Available online at www.sciencedirect.com
ScienceDirect
European Journal of Operational Research 191 (2008) 650–660
EUROPEAN JOURNAL OF OPERATIONAL RESEARCH
www.elsevier.com/locate/ejor

A General Vehicle Routing Problem

Asvin Goel [a,b,*], Volker Gruhn [b]

[a] Zaragoza Logistics Center, Avda. Gómez Laguna 25, 1ª Planta, 50009 Zaragoza, Spain
[b] Chair of Applied Telematics and e-Business, Computer Science Faculty, University of Leipzig, Klostergasse 3, 04109 Leipzig, Germany

Received 15 April 2006; accepted 15 December 2006
Available online 16 February 2007

Abstract

In this paper, we study a rich vehicle routing problem incorporating various complexities found in real-life applications. The General Vehicle Routing Problem (GVRP) is a combined load acceptance and generalised vehicle routing problem. Among the real-life requirements are time window restrictions, a heterogeneous vehicle fleet with different travel times, travel costs and capacity, multi-dimensional capacity constraints, order/vehicle compatibility constraints, orders with multiple pickup, delivery and service locations, different start and end locations for vehicles, and route restrictions for vehicles. The GVRP is highly constrained and the search space is likely to contain many solutions such that it is impossible to go from one solution to another using a single neighbourhood structure. Therefore, we propose iterative improvement approaches based on the idea of changing the neighbourhood structure during the search.
© 2007 Elsevier B.V. All rights reserved.

Keywords: Vehicle Routing Problem; Pickup and Delivery Problem; Variable Neighbourhood Search; Large Neighbourhood Search

1. Introduction

In this paper, we study a rich vehicle routing problem incorporating various complexities found in real-life applications. The General Vehicle Routing Problem (GVRP) is a combined load acceptance and routing problem which generalises the well-known *Vehicle Routing Problem* (VRP) and *Pickup and Delivery Problem* (PDP). Among the real-life requirements are time window restrictions, a heterogeneous vehicle fleet with different travel times, travel costs and capacity, multi-dimensional capacity constraints, order/vehicle compatibility constraints, orders with multiple pickup, delivery and service locations, different start and end locations for vehicles, and route restrictions for vehicles.

This work is motivated by a practical problem arising in air-cargo transport. Within Europe most of the air-cargo is transported by so-called *Road Feeder Services* (*RFS*), i.e., the transport is done on roads, see Heckmann (2002). Although schedules for air-cargo transport are made long before the transport has to begin, the actual demand is not known until shortly before. Therefore, airlines request additional transportation resources or cancel transportation requests only shortly before they are supposed to begin. As a result, operators of RFS not only have to consider the various real-life

* Corresponding author.
E-mail addresses: goel@telematique.eu (A. Goel), gruhn@ebus.informatik.uni-leipzig.de (V. Gruhn).

0377-2217/$ - see front matter © 2007 Elsevier B.V. All rights reserved.
doi:10.1016/j.ejor.2006.12.065

requirements, but also the fact that input data may change dynamically. Algorithms used in interactive dynamic planning systems to support the dispatchers must have very fast *response times*, i.e., the time an algorithm needs for one iteration must be very short. Otherwise, a new solution for a dynamic problem can not be applied if the problem data have changed during the time required for calculation. The heuristics presented in this paper are characterised by very fast response times and can be used in interactive dynamic planning systems.

This paper is organised as follows. First, we discuss related work in Section 2. In Section 3 we verbally describe the GVRP and give a mathematical formulation in Section 4. Then, we present iterative improvement approaches based on the idea of changing the neighbourhood structure during the search. In Section 5 we propose a Reduced Variable Neighbourhood Search approach which is based on various elementary neighbourhood operators. In Section 6 we propose a Large Neighbourhood Search approach which uses fast insertion methods in order to guarantee fast response times required in dynamic planning. Eventually, we present computational results in Section 7.

2. Related work

The problem considered in this paper is a generalisation of the *Vehicle Routing Problem* (VRP) and the *Pickup and Delivery Problem* (PDP), see Cordeau et al. (2004), Mitrović-Minić (1998) and secondary literature given there. The most widely studied vehicle routing problems are the capacitated VRP and the *Vehicle Routing Problem with Time Windows* (VRPTW) which are surveyed by Laporte and Semet (2002), and Cordeau et al. (2002).

Efficient methods for handling complex side constraints in insertion methods are presented in Campbell and Savelsbergh (2004). Comprehensive surveys on construction methods, neighbourhood search methods, and metaheuristics for the VRPTW are given by Bräysy and Gendreau (2005a), and Bräysy and Gendreau (2005b).

