A general heuristic for vehicle routing problems

David Pisinger *
Stefan Ropke *

25th February 2005

Abstract

We present a unified heuristic, which is able to solve five different variants of the vehicle routing problem: the vehicle routing problem with time windows (VRPTW), the capacitated vehicle routing problem (CVRP), the multi-depot vehicle routing problem (MDVRP), the site dependent vehicle routing problem (SDVRP) and the open vehicle routing problem (OVRP).

All problem variants are transformed to a rich pickup and delivery model and solved using the Adaptive Large Neighborhood Search (ALNS) framework presented in Ropke and Pisinger (2004). The ALNS framework is an extension of the Large Neighborhood Search framework by Shaw (1998) with an adaptive layer. This layer adaptively chooses among a number of insertion and removal heuristics, to intensify and diversify the search. The presented approach has a number of advantages: ALNS provides solutions of very high quality, the algorithm is robust, and to some extent self-calibrating. Moreover, the unified model allows the dispatcher to mix various variants of VRP problems for individual customers or vehicles.

As we believe that the ALNS framework can be applied to a large number of tightly constrained optimization problems, a general description of the framework is given, and it is discussed how the various components can be designed in a particular setting.

The paper is concluded with a computational study, in which the five different variants of the vehicle routing problem are considered on standard benchmark tests from the literature. The outcome of the tests is promising as the algorithm is able to improve 183 best known solutions out of 486 benchmark tests. The heuristic has also shown promising results for a large class of vehicle routing problems with backhauls, as demonstrated in Ropke and Pisinger (2005).

Keywords: metaheuristics, large neighborhood search, vehicle routing problem

1 Introduction

Most scientific papers in the area of heuristic solution methods for vehicle routing problems target a specific vehicle routing problem, for example vehicle routing problems with time windows (VRPTW). In such papers a heuristic is designed, implemented and fine-tuned to fit this particular problem type. Only a few papers (see e.g. Cordeau et al. [17, 19]) consider heuristics that "out-of-the-box" can be used to solve several problem types. We believe that general vehicle routing heuristics are an important research area as such heuristics are needed for real life problems, where the transportation needs of different companies often are different and thus call for various types of vehicle routing problems.

The heuristic in this paper is applied to five different problems: the vehicle routing problem with time windows (VRPTW), the capacitated vehicle routing problem (CVRP), the multi-depot vehicle routing problem (MDVRP), the site dependent vehicle routing problem (SDVRP) and the open vehicle routing problem (OVRP). In the CVRP one has to deliver goods to a set of customers with known demands on minimum-cost vehicle routes originating and terminating at a depot. The vehicles are assumed to be homogeneous and having a certain capacity. In some versions of the CVRP one also has to obey a route duration

*DIKU - Department of Computer Science, University of Copenhagen, Universitetsparken 1, DK-2100 Copenhagen Ø, Denmark. E-mail: {pisinger, sropke}@diku.dk

constraint that limits the lengths of the feasible routes. The VRPTW extends the CVRP by associating time windows with the customers. The time window defines an interval where in the customer should be supplied. The OVRP is closely related to the CVRP, but contrary to the CVRP a route ends as soon as the last customer has been served as the vehicles do not need to return to the depot. The MDVRP extends the CVRP by allowing multiple depots. The SDVRP is another generalization of the CVRP. In the SDVRP one can specify that certain customers only can be served by a subset of the vehicles, furthermore, vehicles can have different capacities in the SDVRP. In the CVRP, MDVRP and SDVRP one seeks to minimize the total traveled distance whereas in the OVRP and VRPTW, the first priority is to minimize the number of vehicles and minimizing the traveled distance is the second priority. It should be mentioned that most exact methods and some metaheuristics for the VRPTW minimize total traveled distance instead of minimizing number of vehicles used.

All problem types are transformed to a Rich Pickup and Delivery problem with time windows (RPDPTW) and are solved using the Adaptive Large Neighborhood Search (ALNS) framework presented in [49].

In the RPDPTW we have a number of requests to be carried out by a fixed set of vehicles. Each request consists of picking up a quantity of goods at one location and delivering it to another location. The objective of the problem is to find a feasible set of routes for the vehicles so that all requests are serviced, and such that the overall travel distance is minimized. A feasible route of a vehicle should start at a given location, service a number of requests such that the capacity of the vehicle is not exceeded, and finally end at a given location. A pickup or delivery should take place within a given time window. Each request furthermore has an associated pickup precedence number, and a delivery precedence number. A vehicle must visit the locations in nondecreasing order of precedences (see e.g. Sigurd et al. [54] for various applications of precedence constraints). Since not all vehicles may be able to service all requests (e.g. due to their physical size or the absence of some cooling compartments) we need to ensure that every request is serviced by a given subset of vehicles. Between any two locations we have an associated, nonnegative distance and travel time. It is assumed that travel times satisfy the triangle inequality. This assumption implies that any removal of requests from a feasible route will keep the route feasible with respect to the imposed time windows.

The five vehicle routing problems considered in the present paper have all been intensively studied in the literature. The two best known problems are the VRPTW and the CVRP. The VRPTW has been the target of extensive research and almost any type of metaheuristic has been applied to the problem. For recent surveys on the state of art in VRPTW research we recommend the survey by Cordeau et al [15] that describes both exact and heuristic methods, and the survey by Bräysy and Gendreau [8] that focuses on metaheuristics. It is hard to single out a few VRPTW metaheuristics as the amount of proposed heuristics are enormous, and no heuristic dominates all the other heuristics in all aspects. We would, however, like to mention the metaheuristic by Mester and Bräysy [42] as it has shown outstanding results on larger VRPTW instances with between 200 and 1000 customers. For the smaller VRPTW instances like the Solomon data set, some of the best heuristics in terms of solution quality achieved are the Large Neighborhood Search by Bent and Van Hentenryck [2] and the Hybrid Genetic Algorithm by Homberger and Gehring [32].

Solving the VRPTW to optimality has also received much attention. The current state of the art exact methods are proposed by Kallehauge et al [35], Irnich and Villeneuve [34] and Chabrier [10], and all follow the branch-and-price framework. The two first mentioned approaches furthermore strengthen the obtained lower bound by adding valid inequalities to the master problem. The size of the instances that consistently can be solved to optimality is rather limited as unsolved instances with 50 customers exist, but some large scale instances can be solved. For example, Kallehauge et al. [35] report that a 1000 customer instance has been solved. Solving problems of this size is only possible by current techniques if the instance has a certain structure and the time constraints are very tight. These observations justify the research into heuristics for the VRPTW as industrial routing problems demand robust algorithms for large-sized instances.

The CVRP literature is also vast, classical heuristics for the problem have been surveyed by Laporte and Semet [38], and metaheuristics have been surveyed by Gendreau et al. [29] and more recently by Cordeau et al. [16]. CVRP heuristics have typically been tested on 14 instances containing between 50 and 199 customers. In the early '90s very good metaheuristics for the CVRP were developed like the parallel tabu search by Taillard [23]. Most of the solutions to the 14 classical instances found back then have still not been improved. More recently, some larger instances have been introduced containing between 240 and 1200 customers by Golden et al. [30] and Li et al. [40]. These new instances seem to have spurred a

new interest into metaheuristics for the CVRP as indicated in the survey by Cordeau et al. [16].

Until recently, exact methods for the CVRP were dominated by branch-and-cut methods. One of the best branch-and-cut algorithms for the CVRP was developed by Lysgaard et al. [41]. Recent research results indicate that branch-and-cut-and-price algorithms are a more promising approach as shown by Fukasawa et al. [26]. For the CVRP, the largest problem that has been solved to optimality contains 135 customers.

The OVRP is a variant of the CVRP that has received less attention. The problem appears in various distribution problems, where the vehicle simply stops after the last delivery. The problem was introduced by Sariklis and Powell [51] where a two-phase cluster first-route second heuristic was proposed. Recently tabu search heuristics have been proposed by Fu et al. [25] and Brandão [6].

Tabu search heuristics for the MDVRP have been proposed by Renaud et al. [48] and Cordeau et al. [17]. The last paper deserves special attention as it describes a general heuristic that also solve periodic vehicle routing problems (PVRP) and periodic traveling salesman problems. Earlier, Chao et al. [12] proposed a *record-to-record* improvement heuristic for the MDVRP.

The SDVRP was first studied by Nag et al. [43] who developed several simple heuristics for the problem. Chao et al. [11] developed a more advanced heuristic and constructed several new test instances. Cordeau and Laporte [18] showed that the problem could be seen as a special case of the PVRP and they presented computational results obtained by solving the problem using their PVRP tabu search heuristic.

The main contribution of this paper is to describe a general ALNS heuristic that is able to solve all the above variants of the VRP problem. The computational results are promising as we, for the large scale VRPTW instances suggested by Gehring and Homberger [27], on average are able to decrease the number of vehicles used, and the method becomes more attractive compared to other heuristics as the problem size increases. For the OVRP, MDVRP and SDVRP we are able to improve a large number of best known solutions. The results for the CVRP problems are fair as the ALNS heuristic is comparable to most recently proposed heuristics but it is surpassed by the very best heuristic for the problem type.

Due to the promising results of ALNS, we give a general description of the paradigm, to make it easier to adapt the framework to other problem types. Various strategies for designing construction and removal heuristics are discussed, and the overall framework is presented in a general form.

In the following Section 2 we give a formal mathematical definition of the RPDPTW and in Section 3 we describe how the considered variants of the problem are transformed to the RPDPTW. In Section 4 we give a general presentation of the ALNS algorithm forming the core of our solution approach. Section 5 describes how the general framework has been adapted to solve the RPDPTW. Section 6 presents a number of computational experiments which document that the proposed heuristic performs not worse than state-of-art heuristics specialized to solve the considered variants of the problem. The paper is concluded in Section 7.

2 Formal problem definition

We now present a mathematical formulation of the RPDPTW problem. The mathematical model is used to describe the heuristic in details in later sections and to describe how the considered VRP variants are transformed to the RPDPTW.

Following the terminology of Desaulniers et al. [21], a problem instance of the pickup and delivery problem contains n requests and m vehicles. The problem is defined on a graph where $P = \{1, \ldots, n\}$ is the set of pickup nodes, and $D = \{n+1, \ldots, 2n\}$ is the set of delivery nodes. Request i is represented by node i and $i+n$. $K = \{1, \ldots, m\}$ is the set of all vehicles; one vehicle might not be able to serve all requests. These kinds of limitations are modeled by letting $P_k \subseteq P$ and $D_k \subseteq D$ be the set of pickups and deliveries that can be served by vehicle k. Since every request is serviced by the same vehicle we may assume that $i \in P_k \Leftrightarrow i + n \in D_k$, i.e. that both the pickup and delivery can be serviced by vehicle k. Define $N = P \cup D$ and $N_k = P_k \cup D_k$. Let $\tau_k = 2n + k,\ k \in K$ and $\tau'_k = 2n + m + k,\ k \in K$ be the nodes that represent the start and end terminals of vehicle k. The directed graph $G = (V, A)$ consists of the nodes $V = N \cup \{\tau_1, \ldots, \tau_m\} \cup \{\tau'_1, \ldots, \tau'_m\}$ and the arcs $A = V \times V$. For each vehicle we have a subgraph $G_k = (V_k, A_k)$, where $V_k = N_k \cup \{\tau_k\} \cup \{\tau'_k\}$ and $A_k = V_k \times V_k$. For each edge $(i, j) \in A$

we assign a distance $d_{ij} \geq 0$ and a travel time $t_{ij} \geq 0$. It is assumed that the travel times satisfy the *triangle inequality* i.e. $t_{ij} \leq t_{il} + t_{lj}$ for all $i, j, l \in V$. We assign a service time s_i and a time window $[a_i, b_i]$ to each node $i \in V$. The service time represents the time needed for loading and unloading and the time window indicates when the visit at the particular site should start; a visit to node i can only take place between time a_i and b_i. A vehicle is allowed to arrive to a site before the start of the time window but it has to wait until the start of the time window before the visit can be performed. For each node $i \in N$ we define l_i to be the amount of goods that should be loaded onto the vehicle at the particular node. We have that $l_i \geq 0$ for $i \in P$ and $l_i = -l_{i-n}$ for $i \in D$. Each vehicle $k \in K$ has room for a certain amount of goods, this capacity is given by C_k. Each node has assigned a *precedence* or *priority* Π_i. Nodes with low precedence must always be visited before nodes with higher precedence.

Each vehicle k should follow a legal route from its start terminal τ_k to its destination terminal τ'_k. A legal route $\overline{r}$ is a simple (loop-free) path

$$\overline{r} = (\tau_k = v_1, v_2, \ldots, v_h = \tau'_k) \tag{1}$$

satisfying the precedences and time windows at the customers, the capacity of the vehicle, and ensuring that a pickup takes place before a delivery, and that only requests serviceable by vehicle k are carried out.

More formally, we demand that the vehicle only visits nodes that can be serviced by the vehicle, i.e.

$$v_i \in N_k, \qquad i = 2, \ldots, h-1 \tag{2}$$

A pickup-delivery pair should be served by the same vehicle, and the pickup should take place before the delivery, hence we have

$$i \leq j, \qquad v_i \in P_k, v_j \in D_k, v_j = v_i + n \tag{3}$$

Precedences should be obeyed along the route, this is ensured by the constraints

$$i \leq j, \qquad \Pi_{v_i} \leq \Pi_{v_j} \tag{4}$$

To ensure that time windows are satisfied, we introduce $S_i \in \mathbb{R}_0^+$ to denote when the vehicle starts the service at site v_i. We then have the constraints

$$\begin{aligned}
&a_{v_i} \leq S_i \leq b_{v_i} && i = 1, \ldots, h && (5)\\
&S_{i+1} \geq S_i + s_i + t_{v_i, v_{i+1}} && i = 1, \ldots, h-1 && (6)\\
&a_{\tau_k} \leq S_1 \leq b_{\tau_k} && && (7)\\
&a_{\tau'_k} \leq S_h \leq b_{\tau'_k} && && (8)
\end{aligned}$$

where $[a_{\tau_k}, b_{\tau_k}]$ is the time window of terminal τ_k and $[a_{\tau'_k}, b_{\tau'_k}]$ is the time window of terminal τ'_k. Finally, the capacity of the vehicle should be respected throughout the path. For this purpose we introduce $L_i \in \mathbb{R}_0^+$ to denote the load of the vehicle at node i after serving node i. Then we have

$$\begin{aligned}
&L_i \leq C_k && i = 1, \ldots, h && (9)\\
&L_{i+1} = L_i + l_{i+1} && i = 1, \ldots, h-1 && (10)\\
&L_1 = 0 && && (11)\\
&L_h = 0 && && (12)
\end{aligned}$$

The *travel cost* of a given route $\overline{r}$ is

$$c_{\overline{r}} = \sum_{i=1}^{h-1} d_{v_i, v_{i+1}} \tag{13}$$

Situations may occur where not all requests can be serviced by the available vehicles. To model this situation we create n dummy routes, consisting of a single request. These routes do not make use of any vehicles but they have a large cost, denoted Γ. Requests that are not served by a vehicle are said to be located in the *request bank*.

The whole problem can now be formulated as follows: let R be the set of all feasible routes. The boolean matrix $(\alpha_{j\overline{r}})$ for $\overline{r} \in R$ and $j = 1, \ldots, n$ is used to indicate whether request j is serviced using route $\overline{r}$. The boolean matrix $(\beta_{k\overline{r}})$ for $\overline{r} \in R$ and $k = 1, \ldots, m$ is used to indicate whether the route $\overline{r}$ is carried out by vehicle k. Using binary variables $x_{\overline{r}}$ to indicate whether route $\overline{r}$ is used in the solution we get the following model

$$\min \quad f(x) = \sum_{\overline{r} \in R} c_{\overline{r}} x_{\overline{r}} \tag{14}$$

$$\text{s.t.} \quad \sum_{\overline{r} \in R} \alpha_{j\overline{r}} x_{\overline{r}} = 1 \qquad j = 1, \ldots, n \tag{15}$$

$$\sum_{\overline{r} \in R} \beta_{k\overline{r}} x_{\overline{r}} = 1 \qquad k = 1, \ldots, m \tag{16}$$

$$x_{\overline{r}} \in \{0, 1\} \qquad \overline{r} \in R \tag{17}$$

Note that a dummy route is not assigned to any vehicle, that is for any dummy route $\overline{r}$ we have that $\beta_{k\overline{r}} = 0, \forall k = 1, \ldots, m$

3 Problem transformations

The heuristic in this paper is applied to five different problems — VRPTW, CVRP, OVRP, MDVRP, SDVRP — which all are transformed to a RPDPTW. The conversions which will be described in the following paragraphs are extensions of the transformations presented by Ropke and Pisinger [50] for solving VRP problems with Backhauls.

3.1 Vehicle Routing Problem with Time Windows

In order to transform a VRPTW instance to a RPDPTW instance we map every customer in the VRPTW to a request in the RPDPTW. Such a request consists of a pick up at the depot and delivery at the customer site. The amount of goods that should be carried by the requests is equal to the demand of the corresponding customer. The time window of the pickup is set to $[a_d, a_d]$ where a_d is the start of the time window of the depot in the VRPTW and its service time is set to zero. The time window and service time of the delivery are copied from the corresponding customer in the VRPTW. In order to avoid routes that return to the depot for restocking we let all pickups and deliveries have precedence zero and one respectively. All vehicles in the RPDPTW have the same start and end terminals corresponding to the depot in the VRPTW. Distances and travel times in the RPDPTW are set in the natural way.

3.2 Capacitated Vehicle Routing Problem

A CVRP instance can easily be transformed to a VRPTW instance. This can for example be done by setting all travel and service times to zero and all time windows to [0,0]. If the CVRP contains a route duration constraint then travel times and durations should be set as in the CVRP. All time windows (including the ones at the end terminals) should be set to [0, D] where D is the route duration. The VRPTW is transformed to a RPDPTW as described in Section 3.1.

3.3 Site Dependent Vehicle Routing Problem

In the SDVRP a customer may only be serviced by a given subset of the vehicles, typically because the access paths to the node do not allow given vehicles to pass, or because specific facilities are demanded in the vehicle (e.g. a freezing compartment).

The SDVRP is easily modeled as a RPDPTW by using the transformation from CVRP to RPDPTW and noting that the RPDPTW allows us to specify the pickups P_k and deliveries D_k that can be carried out by vehicle k.

3.4 Open Vehicle Routing Problem

The OVRP is very close to the CVRP. The difference between the two problems is that in the OVRP the vehicles do not have to return to the depot. Thus an OVRP can be solved as an asymmetric CVRP by setting distances and travel times from every customer to the depot to zero.

The travel times in the resulting RPDPTW do not satisfy the triangle inequality, but our method is able to handle the problems anyway since $t_{ij} \leq t_{il} + t_{lj}$ only is violated when l is an end terminal. Our only reason for assuming that the triangle inequality is satisfied for the travel times is that we have to avoid situations where the removal of one or more requests causes the travel time to increase. As the node sequence $i \rightarrow l \rightarrow j$ where $l \in \{\tau'_1, \ldots, \tau'_m\}$ never occurs in a valid route this violation of the triangle inequality does not cause problems.

3.5 Multi-Depot Vehicle Routing Problem

Even though the underlying RPDPTW model supports multiple depots, we cannot use these depots to model the MDVRP. The problem is that we must assign each pickup to a depot and we do not know which depot is going to serve a given request.

Instead we create a dummy base location where all routes start and end and where all ordinary requests are picked up. We also create a dummy request for each vehicle k in the problem. The pickup and delivery locations of these requests are located at the depot of the corresponding vehicle. A dummy request has demand zero, it does not have any service time and it can be served at any time. The set N_k of each vehicle k contains all ordinary requests and the dummy request corresponding to the vehicle. In this way we ensure that each vehicle will carry precisely one dummy request.

The precedences Π_i of a pickup and delivery corresponding to an ordinary request are set to zero and two respectively, the precedence of the pickup and delivery of the dummy requests are set to one and three respectively. This ensures that all ordinary deliveries will be surrounded by the pickup and delivery of a dummy request. The distance and travel time between a pickup of an ordinary request and any other location is set to zero. All other distances and travel times are set as defined by the original MDVRP.

