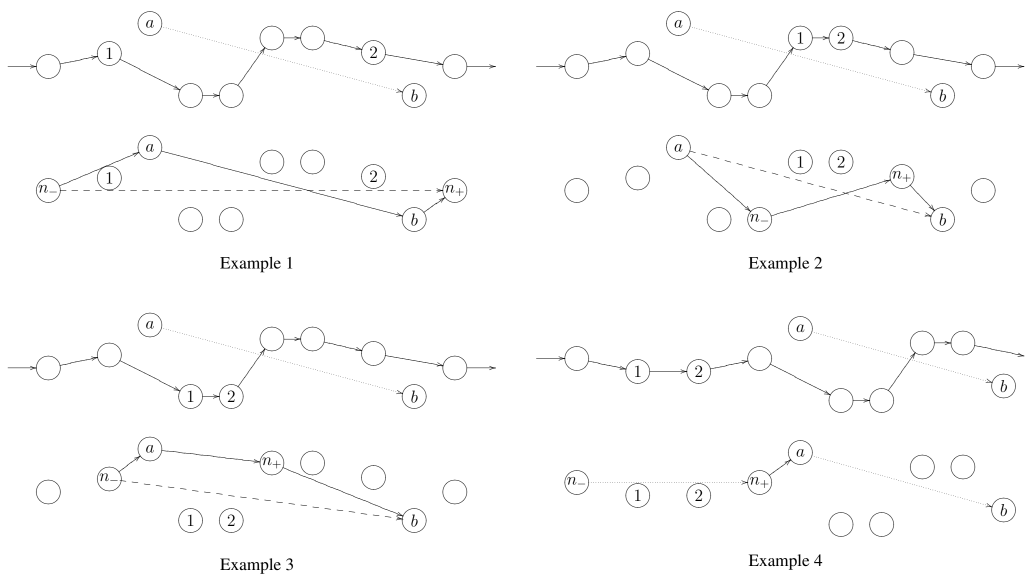

Figure 2: Determination of relatedness values

removed randomly from the tours, but in this case some of them may not be *related* to each other in any way. As a result, they would compete for different vehicles or the same vehicle at different time slots at the next call of the insertion method. Thus, it would be more efficient to remove fewer transportation requests if one could ensure that the unscheduled orders were *related* to each other. For the VRP [18] propose a relatedness measure based on geographical closeness of customer locations. A concept similar to geographical closeness in the VRP however, does not exist for vehicle routing problems with multiple pickup, delivery and/or service locations. If geographical closeness cannot be used, the question is how to define a relatedness measure for our problem. We want to increase to probability that a transportation request which is removed from the tour of a vehicle allows another transportation request to take its "place". Therefore, we propose a tour dependent relatedness measure. An order assigned to the tour of a vehicle which is not suited for an unscheduled order cannot be regarded related to the latter. An order assigned to the tour of a vehicle which is suited for an unscheduled order can be regarded related if the unscheduled order o "fits" to the part of the tour currently occupied by the former. We propose to determine the relatedness measure as illustrated in the examples in figure 2.

To determine the relatedness value of a scheduled order, we don't directly regard the scheduled order itself, but its preceeding and succeeding point in the tour. Let n_- and n_+ denote the predecessor and successor of the scheduled order and let $n_{(o,1)}, \ldots, n_{(o,\lambda_o)}$ denote the locations associated to the unscheduled order o. Let c^v_{nm} denote the cost of vehicle v for travelling from point n to point m. Then the cost of a journey

Remove a randomly chosen transportation request from its tour

Repeat the following until k transportation requests are removed:

(a) Choose an unscheduled transportation request

(b) Determine the corresponding relatedness value for all scheduled transportation requests

(c) Rank the transportation requests according to their relatedness value

(d) Remove some transportation requests with high rank

Figure 3: Removal of transportation requests using the relatedness measure

$(n_1, \ldots, n_{\lambda_o+2})$ containing the subsequences (n_-, n_+) and $(n_{(o,1)}, \ldots, n_{(o,\lambda_o)})$ is

$$\sum_{i=1}^{i<\lambda_o+2} c^v_{n_i n_{i+1}}.$$

Now consider the cheapest of these journeys where time windows restrictions hold. If, as in examples 1-3 of figure 2, n_+ is visited after $n_{(o,1)}$ and n_- is visited before $n_{(o,\lambda_o)}$ the relatedness value is

$$\text{relatedness value} = \sum_{i=1}^{i<\lambda_o+2} c^v_{n_i n_{i+1}} - c^v_{n_1 n_{\lambda_o+2}}.$$