Variable Neighbourhood Search (VNS) is a metaheuristic based on the idea of systematically changing the neighbourhood structure during the search, see Mladenović and Hansen (1997), and Hansen and Mladenović (2003). VNS systematically exploits the following observations: (a) a local optimum with respect to one neighbourhood structure is not necessary so for another; (b) a global optimum is a local optimum with respect to all possible neighbourhood structures; (c) for many problems local optima with respect to one or several neighbourhoods are relatively close to each other. A recent example of a VNS algorithm for vehicle routing problems is the algorithm for the multi-depot VRPTW presented by Polacek et al. (2004). Large Neighbourhood Search (LNS) has been introduced for the VRPTW by Shaw (1997) and can be interpreted as a special case of VNS. Kilby et al. (2000) have shown that LNS is well suited for rich VRP.

In many cases it is assumed that transportation requests are 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 Feillet et al. (2005), but only few attempts have been made to tackle extensions of this problem, for example, by Schönberger et al. (2002). VRP with multiple pickup and delivery locations have been studied by Savelsbergh and Sol (1995), Savelsbergh and Sol (1998), and Hasle (2003).

A comprehensive discussion of dynamic vehicle routing can be found in Psaraftis (1988), and Psaraftis (1995). 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 PDP, which recently has received increasing attention, see Fleischmann et al. (2004), Yang et al. (2004), and Powell et al. (2000), 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 Savelsbergh and Sol (1998). A column generation approach is used to solve the *General Pickup and Delivery Problem* (GPDP) presented by Savelsbergh and Sol (1995).

3. Problem formulation

In the *General Vehicle Routing Problem* (GVRP) 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. 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 a self-operated vehicle (make) or not (buy).

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.

4. Mathematical formulation

Let $\mathcal{O}$ denote the set of transportation requests (orders) and $\mathcal{V}$ denote the set of vehicles. For all $o \in \mathcal{O}$ let $l_{(o,1)}, \ldots, l_{(o,\lambda_o)}$ denote the locations belonging to order $o \in \mathcal{O}$ and for $1 \leqslant \mu \leqslant \lambda_o$ let $n_{(o,\mu)}$ denote a node corresponding to $l_{(o,\mu)}$. For all $v \in \mathcal{V}$ let $l_{(v,1)}, \ldots, l_{(v,\lambda_v)}$ denote the locations which have to be visited by vehicle $v \in \mathcal{V}$, i.e., the start and end location of the tour as well as possible interim locations. For $1 \leqslant \mu \leqslant \lambda_v$ let $n_{(v,\mu)}$ denote a node corresponding to $l_{(v,\mu)}$. Let

$$\mathcal{N} := \bigcup_{o \in \mathcal{O}} \{n_{(o,\mu)} | 1 \leqslant \mu \leqslant \lambda_o\} \cup \bigcup_{v \in \mathcal{V}} \{n_{(v,\mu)} | 1 \leqslant \mu \leqslant \lambda_v\}$$

and

$$\mathcal{A} := \mathcal{N} \times \mathcal{N} \setminus \{(n,n) | n \in \mathcal{N}\}.$$

Note that different nodes $n \in \mathcal{N}$ may correspond to the same geographical location.

For each node $n \in \mathcal{N}$ lower and upper bounds specifying the time windows are denoted by $t_n^{\min}$ and $t_n^{\max}$. For each vehicle $v \in \mathcal{V}$ the travel time for an arc $(n,m) \in \mathcal{A}$ including some possible service time at node n is denoted by d_{nm}^v. For each vehicle $v \in \mathcal{V}$ the cost for travelling from node $n \in \mathcal{N}$ to node $m \in \mathcal{N}$ is denoted by c_{nm}^v. For each order $o \in \mathcal{O}$ the revenue gained when the order is served is denoted by p_o. Let δ_{ov} denote whether order $o \in \mathcal{O}$ may be served by vehicle $v \in \mathcal{V}$ ($\delta_{ov} = 1$), or not ($\delta_{ov} = 0$). Every vehicle supplies some (typically multi-dimensional) non-negative resource r^v (the capacity). At every node some shipments may be loaded or unloaded which require or release a certain amount of the resource the vehicle supplies. For every $n \in \mathcal{N}$ let r_n denote the (typically multi-dimensional) amount of resource requirements for the shipments loaded or unloaded at the node. All operations on resource requirements and supply like summation or comparison can be understood element wise. If a shipment is loaded r_n is non-negative, if it is unloaded r_n is non-positive.