In a solution to the RPDPTW that serves all requests we know that each route will start at a start terminal, located at the dummy base location and perform a number of pickups, then go to the pickup of the dummy request. Next, the ordinary deliveries will be served and the vehicle will return to the delivery of the dummy request and then to the end terminal of the route. Before starting the pickup of the dummy request and after the delivery of it all travel times and distances will be zero. Furthermore travel times and distances are accumulated correctly while carrying the dummy request.

While solving MDVRP problems the cost of dummy routes Γ must be set to a sufficiently high number such that it never will be profitable to leave a dummy request in the request bank.

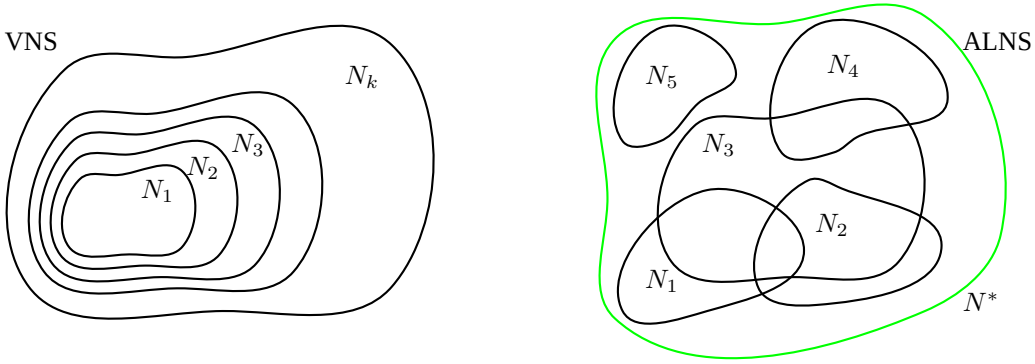

Figure 1: Illustration of neighborhoods used by VNS and ALNS. VNS typically operates on one type of neighborhood with variable depth while ALNS operates on structurally different neighborhoods $N_1, \ldots, N_k$ defined by the corresponding search heuristics. All neighborhoods $N_1, \ldots, N_k$ in ALNS are a subset of the neighborhood N^* defined by modifying q variables.

4 Adaptive Large Neighborhood Search

We will now describe the *Adaptive Large Neighborhood Search* (ALNS) framework used in the present paper. We believe that ALNS can be applied to a large class of difficult optimization problems, hence in the following we consider an optimization problem in the general IP form:

$$\min\{f(x) : Ax \leq b, x \in \mathbb{Z}^n\} \tag{18}$$

ALNS is a local search framework in which a number of simple algorithms compete to modify the current solution. In each iteration an algorithm is chosen to destroy the current solution, and an algorithm is chosen to repair the solution. The new solution is accepted if it satisfies some criteria defined by the local search framework applied at the master level.

To be more formal, we extend the domain of each variable x_i to $\mathbb{Z} \cup \{\perp\}$, where $\perp$ means undefined. A destroy heuristic chooses at most q variables which are assigned the value $\perp$. A repair heuristic assigns feasible values $x_i \in \mathbb{Z}$ to the q variables.

The ALNS framework is an extension of the *Large Neighborhood Search* presented by Shaw [53], where a large collection of variables are modified in each iteration. In ALNS the neighborhoods are searched by simple and fast heuristics. ALNS is also based on the *Ruin and Recreate* paradigm presented by Schrimpf et al. [52], or the *Ripup and Reroute* paradigm applied in [59]. In each iteration the current solution is partially destroyed and then repaired using some heuristics. ALNS also has similarities with *Very Large Neighborhoods Search* (VLNS) presented by Ahuja et al. [1]. In VLNS the algorithm operates on very large neighborhoods chosen in a way so that they still can be searched efficiently.

Variable Neighborhood Search (VNS) was presented by Hansen and Mladenovic [31]. VNS makes use of a parameterized family of neighborhoods, typically obtained by using a given neighborhood with variable depth. When the algorithm reaches a local minimum using one of the neighborhoods, it proceeds with a larger neighborhood from the parameterized family. When the VNS algorithm gets out of the local minimum it proceeds with the smaller neighborhood. On the contrary, ALNS operates on a predefined set of large neighborhoods corresponding to the destroy (removal) and repair (insertion) heuristics. The neighborhoods are not necessarily well-defined in a formal mathematical sense — they are rather defined by the corresponding heuristic algorithm. The difference between VNS and ALNS is graphically illustrated in Figure 1. In the sections that follow, we will identify a neighborhood with the heuristic searching it.

Instead of viewing the ALNS heuristic as a sequence of destroy and repair operations one can alternatively see it as a sequence of *fix* and *optimize* operations. The fix operation selects a number of variables that are fixed at their current value; the optimize operation seeks to find a near-optimal solution that respects the fixed variables, that is, only non-fixed variables can be changed. After the optimization operation, all variables are unlocked again. The fix operation is analogous to the destroy operation and the optimize operation is analogous to the repair operation. The fix/optimize view might be helpful when applying the heuristic to problems where the destroy and repair operations do not seem intuitive.

4.1 Outline of algorithm

ALNS can be based on any local search framework, e.g. simulated annealing, tabu search, guided local search. The general framework is outlined in Figure 2, where lines 2–8 form the main loop of the local search framework at the master level. Implementing a simulated annealing algorithm is straight forward as one solution is sampled in each iteration of the ALNS. A simple tabu search could for example be implemented by randomly sampling a number of candidate solutions and choosing the best non tabu solution.

In each iteration of the main loop we choose one destroy and one repair neighborhood in line 3. An adaptive layer stochastically controls which neighborhoods to choose according to their past performance (score). The more a neighborhood N_i has contributed to the solution process, the larger score π_i it gets, and hence it has a larger probability of being chosen.

The adaptive layer uses roulette wheel selection for choosing a destroy and a repair neighborhood. If the past score of a neighborhood i is denoted π_i and we have $\overline{m}$ neighborhoods, then we choose neighborhood N_j with probability

$$\frac{\pi_j}{\sum_{i=1}^{\overline{m}} \pi_i}$$

Adaptive Large Neighborhood Search

1 Construct a feasible solution x; set $x^* := x$
2 Repeat
3 Choose a destroy neighborhood N^- and a repair neighborhood N^+ using roulette wheel selection based on previously obtained scores $\{\pi_j\}$
4 Generate a new solution x' from x using the heuristics corresponding to the chosen destroy and repair neighborhoods
5 If x' can be accepted then set $x := x'$
6 Update scores π_j of N^- and N^+
7 If $f(x) < f(x^*)$ set $x^* := x$
8 Until stop criteria is met
9 Return x^*

Figure 2: Outline of the ALNS framework

Notice that the destroy and repair neighborhoods are selected independently, and hence two separate roulette wheel selections are performed.

In most applications the neighborhoods are searched by fast heuristics, and hence we can assume that they are equally fast. But if some heuristics are significantly slower than others, one may normalize the score π_i of a neighborhood with a measure of the time consumption t_i of the corresponding heuristic. This ensures a proper trade-off between time consumption and solution quality.

In line 4 of the ALNS-algorithm, we first destroy the current solution x using a heuristic searching the neighborhood N^- and then repair the solution using the heuristic corresponding to neighborhood N^+. It can be advantageous to use noising or randomization in the destroy and repair heuristics to obtain a proper diversification. In traditional local search heuristics the diversification is controlled implicitly by the local search paradigm (accept ratio, tabu list, etc.), but since we use large neighborhoods which are searched by simple heuristics, it is not sufficient to have a diversification operator at the master level. We also need a diversification operator at the sub-level to avoid stagnating search processes where the destroy and repair neighborhoods keep performing the same modifications to a solution.

Finally, in line 6 we update the scores π_i of the neighborhoods. A number of criteria can be used to measure how much a neighborhood contributes to the solution process: new best solutions are obviously given a large score, but also not previously visited solutions are given a score. Depending on the local search framework used on the master level, one may also give specific scores to accepted solutions e.g. in a simulated annealing framework. Since each step of the ALNS heuristic involves two neighborhoods (a destroy and a repair neighborhood), the score obtained in a given iteration is divided equally between them.

Every M iterations of the ALNS algorithm, the scores π_i are reset, and the probabilities for choosing the neighborhoods are recalculated. Each neighborhood is assigned a minimum probability for being chosen to ensure that statistical information about its performance can be collected. The probabilities for choosing a neighborhood can also be a weighted sum of the score during the last M iterations, and the overall score since the beginning of the algorithm.

4.2 Designing an ALNS algorithm

In order to design an ALNS algorithm for a given optimization problem one needs to

- Choose a number of fast construction heuristics which are able to respect a partial solution
- Choose a number of destroy heuristics supporting the chosen construction heuristics
- Choose a local search framework at the master level.

In each iteration the heuristic corresponding to a *destroy neighborhood* should remove a given number q of variables. The destroy neighborhoods (N^-) should be a proper mix of neighborhoods which can

intensify and diversify the search. To diversify the search, one may randomly choose q decision variables, i.e. using a *random removal* neighborhood. To intensify the search one may try to remove q "critical" variables, i.e. variables having a large cost or variables spoiling the current structure of the solution (e.g. edges crossing each other in a Euclidean traveling salesman problem). This is known as *worst removal* or *critical removal*. Concrete examples on *random removal* and *worst removal* neighborhoods in a VRP context are given in Sections 5.1.1–5.1.2.

One may also choose a number of related variables that are easy to interchange while maintaining feasibility of the solution. This *related removal* neighborhood was introduced by Shaw [53]. More formally we can measure the relatedness r_{ij} of two variables x_i and x_j by the deviation of the corresponding coefficients in the constraint matrix A in (18). The smaller r_{ij} the more related are variables x_i and x_j. How exactly r_{ij} should be defined depends on the concrete problem at hand, and one may even have several simultaneous neighborhoods defined by various choices of the relatedness measure (r_{ij}). In order to choose the q most related variables, one needs to solve the NP-hard *dispersion-sum problem* given by

$$\begin{aligned} \text{minimize} \quad & \sum_{i=1}^{n}\sum_{j=1}^{n} r_{ij}x_i x_j \\ \text{subject to} \quad & \sum_{j=1}^{n} x_j = q \\ & x_j \in \{0,1\}, \quad j = 1,\ldots,n \end{aligned} \tag{19}$$

A greedy heuristic for this problem running in $O(n^3)$ was presented in [44] together with a more time-consuming exact algorithm. If n is large, it may be too time-consuming even to compute the whole matrix (r_{ij}) and one will instead choose related variables according to some heuristics. Shaw [53] presented an algorithm running in $O(qn)$ time by initially selecting a variable at random, and then repeatedly selecting an already selected variable i and finding a variable j which minimizes r_{ij} and adding j to the set of chosen variables. An alternative heuristic is based on a modified Kruskal's algorithm for the minimum spanning tree problem, using r_{ij} as edge weights, which stops when a connected component with q or more elements has been constructed. The variables in this component are set to $\perp$. The worst-case running time of this algorithm is $O(n^2 \log n)$ as we have n^2 edges in Kruskal's algorithm. Ropke and Pisinger [49] used a modified version of this algorithm in the VRP for splitting requests on a route into two strongly connected subsets. It should be noted that solving the dispersion sum problem (19) to optimality seldom would be a good idea even if it could be done in a very short time. If r_{ij} is independent of the current solution the destroy neighborhood obtained by solving the dispersion sum problem to optimality would always assign $\perp$ to the same set of variables. Concrete examples on various *related removal* neighborhoods are given in Sections 5.1.3–5.1.5.

Following the same idea as in *related removal* one may choose a number of variables having small coefficients in the resource constraints in (18), as these generally are easy to interchange and loosely speaking can fill up unused resource constraints. We denote this strategy *small removal*.

Finally, one may use *history based removal* where the q variables are chosen according to some historical information as presented in [49]. The historical information could for example count how often setting a given variable (or set of variables) to a specific value leads to a bad solution. One may then try to remove variables that currently are assigned an improper value, based on the historical information. Variants of the *history based removal* neighborhood are discussed in Sections 5.1.6–5.1.7.

Recreate neighborhoods (N^+) are typically based on concrete well-performing heuristics for the given problem. These heuristics can make use of variants of the greedy paradigm, e.g. performing the locally best choice in each step, or performing the least bad choice in each step. An alternative variant of the greedy paradigm is to set all variables to their upper bound in (18), and repeatedly decrease the most expensive variable until a feasible solution is obtained. The repair heuristics can also be based on approximation algorithms or on exact algorithms that in some sense have been truncated to obtain faster solution times. Shaw [53] and Bent and Van Hentenryck [2] proposed more expensive algorithms like searching N^+ based on truncated branch-and-bound methods. Although ALNS mainly is intended to use cheap heuristics, more expensive search methods can be used if the scores of the corresponding neighborhoods are normalized with

respect to the time consumption. In the context of VRP problems, *recreate neighborhoods* are considered in more details in Section 5.2 discussing both simple greedy approaches and variants of regret heuristics.

Some optimization problems can naturally be split into a number of sub-problems, where each subproblem can be solved individually. Such problems include the Bin Packing Problem in which a number of bins are to be filled, or the Vehicle Routing Problem in which a number of routes are to be constructed. For such problems one should decide whether the subproblems should be solved one by one (*sequential heuristics*) or all subproblems should be solved at the same time (*parallel heuristics*). Sequential heuristics are easier to implement but may have the disadvantage that the last subproblem solved will be left with variables that do not fit well together. This is in some extent avoided in parallel heuristics.

A natural extension to the ALNS framework is to have *coupled neighborhoods*. In principle one may, for each destroy neighborhood N_i^-, define a subset $K_i \subseteq \{N^+\}$ of repair neighborhoods that can be used with N_i^-. The roulette wheel selection of repair neighborhoods will then only choose a neighborhood in K_i if N_i^- was chosen.

As a special case, one may have $K_i = \emptyset$ meaning that the neighborhood N_i^- takes care of both the destroy and repair steps. One could use an ordinary local search heuristic to compete with the other destroy and repair neighborhoods, ensuring that a thorough investigation of the solution space close to the current solution is made from time to time.

For some problems it may be sufficient to have a number of destroy and repair heuristics that are selected randomly with equal probability, that is without the adaptive layer. We will denote such a heuristic a *Large Multiple-Neighborhood Search* (LMNS). The LMNS heuristics share the robustness of the ALNS heuristics, while having considerably fewer parameters to calibrate.

4.3 Properties of the ALNS framework

The ALNS framework has several advantages. For most optimization problems we already know a number of well-performing heuristics which can form the core of an ALNS algorithm. Due to the large neighborhoods and diversity of the neighborhoods, the ALNS algorithm will explore large parts of the solution space in a structured way. The resulting algorithm becomes very robust, as it is able to adapt to various characteristics of the individual instances, and seldom is trapped in a local minima.

ALNS is particularly well suited for tightly constrained problems, where small neighborhoods are not sufficient to escape a local minima or certain areas of the solution space. In such problems, the large neighborhood search makes it possible to change many variables each time to reach new, feasible solutions.

The calibration of the ALNS algorithm is quite limited as the adaptive layer automatically adjusts the influence of each neighborhood used. It is still necessary to calibrate the individual sub-heuristics used for searching the destroy and repair neighborhoods, but one may calibrate these individually or even use the parameters used in existing algorithms.

In the design of most local search algorithms the researcher has to choose between a number of possible neighborhoods. In ALNS the question is not “either-or” but rather “both-and”. As a matter of fact, our experience is that the more (reasonable) neighborhoods the ALNS heuristic makes use of, the better it performs [49].

5 ALNS applied to the RPDPTW

We will now describe how the general ALNS framework has been adapted to the RPDPTW problem. The “variables” in the ALNS framework correspond to requests in the RPDPTW. A destroy neighborhood N^- consists of the removal of q requests from the existing routes, and assigning them to the *request bank*. We will describe the various heuristics associated with the destroy neighborhoods in Section 5.1. A repair neighborhood N^+ inserts requests from the request bank into one or more legal routes. The associated insertion heuristics are described in Section 5.2. The local search framework used at the master level is simulated annealing to be described in Section 5.3. Section 5.4 describes the noising method used to diversify the search of the heuristics. Finally, the scheme used for adjusting the weights in the roulette wheel selection is described in Section 5.5.

5.1 Request removal

The ALNS heuristic for the RPDPTW makes use of seven different removal heuristics, each searching a given removal neighborhood N^-. The heuristics take as input a given solution x and outputs q requests that have been removed from the routes.

5.1.1 Random removal

The simplest removal heuristic, `random removal`, selects q requests at random and removes them from the solution. This obviously has the effect of diversifying the search.

5.1.2 Worst removal

The purpose of the `worst removal` heuristic is to choose a number of requests that are very expensive, or which somehow spoil the structure of the current solution. In the RPDPTW it seems reasonable to try to remove requests with high cost and insert them at another place in the solution to obtain a better solution value.

Given a request i served by some vehicle in a solution x we define the cost of the request Δf_{-i} as the difference between the value of $f(x)$ and the cost of solution x where request i is removed completely from the problem.

The `worst removal` heuristic now repeatedly chooses a new request i, having the largest cost Δf_{-i} until q requests have been removed. The removal heuristic is randomized, the randomization is controlled by the parameter p. If p is small, the most expensive request is selected, while less expensive requests may be chosen for larger values of p with a probability that decreases with the cost Δf_{-i}. We refer to [49] for additional details.

5.1.3 Related removal

The purpose of the `related removal` heuristic is to remove a set of requests that in some sense are *related* and hence easy to interchange. For the RPDPTW we define the relatedness r_{ij} of two orders i and j solely by the distance between the requests, as introduced by Ropke and Pisinger [49]. Since each request i consists of a pickup node i and a delivery node $i+n$ we get the expression

$$r_{ij} = \frac{1}{D}\left(d'(i,j) + d'(i,j+n) + d'(i+n,j) + d'(i+n,j+n)\right) \tag{20}$$

where the distance measure $d'(u,v)$ between two nodes in this context is defined as

$$d'(u,v) = \begin{cases} d_{uv} & \text{if } u \text{ and } v \text{ are not located at a terminal} \\ 0 & \text{if } u \text{ or } v \text{ is located at a terminal} \end{cases} \tag{21}$$

The motivation for neglecting the distance from a terminal is that the terminal will be visited in any case, and hence should not contribute to the relatedness measure of two requests.

The denominator D is set to the number of nonzero terms in (20), i.e. the number of pickup and deliveries taking place at a site different from a terminal. Hence if all nodes are different from a terminal, $D := 4$ while if both requests have a pickup at a terminal $D := 1$.

The relatedness measure is used to remove customers as described in Shaw [53]. The algorithm initially selects a request i by random. Then it repeatedly chooses an already selected request j and selects a new request which is most related to j. The algorithm stops when q requests have been chosen. Like in the `worst removal` heuristic (Section 5.1.2) the process is controlled by a randomization parameter p. If p is zero, the most related request is always chosen in the inner loop. If $p > 0$ a less related request may be chosen, where the probability of choosing a request decreases with the relatedness measure r_{ij} and increases with p. The algorithm is described in more details in [49].

5.1.4 Cluster removal

The `cluster removal` heuristic is a variant of the `related removal` heuristic in which we try to remove clusters of related requests from a few routes. As a motivation, consider a route where the requests are grouped into two geographical clusters. When removing requests from such a route it is often important to remove one of these clusters entirely as the insertion methods otherwise would be prone to insert the removed requests back into the route. The `related removal` heuristic from Section (5.1.3) has a tendency to leave requests from such a cluster on the original route so therefore we propose a heuristic that seeks to remove an entire cluster at once.