Otherwise, n_- is visited after $n_{(o,\lambda_o)}$ or n_+ is visited before $n_{(o,1)}$, as shown in example 4 of figure 2. If n_+ is visited before $n_{(o,1)}$ the scheduled order is not regarded related to the unscheduled order if the removal from the tour would not allow n_+ to be visited earlier. Analogously, if n_- is visited after $n_{(o,\lambda_o)}$ the scheduled order is not regarded related to the unscheduled order if the removal from the tour would not allow n_- to be visited later. Otherwise, the relatedness value is the cost for travelling from n_+ to $n_{(o,1)}$ or from $n_{(o,\lambda_o)}$ to n_-.

A small relatedness value indicates that the unscheduled order would be an auspicious candidate for insertion if the considered scheduled order was removed from the tour. Using this relatedness measure we can implement step 2 of the LNS method as illustrated in figure 3. First, a randomly chosen transportation request is removed from some tour. Then, an unscheduled transportation request is chosen randomly and for all scheduled transportation requests the relatedness value is determined. The related transportation requests are ranked according to their relatedness value and some of them with high rank are chosen to be removed from the tour. We will now describe the insertion methods which we use to generate an initial solution as

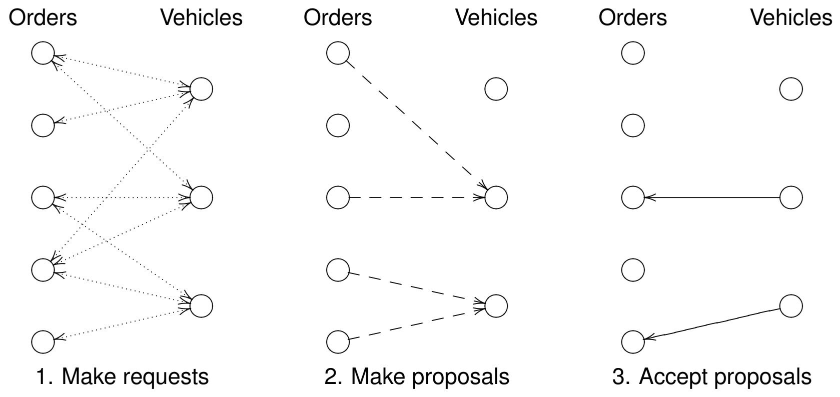

Figure 4: Illustration of the auction method

well as to generate new solutions in step 3 of the LNS method. In both insertion methods, the incremental cost for inserting an order to the tour of some vehicle is used to determine the (local) efficiency of an insertion.

The first method is a sequential insertion method. In the sequential insertion method unscheduled transportation requests are randomly chosen and all feasible and *efficient* insertion possibilities are determined. We assume an infinite incremental cost if no feasible insertion is possible and say that an insertion possibility is *efficient* if the incremental cost is smaller than the revenue of the order. If an efficient insertion possibility is found the transportation request is inserted to a tour with high efficiency. The method continues with the next order until no efficient insertions are possible. The second insertion method is based on the auction method for the vehicle routing problem with time windows by [1]. This method is illustrated in figure 4. In the first phase all unscheduled orders request and receive from each vehicle an insertion possibility and the efficiency of insertion. In the second phase each unscheduled order, which did receive an efficient insertion possibility, chooses a vehicle with low incremental costs and sends a proposal for insertion to this vehicle. In phase three each vehicle which received a proposal chooses an order for insertion to the tour. The method stops if no order can be efficiently inserted and continues otherwise.