Definition. A sequence of distinct nodes $\theta = (n_1, \ldots, n_\lambda)$ is a tour of a vehicle $v \in \mathcal{V}$ if and only if

- $n_1 = n_{(v,1)}$ and $n_\lambda = n_{(v,\lambda_v)}$,
- there exists a subset $\mathcal{O}_\theta \subseteq \mathcal{O}$ with

$$\{n_1, \ldots, n_\lambda\} = \{n_{(v,1)}, \ldots, n_{(v,\lambda_v)}\} \cup \bigcup_{o \in \mathcal{O}_\theta} \{n_{(o,1)}, \ldots, n_{(o,\lambda_o)}\}$$

- there exist times $t_{n_1}, \ldots, t_{n_\lambda}$ such that

$$t_{n_i} + d_{n_i n_{i+1}}^v \leqslant t_{n_{i+1}} \quad \text{for all } 1 \leqslant i < \lambda$$

and

$$t_{n_{(v,i)}} \leqslant t_{n_{(v,i+1)}} \quad \text{for all } 1 \leqslant i < \lambda_v$$

and

$$t_{n_{(o,i)}} \leqslant t_{n_{(o,i+1)}} \quad \text{for all } o \in \mathcal{O}_\theta,\ 1 \leqslant i < \lambda_o.$$

Definition. A tour $\theta = (n_1, \ldots, n_\lambda)$ of a vehicle $v \in \mathcal{V}$ is *feasible* if and only if

- $t_{n_i}^{\min} \leqslant t_{n_i} \leqslant t_{n_i}^{\max}$ for all $1 \leqslant i \leqslant \lambda$,
- $0 \leqslant \sum_{j=1}^{i} r_{n_j} \leqslant r^v$ for all $1 \leqslant i \leqslant \lambda$ and
- $\delta_{ov} = 1$ for all $o \in \mathcal{O}_\theta$.

The GVRP is the problem of finding distinct feasible tours maximising the profit determined by the accumulated revenue of all orders served by a vehicle reduced by the cost for operating the tours.

The GVRP can be modelled using the binary variables x_{nm}^v and y_n^v. x_{nm}^v indicates whether $m \in \mathcal{N}$ is visited immediately after node $n \in \mathcal{N}$ by vehicle $v \in \mathcal{V}$ ($x_{nm}^v = 1$), or not ($x_{nm}^v = 0$). y_n^v indicates whether node $n \in \mathcal{N}$ is visited by vehicle $v \in \mathcal{V}$ ($y_n^v = 1$), or not ($y_n^v = 0$). For each node $n \in \mathcal{N}$ the GVRP contains the variables t_n and ρ_n. If node $n \in \mathcal{N}$ is visited by a vehicle t_n specifies the arrival time and ρ_n specifies the current load of the vehicle. If no vehicle visits node $n \in \mathcal{N}$ both t_n and ρ_n are without any meaning.

The contribution of each vehicle $v \in \mathcal{V}$ to the objective function is

$$\sum_{o \in \mathcal{O}} y_{n_{(o,1)}}^v p_o - \sum_{(n,m) \in \mathcal{A}} x_{nm}^v c_{nm}^v.$$

The first term represents the accumulated revenue of served orders, the second term represents the accumulated costs for vehicle movements.

The GVRP is

$$\text{maximise} \quad \sum_{v \in \mathcal{V}} \left(\sum_{o \in \mathcal{O}} y_{n_{(o,1)}}^v p_o - \sum_{(n,m) \in \mathcal{A}} x_{nm}^v c_{nm}^v \right) \tag{1}$$

subject to

$$\sum_{(n,m) \in \mathcal{A}} x_{nm}^v = \sum_{(m,n) \in \mathcal{A}} x_{mn}^v \quad \text{for all } v \in \mathcal{V}, n \in \mathcal{N} \tag{2}$$

$$y_n^v = \sum_{(n,m) \in \mathcal{A}} x_{nm}^v \quad \text{for all } v \in \mathcal{V}, n \in \mathcal{N} \tag{3a}$$

$$\sum_{v \in \mathcal{V}} y_n^v \leqslant 1 \quad \text{for all } n \in \mathcal{N} \tag{3b}$$

for all $v \in \mathcal{V}, (n,m) \in \mathcal{A}$ with $n \neq n_{(v,\lambda_v)}$:

$$\text{if } x_{nm}^v = 1 \text{ then } t_n + d_{nm}^v \leqslant t_m \tag{4a}$$

$$t_n^{\min} \leqslant t_n \leqslant t_n^{\max} \quad \text{for all } n \in \mathcal{N} \tag{4b}$$

$$t_{n_{(v,\mu)}} \leqslant t_{n_{(v,\mu+1)}} \quad \text{for all } v \in \mathcal{V}, 1 \leqslant \mu < \lambda_v \tag{5a}$$

$$t_{n_{(o,\mu)}} \leqslant t_{n_{(o,\mu+1)}} \quad \text{for all } o \in \mathcal{O}, 1 \leqslant \mu < \lambda_o \tag{5b}$$

$$y_{n_{(v,\mu)}}^v = 1 \quad \text{for all } v \in \mathcal{V}, 1 \leqslant \mu \leqslant \lambda_v \tag{6a}$$

$$\sum_{\mu=1}^{\lambda_o} y_{n_{(o,\mu)}}^v = \lambda_o y_{n_{(o,1)}}^v \quad \text{for all } o \in \mathcal{O}, v \in \mathcal{V} \tag{6b}$$

$$\rho_{n_{(v,1)}} = r_{n_{(v,1)}} \quad \text{for all } v \in \mathcal{V} \tag{7a}$$