Although we could use the same algorithm as above for selecting related requests — just restricted to a single route — we have chosen to use a heuristic based on strongly connected components, as described in Section 4.2. We simply run Kruskal's algorithm for the minimum spanning tree problem (using r_{ij} for the edge distances) and terminate the algorithm when two connected components remain. One of these clusters is chosen at random and the requests from the chosen cluster are removed. If less than q requests have been selected, we randomly pick a removed request and choose a request from a different route, that is most related to the given request. The route of the new request is partitioned into two clusters and so the process continues until the desired number of requests has been removed. We refer the reader to Ropke and Pisinger [50] for more details.

5.1.5 Time-oriented removal

The `time oriented removal` is another variant of the `related removal` heuristic. In this heuristic we try to remove requests that are served at roughly the same time as we hope that these requests are easy to interchange.

The heuristic works as follows. A request $\tilde{r}$ is chosen at random and the B requests that are closest to $\tilde{r}$ (according to the distance r_{ij} defined in (20)) are marked. We define a time-oriented distance between two requests as

$$\Delta t_{ij} = |t_{p_i} - t_{p_j}| + |t_{d_i} - t_{d_j}| \tag{22}$$

where t_{p_i} and t_{d_i} are the time when the pickup and the delivery of request i are served in the current solution. Among the B marked requests we select the $q-1$ that are closest to $\tilde{r}$ according to Δt_{ij}. The process is controlled by a randomization parameter p like in the `related removal` heuristic described in Section 5.1.3. These requests are removed together with $\tilde{r}$.

Before running the removal heuristic we first select a subset of all requests that are geographically close to the chosen request, as we observed that this selection made the heuristic perform better on large instances. The reason for this is that if the heuristic only considered requests that are close to the chosen request time-wise, then only one or two requests would be removed from each route in the larger problems, and this makes it hard to make any major improvements to the solution.

5.1.6 Historical node-pair removal

It is well-known from several metaheuristics that using historical information from the past local search (e.g. the long term memory or the aspiration level in tabu search) may improve the performance of a local search algorithm. In the present heuristic we look at the historical success of visiting two nodes right after each other in a route, while the heuristic in Section 5.1.7 looks at the historical success of servicing two requests by the same vehicle.

The `historical node-pair removal` heuristic (denoted the *neighbor graph removal heuristic* in [50]) makes use of both historical information and the present solution when removing the requests. With each pair of nodes $(u, v) \in A$ we associate a weight $f^*_{(u,v)}$ which indicates the best solution value found so far, in a solution which used edge (u, v). Initially $f^*_{(u,v)}$ is set to infinity, and each time a new solution is found in the ALNS algorithm, we update the weights $f^*_{(u,v)}$ of all edges used in the given solution, for which the edge weight can be improved.

We may use the edge weights $f^*_{(u,v)}$ to remove requests that seem to be misplaced. The removal heuristic simply calculates the cost of a request $(i, i+n)$ in the current solution by summing the weights of edges incident to i and $i+n$. The most costly request is removed, and the process is repeated until q

requests have been extracted. To ensure some variation in the extracted requests, randomness is introduced in the removal process.

5.1.7 Historical request-pair removal

An alternative history-based removal heuristic can make use of the historical success of placing pairs of requests in the same route. We will call this approach `historical request-pair removal` (denoted *request graph removal* in [50]).

For this purpose we introduce the weight $h_{(a,b)}$ with each pair of requests $(a, b) \in \{1, \ldots, n\} \times \{1, \ldots, n\}$. The weight $h_{(a,b)}$ denotes the number of times the two requests a and b have been served by the same vehicle in the B best unique solutions observed so far in the search. Initially $h_{(a,b)}$ is set to zero, and each time a new unique top-B solution is observed, the weights are incremented and decremented according to the solutions entering and leaving the top-B solutions. A proper value of B was experimentally found to be $B = 100$.

The weights $h_{(a,b)}$ could be used in a similar way as in the historical node-pair removal heuristic described above, but initial experiments indicated that this was an unpromising approach. Instead, the graph is used to define the relatedness between two requests, such that two requests are considered to be related if the weight of the corresponding edge in the request graph is high. This relatedness measure is used as in the related removal heuristic described in Section 5.1.3.

5.2 Inserting requests

The considered insertion heuristics all construct a number of routes for the vehicles. As each route can be considered as an individual sub-problem the heuristics can build the routes *sequentially* or *in parallel* as discussed in Section 4.2. The first-mentioned heuristics build one route at a time while parallel heuristics construct several routes at the same time. The heuristics presented in this paper are all parallel, as they are used in a context where a number of partial routes $k \in R$ are given, and a number of unplaced requests U should be inserted from the request bank.

5.2.1 Basic greedy heuristic

A simple greedy approach is to repeatedly insert a request in the cheapest possible route. More formally, let $\Delta f_{i,k}$ denote the change in the objective value incurred by inserting request i at the *cheapest* position in route k. We set $\Delta f_{i,k} = \infty$ if request i cannot be inserted in route k. Following the greedy approach we calculate

$$(i, k) := \arg \min_{i \in U, k \in R} \Delta f_{i,k} \tag{23}$$

and insert request i in route k at its minimum cost position. This process continues until all requests have been inserted or no more requests are feasible. The time complexity of this `basic greedy` heuristic is decreased by tabulating all values of $\Delta f_{i,k}$ and noting that only one route is changed in each iteration.

5.2.2 Regret heuristics

An obvious problem with the `basic greedy` heuristic is that it often postpones the placement of difficult requests to the last iterations where we do not have much freedom of action. The `regret` heuristic tries to circumvent the problem by incorporating a kind of look-ahead information when selecting the request to insert. Regret heuristics have been used by Potvin and Rousseau [45] for the VRPTW and in the context of the Generalized Assignment Problem by Trick [58].

Let Δf_i^q denote the change in the objective value incurred by inserting request i into its best position in the qth *cheapest* route for request i. For example Δf_i^2 denotes the change in the objective value by inserting request i in the route where the request can be inserted *second cheapest*. In each iteration, the regret heuristic chooses to insert the request i that maximizes

$$i := \arg \max_{i \in U} \left(\Delta f_i^2 - \Delta f_i^1 \right) \tag{24}$$

The request is inserted in the best possible route, at the minimum cost position. In other words, we maximize the difference of cost of inserting the request i in its best route and its second best route. We repeat the process until no more requests can be inserted.

The heuristic can be extended in a natural way to define a class of regret heuristics: the `regret-`q heuristic is the construction heuristic that in each construction step chooses to insert the request i that maximizes

$$i := \arg\max_{i \in U} \left(\sum_{h=2}^{q} \Delta f_i^h - \Delta f_i^1 \right) \tag{25}$$

Ties are broken by selecting the request with smallest insertion cost. The request i is inserted at its minimum cost position, in its best route.

The `regret` heuristic based on criteria (24) is obviously a `regret`-2 heuristic and the `basic greedy` heuristic from Section 5.2.1 is a `regret`-1 heuristic due to the tie-breaking rules. Informally speaking, heuristics with $q > 2$ investigate the cost of inserting a request on the q best routes and chooses to insert the request whose cost difference between inserting it into the best route and the $q - 1$ best routes is largest. Compared to a `regret`-2 heuristic, `regret-`q heuristics with large values of q discover earlier when the possibilities for inserting a request at a favorable place becomes limited.

5.3 Master local search framework

At the master level we have chosen to use simulated annealing as our local search framework. Our acceptance criteria in Line 5 of the main algorithm depicted in Figure 2 thus becomes to accept a candidate solution x' given the current solution x with probability

$$e^{-\frac{f(x')-f(x)}{T}}, \tag{26}$$

where $T > 0$ is the *temperature*. We use a standard exponential cooling rate, starting from the temperature T_{start} and decreasing T according to the expression $T = T \cdot c$, where $0 < c < 1$ is the *cooling rate*. We calculate T_{start} by inspecting our initial solution. The following method was developed in [50] and works well when the number of requests in the problems to be solved is relatively constant. First the cost z' of the initial solution is calculated using a modified objective function. In the modified objective function, Γ (cost of having requests in the request bank) is set to zero. The start temperature is now set such that a solution that is w percent worse than the current solution is accepted with probability 0.5. The reason for setting Γ to zero is that this parameter typically is large and could cause us to set the starting temperature to a too large number if the initial solution had some requests in the request bank. Now w is a parameter that has to be set. We denote this parameter the *start temperature control parameter*. We have observed that this approach is able to cope with instances of different sizes better if we divide the start temperature found by the number n of requests in the instance.

5.4 Applying noise to the objective function

As mentioned in Section 4.1 it can be necessary to use noising or randomization in the destroy and repair heuristics, as a diversification operator at the master level is not sufficient.

For the RPDPTW problem we have chosen to add a noise term to the objective function of the insertion heuristics. Every time we calculate the cost C of a request insertion into a route, we add some noise δ and calculate a modified insertion cost $C' = max\{0, C + \delta\}$. The noise δ is chosen as a random number in the interval $[-N_{\max}, N_{\max}]$, where $N_{\max} = \eta \cdot \max_{i,j \in V}\{d_{ij}\}$, and η is a parameter that controls the amount of noise. We use the maximum distance to make the noise level proportional to the objective value. The distances form part of the objective function in all problems considered, hence the noise level is somehow proportional to the objective function.

Every insertion heuristic is split into two heuristics — one using noise, and one using the original objective function only. After selecting which removal and insertion heuristic to use, it is decided if the noise applied insertion heuristic should be used. This is again done using the roulette wheel selection principle as we keep track of how well the insertion heuristics with and without noise have been performing

recently. Notice that we do not keep track of how well each individual insertion heuristic is performing with and without noise, but only the insertion heuristics in general.

5.5 Adaptive weights adjustment

The roulette wheel selection mechanism in the ALNS framework presented in Section 4.1 is based on the scores π_i of the respective heuristics. A high score corresponds to a successful heuristic, and hence the heuristic should be chosen with larger probability.

The scores are collected during some small time segments, defined as 100 iterations. The *observed* score $\overline{\pi}_{i,j}$ of a heuristic i in time segment j is incremented with the following values depending on the new solution x':

- σ_1 The last remove-insert operation resulted in a new global best solution x'.
- σ_2 The last remove-insert operation resulted in a solution x' that has not been accepted before, and the cost of the new solution is better than the cost of current solution.
- σ_3 The last remove-insert operation resulted in a solution x' that has not been accepted before. The cost of the new solution is worse than the cost of current solution, but the solution was accepted.

We distinguish between the two latter situations since we prefer heuristics that are able to improve on the solution, but we also want to reward heuristics that can diversify the search to some extent. We keep track of visited solutions by assigning a hash key to each solution and storing the key in a hash table.

At the end of each segment we calculate the *smoothened* scores to be used in the roulette wheel selection as

$$\pi_{i,j+1} = \rho\frac{\overline{\pi}_{i,j}}{a_i} + (1-\rho)\pi_{i,j} \tag{27}$$

where a_i is the number of times the heuristic has been called in the time segment. The *reaction factor* ρ controls how quickly the weight adjustment algorithm reacts to changes in the scores. If $\rho = 1$ then the roulette wheel selection is only based on the scores in the most recent segment, while if $\rho < 1$ the scores of past segments is also taken into account. For an illustration that shows how the scores evolve during a search we refer the reader to [49].

5.6 Minimizing the number of vehicles used

The presented heuristic minimizes the travel costs, hence in order to minimize the number of vehicles also, we use a two-stage approach.

Starting from a heuristic solution which makes use of m vehicles, we repeatedly remove one route and place the corresponding requests in the request bank. If the ALNS heuristic is able to find a solution that serves all requests we proceed with a lower number of routes. We assign a large cost Γ to requests in the request bank to encourage solutions with all requests serviced.

If the ALNS heuristic fails to find a solution with all requests serviced, the algorithm steps back to the last feasible solution encountered and proceeds with the second stage of the algorithm which consists of the ordinary ALNS heuristic with the last found feasible solution as a starting point. For additional detail on the two-stage algorithm see [49].

A different two-stage approach was used by Bent and Van Hentenryck [2], in which two distinct neighborhoods and metaheuristics were used for the two stages.

6 Computational experiments

6.1 Parameter tuning

In order to keep the parameter tuning to a minimum we have used almost the same parameter setting as determined in [49], with the exception of the cooling rate c and the start temperature control parameter w. These were calibrated by selecting 5 reasonable values for each parameter and testing the 25 possible

combinations on 8 VRPTW instances with between 100 and 1000 customers. This was done separately for both the vehicle minimizing ALNS and the ordinary distance minimizing ALNS, so different values for c and w are used when trying to find a feasible solution and when minimizing the distance.

6.1.1 Selecting the number of requests to remove

In our past work [49, 50] we have removed up to 100 requests in each iteration. Experiments indicated that we seldom accepted the moves resulting from such removals as the insertion heuristics are too weak. Consequently the maximum number of requests that can be removed in a single iteration has been reduced to 60. It was also observed that moves resulting from removing a small number of requests often were accepted, but seldom lead to any major improvements of the solution. Therefore we now remove at least $0.1n$ requests in each iteration. To be precise, the number of requests to remove is found as a random number between $\min\{0.1n, 30\}$ and $\min\{0.4n, 60\}$. That is, for small instances the number of requests to remove will be in the interval $[0.1n, 0.4n]$ while for larger instances the interval is $[30, 60]$.

6.2 Analysis of typical search

In order to illustrate how the present ALNS heuristic works, we have produced a number of figures by running the heuristic on a 200 customer VRPTW instance and minimizing the traveled distance. All figures are from the same search.

Figure 3 shows the cost of the accepted solutions and the best known solution as a function of the iteration count. The figure is very typical for a Simulated Annealing metaheuristic. Initially very poor moves are accepted and consequently the graph of accepted solutions is fluctuating wildly. As the temperature is decreased the fluctuations become smaller and they eventually nearly die out such that only improving solutions or very mild deteriorating solutions are accepted.

The next sequence of figures all show the distance between selected solutions. We have chosen to define the distance between two solutions x and x' as the Hamming distance between the corresponding binary edge-variables. Figure 4 (left) shows the distance between each new accepted solution and the previously accepted solution (the current solution). The figure shows that the ALNS in the first half of the search can make huge changes to the solution in a single move as discussed in Section 4.3. In the last half of the search only smaller moves are accepted. Figure 4 (right) depicts the difference between each proposed solution and the last accepted solution. The figure shows that large moves are proposed throughout the search process, but toward the end of the search these large moves are not accepted.

Figure 4 suggests some possible improvements to the algorithm: (1) Towards the end of the search it seems to be beneficial to reduce the number of requests q that are removed in each iteration as the simulated annealing framework generally only will accept minor changes. This could speed up the algorithm or allow us to perform more iterations within the same amount of time. (2) Several moves have distance zero, meaning that no changes were made to the solution vector. Obviously, such moves should be avoided, possibly by incorporating a tabu-like principle in the insertion heuristics.

Figure 5 (top left) shows the Hamming distance between the accepted solutions and the previously best known solution. Every time the distance reaches zero we have most likely found a new best solution (or we have returned to the previously best known solution). It is interesting to see how quickly the search moves away from the currently best known solution. This behavior is contrary to some of the ideas behind the Variable Neighborhood metaheuristics and the Noising Method, where one tries to stick around the currently best known solution or return to it if the current search direction seems fruitless. Also notice that we move very far away from the best solutions. This can be seen as the number of edges in a solution is equal to $2n + m$. The maximum Hamming distance between two solutions is therefore $2(2n + m)$. In the instance studied in this section $n = 200$ and $m = 20$, thus the maximum hamming distance for this instance is 840.

Figure 5 (top right) shows the Hamming distance from each accepted solution and the best solution found throughout the search. It is interesting to see that this plot is much more steady compared to the plot in Figure 5 (top left) and that even though we are moving very far away from the previously best known solution, the distance to the overall best solution (which of course is unknown early in the search) remains roughly stable.

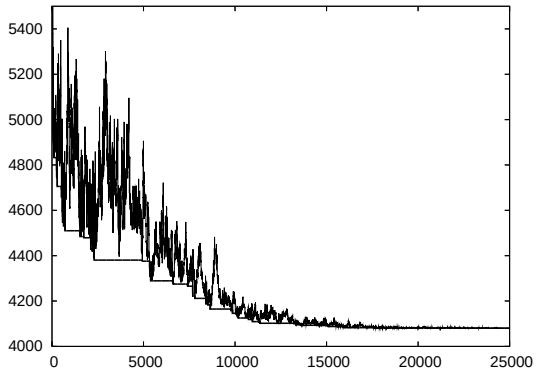

Figure 3: Solution cost as function of iteration count. Along the x-axis we show the iteration count while the y-axis shows solution cost. The upper graph shows the cost of the accepted solutions while the lower graph shows the cost of currently best known solution.

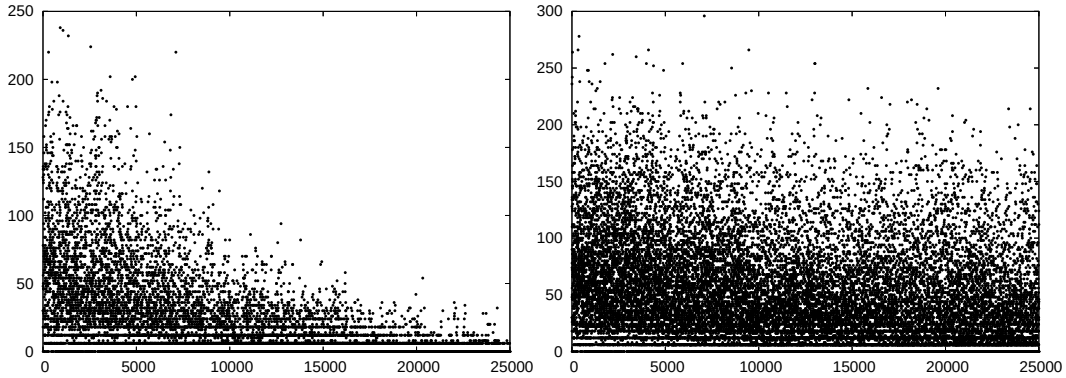

Figure 4: **Left:** Difference between accepted solutions. The figure shows the Hamming distance between an accepted solution and the last accepted solution. **Right:** Difference between proposed solution and last accepted solution. The figure shows the Hamming distance between each proposed solution and the last accepted solution. The x-axis shows iteration count and the y-axis shows solution distance.

Figure 5 (bottom) combines the two previous plots. The upper contours of the two plots fit each other surprisingly well. This indicates that the ALNS heuristic quickly moves away from the currently best known solution until the distance to the currently best known solution is roughly the same as the distance to the final best known solution. The search then visits solutions where the two distances are roughly the same until a new best solution is found. We believe that the Simulated Annealing framework is responsible for this behavior.

6.3 Application of the heuristic to standard benchmark problems

This section examines how the proposed heuristic performs on standard benchmark instances for the five problem types considered in this paper. In order to investigate how much influence the number of LNS iterations has on the solution quality, we have tested two configurations of our algorithm. One version (*ALNS-25K*) that does 25000 iterations while minimizing the total traveled distance and one that does 50000 iterations (*ALNS-50K*). Both configurations use up to 25000 iterations in the vehicle minimization stage. The cooling rate c in the simulated annealing algorithm described in Section 5.3 was adjusted such that both configurations go through the same temperature span.