5 Computational experiments

Computational experiments were performed on test cases derived from the real-life problem. We generated test problem with $|\mathcal{V}|$ vehicles and $|\mathcal{O}_0|$ orders which are known at the beginning of the planning horizon. In every hour of the planning horizon of one week $|\mathcal{O}_t|$ new orders become known. The length of time windows was set to a fix value τ for all locations associated to transportation requests. We generated a heterogeneous vehicle fleet and transportation requests with different requirements to the vehicles and pickup and delivery

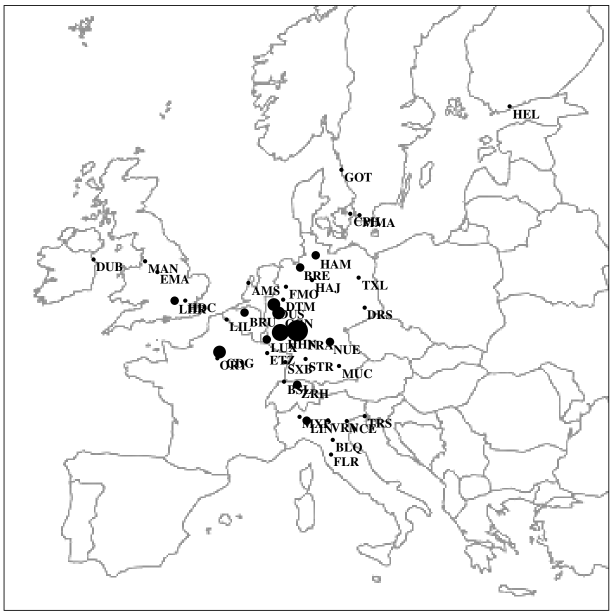

Figure 5: Distribution of pickup and delivery locations

locations distributed as indicated in figure 5.

At each timestep all transportation requests which were unscheduled were permanently rejected before new transportation requests were added to the problem. New transportation requests were inserted to the tours by the auction method afterwards. The solution obtained hereby was used as a reference solution. To test our algorithms we only allowed 30 seconds of computation time per timestep on a personal computer with Intel Pentium 4 processor with 3.00 GHz and linux operating system. The average time per iteration of our LNS algorithms was below one second for all test problems except for those with 500 vehicles where the average time per iteration was below 1.75 seconds. In table 1 we show the percentage of improvement over the reference solutions. The LNS method using the auction method for reinsertion is denoted by LNS-A, the LNS method using the sequential method for reinsertion is denoted by LNS-S. The LNS method denoted by LNS-AS randomly switches between the sequential and the auction method. We can see that in almost all cases the LNS algorithms using random removals are outperformed by LNS using the relatedness measure presented in this paper. Furthermore, we can see that in most cases LNS-A outperforms the other algorithms. However, our assumption that in some cases the sequential insertion method can produce better results due to higher diversification is also confirmed. No significant changes in the performance of our algorithms can be identified between the test cases with very short time windows and longer time windows.

Problem				Random removals			Related removals		
$\|\mathcal{V}\|$	$\|\mathcal{O}_0\|$	$\|\mathcal{O}_t\|$	τ	LNS-A	LNS-AS	LNS-S	LNS-A	LNS-AS	LNS-S
50	150	5	2h	50.96	51.81	45.19	52.90	52.56	54.68
100	300	10	2h	37.75	34.66	32.32	36.26	35.88	38.23
250	750	25	2h	19.87	17.89	13.09	23.02	21.55	17.22
500	1500	50	2h	8.92	6.09	5.90	11.99	10.24	7.69
50	150	5	12h	43.89	42.18	42.52	45.89	42.37	45.82
100	300	10	12h	32.87	32.44	30.23	34.87	34.57	34.04
250	750	25	12h	20.39	18.20	16.96	23.51	21.15	21.06
500	1500	50	12h	7.75	8.09	5.84	11.43	11.69	10.25

Table 1: Results

6 Conclusions

In this paper we considered a dynamic real-life problem which is a combined load acceptance and generalised vehicle routing problem. The problem incorporates some practical complexities which received only little attention in the vehicle routing literature. Among those are transportation requests with multiple pickup and delivery locations. We presented algorithms based on Large Neighborhood Search which are capable of handling these complexities. In VRP with multiple pickup and delivery locations geographical closeness cannot be used as a relatedness measure. We have proposed a tour dependent relatedness measure for transportation requests with multiple pickup and delivery locations and our computational experiments have shown that this relatedness measure almost always outperforms random removals independently of the insertion method used. To guarantee fast response times fast insertion methods are proposed for solving the dynamic problem. Our algorithms perform well for problems with hundreds of vehicles and several hundreds of transportation requests and response times were often less than a second. The combination of fast response times and the capability of handling the practical complexities allows the use of our algorithms in dynamic routing systems.