for all $v \in \mathcal{V}, (n,m) \in \mathcal{A}$ with $n \neq n_{(v,\lambda_v)}$:

$$\text{if } x_{nm}^v = 1 \text{ then } \rho_m = \rho_n + r_m \tag{7b}$$

$$\text{for all } v \in \mathcal{V}, n \in \mathcal{N}: \text{ if } y_n^v = 1 \text{ then } 0 \leqslant \rho_n \leqslant r^v \tag{7c}$$

$$y_{n_{(o,1)}}^v \leqslant \delta_{ov} \quad \text{for all } o \in \mathcal{O}, v \in \mathcal{V} \tag{8}$$

$$x_{nm}^v \in \{0,1\} \quad \text{for all } v \in \mathcal{V}, (n,m) \in \mathcal{A}, \tag{9}$$

$$y_n^v \in \{0,1\} \text{ for all } v \in \mathcal{V}, n \in \mathcal{N} \tag{10}$$

The objective function is represented by (1). Eq. (2) represents the flow conservation constraints which impose that each vehicle which reaches a node $n \in \mathcal{N}$ also departs from the node. Constraints (3a) and (3b) impose that each node is visited at most once. Inequality (4a) imposes that each node which is not the starting point of a tour is reached no earlier than the preceding node plus the time required to travel from the preceding node to the node. Inequality (4b) impose that each arrival time is within the time windows of the node. Constraints (5a) and (5b) are the precedence constraints imposed on the sequence in which nodes associated to vehicles and orders are visited. Eq. (6a) imposes that all nodes which must be visited by a vehicle are visited by this vehicle. Eq. (6b) represents the grouping constraint which imposes that, if the first node of an order is visited by some vehicle, all other nodes belonging to the order are visited by the same vehicle. Constraints (7a) to (7c) are the capacity constraints which impose that the load at each node equals the load at the preceding node plus the additional load at the node and that at each node the load is below the capacity of the vehicle and non-negative. Inequality (8) represents the compatibility constraint which imposes that orders are only assigned to vehicles capable of serving the order. Finally, constraints (9) and (10) impose that the values of x_{nm}^v and y_n^v are binary.

The purpose of this mathematical formulation is to give a precise description of the problem. For many practical instances the number of variables and constraints is obviously too large to solve the

instance with a standard mixed integer programming solver. For example, an instance of a vehicle routing problem with 10 vehicles and 100 customers has $|\mathcal{N}| = 120$ nodes and $|\mathcal{A}| = |\mathcal{N}| \cdot (|\mathcal{N}| - 1) = 14\,280$ arcs. Therefore, it has 142 800 binary variables x_{nm}^{v} and even more constraints. As standard mixed integer programming solver cannot be used to solve problems of this size, this paper presents heuristic approaches for determining feasible solutions of high quality by iteratively improving the current (feasible) solution. The heuristics presented in this paper make use of the following assumptions:

$$\text{for all } v \in \mathcal{V} : (n_{(v,1)}, n_{(v,2)} \ldots, n_{(v,\lambda_v)}) \quad \text{is a feasible tour} \tag{A1}$$

and

$$\text{for all } o \in \mathcal{O} : \sum_{\mu=1}^{i} r_{n_{(o,\mu)}} \geqslant 0 \quad \text{for all } 1 \leqslant i \leqslant \lambda_o. \tag{A2}$$

Assumption (A1) guarantees that all initial tours are feasible. Assumption (A2) guarantees that all loads which are delivered to a location belonging to some order $o \in \mathcal{O}$ are picked up at a preceding location belonging to the same order.

5. Reduced Variable Neighbourhood Search

This section presents an algorithm following the Reduced Variable Neighbourhood Search (RVNS) scheme described in Hansen and Mladenović (2003). The RVNS algorithm changes the neighbourhood structure during the search by selecting different neighbourhoods defined by the following elementary neighbourhood operators:

INSERT: The INSERT-operator randomly chooses an unscheduled order and inserts it to the tour of the vehicle with lowest incremental costs. If the order cannot be feasibly inserted the current solution is not changed.

REMOVE: The REMOVE-operator randomly chooses a scheduled order and removes it from the tour it is assigned to.

RELOCATE: The RELOCATE-operator randomly chooses a scheduled order and removes it from the tour it is assigned to. The order is inserted to the tour of the vehicle with lowest incremental costs.

REPLACE: The REPLACE-operator randomly chooses an order o_1 and removes it from its tour if it is scheduled. Another order o_2 assigned to a vehicle v is chosen and removed from its tour. Then, order o_1 is inserted to the tour of vehicle v. If o_1 cannot be feasibly inserted the current solution is not changed.

SWAP: The SWAP-operator randomly chooses two scheduled orders o_1 and o_2. It removes both from their tour and inserts them to the respective other tour. If either o_1 or o_2 cannot be feasibly inserted the current solution is not changed.