We have applied the heuristic to each instance 5 or 10 times, depending on the instance size. We report

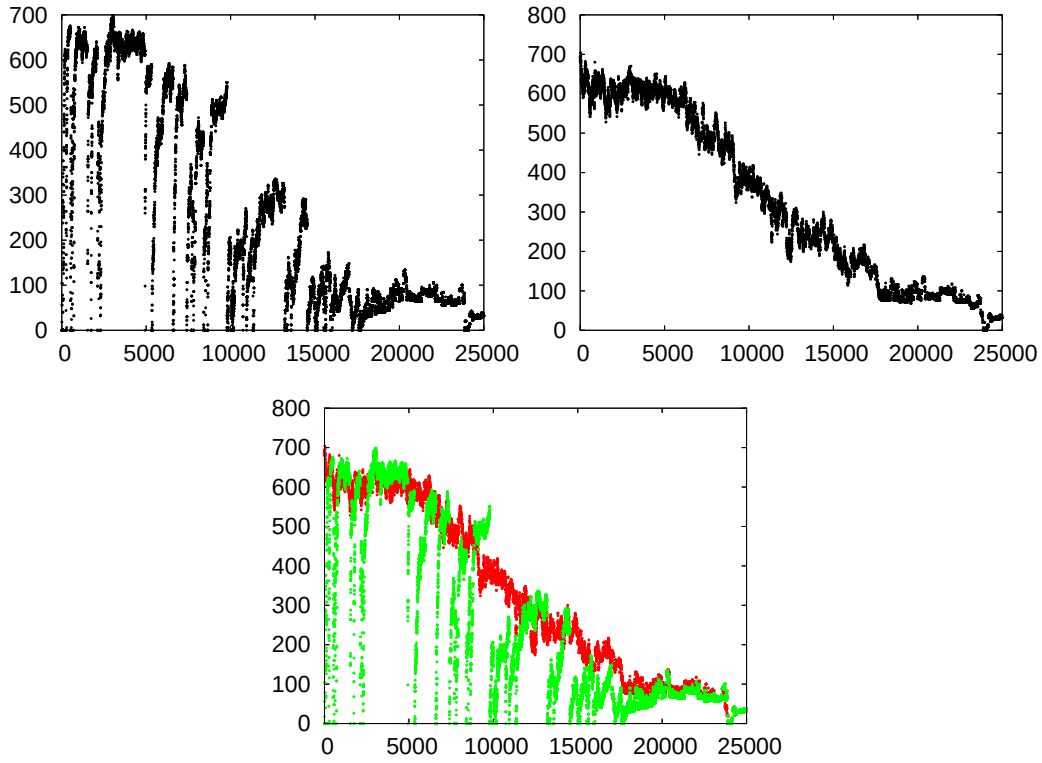

Figure 5: **Top left:** Hamming distance between accepted solutions and the currently best known solution. **Top right:** Hamming distance between accepted solutions and the best solution found during the search. **Bottom:** The two plots showed in the same diagram. The x-axis shows iteration count and the y-axis shows Hamming distance.

the best solution value out of the 5 or 10 experiments as well as the average solution value.

All experiments were performed on a 3GHz Pentium 4 computer. Detailed results from the experiments can be found in the appendix. As mentioned before, the same parameter configuration has been used for all experiments.

6.3.1 Vehicle routing problems with time windows (VRPTW)

An large number of metaheuristics have been proposed for solving the VRPTW. Bräysy and Gendreau [8] have surveyed most of these approaches, and their survey contains 47 metaheuristics. Most of these metaheuristics have been applied to the *Solomon data set* [55]. The Solomon data set contains 56 VRPTW instances that all contain 100 customers. The instances contain a variety of customer and time window distributions and have proved to be a challenge for both heuristics and exact methods since their introduction. Most of the proposed metaheuristics use vehicle minimization as primary objective and travel distance minimization as secondary objective, we prioritize our objectives in the same way. In this section we compare the ALNS heuristic to the "best" of the previously proposed metaheuristics. It is hard to decide which of the previously proposed metaheuristics are the best, as several criteria for comparing the heuristics could be used. In this paper we have selected the metaheuristics that have been able to reach the minimum number of total vehicles used for all of the instances in the Solomon data set, as these in a certain sense can be regarded as the best heuristics in terms of solution quality. Table 1 summarizes this comparison.

The table shows that the ALNS heuristic is able to compete with the best heuristics for the VRPTW when considering the moderate sized Solomon instances, even though it was not specifically designed for this problem type. The heuristics by Homberger and Gehring [32] and Bent and Van Hentenryck [2] obtain slightly better results compared to the best solutions obtained by ALNS-25K, but the papers do not state how many experiments that were performed to reach these results. On the other hand, ALNS-25K reaches slightly better solutions than the three remaining heuristics and the computational time is reasonable. The column showing the average performance of ALNS-25K indicates that one run of the heuristic can be performed quite fast but then one should not expect to reach the minimum number of vehicles. It does not seem worthwhile to spend 50000 iteration instead of 25000 for these rather small problems. During the calibration of the algorithm we discovered a new best solution to problem R207. This solution can be found in the Appendix.

When the VRPTW has been solved by exact methods in the literature one has usually considered minimizing the traveled distance without putting any limits on the number of vehicles. Furthermore all distances are usually truncated to one decimal (see for example the work by Larsen [39]). In Table 2 we summarize the result of applying the ALNS-25K heuristic to the Solomon VRPTW instances using the same objective and rounding criteria as the exact methods. The heuristic has been applied to each instance 10 times and the table reports the best and average performances. The table shows that the heuristic is able to find solutions that are very close to the optimal solutions and in many cases the heuristic is able to identify the optimal solution in at least one of the test runs.

The optimal solutions have been reported in Chabrier [10], Cook and Rich [14], Danna and Le Pape [20], Feilet et al. [24], Irnich and Villeneuve [34], Kallehauge et al. [35], Kohl et al. [36] and Larsen [39].

Larger VRPTW instances have been proposed by Gehring and Homberger [27]. The Gehring/Homberger data set contains 300 instances with between 200 and 1000 customers. In Tables 3–7 we compare the ALNS heuristic to the best heuristics that have been applied to these problems. The two heuristics that reach the best solution quality is the heuristic by Mester and Bräysy [42] and the ALNS heuristic. Overall the ALNS heuristic is better at minimizing the number of vehicles which is the primary objective of these problems. The heuristic of Mester and Bräysy is very good at minimizing the traveled distance though. The experiments show that the time used by the ALNS heuristic scales quite well with the problem size. The 50000 iteration ALNS configuration becomes worthwhile for the larger problems. For problems with 600 customers or more the difference in total traveled distance obtained by the ALNS-25K and ALNS-50K configurations become quite large, as the simulated annealing metaheuristic needs more iterations to obtain a good solution for large problems.

The ALNS heuristic has been able to improve the best known solution for 122 out of the 300 large scale VRPTW instances. The best solutions for the large VRPTW instances obtained by the ALNS-25K and ALNS-50K configurations are shown in Table 8.

	BBB	HG	B	BH	IIKMUY	ALNS 25K		ALNS 50K	
R1	11.92	11.92	11.92	11.92	11.92	11.92	12.03	11.92	12.03
	1221.10	1212.73	1222.12	**1211.10**	1217.40	1213.39	1216.93	1212.39	1215.16
R2	2.73	2.73	2.73	2.73	2.73	2.73	2.75	2.73	2.75
	975.43	955.03	975.12	**954.27**	959.11	958.60	968.01	957.72	965.94
C1	10.00	10.00	10.00	10.00	10.00	10.00	10.00	10.00	10.00
	828.48	**828.38**	**828.38**	**828.38**	**828.38**	**828.38**	**828.38**	**828.38**	**828.38**
C2	3.00	3.00	3.00	3.00	3.00	3.00	3.00	3.00	3.00
	589.93	**589.86**	**589.86**	**589.86**	**589.86**	**589.86**	**589.86**	**589.86**	**589.86**
RC1	11.50	11.50	11.50	11.50	11.50	11.50	11.60	11.50	11.60
	1389.89	1386.44	1389.58	**1384.17**	1391.03	1385.39	1386.91	1385.78	1385.56
RC2	3.25	3.25	3.25	3.25	3.25	3.25	3.25	3.25	3.25
	1159.37	**1108.52**	1128.38	1124.46	1122.79	1124.77	1140.06	1123.49	1135.46
CNV	405	405	405	405	405	405	407.5	405	407.5
CTD	57952	**57192**	57710	57273	57444	57360	57641	57332	57550
CPU	P-400 Mhz	P-400 Mhz	P-200Mhz	SU 10	P3 1Ghz	P4 3Ghz	P4 3Ghz	P4 3Ghz	P4 3Ghz
T. (s)	1800	N/A	4950	7200	15000	86	86	146	146
Exp.	3	N/A	1	> 5	1	10	1	10	1

Table 1: Solomon instances with 100 customers. The table compares the ALNS heuristic to the heuristics by Berger et al. (BBB) [4], Homberger and Gehring (HG) [32], Bräysy (B) [7], Bent and Van Hentenryck (BH) [2] and Ibaraki et al. (IIKMUY) [33]. The data set is divided into six groups: R1, R2, C1, C2, RC1, RC2. For each group we report two numbers per heuristic. The top number is the number of vehicles used and the bottom number is the distance traveled. These numbers have been averaged over all the instance in the given group. The rows named *CNV* and *CTD* show the cumulative number of vehicles and distances respectively. The row *CPU* shows the computer used in the experiment, the row *T. (s)* shows the number of CPU seconds used for finding the solutions. The last row shows the number of experiments that were performed in order to obtain the results presented in the table (if multiple experiments were performed, the table shows the best results obtained). The two columns for the ALNS heuristic show the results obtained with the 25000 iteration configuration and the 50000 iteration configuration. For each configuration we show two columns. The first column shows the best result out of ten experiments, and the second column show the average solution quality (averaged over the ten experiments). Bold entries mark the best solution quality obtained among the heuristics in the comparison.

exact methods			ALNS			
Customers	Instances	Solved to optimality	Optimums found	Avg. gap all (%)	Avg. gap opt.(%)	Avg. time (s)
25	56	56	56	0.02	0.02	5
50	56	53	48	0.19	0.13	15
100	56	37	27	0.36	0.26	47

Table 2: Comparison of ALNS to exact methods. The columns should be interpreted as follows: *Customers* — the number of customers in the test set, *Instances* — the number of instances in the test set, *Solved to optimality* — the number of instances that has been solved to optimality in the literature, *Optimums found* — the number of optimal solutions that were found by the heuristic, *Avg. gap all (%)* — the average gap over all instances, *Avg. gap opt. (%)* — the average gap over instances solved to optimality in the literature, *Avg. time (s)* — the average time in seconds spent on performing one experiment.

	GH99	GH01	BH	LL	LC	BHD	MB	ALNS 25K		ALNS 50K	
R1	18.2	18.2	18.2	18.3	18.2	18.2	**18.2**	18.2	18.20	18.2	18.20
	3705.00	3855.03	3677.96	3736.20	3676.95	3718.30	**3618.68**	3635.94	3664.648	3631.226	3652.747
R2	4.0	4.0	4.1	4.1	4.0	4.0	**4.0**	4.0	4.05	4.0	4.05
	3055.00	3032.49	3023.62	3023.00	2986.01	3014.28	**2942.92**	2950.30	2950.04	2949.368	2942.594
C1	18.9	18.9	18.9	19.1	18.9	18.9	**18.8**	18.9	18.90	18.9	18.90
	2782.00	2842.08	2726.63	2728.60	2743.66	2749.83	**2717.21**	2723.10	2732.458	2721.522	2728.382
C2	6.0	6.0	6.0	6.0	6.0	6.0	6.0	6.0	6.00	**6.0**	6.00
	1846.00	1856.99	1860.17	1854.90	1836.10	1842.65	1833.57	1833.33	1836.4	**1832.947**	1834.675
RC1	18.0	18.1	18.0	18.3	18.0	18.0	18.0	18.0	18.00	**18.0**	18.00
	3555.00	3674.91	3279.99	3385.80	3449.71	3329.62	3221.34	3233.76	3282.989	**3212.282**	3257.168
RC2	4.3	4.4	4.5	4.9	4.3	4.4	4.4	4.3	4.33	**4.3**	4.33
	2675.00	2671.34	2603.08	2518.70	2613.75	2585.89	2519.79	2560.59	2592.39	**2556.874**	2578.575
CNV	694	696	697	707	694	695	**694**	694	694.8	694	694.8
CTD	176180	179328	171715	172472	173061	172406	**168573**	169370	170589	169042	169941
CPU	P-200Mhz	P-400Mhz	SU 10	P-545Mhz	P-933Mhz	A-700Mhz	P4 2Ghz	P4-3Ghz	P4-3Ghz	P4-3Ghz	P4-3Ghz
T. (min)	4x10	4x2.1	n/a	182.1	5x10	2.4	8	4.3	4.3	7.7	7.7
Exp.	1	3	n/a	3	1	3	1	10	1	10	1

Table 3: Gehring/Homberger VRPTW instances with 200 customers. The table compares the ALNS heuristic to the heuristics by Gehring and Homberger (GH99) [27] and (GH01) [28], Bent and Van Hentenryck (BH) [2], Le Bouthillier and Cranic (LC) [5], Bräysy et al (BHD) [9] and Mester and Bräysy (MB) [42]. The table should be interpreted like Table 1. Notice that computing times are reported in minutes. Entries of the form $x \times y$ appearing in the *T. (min)* row indicate that the experiment was run for y minutes on a parallel computer with x processors.

	GH99	GH01	BH	LL	LC	BHD	MB	ALNS 25K		ALNS 50K	
R1	36.4	36.4	36.4	36.6	36.5	36.4	**36.3**	36.4	36.40	36.4	36.40
	8925.00	9478.22	8713.37	8912.40	8839.28	8692.17	**8530.03**	8609.38	8663.57	8540.04	8589.90
R2	8.0	8.0	8.0	8.0	8.0	8.0	**8.0**	8.0	8.00	8.0	8.00
	6502.00	6650.28	6959.75	6610.60	6437.68	6382.63	**6209.94**	6252.01	6309.84	6241.72	6277.07
C1	38.0	38.0	38.0	38.7	37.9	37.9	37.9	37.6	37.62	**37.6**	37.62
	7584.00	7855.82	7220.96	7181.40	7447.09	7230.48	7148.27	7369.88	7450.84	**7290.16**	7372.49
C2	12.0	12.0	12.0	12.1	12.0	12.0	**12.0**	12.0	12.00	12.0	12.00
	3935.00	3940.19	4154.40	4017.10	3940.87	3894.48	**3840.85**	3849.27	3884.44	3844.69	3875.95
RC1	36.1	36.1	36.1	36.5	36.0	36.0	**36.0**	36.0	36.00	36.0	36.00
	8763.00	9294.99	8330.98	8377.90	8652.01	8305.55	**8066.44**	8149.61	8240.28	8069.30	8148.81
RC2	8.6	8.8	8.9	9.5	8.6	8.9	8.8	8.5	8.64	**8.5**	8.64
	5518.00	5629.43	5631.70	5466.20	5511.22	5407.87	5243.06	5366.82	5388.76	**5335.09**	5351.56
CNV	1390	1392	1393	1414	1390	1391	1389	1385	1386.6	**1385**	1386.6
CTD	412270	428489	410112	405656	408281	399132	390386	395970	399377	**393210**	396158
CPU	P-200Mhz	P-400Mhz	SU 10	P-545Mhz	P-933Mhz	A-700Mhz	P4 2Ghz	P4-3Ghz	P4-3Ghz	P4-3Ghz	P4-3Ghz
T. (min)	4x20	4x7.1	n/a	359	5x20	7.9	17	9.7	9.7	15.8	15.8
Exp.	1	3	n/a	3	1	3	1	5	1	5	1

Table 4: Gehring/Homberger VRPTW instances with 400 customers

	GH99	GH01	BH	LL	LC	BHD	MB	ALNS 25K		ALNS 50K	
R1	54.5	54.5	55.0	55.2	54.8	54.5	**54.5**	54.5	54.50	54.5	54.50
	20854.00	21864.47	19308.62	19744.80	19869.82	19081.18	**18358.68**	19370.04	19562.34	18888.52	19048.49
R2	11.0	11.0	11.0	11.1	11.2	11.0	11.0	11.0	11.00	**11.0**	11.00
	13335.00	13656.15	14855.43	13592.40	13093.97	13054.83	12703.52	12729.51	12826.39	**12619.26**	12721.32
C1	57.9	57.7	57.8	58.2	57.9	57.8	57.8	57.5	57.56	**57.5**	57.56
	14792.00	14817.25	14357.11	14267.30	14205.58	14165.90	14003.09	14125.94	14212.71	**14065.89**	14098.04
C2	17.9	17.8	17.8	18.2	17.9	18.0	17.8	17.5	17.80	**17.5**	17.80
	7787.00	7889.96	8259.04	8202.60	7743.92	7528.73	7455.83	7891.70	7834.72	**7801.296**	7682.61
RC1	55.1	55.0	55.1	55.5	55.2	55.0	**55.0**	55.0	55.00	55.0	55.00
	18411.00	19114.02	17035.91	17320.00	17678.13	16994.22	**16418.63**	16846.71	17006.94	16594.94	16722.51
RC2	11.8	11.9	12.4	13.0	11.8	12.1	12.1	11.6	11.78	**11.6**	11.78
	11522.00	11670.29	11987.89	11204.90	11034.71	11212.36	10677.46	10922.44	10938.30	**10777.12**	10828.45
CNV	2082	2079	2091	2112	2088	2084	2082	2071	2076.4	**2071**	2076.4
CTD	867010	890121	858040	843320	836261	820372	796172	818863	823814	**807470**	811014
CPU	P-200Mhz	P-400Mhz	SU 10	P-545Mhz	P-933Mhz	A-700Mhz	P4 2Ghz	P4-3Ghz	P4-3Ghz	P4-3Ghz	P4-3Ghz
T. (min)	4x30	4x12.9	n/a	399.8	5x30	16.2	40	10.5	10.5	18.3	18.3
Exp.	1	3	n/a	3	1	3	1	5	1	5	1

Table 5: Gehring/Homberger VRPTW instances with 600 customers.

	GH99	GH01	BH	LL	LC	BHD	MB	ALNS 25K		ALNS 50K	
R1	72.8	72.8	**72.7**	73.0	73.1	72.8	72.8	72.8	72.80	72.8	72.80
	34586.00	34653.88	**33337.91**	33806.34	33552.40	32748.06	31918.47	32697.85	32905.52	32316.79	32528.76
R2	15.0	15.0	15.0	15.1	15.0	15.0	**15.0**	15.0	15.00	15.0	15.00
	21697.00	21672.85	24554.63	21709.39	21157.56	21170.15	**20295.28**	20477.77	20627.40	20353.51	20499.72
C1	76.7	76.1	76.1	77.4	76.3	76.3	76.2	75.6	75.66	**75.6**	75.66
	26528.00	26936.68	25391.67	25337.02	25668.82	25170.88	25132.27	25365.59	25547.82	**25193.13**	25269.64
C2	24.0	23.7	24.4	24.4	24.1	24.2	**23.7**	23.7	23.98	23.7	23.94
	12451.00	11847.92	14253.83	11956.60	11985.11	11648.92	**11352.29**	11985.80	11999.28	11725.46	11741.73
RC1	72.4	72.3	73.0	73.2	**72.3**	73.0	73.0	73.0	73.00	73.0	73.00
	38509.00	40532.35	30500.15	31282.54	**37722.62**	30005.95	30731.07	29864.06	30016.05	29478.3	29625.04
RC2	16.1	16.1	16.6	17.1	15.8	16.3	15.8	15.7	15.82	**15.7**	15.82
	17741.00	17941.23	18940.84	17561.22	17441.60	17686.65	16729.18	16870.87	17022.33	**16761.95**	16852.95
CNV	2770	2760	2778	2802	2766	2776	2765	2758	2762.6	**2758**	2762.2
CTD	1515120	1535849	1469790	1416531	1475281	1384306	1361586	1372619	1381184	**1358291**	1365178
CPU	P-200Mhz	P-400Mhz	SU 10	P-545Mhz	P-933Mhz	A-700Mhz	P4-2Ghz	P4-3Ghz	P4-3Ghz	P4-3Ghz	P4-3Ghz
T. (min)	4x40	4x23.2	n/a	512.9	5x40	26.2	145	13.5	13.5	22.7	22.7
Exp.	1	3	n/a	3	1	3	1	5	1	5	1

Table 6: Gehring/Homberger VRPTW instances with 800 customers

	GH99	GH01	BH	LL	LC	BHD	MB	ALNS 25K		ALNS 50K	
R1	**91.9**	91.9	92.8	92.7	92.2	92.1	92.1	92.2	92.30	92.2	92.30
	57186.00	58069.61	51193.47	50990.80	55176.95	50025.64	49281.48	52131.96	51900.53	50751.25	50584.55
R2	19.0	19.0	19.0	19.0	19.2	19.0	19.0	19.0	19.00	**19.0**	19.00
	31930.00	31873.62	36736.97	31990.90	30919.77	31458.23	29860.32	30108.84	30327.34	**29780.82**	30016.08
C1	96.0	95.4	95.1	96.3	95.3	95.8	95.1	94.6	94.72	**94.6**	94.72
	43273.00	43392.59	42505.35	42428.50	43283.92	42086.77	41569.67	42123.87	42266.42	**41877.00**	42034.65
C2	30.2	29.7	30.3	30.8	29.9	30.6	**29.7**	29.7	29.90	29.7	29.86
	17570.00	17574.72	18546.13	17294.90	17443.50	17035.88	**16639.54**	17307.16	17589.70	16840.37	17052.62
RC1	90.0	90.1	90.2	90.4	90.0	90.0	**90.0**	90.0	90.00	90.0	90.00
	50668.00	50950.14	48634.15	48892.40	49711.36	46736.92	**45396.41**	47735.43	48168.74	46752.15	47081.64
RC2	19.0	18.5	19.4	19.8	18.5	19.0	18.7	18.3	18.46	**18.3**	18.46
	27012.00	27175.98	29079.78	26042.30	26001.11	25994.12	25063.51	25267.93	25466.13	**25090.88**	25185.45
CNV	3461	3446	3468	3490	3451	3465	3446	3438	3443.8	**3438**	3443.4
CTD	2276390	2290367	2266959	2176398	2225366	2133376	2078110	2146752	2157189	**2110925**	2119550
CPU	P-200Mhz	P-400Mhz	SU 10	P-545Mhz	P-933Mhz	A-700Mhz	P4 2Ghz	P4-3Ghz	P4-3Ghz	P4-3Ghz	P4-3Ghz
T. (min)	4x50	4x30.1	n/a	606.3	5x50	39.6	600	16	16	26.6	26.6
Exp.	1	3	n/a	3	1	3	1	5	1	5	1

Table 7: Gehring/Homberger VRPTW instances with 1000 customers.