References

[1] J. Antes and U. Derigs. A new parallel tour construction algorithm for the vehicle routing problem with time windows. Department of Information Systems and Operations Research, University of Cologne, Cologne, Germany, 1995.

[2] J.-F. Cordeau, G. Desaulniers, J. Desrosiers, M.M. Solomon, and F. Soumis. VRP with time windows. In P. Toth and D. Vigo, editors, *The Vehicle Routing Problem*, pages 157–193. SIAM Monographs on Discrete Mathematics and Applications, Philadelphia, 2002.

[3] J.-F. Cordeau, M. Gendreau, A. Hertz, G. Laporte, and J.-S. Sormany. New heuristics for the vehicle routing problem. Les cahiers du GERAD G-2004-33, Université de Montreal HEC, Montréal, Canada, 2004.

[4] D. Feillet, P. Dejax, and M. Gendreau. Traveling Salesman Problems with Profits. *Transportation Science*, 39(2):188–205, 2005.

[5] B. Fleischmann, S. Gnutzmann, and E. Sandvoß. Dynamic vehicle routing based on on-line traffic information. *Transportation Science*, 38(4):420–433, 2004.

[6] P. Hansen and N. Mladenović. A tutorial on Variable Neighborhood Search. Les cahiers du GERAD G-2003-46, Université de Montreal HEC, Montréal, Canada, 2003.

[7] G. Hasle. Heuristics for rich VRP models. Presented at the Seminar at GERAD, 30.10.2003, Montréal, Canada, 2003.

[8] M. Heckmann. DV-gestütztes Geschäftsprozeßmanagement in der Luftfrachtlogistik. Dissertation, Shaker Verlag Aachen, 2002.

[9] P. Kilby, P. Prosser, and P. Shaw. A comparison of traditional and constraint-based heuristic methods on vehicle routing problems with side constraints. *Constraints*, 5:389–414, 2000.

[10] G. Laporte and F. Semet. Classical heuristics for the capacitated VRP. In P. Toth and D. Vigo, editors, *The Vehicle Routing Problem*, pages 109–128. SIAM Monographs on Discrete Mathematics and Applications, Philadelphia, 2002.

[11] S. Mitrović-Minić. Pickup and delivery problem with time windows: A survey. Technical report TR 1998-12, School of Computing Science, Simon Fraser University, Burnaby, BC, Canada, 1998.

[12] N. Mladenović and P. Hansen. Variable Neighborhood Search. *Computers and Operations Research*, 24:1097–1100, 1997.

[13] W.B. Powell, W. Snow, and R.K. Cheung. Adaptive labeling algorithms for the dynamic assignment problem. *Transportation Science*, 34(1):50–66, 2000.

[14] H.N. Psaraftis. Dynamic vehicle routing problems. In B.L. Golden and A.A. Assad, editors, *Vehicle routing: Methods and studies*, pages 233–248. North-Holland Amsterdam, 1988.

[15] H.N. Psaraftis. Dynamic vehicle routing: Status and prospects. *Annals of Operations Research*, 61:143–164, 1995.

[16] M.W.P. Savelsbergh and M. Sol. The general pickup and delivery problem. *Transportation Science*, 29(1):17–30, 1995.

[17] M.W.P. Savelsbergh and M. Sol. DRIVE: dynamic routing of independent vehicles. *Operations Research*, 46:474–490, 1998.

[18] P. Shaw. A new local search algorithm providing high quality solutions to vehicle routing problems. Technical report, APES group, Department of Computer Sciences, University of Strathclyde, Glasgow, Scottland, 1997.

[19] J. Yang, P. Jaillet, and H. Mahmassani. Real-time multi-vehicle truckload pickup-and-delivery problems. *Transportation Science*, 38(2):135–148, 2004.