Given the neighbourhood structures defined by these operators, the RVNS algorithm can be outlined as illustrated in Fig. 1.

The algorithm starts with the determination of an initial solution s. Until a stopping criterion is met, e.g. the maximum computing time, the RVNS algorithm repeats the following steps. First, the next neighbourhood $N_k^{\mathrm{VNS}}(s)$ to be considered is chosen randomly. Then, a new solution $s^* \in N_k^{\mathrm{VNS}}(s)$ is generated. This new found solution s^* is accepted as the

Initialisation: Find an initial solution s; choose a stopping condition
Repeat the following until the stopping condition is met:
(1) Choose $k \in \{1, \ldots, 5\}$
(2) *Shaking:* Generate a point $s^* \in N_k^{\mathrm{VNS}}(s)$
(3) *Move or not:* If the new solution s^* is better than s, move there $(s \leftarrow s^*)$

Fig. 1. Reduced Variable Neighbourhood Search.

next current solution if the objective value is improved.

6. Large Neighbourhood Search

This section presents a Large Neighbourhood Search (LNS) algorithm for the GVRP. Let s denote a feasible solution of the GVRP and let $|\mathcal{O}_s|$ denote the number of transportation requests which are assigned to the tour of some vehicle. For each solution s let $N_k^{\mathrm{LNS}}(s)$ denote the kth neighbourhood of s which is defined by removing k transportation requests from their tours. The LNS algorithm can be outlined as illustrated in Fig. 2.

The algorithm starts with an initial solution s which can be obtained by any tour construction method. In each iteration the number k of transportation requests to be removed is chosen. Then, k transportation requests are removed from the tours they are currently assigned to. A new solution s^* is generated by re-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 choice of the next neighbourhood is, differently as proposed in Hansen and Mladenović (2003), completely undetermined. The probability that a better solution can be found is strongly connected to the choice of the neighbourhood $N_k^{\mathrm{LNS}}(s)$. 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. Therefore, it is not clear how to change k effectively and we choose it randomly in our implementation.

6.1. Removals

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 removed randomly from the tours, but in this case, some of them may not be *related* to each other in any way. Hence, the re-insertions of these transportation requests are independent of another and the same effect can be achieved by removing less transportation requests and performing the re-insertions sequentially.

For the VRP Shaw (1997) 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 the GVRP, as transportation requests may have 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 the 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 Fig. 3. In the illus-

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 $s' \in N_k^{\mathrm{LNS}}(s)$
(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 $(s \leftarrow s^*)$

Fig. 2. Large Neighbourhood Search.

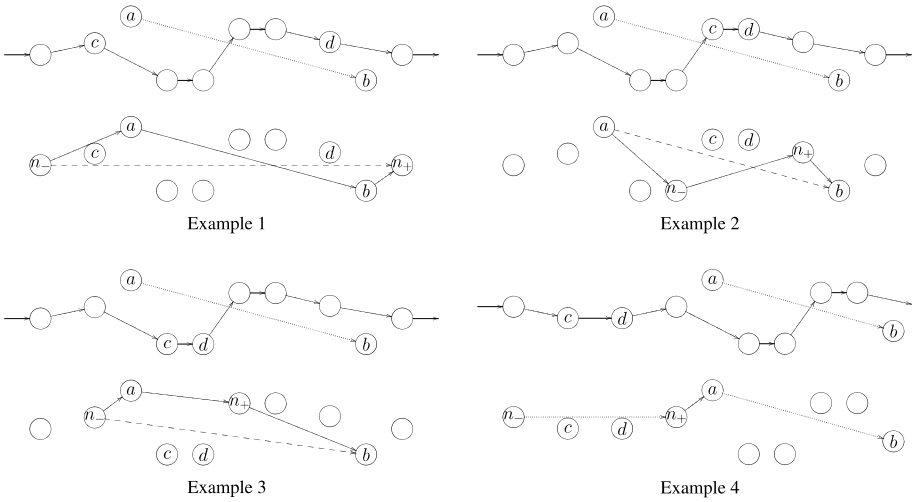


Fig. 3. Determination of relatedness values.

tration (a) represents node $n_{(o,1)}$ and (b) represents node $n_{(o,\lambda_o)}$. To determine the relatedness value of a scheduled order (represented by nodes (c) and (d) in the illustration), we do not directly regard the scheduled order itself, but its preceding and succeeding point in the tour. Let n_- and n_+ denote the predecessor and successor of the scheduled order. Now determine all sequences $\theta = (n_1, \ldots, n_{\lambda_o+2})$ containing the subsequences (n_-, n_+) and $(n_{(o,1)}, \ldots, n_{(o,\lambda_o)})$ such that there exist arrival times $t_{n_1}, \ldots, t_{n_{\lambda_o+2}}$ with

$$t_{n_i} + d^v_{n_i n_{i+1}} \leqslant t_{n_{i+1}} \quad \text{for all } 1 \leqslant i < \lambda_o + 2$$

and

$$t^{\min}_{n_i} \leqslant t_{n_i} \leqslant t^{\max}_{n_i} \quad \text{for all } 1 \leqslant i \leqslant \lambda_o + 2.$$