	R1		R2		C1		C2		RC1		RC2	
#	Veh.	Dist.	Veh.	Dist.	Veh.	Dist.	Veh.	Dist.	Veh.	Dist.	Veh.	Dist.
200 customers												
1	20	4785.96	4	4563.55	**20**	**2704.57**	**6**	**1931.44**	**18**	**3647.56**	6	3126.03
2	18	4059.57	4	3650.54	18	2943.83	**6**	**1863.16**	**18**	**3269.91**	5	2828.39
3	18	3387.64	**4**	**2892.07**	18	2710.21	6	1776.96	18	3034.45	**4**	**2613.12**
4	18	3086.11	4	1981.30	18	2644.92	**6**	**1713.46**	**18**	**2869.74**	**4**	**2052.74**
5	18	4125.19	4	3377.18	**20**	**2702.05**	**6**	**1878.85**	18	3430.03	**4**	**2912.13**
6	**18**	**3586.80**	4	2929.72	**20**	**2701.04**	**6**	**1857.35**	**18**	**3357.90**	**4**	**2975.13**
7	18	3160.44	4	2456.71	**20**	**2701.04**	**6**	**1849.46**	**18**	**3233.29**	**4**	**2539.85**
8	18	2971.66	**4**	**1849.87**	19	2775.48	**6**	**1820.53**	**18**	**3110.46**	**4**	**2314.61**
9	18	3802.55	4	3113.74	18	2687.83	**6**	**1830.05**	18	3114.02	4	2175.98
10	18	3312.44	4	2666.10	**18**	**2644.25**	6	1808.21	**18**	**3020.24**	4	2015.61
400 customers												
1	40	10432.30	8	9338.49	40	7152.06	12	4116.33	**36**	**8813.43**	**11**	**6834.02**
2	**36**	**9115.68**	**8**	**7649.87**	**36**	**7733.55**	**12**	**3930.05**	**36**	**8118.43**	**9**	**6355.59**
3	36	7988.22	8	5998.04	**36**	**7082.13**	12	3775.32	**36**	**7663.73**	**8**	**5055.02**
4	36	7415.81	**8**	**4326.48**	**36**	**6816.17**	12	3543.60	**36**	**7368.47**	**8**	**3647.39**
5	**36**	**9479.10**	8	7252.64	40	7152.06	12	3946.14	**36**	**8426.57**	9	6119.44
6	36	8556.38	8	6212.37	40	7153.45	**12**	**3875.94**	36	8390.24	**8**	**5997.24**
7	36	7725.97	8	5136.74	**39**	**7546.78**	12	3894.98	36	8223.65	**8**	**5476.57**
8	**36**	**7390.76**	**8**	**4055.22**	**37**	**7546.32**	12	3796.00	36	7922.67	8	4877.39
9	36	8970.98	8	6507.40	36	7573.18	12	3881.21	36	7953.20	**8**	**4601.30**
10	36	8325.16	**8**	**5894.40**	36	7145.92	12	3687.13	36	7774.83	8	4355.52
600 customers												
1	59	21677.41	11	18837.28	**60**	**14095.64**	18	7780.84	55	17751.33	**15**	**13163.03**
2	54	20045.49	**11**	**15069.24**	**56**	**14174.12**	**17**	**8799.38**	55	16548.43	**12**	**11853.72**
3	54	17733.91	**11**	**11291.52**	**56**	**13803.50**	**17**	**7604.00**	**55**	**15499.02**	**11**	**9863.35**
4	54	16374.29	**11**	**8163.24**	**56**	**13578.66**	**17**	**6993.77**	**55**	**15072.90**	**11**	**7231.64**
5	54	21243.24	**11**	**15418.00**	60	14085.72	18	7578.12	**55**	**17401.34**	12	12560.43
6	54	18948.53	**11**	**12936.28**	**60**	**14089.66**	18	7554.61	55	17355.10	**11**	**12282.52**
7	54	17438.28	**11**	**10269.96**	**58**	**15017.03**	**18**	**7520.34**	55	17058.40	**11**	**11052.49**
8	54	16146.17	**11**	**7752.78**	**57**	**14343.05**	**17**	**8696.15**	55	16510.65	11	10488.75
9	54	20375.70	11	13885.52	56	13767.45	18	7356.19	55	16435.71	**11**	**9882.71**
10	54	18902.19	**11**	**12568.79**	**56**	**13688.57**	17	7938.94	55	16316.51	11	9340.06
800 customers												
1	80	37492.04	15	28822.48	80	25184.38	24	11664.00	**73**	**31275.38**	**19**	**20954.95**
2	72	33816.69	**15**	**23274.22**	**74**	**25536.76**	24	11428.07	73	29172.08	**17**	**18032.89**
3	72	30317.49	15	18078.82	**72**	**24629.86**	24	11184.67	73	28164.66	**15**	**14800.78**
4	72	28568.78	**15**	**13413.79**	**72**	**23938.33**	23	10999.42	73	27201.39	**15**	**11368.19**
5	72	35503.63	15	25077.09	**80**	**25166.28**	24	11451.57	73	30548.23	16	19180.13
6	72	32360.07	15	20969.81	**80**	**25160.85**	24	11403.57	73	30511.07	15	19075.89
7	72	29979.63	**15**	**16977.49**	79	25425.92	24	11412.08	73	30007.82	**15**	**17329.32**
8	**72**	**28341.21**	**15**	**12945.52**	**75**	**25450.99**	**23**	**13878.40**	73	29547.96	**15**	**16226.78**
9	**72**	**34218.41**	15	22877.21	**72**	**25737.46**	24	11650.10	73	29360.93	**15**	**15687.20**
10	72	32569.97	**15**	**21092.27**	**72**	**25697.68**	23	12103.56	73	28993.52	15	14944.14
1000 customers												
1	100	54720.19	19	43264.68	**100**	**42478.95**	**30**	**16879.24**	90	48933.68	21	30396.13
2	**91**	**55428.79**	**19**	**34417.47**	**91**	**42249.60**	29	17563.06	90	46165.33	**18**	**27552.05**
3	91	49634.84	**19**	**25400.16**	**90**	**40376.43**	30	16109.71	90	44014.81	18	20811.18
4	91	45303.47	**19**	**18332.77**	**90**	**39980.07**	**29**	**16011.30**	90	42607.34	**18**	**16007.59**
5	92	53089.15	19	37746.01	**100**	**42469.18**	30	16596.69	90	48934.53	**18**	**28368.48**
6	91	54555.32	19	30778.85	**100**	**42471.29**	**30**	**16369.10**	90	48766.98	18	28746.61
7	91	48141.47	**19**	**23991.71**	**99**	**42673.51**	31	16590.48	90	48005.94	18	26765.43
8	91	44853.70	**19**	**17844.36**	**95**	**42359.27**	29	18407.27	90	47122.61	**18**	**24961.29**
9	92	52015.72	19	34349.70	**91**	**41482.00**	**30**	**16294.72**	90	46889.79	18	24113.72
10	92	49769.85	19	31682.52	90	42214.60	29	17582.15	90	46080.51	18	23056.75

Table 8: The table shows the best solutions to large VRPTW instances identified by the ALNS heuristic. The first column shows the problem number. The columns *veh.* and *dist.* show the number of vehicles and total distance traveled in the best solution found. The table is grouped by instance type and instance size. Bold entries indicate a best solution (either a tie with one of the heuristics from the literature or a new best solution).

6.3.2 Multi depot vehicle routing problem (MDVRP)

Table 9 shows the results obtained on 33 MDVRP instances from the literature. Both ALNS configurations have been applied 10 times to each instance. The results obtained by the ALNS heuristic are compared to the best results obtained by heuristics proposed by Chao et al. [12], Renaud et al. [48] and Cordeau et al. [17]. The heuristic that previously has achieved the best solution quality is the one proposed by Cordeau et al. The cost of a solution is defined as the total distance traveled by the vehicles. The table shows that the ALNS heuristic has been able to improve upon the best solution for a considerable number of instances. Each configuration has found 14 new best solutions, but as most of these overlap, the total number of new best solutions is 15. The individual improvements are typically rather small though. The table also shows that the ALNS heuristic is quite stable as the average gap from the best known solution never surpasses 2% and 1% in the ALNS-25K and ALNS-50K configurations, respectively. It should be mentioned that the ALNS heuristic is slower than the previously proposed heuristics. The ALNS-25K and ALNS-50K configurations use on average two and four minutes respectively to perform one experiment on a 3GHz Pentium 4. The heuristic by Cordeau et al. on average used 11.7 minutes to perform one experiment on a Sun Sparcstation 10 which is considerably slower than our computer.

6.3.3 Site dependent vehicle routing problem (SDVRP)

The results obtained on the SDVRP instances are summarized in Table 10. The results are promising as the average solution quality of ALNS-25K overall is better than results previously published. Also the sum of the costs of the best known solutions found by the ALNS-50K configuration is more than 2% better than the previous best known solution and the best known solution was improved for 30 out of the 35 instances. The computational time for performing one experiment with the ALNS-25K configuration seems to be roughly comparable with the time needed for performing one experiment with the heuristic proposed by Cordeau and Laporte [18]. The ALNS-25K configuration spends on average 1.4 minutes to perform one experiment while the heuristic by Cordeau and Laporte spent around 12 minutes to perform the same task on a Sun Ultra 2, 300 MHz. It should be mentioned that the problem PR02 caused the ALNS heuristic some difficulties, as it was only able to find a feasible solution in one out of ten experiments for the ALNS-25K configuration and three out of ten experiments for the ALNS-50K configuration.

6.3.4 Capacitated vehicle routing problem (CVRP)

For the CVRP we have chosen to test the ALNS heuristic on three datasets. The first dataset was proposed by Christofides et al. [13] and contains instances with between 50 and 200 customers, the second dataset was proposed by Golden et al. [30] and contains instances with up to 483 customers. The last dataset was proposed by Li et al. [40] and contains instances with up to 1200 customers. These are the so-far largest instances that the ALNS heuristic has been applied to. Table 11 summarizes these experiments. Notice that we only compare the ALNS heuristic to a subset of all the CVRP heuristics that have been proposed in the literature. The heuristics used for benchmarking are the most recent heuristics that were surveyed by Cordeau et al. [16].

The table shows that the ALNS heuristic cannot compete with the well-performing heuristic by Mester and Bräysy [42], but its performance is comparable to the rest of the heuristics. For the last dataset, the heuristic proposed by Li et al. must be considered to be the best as it is very fast compared to the ALNS heuristic although the ALNS heuristic overall is able to reach better solutions. We discovered one new best solution for the Golden et al. dataset and three new best solutions for the Li et al. dataset.

6.3.5 Open vehicle routing problem (OVRP)

The results on the OVRP are summarized in Table 12. The heuristic was tested on the same 16 instances that were used by Brandão [6] and Fu et al. [25]. The primary objective considered was to minimize the number of vehicles used, while the secondary objective was to minimize the traveled distance. The solutions obtained by the ALNS heuristic are promising as the best known solution to 11 out of the 16 instances has been improved. The running time of the ALNS heuristic is quite respectable compared to the two other heuristics. The configuration of Brandão's heuristic that obtains the best results spends on average

9.6 minutes to solve an instance on a 500MHz Pentium III. In the paper by Fu et al. two configurations of their heuristic are tested. These configurations spend on average 6.6 and 13.9 minutes respectively to solve an instance on a 600 MHz Pentium II. The ALNS-25K and ALNS-50K configurations use 1.4 and 2.3 minutes respectively to solve an instance on a 3GHz Pentium IV.

6.3.6 Computational results conclusion

The computational results presented in this section are very encouraging. The results show that the general ALNS heuristic is on par with the best specialized heuristics for the VRPTW and that the heuristic currently is the best when it comes to minimizing the number of vehicles in large VRPTW instances. One should keep in mind that numerous specialized heuristics have been proposed for the VRPTW making it difficult for a general heuristic to compete on these instances.

For the MDVRP, SDVRP and OVRP the ALNS heuristic has been able to find many new best solutions and the results on the SDVRP are especially promising. For the CVRP the proposed heuristic is able to compete with many of the most recent heuristics, but it is outperformed by a more specialized heuristic for this problem. Nevertheless, a couple of new best solutions were found for this problem type also. One should also keep in mind that the heuristic was not tuned for each problem type, but a general parameter setting was used for all experiments.

The comparison between the fast and the slow version of the ALNS heuristic showed that it did not pay off to use the ALNS-50K variant for the smaller instances, while for instances with around 400 to 600 or more customers it seemed worthwhile to use the ALNS-50K configuration. Consequently, it might be useful to use a variable number of iterations I which depends on the number n of requests as e.g. $I := 20000 + 50n$.

7 Conclusion

A new, general heuristic framework, denoted Adaptive Large Neighborhood Search has been presented. The framework has been used to several variants of vehicle routing problems in the present paper as well as in [49, 50]. This includes the vehicle routing problem with time windows (VRPTW), the capacitated vehicle routing problem (CVRP), the multi-depot vehicle routing problem (MDVRP), the site dependent vehicle routing problem (SDVRP), the open vehicle routing problem (OVRP), the pickup and delivery problem with time windows (PDPTW), the vehicle routing problem with backhauls (VRPB), the mixed vehicle routing problem with backhauls (MVRPB), the multi-depot mixed vehicle routing problem with backhauls (MDMVRPB), the vehicle routing problem with backhauls and time windows (VRPBTW), the mixed vehicle routing problem with backhauls and time windows (MVRPBTW) and the vehicle routing problem with simultaneous deliveries and pickups (VRPSDP).

Due to the generality of the ALNS framework, and the encouraging results demonstrated for a wide spectrum of VRP problems, we believe that ALNS should be considered as one of the standard frameworks for solving large-sized optimization problems.

Supply chain management is a research area getting increasing attention [37]. By co-ordinating activities in the supply chain, companies can rationalize the process resulting in mutual gains. If the involved companies co-ordinate their transportation activities we will see a need for solving mixed transportation problems, where the instances for example consist of a mixture of PDPTW, MDRP and SDVRP problems. In order to handle future changes in the distribution structure, these algorithms need to be stable for various input types, and should not need to be tuned for particular problem characteristics. It should be clear that the ALNS framework may play an important role.

In conclusion we may add a philosophical observation: We have seen that a mixture of good and less good heuristics lead to better solutions than using good heuristics solely. It is however necessary to hierarchically controls the search, such that well-performing heuristics are given most influence, but such that all heuristics participate to the solution process. Using this principle one gets a robust and well-performing solution approach.

8 Appendix

New best solution to Solomon R207 instance

Route	Length	Visit sequence
1	437.339	42 92 45 46 36 64 11 62 88 30 20 65 71 9 81 34 78 79 3 76 28 53 40 2 87 57 41 22 73 21 72 74 75 56 4 25 55 54 80 68 77 12 26 58 13 97 37 100 98 93 59 95 94
2	453.269	27 1 69 50 33 29 24 39 67 23 15 43 14 44 38 86 16 61 91 85 99 96 6 84 8 82 7 48 47 49 19 10 63 90 32 66 35 51 70 31 52 18 83 17 5 60 89

Total length 890.61

Full VRPTW tables

The full tables documenting the VRPTW experiments described in section 6.3.1 can be found in Tables 13 – 18.

Tables 19 – 21 contain detailed result from the experiment comparing the ALNS heuristic to exact methods.

Full CVRP tables

The detailed results for the CVRP experiments described in section 6.3.4 can be found in Tables 22 – 24.

References

[1] Ravindra K. Ahuja, Özlem Ergun, James b. Orlin, and Abraham P. Punnen. A survey of very large-scale neighborhood search techniques. *Discrete Applied Mathematics*, 123:72–102, 2002.

[2] Russel Bent and Pascal Van Hentenryck. A two-stage hybrid local search for the vehicle routing problem with time windows, 2004. To appear in Transportation Science.

[3] J. Berger and M. Barkaoui. A new hybrid genetic algorithm for the capacitated vehicle routing problem. *Journal of the Operational Research Society*, 54:1254–1262, 2003.

[4] Jean Berger, Mohamed Barkaoui, and Olli Bräysy. A route hybrid genetic approach for the vehicle routing problem with time windows. *INFOR*, 41(2), 2003.

[5] Alexandre Le Bouthillier and Teodor Gabriel Crainic. A cooperative parallel meta-heuristic for the vehicle routing problem with time windows. *Computers and Operations Research*, 2004. to appear.

[6] José Brandão. A tabu search algorithm for the open vehicle routing problem. *European Journal of Operational Research*, 157(3):552–564, 2004.

[7] Olli Bräysy. A reactive variable neighborhood search for the vehicle-routing problem with time windows. *INFORMS Journal on Computing*, 15(4):347–368, 2003.

[8] Olli Bräysy and Michel Gendreau. Vehicle routing problem with time windows, part ii: Metaheuristics. *Transportation Science*. To appear.

[9] Olli Bräysy, Geir Hasle, and Wout Dullaert. A multi-start local search algorithm for the vehicle routing problem with time windows. *European Journal of Operational Research*, 2003. to appear.

[10] Alain Chabrier. Vehicle routing problem with elementary shortest path based column generation. Working Paper, ILOG, Madrid, 2003, 2003.

[11] I-Ming Chao, Bruce Golden, and Edward Wasil. A computational study of a new heuristic for the site-dependent vehicle routing problem. *INFOR*, 37(3):319–336, 1999.

[12] I.M. Chao, B. L. Golden, and E.A. Wasil. A new heuristic for the multi-depot vehicle routing problem that improves upon best-known solutions. *Am. J. Math. Mgmt. Sci.*, 13:371–406, 1993.

[13] Nicos Christofides, Aristide Mingozzi, and Paolo Toth. The vehicle routing problem. In Nicos Christofides, Aristide Mingozzi, Paolo Toth, and Claudio Sandi, editors, *Combinatorial Optimization*, chapter 11, pages 315 – 338. John Wiley & Sons, 1979.

[14] William Cook and Jennifer L. Rich. A parallel cutting-plane algorithm for the vehicle routing problem with time windows. Technical Report TR99-04, Departement of Computational and Applied Mathematics, Rice University, 1999.