Obviously, there may be various such sequences. Let us consider the sequence $\theta = (n_1, \ldots, n_{\lambda_o+2})$ with least costs

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

If, as in examples 1–3 of Fig. 3, n_+ is visited after $n_{(o,1)}$ and n_- is visited before $n_{(o,\lambda_o)}$ the relatedness value is

$$\text{relatedness value} := c_\theta - c^v_{n_1 n_{\lambda_o+2}}.$$

Otherwise, n_- is visited after $n_{(o,\lambda_o)}$ or n_+ is visited before $n_{(o,1)}$, as shown in example 4 of Fig. 3. 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 candi-

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

Fig. 4. Removal of transportation requests using the relatedness measure.

date 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 Fig. 4.

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.

6.2. Insertions

In order to guarantee fast response times required in dynamic planning, we propose to use fast insertion heuristics to generate new solutions in step 3 of the LNS method. The auction method for the VRPTW proposed by Antes and Derigs (1995) can be easily modified in order to consider the various real-life requirements found in the GVRP.

The (local) efficiency of an insertion is determined by the difference between the revenue of the inserted order and the incremental cost for insertion. We say that an insertion possibility is *efficient* if the incremental cost is smaller than the revenue of the order. If no feasible insertion is possible an infinite incremental cost is assumed.

Each iteration of the auction method can be divided into three phases which are illustrated in Fig. 5. In the first phase all unscheduled orders request and receive from each suitable 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 with high efficiency to be inserted to the tour. The method stops if no order can be efficiently inserted and continues otherwise with the next iteration.

7. Computational experiments

In order to evaluate our algorithms, test problems have been generated incorporating most of the complexities found in the dynamic real-life problem. In our problem the carrier has to transport shipments between European airports. We assumed a frequency distribution of pickups and deliveries at these airports as illustrated in Fig. 6. The frequency of pickups and deliveries at the airports is indicated by the size of the circle. Most of the shipments have to be picked up or delivered to an airport in the region between Paris, Düsseldorf, and Frankfurt. Some shipments, however, have very remote origins or destinations, for example Florence, Dublin, Gothenburg, and Helsinki.

A heterogeneous vehicle fleet has been generated where some of the vehicles have refrigerated cargo bodies and some are manned by two drivers. We assume that all vehicles are en-route when planning starts and each vehicle becomes available at one of the airports during some time of the day. All vehicles eventually have to return to the depot in Frankfurt.

We randomly generated shipments such that the frequency distribution illustrated in Fig. 6 is achieved. Transportation requests have been generated by choosing one full or half truckload shipment or by combining two half truckload shipments with

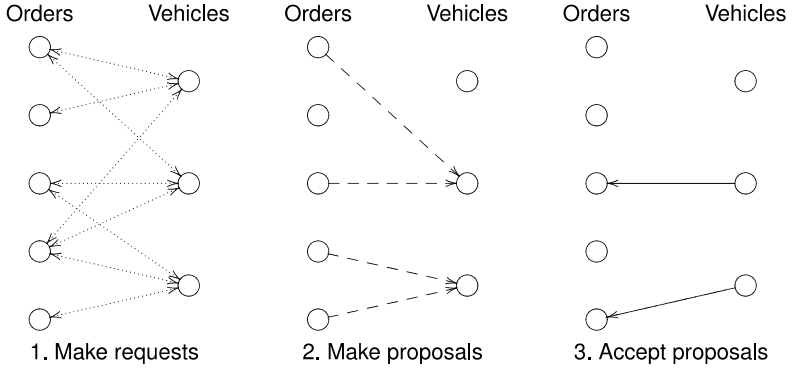


Fig. 5. Illustration of the auction method.



Fig. 6. Distribution of pickup and delivery locations.

identical pickup or delivery location. Some of the transportation requests require a vehicle with refrigerated cargo body, some require a vehicle manned by two drivers.

Travel distances are based on the direct distances between the airports. As in the real-life problem, they are multiplied by 1.3 to consider the average deviation occurring in road transport. Travel costs are proportional to the travel distance. Vehicles with refrigerated cargo bodies and vehicles manned by two drivers are more expensive than vehicles with standard cargo bodies and those manned by one driver. The revenue of the transportation requests is set to double the costs of the cheapest vehicle capable of transporting the shipments. That is, the shippers are not only willing to pay for the transport itself, but also for the return trip to the start location.

For our test cases we generated $|\mathscr{V}|$ vehicles and transportation request which become known at some time during our simulation of 10 hours. In the beginning only $|\mathscr{C}_0|$ orders are known to the carrier and every hour $|\mathscr{C}_t|$ new orders become known. For all orders we set the length of the time windows at each location to the same value τ, i.e., either 2 hours or 12 hours.