[15] Jean-François Cordeau, Guy Desaulniers, Jacques Desrosiers, Marius M. Solomon, and François Soumis. Vrp with time windows. In Paulo Toth and Daniele Vigo, editors, *The Vehicle Routing Problem*, volume 9 of *SIAM Monographs on Discrete Mathematics and Applications*, chapter 7, pages 157–193. SIAM, Philidelphia, 2002.

[16] Jean-François Cordeau, Michel Gendreau, Alain Hertz, Gilbert Laporte, and Jean-Sylvain Sormany. New heuristics for the vehicle routing problem. Technical Report G-2004-33, GERAD, Montreal, Canada, 2004.

[17] Jean-François Cordeau, Michel Gendreau, and Gilbert Laporte. A tabu search heuristic for periodic and multi-depot vehicle routing problems. *Networks*, 30:105–119, 1997.

[18] Jean-François Cordeau and Gilbert Laporte. A tabu search algorithm for the site dependent vehicle routing problem with time windows. *INFOR*, 39:292–298, 2001.

[19] Jean-François Cordeau, Gilbert Laporte, and Anne Mercier. A unified tabu search heuristic for vehicle routing problems with time windows. *Journal of the Operational Research Society*, 52:928–936, 2000.

[20] Emilie Danna and Claude Le Pape. Accelerating branch-and-price with local search: A case study on the vehicle routing problem with time windows. Technical Report 03-006, ILOG, ILOG S.A, 9, rue de Verdun, F-94253 Gentilly Cédex, Septemper 2003.

[21] Guy Desaulniers, Jacques Desrosiers, Andreas Erdmann, Marius M. Solomon, and François Soumis. Vrp with pickup and delivery. In Paulo Toth and Daniele Vigo, editors, *The Vehicle Routing Problem*, volume 9 of *SIAM Monographs on Discrete Mathematics and Applications*, chapter 9, pages 225–242. SIAM, Philidelphia, 2002.

[22] Ozlem Ergun, James B. Orlin, and Abran Steele-Feldman. Creating very large scale neighborhoods out of smaller ones by compounding moves: a study on the vehicle routing problem. Working Paper, Massachusetts Institute of Technology, 2003.

[23] Éric Taillard. Parallel iterative search methods for vehicle routing problems. *Networks*, 23:661–673, 1993.

[24] Dominique Feillet, Pierre Dejax, Michel Gendreau, and Cyrille Gueguen. An exact algorithm for the elementary shortest path problem with resource constraints: Application to some vehicle routing problems. *Networks*, 44(3):216–229, 2004.

[25] Zhuo Fu, Richard W. Eglese, and Leon Li. A tabu search heuristic for the open vehicle routing problem. Technical Report 2003/042, Lancaster University Management School, Lancaster, United Kingdom, 2003.

[26] Ricardo Fukasawa, Jens Lysgaard, Marcus Poggi de Aragão, Marcelo Reis, Eduardo Uchoa, and Renato F. Werneck. Robust branch-and-cut-and-price for the capacitated vehicle routing problem. In *Proceedings of IPCO X*. Columbia University, 2004.

[27] Hermann Gehring and Jörg Homberger. A parallel hybrid evolutionary metaheuristic for the vehicle routing problem with time windows. In K. Miettinen, M. Mäkelä, and J. Toivanen, editors, *Proceedings of EUROGEN99 – Short Course on Evolutionary Algorithms in Engineering and Computer Science*, pages 57–64. University of Jyväskylä, 1999.

[28] Hermann Gehring and Jörg Homberger. A parallel two-phase metaheuristic for routing problems with time windows. *Asia-Pacific Journal of Operational Research*, 18:35–47, 2001.

[29] Michel Gendreau, Gilbert Laporte, and Jean-Yves Potvin. Metaheuristics for the capacitated vrp. In Paulo Toth and Daniele Vigo, editors, *The Vehicle Routing Problem*, volume 9 of *SIAM Monographs on Discrete Mathematics and Applications*, chapter 6, pages 129–154. SIAM, Philidelphia, 2002.

[30] Bruce L. Golden, Edward A. Wasil, James P. Kelly, and I-Ming Chao. Metaheuristics in vehicle routing. In T.G. Crainic and G. Laporte, editors, *Fleet Management and Logistics*, pages 33–56. Kluwer, Boston (MA), 1998.

[31] Pierre Hansen and Nenad Mladenovic. An introduction to variable neighborhood search. In Stefan Voss et al., editor, *Metaheuristics, Advances and Trends in Local Search Paradigms for Optimization*, pages 433–458. Kluwer Academic Publishers, Dordrecht, 1999.

[32] Jörg Homberger and Hermann Gehring. A two-phase hybrid metaheuristic for the vehicle routing problem with time windows. *European Journal of Operational Research*, 2004. To appear.

[33] T. Ibaraki, S. Imahori, T. Masuda, T. Uno, and M. Yagiura. A route hybrid genetic approach for the vehicle routing problem with time windows. *Transportation Science*. To appear.

[34] Stefan Irnich and Daniel Villeneuve. The shortest path problem with resource constraints and k-cycle elimination for $k \geq 3$. Technical Report G-2003-55, GERAD, Montreal, Canada, September 2003.

[35] B. Kallehauge, J. Larsen, and O. B. G. Madsen. Lagrangean duality applied on vehicle routing with time windows - experimental results. Technical Report IMM-REP-2000-8, Informatics and Mathematical Modelling, Technical University of Denmark, DTU, Richard Petersens Plads, Building 321, DK-2800 Kgs. Lyngby, 2001.

[36] Niklas Kohl, Jacques Desrosiers, Oli B. G. Madsen, Marius M. Solomon, and François Soumis. 2-path cuts for the vehicle routing problem with time windows. *Transportation Science*, 33(1):101 – 116, 1999.

[37] D.M. Lambert and M.C. Cooper. Issues in supply chain management. *Marketing Management*, 29:65–83, 2000.

[38] Gilbert Laporte and Frédéric Semet. Classical heuristics for the capacitated vrp. In Paulo Toth and Daniele Vigo, editors, *The Vehicle Routing Problem*, volume 9 of *SIAM Monographs on Discrete Mathematics and Applications*, chapter 5, pages 109–128. SIAM, Philidelphia, 2002.

[39] Jesper Larsen. *Parallellization of the Vehicle Routing Problem with Time Windows*. PhD thesis, Department of Mathematical Modelling, Technical University of Denmark, 1999. IMM-PHD-1999-62.

[40] Feiyue Li, Bruce Golden, and Edward Wasil. Very large-scale vehicle routing: new test problems, algorithms, and results. *Computers and Operations Research*, 2004. To appear.

[41] Jens Lysgaard, Adam N. Letchford, and Richard W. Eglese. A new branch-and-cut algorithm for the capacitated vehicle routing problem. *Mathematical Programming, Ser. A*, 100:423–445, 2004.

[42] David Mester and Olli Bräysy. Active guided evolution strategies for large-scale vehicle routing problems with time windows. *Computers and Operations Research*, 2004. to appear.

[43] Barindra Nag, Bruce L. Golden, and Arjang Assad. Vehicle routing with site dependencies. In B.L. Golden and A.A. Assad, editors, *Vehicle Routing: Methods and Studies*, pages 149–159. Elsevier Science Publishers B.V, 1988.

[44] David Pisinger. Upper bounds and exact algorithms for p-dispersion problems. *Computers and Operations Research*, 2004. to appear.

[45] Jean-Yves Potvin and Jean-Marc Rousseau. A parallel route building algorithm for the vehicle routing and scheduling problem with time windows. *European Journal of Operational Research*, 66:331–340, 1993.

[46] Christian Prins. A simple and effective evolutionary algorithm for the vehicle routing problem. *Computers and Operations Research*, 31:1985–2002, 2004.

[47] M. Reimann, K. Doerner, and R.F. Hartl. D-ants: Savings based ants divide and conquer the vrp. *Computers and Operations Research*, 31:563–591, 2004.

[48] Jacques Renaud, Gilbert Laporte, and Fayez F. Boctor. A tabu search heuristic for the multi-depot vehicle routing problem. *Computers and Operations Research*, 23(3):229–235, 1996.

[49] Stefan Ropke and David Pisinger. An adaptive large neighborhood search heuristic for the pickup and delivery problem with time windows, 2004. Submitted to Transportation Science.

[50] Stefan Ropke and David Pisinger. A unified heuristic for a large class of vehicle routing problems with backhauls. *European Journal of Operational Research*, 2004. to appear.

[51] D. Sariklis and S. Powell. A heuristic method for the open vehicle routing problem. *Journal of the Operational Research Society*, 51:564–573, 2000.

[52] Gerhard Schrimpf, Johannes Schneider, Hermann Stamm-Wilbrandt, and Gunter Dueck. Record breaking optimization results using the ruin and recreate principle. *Journal of Computational Physics*, 159(2):139–171, 2000.

[53] Paul Shaw. Using constraint programming and local search methods to solve vehicle routing problems. In *CP-98 (Fourth International Conference on Principles and Practice of Constraint Programming)*, volume 1520 of *Lecture Notes in Computer Science*, pages 417–431, 1998.

[54] Mikkel Sigurd, David Pisinger, and Michael Sig. The pickup and delivery problem with time windows and precedences. *Transportation Science*, 38:197–209, 2004.

[55] Marius M. Solomon. Algorithms for the vehicle routing and scheduling problems with time window constraints. *Operations Research*, 35(2):254–265, 1987.

[56] C.D. Tarantilis and C.T. Kiranoudis. Boneroute: an adaptive memory-based method for effective fleet management. *Annals of Operations Research*, 115:227–241, 2002.

[57] Paulo Toth and Daniele Vigo. The granular tabu search and its application to the vehicle-routing problem. *INFORMS Journal on Computing*, 15(4):333–346, 2003.

[58] Michael A. Trick. A linear relaxation heuristic for the generalized assigment problem. *Naval Research Logistics*, 39:137–152, 1992.

[59] Jr. William A. Dees and Patrick G. Karger. Automated rip-up and reroute techniques. In *Proceedings of the 19th conference on Design automation*, pages 432–439. IEEE Press, 1982.

	Best known					ALNS 25K				ALNS 50K			
	n	t	type	cost	ref	avg. sol.	best sol.	avg. gap (%)	avg. time (s)	avg. sol.	best sol.	avg. gap (%)	avg. time (s)
P01	50	4	C	**576.87**	CGW	576.87	**576.87**	0.00	14	576.87	**576.87**	0.00	29
P02	50	4	C	**473.53**	RLB	473.53	**473.53**	0.00	14	473.53	**473.53**	0.00	28
P03	75	2	C	**641.19**	CGW	641.19	**641.19**	0.00	32	641.19	**641.19**	0.00	64
P04	100	2	C	1001.59	CGL	1008.49	1001.59	0.74	42	1006.09	**1001.04**	0.50	88
P05	100	2	C	**750.03**	CGL	753.04	751.86	0.40	58	752.34	751.26	0.31	120
P06	100	3	C	**876.5**	RLB	884.36	880.42	0.90	47	883.01	876.70	0.74	93
P07	100	4	C	885.8	CGL	889.14	**881.97**	0.81	43	889.36	**881.97**	0.84	88
P08	249	2	CD	4437.68	CGL	4426.86	**4387.38**	0.90	166	4421.03	4390.80	0.77	333
P09	249	3	CD	3900.22	CGL	3902.18	3874.75	0.74	182	3892.50	**3873.64**	0.49	361
P10	249	4	CD	3663.02	CGL	3676.93	3655.18	0.74	180	3666.85	**3650.04**	0.46	363
P11	249	5	CD	3554.18	CGL	3592.82	3552.27	1.32	174	3573.23	**3546.06**	0.77	357
P12	80	2	C	**1318.95**	RLB	1319.70	**1318.95**	0.06	38	1319.13	**1318.95**	0.01	75
P13	80	2	CD	**1318.95**	RLB	1321.10	**1318.95**	0.16	30	1318.95	**1318.95**	0.00	60
P14	80	2	CD	**1360.12**	CGL	1360.12	**1360.12**	0.00	29	1360.12	**1360.12**	0.00	58
P15	160	4	C	**2505.42**	CGL	2517.96	**2505.42**	0.50	125	2519.64	**2505.42**	0.57	253
P16	160	4	CD	**2572.23**	RLB	2577.28	**2572.23**	0.20	92	2573.95	**2572.23**	0.07	188
P17	160	4	CD	**2709.09**	CGL	2709.65	**2709.09**	0.02	90	2709.09	**2709.09**	0.00	179
P18	240	6	C	**3702.85**	CGL	3751.85	3727.58	1.32	209	3736.53	**3702.85**	0.91	419
P19	240	6	CD	**3827.06**	RLB	3846.35	3839.36	0.50	158	3838.76	**3827.06**	0.31	315
P20	240	6	CD	**4058.07**	CGL	4065.32	**4058.07**	0.18	151	4064.76	**4058.07**	0.16	300
P21	360	9	C	**5474.84**	CGL	5576.82	5519.47	1.86	293	5501.58	**5474.84**	0.49	582
P22	360	9	CD	**5702.16**	CGL	5731.10	5714.46	0.51	228	5722.19	**5702.16**	0.35	462
P23	360	9	CD	6095.46	CGL	6107.84	**6078.75**	0.48	223	6092.66	**6078.75**	0.23	443
PR01	48	4	CD	**861.32**	CGL	861.32	**861.32**	0.00	16	861.32	**861.32**	0.00	30
PR02	96	4	CD	1307.61	CGL	1311.54	**1307.34**	0.32	52	1308.17	**1307.34**	0.06	103
PR03	144	4	CD	1806.6	CGL	1810.90	**1806.53**	0.24	106	1810.66	1806.60	0.23	214
PR04	192	4	CD	2072.52	CGL	2080.55	2066.64	0.95	146	2073.16	**2060.93**	0.59	296
PR05	240	4	CD	2385.77	CGL	2352.59	2341.65	0.63	188	2350.31	**2337.84**	0.53	372
PR06	288	4	CD	2723.27	CGL	2695.15	**2685.35**	0.36	232	2695.74	2687.60	0.39	465
PR07	72	6	CD	**1089.56**	CGL	1089.56	**1089.56**	0.00	29	1089.56	**1089.56**	0.00	58
PR08	144	6	CD	1666.6	CGL	1677.31	1665.80	0.75	105	1675.74	**1664.85**	0.65	207
PR09	216	6	CD	2153.1	CGL	2148.85	**2136.42**	0.58	173	2144.84	**2136.42**	0.39	350
PR10	288	6	CD	2921.85	CGL	2913.34	**2889.49**	0.83	228	2905.43	2889.82	0.55	455
Tot.				80394		80651.59	80249.57		3894	80448.26	80133.89		7809
Avg.								0.52	118			0.34	237
< PB							14				14		
#B				18			20				27		

Table 9: Multi depot vehicle routing problems. The leftmost column shows the problem name, while the rest of the table is divided into three major columns that display the previously best known results, and the results obtained by the ALNS-25K and ALNS-50K configurations. The sub columns should be interpreted like this: *n* — number of customers, *t* — number of depots, *type* — the type of the instance (*C* indicates that the instance is capacity constrained, while *D* indicates that route duration constraints are present), *cost* — the cost of the previously best known solution (the cost is calculated as the total distance traveled), *ref* — where the solution first was reported. The following abbreviations are used: *CGW* — Chao et al. [12], *RLB* — Renaud et al. [48], *CGL* — Cordeau et al. [17]. The last 10 instances were introduced by Cordeau et al. [17] and the two other heuristics have not been applied to these instances. The Columns *best sol.* and *avg. sol.* show the cost of the best solution and the average cost of the solutions obtained during 10 experiments. *avg. gap* shows how far the average solution cost is from the best known solution. *avg. time* shows how much time the heuristic spends to perform one experiment. The rows *Tot.* and *Avg.* sums and averages key columns. *<PB* indicates for how many instances an improved solution was found and *#B* displays the number of best known solutions obtained. Entries written in bold indicate best known solutions.

	Best known				ALNS 25K				ALNS 50K			
	n	t	cost	ref	avg. sol.	best sol.	avg. gap (%)	avg. time (s)	avg. sol.	best sol.	avg. gap (%)	avg. time (s)
P01	50	3	642.66	CL	645.04	**640.32**	0.74	10	642.93	**640.32**	0.41	20
P02	50	2	**598.1**	CL	599.40	**598.10**	0.22	10	598.82	**598.10**	0.12	19
P03	75	3	959.36	CL	962.36	958.14	0.56	20	963.14	**957.04**	0.64	40
P04	75	2	**854.43**	CL	858.05	**854.43**	0.42	18	856.22	**854.43**	0.21	36
P05	100	3	1020.22	CL	1012.46	1007.51	0.89	34	1009.08	**1003.57**	0.55	68
P06	100	2	1036.02	CL	1034.09	1028.70	0.54	35	1032.67	**1028.52**	0.40	69
P07	27	3	**391.3**	CGW	391.30	**391.30**	0.00	4	391.30	**391.30**	0.00	8
P08	54	3	**664.46**	CGW	664.46	**664.46**	0.00	12	664.46	**664.46**	0.00	24
P09	81	3	**948.23**	CGW	958.69	**948.23**	1.10	24	961.36	**948.23**	1.38	47
P10	108	3	1223.88	CL	1229.42	**1218.75**	0.88	38	1225.28	**1218.75**	0.54	76
P11	135	3	1464.98	CL	1488.28	1468.38	1.70	58	1475.85	**1463.33**	0.86	116
P12	162	3	1695.67	CL	1697.98	1690.56	1.17	78	1689.62	**1678.40**	0.67	157
P13	54	3	1196.73	CL	1194.40	**1194.18**	0.02	12	1194.91	**1194.18**	0.06	24
P14	108	3	1962.66	CL	1961.11	**1960.62**	0.02	36	1960.83	**1960.62**	0.01	72
P15	162	3	2751.45	CL	2712.10	2695.22	1.01	77	2701.61	**2685.09**	0.61	152
P16	216	3	3491.18	CL	3421.74	3402.94	0.75	109	3411.50	**3396.36**	0.45	213
P17	270	3	4230.96	CL	4109.62	**4084.92**	0.60	146	4114.26	4085.61	0.72	291
P18	324	3	4929.71	CL	4821.55	4775.35	1.39	177	4795.31	**4755.50**	0.84	346
P19	100	3	850.39	CL	852.09	846.35	0.71	43	848.54	**846.07**	0.29	85
P20	150	3	1046.14	CL	1048.75	1042.21	1.74	83	1042.10	**1030.78**	1.10	168
P21	199	3	1337.83	CL	1281.58	1272.41	0.77	110	1283.03	**1271.75**	0.89	217
P22	120	3	1012.17	CL	1010.30	1008.78	0.16	65	1008.81	**1008.71**	0.01	130
P23	100	3	818.75	CL	807.67	**803.29**	0.55	37	807.00	**803.29**	0.46	73
PR01	48	4	1384.15	CL	1387.37	**1380.77**	0.48	10	1393.85	**1380.77**	0.95	19
PR02	96	4	2320.97	CL	2311.54	**2311.54**	0.00	32	2330.60	**2311.54**	0.82	63
PR03	144	4	2623.31	CL	2608.31	**2590.01**	0.71	71	2607.66	2602.13	0.68	140
PR04	192	4	3500.79	CL	3510.26	3481.44	1.04	98	3489.51	**3474.01**	0.45	191
PR05	240	4	4479.34	CL	4430.28	**4382.65**	1.09	123	4431.16	4416.38	1.11	251
PR06	288	4	4546.79	CL	4475.52	4452.93	0.70	159	4465.18	**4444.52**	0.47	314
PR07	72	6	1955.11	CL	1926.52	**1889.82**	1.94	19	1916.50	**1889.82**	1.41	39
PR08	144	6	3082.32	CL	3001.88	**2976.76**	0.84	66	3007.99	2977.50	1.05	135
PR09	216	6	3664.22	CL	3581.58	3548.22	1.28	113	3567.15	**3536.20**	0.88	226
PR10	288	6	4739.43	CL	4675.65	**4646.96**	0.62	162	4673.67	4648.76	0.57	322
PR11	1008	4	13227.96	CL	12987.58	12888.47	2.11	433	12810.71	**12719.65**	0.72	847
PR12	720	6	9621.99	CL	9510.37	9437.14	1.30	332	9437.56	**9388.07**	0.53	658
Tot.			90274		89169.30	88541.88		2853	88810.17	88273.77		5658
Avg.							0.80	81			0.60	162
< PB						29				30		
#B			5			18				30		

Table 10: Site dependent vehicle routing problems. The table should be interpreted like Table 9. Column *t* shows the number of vehicle types. *CL* refers to the the heuristic by Cordeau and Laporte [18] and *CGW* refers to the heuristic by Chao et al [11]. The ALNS heuristic was applied 10 times for each problem.

		Christofides		Golden et al.		Li et al.	
Heuristic	CPU	%	Minutes	%	Minutes	%	Minutes
TV	P-200MHz	0.64	3.84	2.88	17.55	-	-
LGV	Athlon 1Ghz	-	-	1.05	-	1.20	3.16
CGLM	P4-2GHz	0.56	24.62	1.46	56.11	-	-
EOS	P3-733MHz	0.24	30.95	3.77	137.95	-	-
P	P3-1GHz	0.24	5.19	0.92	66.90	-	-
TK	P2-400MHz	0.23	5.22	-	-	-	-
MB Best	P4-2GHz	0.03	7.72	0.01	72.94	-	-
MB Fast	P4-2GHz	0.07	0.27	0.94	0.63	-	-
BB	P-400MHz	0.49	21.25	-	-	-	-
RDH	P-900Mhz	-	-	0.67	49.33	-	-
ALNS 25K Best of 10	P4-3GHz	0.15	9.33	0.67	53.00	0.88	243.17
ALNS 25K Avg.	P4-3GHz	0.39	0.93	1.25	5.30	2.40	24.32
ALNS 50K Best of 10	P4-3GHz	0.11	17.50	0.49	107.67	0.50	497.90
ALNS 50K Avg.	P4-3GHz	0.31	1.75	1.02	10.77	1.90	49.79
		14 instances		20 instances		12 instances	
		50-200 customers		240-483 customers		560-1200 customers	

Table 11: Capacitated vehicle routing problems. The table compares the ALNS heuristic to nine heuristics proposed in the literature recently. The first column indicates the heuristic considered. *TV* — granular tabu search by Toth and Vigo [57], *LGV* — variable-length neighbor list record-to-record travel heuristic by Li et al. [40], *CGLM* — unified tabu search by Cordeau et al. [17, 19], *EOS* — very large scale neighborhood search by Ergun et al [22], *P* — evolutionary algorithm by Prins [46], *TK* — bone route heuristic by Tarantilis and Kiranoudis [56], *MB* — AGES heuristic by Mester and Bräysy [42] (two configurations of this heuristic is included in the table), *BB* — hybrid genetic algorithm by Berger and Barkaoui [3], *RDH* — ants system algorithm by Reimann et al. [47]. The table contains four rows for the ALNS heuristic. For each of the configurations ALNS-25K and ALNS-50K we report the best solution quality in ten experiments and the average solution quality (averaged over the same ten experiments). The *CPU* column lists the CPU used, *P* is used as an abbreviation for Pentium. The rest of the table contains three major columns, one for each dataset. For each of the datasets we report the gap between the solution obtained by the heuristic and the best known solution and we report the time spend on average by the heuristic to solve one instance. When reporting solution times for finding the best solution of ten runs, the time of all runs has been included. The ALNS heuristic is the only heuristic that has been applied to all datasets, which explains the missing entries. It should be noted that some of the numbers reported in the table were obtained from the survey by Cordeau et al. [16].

	Best known				ALNS 25K						ALNS 50K					
	n	veh.	cost	References	avg. sol.	avg. #veh.	best sol.	best #veh.	avg. gap (%)	avg. time (s)	avg. sol.	avg. #veh.	best sol.	best #veh.	avg. gap (%)	avg. time (s)
P01	50	**5**	**408.5**	FEL	416.67	5.0	416.06	5	2.00	12	416.45	5.0	416.06	5	1.95	23
P02	75	10	570.6	FEL	570.81	10.0	**567.14**	**10**	0.65	36	568.86	10.0	**567.14**	**10**	0.30	53
P03	100	**8**	**617**	FEL	642.93	8.0	641.76	8	4.20	85	642.32	8.0	641.76	8	4.10	128
P04	150	12	734.5	FEL	734.34	12.0	**733.13**	**12**	0.17	179	733.49	12.0	**733.13**	**12**	0.05	279
P05	199	16	953.4	B	912.54	16.0	897.93	16	1.84	124	907.03	16.0	**896.08**	**16**	1.22	237
P06	50	**6**	**400.6**	FEL	412.96	6.0	412.96	6	3.08	20	412.96	6.0	412.96	6	3.08	31
P07	75	10	634.5	B	592.16	10.0	584.15	10	1.54	18	588.72	10.0	**583.19**	**10**	0.95	33
P08	100	**9**	**638.2**	FEL	646.23	9.0	645.31	9	1.26	73	646.28	9.0	645.16	9	1.27	114
P09	150	13	785.2	B	766.42	13.1	759.35	13	1.13	108	764.32	13.1	**757.84**	**13**	0.85	185
P10	199	17	884.6	B	882.33	17.0	**875.67**	**17**	0.76	120	878.42	17.0	**875.67**	**17**	0.31	224
P11	120	7	683.4	B	682.68	7.0	**682.12**	**7**	0.08	73	682.39	7.0	**682.12**	**7**	0.04	141
P12	100	10	534.8	FEL	534.81	10.0	**534.24**	**10**	0.11	80	534.44	10.0	**534.24**	**10**	0.04	118
P13	120	11	943.7	B	911.98	11.0	**909.80**	**11**	0.24	61	911.12	11.0	**909.80**	**11**	0.15	116
P14	100	11	597.3	B	591.87	11.0	**591.87**	**11**	0.00	40	591.89	11.0	**591.87**	**11**	0.00	75
F11	71	**4**	**175**	FEL	177.00	4.0	177.00	4	1.14	69	177.00	4.0	177.00	4	1.14	104
F12	134	7	778.5	FEL	770.59	7.0	**770.17**	**7**	0.06	237	770.31	7.0	**770.17**	**7**	0.02	359
Tot.		156	10340		10246.32	156.10	10198.67	156		1336	10225.99	156.10	10194.19	156		2222
Avg.									1.14	83					0.97	139
< PB							11						11			
#B			5				8						11			

Table 12: Open vehicle routing problem instances. The table should be interpreted like Table 9. The abbreviations used in the *References* column are: *B* - Brandao's heuristic [6], *FEL* - the heuristic by Fu et al. [25]. The column *veh.* indicates the number of vehicles used in the previous best solution, *avg. #veh.* indicates the number of vehicles used on average by the particular ALNS configuration (averaged over ten experiments). The column *best #veh.* indicates the number of vehicles used in the best found solution (out of 10 experiments).

	Best known			ALNS 25K						ALNS 50K					
	veh.	cost	References	avg. sol.	avg. #veh.	best sol.	best #veh.	avg. gap (%)	avg. time (s)	avg. sol.	avg. #veh.	best sol.	best #veh.	avg. gap (%)	avg. time (s)
R101	**19**	**1645.79**	H (2000)	1650.86	19.0	1650.80	19	0.31	55	1650.80	19.0	1650.80	19	0.30	85
R102	**17**	**1486.12**	RT (1995)	1486.89	17.0	**1486.12**	**17**	0.05	62	1486.75	17.0	**1486.12**	**17**	0.04	94
R103	**13**	**1292.68**	LLH (2001)	1294.89	13.0	**1292.68**	**13**	0.17	64	1294.04	13.0	**1292.68**	**13**	0.11	97
R104	**9**	**1007.24**	M (2002)	987.85	9.8	1013.13	9	-1.93	61	987.85	9.8	1013.13	9	-1.93	96
R105	**14**	**1377.11**	RT (1995)	1378.77	14.0	**1377.11**	**14**	0.12	56	1378.11	14.0	**1377.11**	**14**	0.07	85
R106	**12**	**1251.98**	M (2002)	1258.40	12.0	1252.03	12	0.51	61	1255.52	12.0	1252.03	12	0.28	92
R107	**10**	**1104.55**	S97 (1997)	1118.18	10.0	1113.70	10	1.23	52	1115.19	10.0	1104.76	10	0.96	85
R108	**9**	**960.88**	BBB (2001)	969.37	9.0	963.91	9	0.88	40	965.36	9.0	**960.88**	**9**	0.47	75
R109	**11**	**1194.73**	HG (1999)	1213.09	11.1	**1194.73**	**11**	1.54	47	1211.44	11.1	**1194.73**	**11**	1.40	77
R110	**10**	**1118.59**	M (2002)	1149.56	10.0	1119.14	10	2.77	41	1148.92	10.0	1119.14	10	2.71	71
R111	**10**	**1096.72**	RGP (2001?)	1112.14	10.0	1096.74	10	1.41	46	1105.36	10.0	1096.73	10	0.79	78
R112	**9**	**982.14**	GTA (1999)	983.16	9.5	1000.60	9	0.10	58	982.62	9.5	1000.60	9	0.05	91
C101	**10**	**828.94**	RT (1995)	828.94	10.0	**828.94**	**10**	0.00	29	828.94	10.0	**828.94**	**10**	0.00	57
C102	**10**	**828.94**	RT (1995)	828.94	10.0	**828.94**	**10**	0.00	59	828.94	10.0	**828.94**	**10**	0.00	91
C103	**10**	**828.06**	RT (1995)	828.06	10.0	**828.06**	**10**	0.00	65	828.06	10.0	**828.06**	**10**	0.00	99
C104	**10**	**824.78**	RT (1995)	824.78	10.0	**824.78**	**10**	0.00	69	824.78	10.0	**824.78**	**10**	0.00	105
C105	**10**	**828.94**	RT (1995)	828.94	10.0	**828.94**	**10**	0.00	31	828.94	10.0	**828.94**	**10**	0.00	59
C106	**10**	**828.94**	RT (1995)	828.94	10.0	**828.94**	**10**	0.00	32	828.94	10.0	**828.94**	**10**	0.00	62
C107	**10**	**828.94**	RT (1995)	828.94	10.0	**828.94**	**10**	0.00	32	828.94	10.0	**828.94**	**10**	0.00	62
C108	**10**	**828.94**	RT (1995)	828.94	10.0	**828.94**	**10**	0.00	61	828.94	10.0	**828.94**	**10**	0.00	93
C109	**10**	**828.94**	RT (1995)	828.94	10.0	**828.94**	**10**	0.00	64	828.94	10.0	**828.94**	**10**	0.00	99
RC101	**14**	**1696.94**	TBGGP (1997)	1688.35	14.2	1697.43	14	-0.51	53	1688.17	14.2	1697.43	14	-0.52	80
RC102	**12**	**1554.75**	TBGGP (1997)	1547.04	12.1	**1554.75**	**12**	-0.50	56	1555.06	12.1	**1554.75**	**12**	0.02	84
RC103	**11**	**1261.67**	S98 (1998)	1270.78	11.0	1262.02	11	0.72	58	1268.53	11.0	1262.02	11	0.54	90
RC104	**10**	**1135.48**	CLM (2000)	1135.80	10.0	1135.52	10	0.03	60	1135.89	10.0	1135.83	10	0.04	92
RC105	**13**	**1629.44**	BBB (2001)	1640.18	13.0	**1629.44**	**13**	0.66	54	1640.92	13.0	1633.72	13	0.70	83
RC106	**11**	**1424.73**	BBB (2001)	1413.07	11.5	1432.12	11	-0.82	49	1411.92	11.5	1432.12	11	-0.90	76
RC107	**11**	**1230.48**	S97 (1997)	1232.48	11.0	1230.95	11	0.16	56	1231.65	11.0	1230.54	11	0.09	86
RC108	**10**	**1139.82**	TBGGP (1997)	1167.55	10.0	1140.87	10	2.43	41	1152.30	10.0	**1139.82**	**10**	1.10	71
R201	**4**	**1252.37**	HG (1999)	1253.23	4.0	1253.23	4	0.07	133	1253.23	4.0	1253.23	4	0.07	193
R202	**3**	**1191.7**	RGP (2001?)	1229.81	3.0	1195.30	3	3.20	96	1223.62	3.0	1195.30	3	2.68	181
R203	**3**	**939.54**	M (2002)	944.64	3.0	939.58	3	0.54	164	943.57	3.0	941.08	3	0.43	256
R204	**2**	**825.52**	BVH (2001)	841.48	2.0	833.09	2	1.93	182	843.39	2.0	833.09	2	2.16	346
R205	**3**	**994.42**	RGP (2001?)	1018.90	3.0	994.43	3	2.46	97	1010.43	3.0	994.43	3	1.61	186
R206	**3**	**906.14**	SSSD (2000)	923.91	3.0	915.27	3	1.96	192	921.07	3.0	**906.14**	**3**	1.65	282
R207	**2**	**893.33**	BVH (2001)	928.28	2.0	**893.33**	**2**	3.91	180	927.62	2.0	**893.33**	**2**	3.84	332
R208	**2**	**726.75**	M (2002)	736.12	2.0	726.82	2	1.29	185	735.76	2.0	726.82	2	1.24	369
R209	**3**	**909.16**	H (2000)	926.72	3.0	914.45	3	1.93	101	923.48	3.0	914.13	3	1.58	185
R210	**3**	**939.34**	M (2002)	955.02	3.0	954.12	3	1.67	112	955.29	3.0	950.52	3	1.70	204
R211	**2**	**892.71**	BVH (2001)	889.99	2.3	925.03	2	-0.30	216	887.93	2.3	926.83	2	-0.54	349
C201	**3**	**591.56**	RT (1995)	591.56	3.0	**591.56**	**3**	0.00	78	591.56	3.0	**591.56**	**3**	0.00	147
C202	**3**	**591.56**	RT (1995)	591.56	3.0	**591.56**	**3**	0.00	88	591.56	3.0	**591.56**	**3**	0.00	163
C203	**3**	**591.17**	RT (1995)	591.17	3.0	**591.17**	**3**	0.00	96	591.17	3.0	**591.17**	**3**	0.00	181
C204	**3**	**590.6**	RT (1995)	590.60	3.0	**590.60**	**3**	0.00	102	590.60	3.0	**590.60**	**3**	0.00	189
C205	**3**	**588.88**	RT (1995)	588.88	3.0	**588.88**	**3**	0.00	81	588.88	3.0	**588.88**	**3**	0.00	155
C206	**3**	**588.49**	RT (1995)	588.49	3.0	**588.49**	**3**	0.00	83	588.49	3.0	**588.49**	**3**	0.00	156
C207	**3**	**588.29**	RT (1995)	588.29	3.0	**588.29**	**3**	0.00	84	588.29	3.0	**588.29**	**3**	0.00	167
C208	**3**	**588.32**	RT (1995)	588.32	3.0	**588.32**	**3**	0.00	85	588.32	3.0	**588.32**	**3**	0.00	161
RC201	**4**	**1406.91**	M (2002)	1417.80	4.0	1413.52	4	0.77	83	1414.69	4.0	1413.52	4	0.55	140
RC202	**3**	**1367.09**	CC (2002)	1405.16	3.0	1368.04	3	2.78	96	1403.60	3.0	**1367.09**	**3**	2.67	177
RC203	**3**	**1049.62**	CC (2002)	1075.51	3.0	1068.08	3	2.47	100	1072.57	3.0	1068.60	3	2.19	192
RC204	**3**	**798.41**	M (2002)	818.00	3.0	799.27	3	2.45	228	806.81	3.0	798.46	3	1.05	320
RC205	**4**	**1297.19**	M (2002)	1318.01	4.0	1302.42	4	1.60	134	1312.75	4.0	1302.42	4	1.20	194
RC206	**3**	**1146.32**	H (2000)	1155.91	3.0	**1146.32**	**3**	0.84	87	1155.16	3.0	**1146.32**	**3**	0.77	166
RC207	**3**	**1061.14**	BVH (2001)	1095.29	3.0	1070.85	3	3.22	96	1088.15	3.0	1061.84	3	2.55	182
RC208	**3**	**828.14**	IKMUY (2001)	834.83	3.0	829.69	3	0.81	109	829.96	3.0	829.69	3	0.22	196
Tot.	405	57192		57641.28	407.50	57360.86	405		4800	57549.75	407.50	57332.03	405		8182
Avg.								0.77	86					0.61	146
< PB						0						0			
#B		56				25						28			

Table 13: Solomon VRPTW instances. The table should be interpreted as table 12. The best known solutions were gathered from the web page: http://www.sintef.no/static/am/opti/projects/top/vrp/benchmarks.html. See this page for complete references to where the best known solutions first were identified.