In our simulations the algorithms were only allowed 60 seconds of computing time per timestep (representing one hour in our simulation scenario) on a personal computer with Intel Pentium 4 processor with 3.00 GHz. All decisions taken in a preceding timestep may be revised in a subsequent timestep. However, at the end of each timestep all unscheduled transportation requests are assumed to be rejected or subcontracted by external carriers, i.e., whenever a transportation request is not assigned to the tour of a vehicle at the end of a timestep the transportation request is removed from the model. With the beginning of a new timestep the start location of every vehicle is updated in order to consider the vehicle's movement. Whenever the first location belonging to a transportation request is visited by a vehicle the transportation request is removed from the model and the corresponding locations are added to the sequence of locations the vehicle must visit before reaching the final location of the tour. New transportation requests are added to the model in the beginning of each timestep and are inserted to the tours by the auction method before the Reduced Variable Neighbourhood Search or Large Neighbourhood Search method is invoked.

Table 1 shows the results of our computational experiments on the instances p1–p24. The average values of the objective function values obtained by multiple runs of our heuristics are listed in column AVG(f) and the standard deviation in column σ. The LNS method using unrelated removals is denoted by LNS-U. The LNS method using related removals is denoted by LNS-R. In each iteration of the LNS algorithms the number $k \in [2, 30]$ of transportation requests to be removed was chosen randomly.

We can see that neither RVNS nor LNS-U clearly dominate each other. While LNS-U seems to perform better for small problems, RVNS seems to perform better for the large instances. The use of the relatedness criterion significantly improves the performance of the LNS method and LNS-R produces the best average results in almost all cases. However, in some cases the RVNS still performs slightly better.

Response times of all algorithms were below a few seconds and average response times were mostly below one second. Due to the smaller size of the neighbourhoods to be explored, the RVNS algorithm has much smaller response times than the LNS algorithms.

As the GVRP generalises the classical models VRP and PDP, the heuristics presented in this paper may also be used for instances of these problems. However, in these problems all transportation requests must be served. Therefore, the revenues

Table 1
Results

Nos.	Problem				RVNS		LNS-U		LNS-R	
	$\lvert\mathscr{V}\rvert$	$\lvert\mathscr{O}_0\rvert$	$\lvert\mathscr{O}_t\rvert$	τ	AVG(f)	σ	AVG(f)	σ	AVG(f)	σ
p1	100	300	20	2	181038.00	2267.39	186550.44	482.69	193658.71	1221.44
p2	100	300	20	2	197575.33	1334.72	205134.57	2524.10	219957.12	824.21
p3	100	300	20	2	188633.12	1295.13	196017.53	2445.82	198857.20	1501.96
p4	100	300	20	2	185183.01	1329.64	191217.30	3238.03	196001.81	2157.83
p5	100	300	20	12	289579.38	1985.62	297616.97	1577.71	299628.28	1977.18
p6	100	300	20	12	273997.70	1742.22	278874.72	2889.00	289026.76	1303.75
p7	100	300	20	12	262702.54	1415.41	268784.42	1229.85	272337.89	1037.70
p8	100	300	20	12	271150.15	1564.48	281425.79	951.06	287839.02	2907.92
p9	250	750	50	2	505072.76	5866.67	506442.17	2300.82	527543.21	6252.58
p10	250	750	50	2	487759.72	2217.41	488689.09	1678.84	497543.61	3330.79
p11	250	750	50	2	502367.70	2341.00	498615.52	3114.95	511879.04	3732.24
p12	250	750	50	2	526699.71	7271.28	518665.58	3433.58	532709.90	5678.19
p13	250	750	50	12	713547.18	4454.70	717727.04	2492.39	738878.17	5546.93
p14	250	750	50	12	722309.20	2690.77	712107.92	5874.45	721670.85	3626.29
p15	250	750	50	12	689475.10	3151.38	679460.74	2079.40	692698.84	3795.64
p16	250	750	50	12	702658.55	7443.01	682183.45	3281.88	692122.23	1583.69
p17	500	1500	100	2	1083725.74	4825.70	1069182.97	3426.54	1106940.06	11194.71
p18	500	1500	100	2	1062635.10	9685.07	1065859.76	10432.19	1089812.91	3170.01
p19	500	1500	100	2	1022168.34	7713.50	1022978.35	4348.34	1032741.35	4922.99
p20	500	1500	100	2	1059648.63	9481.78	1026574.20	5672.72	1065159.95	15579.42
p21	500	1500	100	12	1395467.83	9011.79	1377983.56	13635.94	1399397.98	5951.14
p22	500	1500	100	12	1430855.62	7125.78	1400686.31	5552.82	1439543.12	10026.49
p23	500	1500	100	12	1432836.25	5338.43	1415294.39	8211.29	1430347.23	7313.15
p24	500	1500	100	12	1431020.98	3362.37	1434010.56	9408.59	1438788.69	9012.45

p_o of all orders $o \in \mathscr{O}$ must be set to very large values, such that the heuristics are encouraged to only produce solutions in which all orders are served. The RVNS method will not be very effective as only the neighbourhood operators RELOCATE and SWAP allow to move from one solution in which all transportation requests are served to another solution in which all transportation requests are served. Furthermore, our heuristics do not aim at reducing the number of vehicles used, as this is usually not a goal in dynamic planning where the number of available vehicles is fixed. Note that highly specialised algorithms developed for the classical vehicle routing problems can exploit problem-specific knowledge and should have better performance than any method developed for the GVRP. The goal of this work, however, was to develop algorithms that can be used for dynamic problems considering a variety of practical complexities that are not considered by the classical models.

8. Conclusions

Many practical routing problems encounter complexities which are not considered in the classical models. In this paper we presented the General Vehicle Routing Problem (GVRP) which is capable of handling a variety of real-life requirements. It generalises the well-known and well-studied classical models VRP and PDP. Furthermore, it amalgamates some extensions of the classical models which, up to now, have only been treated independently.

As the GVRP is highly constrained, the search space is likely to contain many solutions such that it is impossible to go from one solution to another using a single neighbourhood structure. Therefore, we propose iterative improvement approaches based on the idea of changing the neighbourhood structure during the search. To avoid getting trapped in a local optimum with respect to one neighbourhood structure, the RVNS algorithm changes between various elementary neighbourhood operators and the LNS approach uses nested neighbourhoods of different size. We have proposed a tour dependent relatedness measure for the LNS, as geographical closeness cannot be used in the GVRP. Our computational experiments have shown that this relatedness measure significantly outperforms random removals.

Our algorithms perform well concerning response times for problems with hundreds of vehicles and several hundreds of transportation

requests. As average response times were mostly less than a second our algorithms may be used in interactive dynamic planning systems.

References

Antes, J., Derigs, U., 1995. 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.

Bräysy, O., Gendreau, M., 2005a. Vehicle routing problem with time windows. Part I: Route construction and local search algorithms. Transportation Science 39 (1), 104–118.

Bräysy, O., Gendreau, M., 2005b. Vehicle routing problem with time windows. Part II: Metaheuristics. Transportation Science 39 (1), 119–139.

Campbell, A., Savelsbergh, M., 2004. Efficient insertion heuristics for vehicle routing and scheduling problems. Transportation Science 38 (3), 369–378.

Cordeau, J.-F., Desaulniers, G., Desrosiers, J., Solomon, M., Soumis, F., 2002. VRP with time windows. In: Toth, P., Vigo, D. (Eds.), The Vehicle Routing Problem. SIAM Monographs on Discrete Mathematics and Applications, Philadelphia, pp. 157–193.

Cordeau, J.-F., Gendreau, M., Hertz, A., Laporte, G., Sormany, J.-S., 2004. New heuristics for the vehicle routing problem. Les Cahiers du GERAD, G-2004-33, HEC Montréal, Canada.

Feillet, D., Dejax, P., Gendreau, M., 2005. Traveling Salesman Problems with Profits. Transportation Science 39 (2), 188–205.

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

Hansen, P., Mladenović, N., 2003. A tutorial on Variable Neighborhood Search. Les Cahiers du GERAD, G-2003-46, HEC Montréal, Canada.

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

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

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

Laporte, G., Semet, F., 2002. Classical heuristics for the capacitated VRP. In: Toth, P., Vigo, D. (Eds.), The Vehicle Routing Problem. SIAM Monographs on Discrete Mathematics and Applications, Philadelphia, pp. 109–128.

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

Mladenović, N., Hansen, P., 1997. Variable Neighborhood Search. Computers and Operations Research 24, 1097–1100.

Polacek, M., Hartl, R., Doerner, K., Reimann, M., 2004. A variable neighborhood search for the multi depot vehicle routing problem with time windows. Journal of Heuristics 10, 613–627.

Powell, W., Snow, W., Cheung, R., 2000. Adaptive labeling algorithms for the dynamic assignment problem. Transportation Science 34 (1), 50–66.

Psaraftis, H., 1988. Dynamic vehicle routing problems. In: Golden, B., Assad, A. (Eds.), Vehicle routing: Methods and studies. North-Holland, Amsterdam, pp. 233–248.

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

Savelsbergh, M., Sol, M., 1995. The general pickup and delivery problem. Transportation Science 29 (1), 17–30.

Savelsbergh, M., Sol, M., 1998. DRIVE: Dynamic routing of independent vehicles. Operations Research 46, 474–490.

Schönberger, J., Kopfer, H., Mattfeld, H., 2002. A combined approach to solve the pickup and delivery selection problem. In: Operations Research Proceedings 2002. Springer, pp. 150–155.

Shaw, P., 1997. 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.

Yang, J., Jaillet, P., Mahmassani, H., 2004. Real-time multivehicle truckload pickup-and-delivery problems. Transportation Science 38 (2), 135–148.