	Best known			ALNS 25K						ALNS 50K					
	veh.	cost	References	avg. sol.	avg. #veh.	best sol.	best #veh.	avg. gap (%)	avg. time (s)	avg. sol.	avg. #veh.	best sol.	best #veh.	avg. gap (%)	avg. time (s)
R1_2_1	**19**	**5024.65**	B	4809.44	20.0	4798.22	20	-4.28	170	4798.77	20.0	4785.96	20	-4.50	257
R1_2_2	**18**	**4054.44**	MB	4091.46	18.0	4066.91	18	0.91	165	4079.18	18.0	4059.57	18	0.61	256
R1_2_3	**18**	**3164.41**	LC	3414.89	18.0	3387.64	18	7.92	160	3407.48	18.0	3396.47	18	7.68	260
R1_2_4	**18**	**3067.93**	MB	3104.90	18.0	3086.11	18	1.20	200	3100.94	18.0	3086.65	18	1.08	308
R1_2_5	**18**	**4112.88**	MB	4184.89	18.0	4125.19	18	1.75	147	4157.28	18.0	4125.19	18	1.08	231
R1_2_6	18	3599.84	MB	3643.10	18.0	3616.52	18	1.57	167	3631.74	18.0	**3586.80**	**18**	1.25	258
R1_2_7	**18**	**3151.42**	MB	3187.56	18.0	3170.98	18	1.15	169	3186.04	18.0	3160.44	18	1.10	271
R1_2_8	**18**	**2963.9**	MB	2992.96	18.0	2971.66	18	0.98	200	2989.62	18.0	2975.59	18	0.87	310
R1_2_9	**18**	**3784.33**	MB	3853.46	18.0	3802.55	18	1.83	135	3840.07	18.0	3823.15	18	1.47	223
R1_210	**18**	**3307.78**	MB	3363.82	18.0	3333.66	18	1.69	148	3336.35	18.0	3312.44	18	0.86	241
C1_2_1	**20**	**2704.57**	GH	2704.57	20.0	**2704.57**	**20**	0.00	94	2704.57	20.0	**2704.57**	**20**	0.00	181
C1_2_2	**18**	**2917.89**	BVH	2977.48	18.0	2948.73	18	2.04	152	2969.62	18.0	2943.83	18	1.77	242
C1_2_3	**18**	**2708.08**	MB	2744.41	18.0	2719.62	18	1.34	145	2729.39	18.0	2710.21	18	0.79	245
C1_2_4	**18**	**2644.61**	MB	2646.94	18.0	2645.60	18	0.09	146	2646.36	18.0	2644.92	18	0.07	253
C1_2_5	**20**	**2702.05**	GH	2702.05	20.0	**2702.05**	**20**	0.00	96	2702.05	20.0	**2702.05**	**20**	0.00	186
C1_2_6	**20**	**2701.04**	GH	2701.04	20.0	**2701.04**	**20**	0.00	101	2701.04	20.0	**2701.04**	**20**	0.00	193
C1_2_7	**20**	**2701.04**	GH	2701.04	20.0	**2701.04**	**20**	0.00	184	2701.04	20.0	**2701.04**	**20**	0.00	281
C1_2_8	**18**	**2769.19**	MB	2791.15	19.0	2775.48	19	0.79	157	2789.38	19.0	2775.48	19	0.73	250
C1_2_9	**18**	**2642.82**	MB	2705.26	18.0	2687.83	18	2.36	104	2688.82	18.0	2687.83	18	1.74	196
C1_210	18	2649.26	MB	2650.64	18.0	2645.08	18	0.24	117	2651.55	18.0	**2644.25**	**18**	0.28	214
RC1_2_1	18	3691.99	MB	3812.41	18.0	3727.17	18	4.52	93	3731.52	18.0	**3647.56**	**18**	2.30	175
RC1_2_2	18	3298.68	MB	3342.07	18.0	**3269.91**	**18**	2.21	96	3309.57	18.0	3276.88	18	1.21	185
RC1_2_3	**18**	**3025.9**	MB	3053.11	18.0	3036.32	18	0.90	104	3051.91	18.0	3034.45	18	0.86	201
RC1_2_4	18	2879.4	MB	2906.27	18.0	**2869.74**	**18**	1.27	109	2887.58	18.0	2873.54	18	0.62	215
RC1_2_5	**18**	**3419.81**	MB	3509.40	18.0	3463.01	18	2.62	90	3500.46	18.0	3430.03	18	2.36	173
RC1_2_6	18	3393.09	MB	3473.96	18.0	3398.67	18	3.46	91	3431.75	18.0	**3357.90**	**18**	2.20	174
RC1_2_7	18	3266.48	MB	3353.23	18.0	3290.65	18	3.71	93	3302.54	18.0	**3233.29**	**18**	2.14	179
RC1_2_8	18	3115.82	MB	3163.78	18.0	3147.87	18	1.71	95	3149.37	18.0	**3110.46**	**18**	1.25	183
RC1_2_9	**18**	**3083.41**	MB	3152.09	18.0	3114.02	18	2.23	94	3150.15	18.0	3116.47	18	2.16	183
RC1_210	18	3038.85	MB	3063.57	18.0	**3020.24**	**18**	1.43	98	3056.83	18.0	3042.24	18	1.21	190
R2_2_1	**4**	**4501.8**	MB	4340.82	4.5	4563.55	4	-3.58	527	4329.15	4.5	4571.67	4	-3.84	821
R2_2_2	**4**	**3645.38**	MB	3683.64	4.0	3666.72	4	1.05	416	3669.25	4.0	3650.54	4	0.65	795
R2_2_3	4	2932.44	MB	2928.17	4.0	**2892.07**	**4**	1.25	458	2924.73	4.0	**2892.07**	**4**	1.13	890
R2_2_4	**4**	**1981.29**	MB	1992.90	4.0	1981.30	4	0.59	482	1989.24	4.0	1981.30	4	0.40	910
R2_2_5	**4**	**3367.55**	SAM::OPT	3431.26	4.0	3382.22	4	1.89	385	3417.75	4.0	3377.18	4	1.49	723
R2_2_6	**4**	**2914.56**	MB	2957.14	4.0	2929.72	4	1.46	414	2947.20	4.0	2931.14	4	1.12	807
R2_2_7	**4**	**2453.62**	MB	2461.82	4.0	2456.71	4	0.33	461	2465.00	4.0	2459.82	4	0.46	883
R2_2_8	**4**	**1849.87**	MB	1874.00	4.0	1850.85	4	1.30	495	1866.03	4.0	**1849.87**	**4**	0.87	959
R2_2_9	**4**	**3111.41**	MB	3134.41	4.0	3113.74	4	0.74	405	3126.66	4.0	3113.74	4	0.49	768
R2_210	**4**	**2657**	MB	2696.24	4.0	2666.10	4	1.48	399	2690.93	4.0	2666.35	4	1.28	784
C2_2_1	**6**	**1931.44**	GH	1931.44	6.0	**1931.44**	**6**	0.00	214	1931.44	6.0	**1931.44**	**6**	0.00	391
C2_2_2	**6**	**1863.16**	GH	1863.16	6.0	**1863.16**	**6**	0.00	233	1863.16	6.0	**1863.16**	**6**	0.00	445
C2_2_3	**6**	**1775.11**	M	1784.79	6.0	1776.96	6	0.55	263	1783.42	6.0	1776.96	6	0.47	497
C2_2_4	6	1720.09	MB	1719.58	6.0	**1713.46**	**6**	0.36	275	1715.66	6.0	**1713.46**	**6**	0.13	527
C2_2_5	**6**	**1878.85**	BVH	1881.87	6.0	1879.31	6	0.16	224	1879.27	6.0	**1878.85**	**6**	0.02	413
C2_2_6	**6**	**1857.35**	B	1859.74	6.0	**1857.35**	**6**	0.13	224	1857.35	6.0	**1857.35**	**6**	0.00	425
C2_2_7	**6**	**1849.46**	GH	1851.62	6.0	**1849.46**	**6**	0.12	231	1849.46	6.0	**1849.46**	**6**	0.00	431
C2_2_8	6	1820.59	MB	1828.56	6.0	1823.88	6	0.44	233	1823.21	6.0	**1820.53**	**6**	0.15	442
C2_2_9	6	1830.18	SAM::OPT	1833.78	6.0	**1830.05**	**6**	0.20	241	1834.31	6.0	**1830.05**	**6**	0.23	449
C2_210	**6**	**1806.6**	M	1809.46	6.0	1808.21	6	0.16	249	1809.47	6.0	1808.21	6	0.16	466
RC2_2_1	**6**	**3103.48**	MB	3146.36	6.0	3126.03	6	1.38	428	3143.65	6.0	3129.07	6	1.29	635
RC2_2_2	**5**	**2827.45**	M	2870.43	5.0	2828.39	5	1.52	629	2856.08	5.0	2835.67	5	1.01	916
RC2_2_3	4	2617.9	MB	2652.00	4.0	2620.87	4	1.49	444	2631.97	4.0	**2613.12**	**4**	0.72	849
RC2_2_4	4	2055.97	MB	2080.99	4.0	2056.93	4	1.38	476	2063.32	4.0	**2052.74**	**4**	0.52	923
RC2_2_5	4	2912.57	MB	3039.09	4.0	2913.21	4	4.36	433	3041.01	4.0	**2912.13**	**4**	4.43	803
RC2_2_6	4	3086.76	LC	2920.37	4.3	2977.41	4	-1.84	525	2900.85	4.3	**2975.13**	**4**	-2.50	846
RC2_2_7	4	2550.56	M	2609.93	4.0	2563.90	4	2.76	408	2572.96	4.0	**2539.85**	**4**	1.30	789
RC2_2_8	4	2317.8	MB	2341.46	4.0	2322.52	4	1.16	413	2330.66	4.0	**2314.61**	**4**	0.69	788
RC2_2_9	**4**	**2175.61**	MB	2216.32	4.0	2175.98	4	1.87	417	2214.61	4.0	2180.81	4	1.79	795
RC2_210	**4**	**2015.6**	MB	2046.95	4.0	2020.68	4	1.56	424	2030.64	4.0	2015.61	4	0.75	808
Tot.	692	168997		170589.23	694.80	169370.28	694		15341	169941.42	694.80	169042.17	694		27688
Avg.								1.17	256					0.81	461
< PB						8						18			
#B		41				14						26			

Table 14: Gehring/Homberger VRPTW instances, 200 customers.. The best known solutions were gathered from the web page: http://www.sintef.no/static/am/opti/projects/top/vrp/benchmarks.html in January 2005. This list of best known solutions was supplemented by the solutions found by Mester and Bräysy [42] (MB) and Le Bouthillier and Crainic [**?**] (LC). See the aforementioned web page for full references. The same sources were used for the best known solution columns in tables 4 to 7.

	Best known			ALNS 25K						ALNS 50K					
	veh.	cost	References	avg. sol.	avg. #veh.	best sol.	best #veh.	avg. gap (%)	avg. time (s)	avg. sol.	avg. #veh.	best sol.	best #veh.	avg. gap (%)	avg. time (s)
R1_4_1	**38**	**11084**	B	10557.71	40.0	10502.22	40	-4.75	368	10485.89	40.0	10432.30	40	-5.40	554
R1_4_2	36	9161.26	MB	9277.12	36.0	9239.87	36	1.77	224	9166.43	36.0	**9115.68**	**36**	0.56	401
R1_4_3	**36**	**7941.53**	MB	8029.69	36.0	7996.33	36	1.11	269	8053.08	36.0	7988.22	36	1.40	464
R1_4_4	**36**	**7332.93**	MB	7468.64	36.0	7449.60	36	1.85	227	7441.43	36.0	7415.81	36	1.48	439
R1_4_5	36	9512.25	MB	9738.05	36.0	9588.45	36	2.73	182	9560.46	36.0	**9479.10**	**36**	0.86	347
R1_4_6	**36**	**8534.05**	MB	8740.69	36.0	8677.13	36	2.42	227	8613.60	36.0	8556.38	36	0.93	407
R1_4_7	**36**	**7710.41**	MB	7812.04	36.0	7769.68	36	1.32	245	7763.04	36.0	7725.97	36	0.68	439
R1_4_8	36	7398.68	MB	7468.69	36.0	7425.43	36	1.05	231	7398.80	36.0	**7390.76**	**36**	0.11	444
R1_4_9	**36**	**8878.19**	MB	9125.58	36.0	9058.30	36	2.79	217	9053.20	36.0	8970.98	36	1.97	386
R1_410	**36**	**8227.49**	MB	8417.50	36.0	8386.75	36	2.31	194	8363.10	36.0	8325.16	36	1.65	377
C1_4_1	**40**	**7152.02**	M	7152.06	40.0	7152.06	40	0.00	203	7152.06	40.0	7152.06	40	0.00	387
C1_4_2	37	7357.45	MB	7815.71	36.2	7830.99	36	1.06	260	7759.63	36.2	**7733.55**	**36**	0.34	437
C1_4_3	36	7151.17	MB	7208.25	36.0	7174.23	36	1.78	212	7104.35	36.0	**7082.13**	**36**	0.31	403
C1_4_4	36	6822.18	MB	6909.71	36.0	6833.32	36	1.37	224	6861.13	36.0	**6816.17**	**36**	0.66	431
C1_4_5	**40**	**7152.02**	M	7152.06	40.0	7152.06	40	0.00	215	7152.06	40.0	7152.06	40	0.00	404
C1_4_6	**40**	**7153.41**	M	7153.45	40.0	7153.45	40	0.00	286	7153.45	40.0	7153.45	40	0.00	479
C1_4_7	39	7668.33	LC	7643.60	39.0	7620.09	39	1.28	297	7621.62	39.0	**7546.78**	**39**	0.99	485
C1_4_8	38	7113.4	MB	7814.18	37.0	7661.98	37	3.55	284	7794.27	37.0	**7546.32**	**37**	3.29	464
C1_4_9	**36**	**7524.32**	MB	8042.29	36.0	7673.65	36	6.88	255	7800.59	36.0	7573.18	36	3.67	428
C1_410	**36**	**6907.26**	MB	7617.12	36.0	7446.94	36	10.28	214	7325.70	36.0	7145.92	36	6.06	398
RC1_4_1	36	8960.82	MB	9139.22	36.0	9044.65	36	3.70	207	8939.82	36.0	**8813.43**	**36**	1.43	371
RC1_4_2	36	8174.27	MB	8287.21	36.0	8181.05	36	2.08	197	8176.96	36.0	**8118.43**	**36**	0.72	370
RC1_4_3	36	7737.99	MB	7744.57	36.0	7668.27	36	1.05	214	7729.95	36.0	**7663.73**	**36**	0.86	403
RC1_4_4	36	7411.02	MB	7497.41	36.0	7447.70	36	1.75	226	7433.65	36.0	**7368.47**	**36**	0.88	436
RC1_4_5	36	8499.15	MB	8634.51	36.0	8503.19	36	2.47	190	8520.69	36.0	**8426.57**	**36**	1.12	356
RC1_4_6	**36**	**8304.99**	MB	8640.29	36.0	8533.72	36	4.04	185	8445.05	36.0	8390.24	36	1.69	351
RC1_4_7	**36**	**8051.71**	MB	8355.82	36.0	8223.65	36	3.78	192	8331.40	36.0	8227.10	36	3.47	360
RC1_4_8	**36**	**7917.68**	MB	8174.94	36.0	8135.05	36	3.25	192	8070.47	36.0	7922.67	36	1.93	363
RC1_4_9	**36**	**7890.45**	MB	8067.40	36.0	7953.20	36	2.24	194	8016.28	36.0	7987.55	36	1.59	370
RC1_410	**36**	**7716.32**	MB	7861.40	36.0	7805.59	36	1.88	199	7823.83	36.0	7774.83	36	1.39	376
R2_4_1	**8**	**9257.92**	MB	9513.88	8.0	9375.10	8	2.76	1002	9432.87	8.0	9338.49	8	1.89	1574
R2_4_2	8	7674.9	MB	7762.67	8.0	7728.27	8	1.47	1313	7744.54	8.0	**7649.87**	**8**	1.24	1942
R2_4_3	**8**	**5988.02**	MB	6078.27	8.0	5998.04	8	1.51	1426	6053.22	8.0	6034.08	8	1.09	2120
R2_4_4	8	4331.07	MB	4356.73	8.0	**4326.48**	**8**	0.70	1565	4345.23	8.0	4327.61	8	0.43	2333
R2_4_5	**8**	**7143.55**	MB	7305.24	8.0	7255.52	8	2.26	1207	7277.89	8.0	7252.64	8	1.88	1841
R2_4_6	**8**	**6163.81**	MB	6284.34	8.0	6222.32	8	1.96	1326	6229.61	8.0	6212.37	8	1.07	1986
R2_4_7	**8**	**5082.1**	MB	5182.15	8.0	5138.58	8	1.97	1441	5154.64	8.0	5136.74	8	1.43	2164
R2_4_8	8	4068.97	MB	4090.90	8.0	**4055.22**	**8**	0.88	1587	4076.34	8.0	4060.51	8	0.52	2384
R2_4_9	**8**	**6493.13**	MB	6565.87	8.0	6526.20	8	1.12	1222	6537.26	8.0	6507.40	8	0.68	1817
R2_410	8	5895.93	MB	5958.31	8.0	**5894.40**	**8**	1.08	1283	5919.14	8.0	5897.46	8	0.42	1891
C2_4_1	**12**	**4116.05**	M	4125.50	12.0	4116.33	12	0.23	403	4116.93	12.0	4116.33	12	0.02	753
C2_4_2	12	3930.29	MB	3930.22	12.0	**3930.05**	**12**	0.00	477	3930.13	12.0	**3930.05**	**12**	0.00	858
C2_4_3	**12**	**3739.72**	GH	3782.86	12.0	3775.32	12	1.15	525	3780.81	12.0	3775.54	12	1.10	952
C2_4_4	**12**	**3535.99**	MB	3549.80	12.0	3546.66	12	0.39	520	3568.37	12.0	3543.60	12	0.92	965
C2_4_5	**12**	**3939.42**	MB	3981.35	12.0	3946.94	12	1.06	434	3951.72	12.0	3946.14	12	0.31	783
C2_4_6	**12**	**3875.94**	MB	3883.95	12.0	**3875.94**	**12**	0.21	457	3921.04	12.0	**3875.94**	**12**	1.16	811
C2_4_7	**12**	**3894.13**	M	3937.44	12.0	3903.46	12	1.11	453	3960.36	12.0	3894.98	12	1.70	829
C2_4_8	**12**	**3787.08**	MB	3863.49	12.0	3804.12	12	2.02	498	3850.01	12.0	3796.00	12	1.66	884
C2_4_9	**12**	**3876.1**	MB	4025.46	12.0	3887.00	12	3.85	471	3964.79	12.0	3881.21	12	2.29	851
C2_410	**12**	**3684.89**	MB	3764.34	12.0	3706.87	12	2.16	502	3715.36	12.0	3687.13	12	0.83	896
RC2_4_1	11	7019.89	GH	6876.33	11.2	**6834.02**	**11**	0.62	786	6857.62	11.2	6840.51	11	0.35	1198
RC2_4_2	10	5924.84	MB	6166.44	9.8	6356.23	9	-2.98	1029	6125.49	9.8	**6355.59**	**9**	-3.62	1553
RC2_4_3	8	5114.76	MB	5139.79	8.0	5073.80	8	1.68	820	5109.29	8.0	**5055.02**	**8**	1.07	1503
RC2_4_4	8	3648.64	MB	3737.66	8.0	3666.70	8	2.47	959	3692.45	8.0	**3647.39**	**8**	1.24	1694
RC2_4_5	**9**	**6063.46**	MB	6107.39	9.4	6257.87	9	0.72	901	6019.04	9.4	6119.44	9	-0.73	1416
RC2_4_6	8	6054.21	GH	6093.66	8.0	**5997.24**	**8**	1.61	835	6092.17	8.0	6008.41	8	1.58	1425
RC2_4_7	8	5519.25	MB	5664.90	8.0	5529.42	8	3.44	714	5623.09	8.0	**5476.57**	**8**	2.68	1324
RC2_4_8	**8**	**4854.16**	MB	4949.32	8.0	4877.39	8	1.96	1284	4933.15	8.0	4891.18	8	1.63	1903
RC2_4_9	8	4628.26	MB	4736.64	8.0	4674.88	8	2.94	1168	4662.33	8.0	**4601.30**	**8**	1.33	1779
RC2_410	**8**	**4316.36**	MB	4415.46	8.0	4400.68	8	2.30	1290	4401.00	8.0	4355.52	8	1.96	1937
Tot.	1386	392070		399377.24	1386.60	395969.66	1385		34732	396157.93	1386.60	393210.00	1385		56699
Avg.								1.80	579					1.05	945
< PB						12						24			
#B		35				7						21			

Table 15: Gehring/Homberger VRPTW instances, 400 customers..

	Best known			ALNS 25K						ALNS 50K					
	veh.	cost	References	avg. sol.	avg. #veh.	best sol.	best #veh.	avg. gap (%)	avg. time (s)	avg. sol.	avg. #veh.	best sol.	best #veh.	avg. gap (%)	avg. time (s)
R1_6_1	**59**	**21131.09**	MB	21881.08	59.0	21767.25	59	3.55	514	21743.91	59.0	21677.41	59	2.90	763
R1_6_2	**54**	**19603.7**	MB	20892.38	54.0	20719.50	54	6.57	276	20253.42	54.0	20045.49	54	3.31	504
R1_6_3	**54**	**17400.6**	MB	18399.70	54.0	18154.60	54	5.74	289	17886.87	54.0	17733.91	54	2.79	535
R1_6_4	**54**	**15993.8**	MB	16640.46	54.0	16550.00	54	4.04	300	16459.06	54.0	16374.29	54	2.91	569
R1_6_5	**54**	**20395**	MB	22399.39	54.0	22051.85	54	9.83	359	21462.92	54.0	21243.24	54	5.24	577
R1_6_6	**54**	**18620.26**	MB	19759.95	54.0	19610.14	54	6.12	263	19206.68	54.0	18948.53	54	3.15	494
R1_6_7	**54**	**17107.91**	MB	17915.15	54.0	17773.37	54	4.72	279	17483.82	54.0	17438.28	54	2.20	527
R1_6_8	**54**	**15725.86**	MB	16509.06	54.0	16436.50	54	4.98	295	16245.90	54.0	16146.17	54	3.31	560
R1_6_9	**54**	**19372.96**	MB	21316.90	54.0	20860.58	54	10.03	276	20548.47	54.0	20375.70	54	6.07	494
R1_610	**54**	**18235.57**	MB	19909.33	54.0	19776.64	54	9.18	258	19193.80	54.0	18902.19	54	5.25	485
R2_6_1	**11**	**18325.6**	MB	19066.50	11.0	18865.57	11	4.04	872	18937.51	11.0	18837.28	11	3.34	1622
R2_6_2	11	15346.42	MB	15318.18	11.0	15222.07	11	1.65	928	15187.30	11.0	**15069.24**	**11**	0.78	1727
R2_6_3	11	11663.06	MB	11422.68	11.0	11395.17	11	1.16	1001	11386.17	11.0	**11291.52**	**11**	0.84	1903
R2_6_4	11	8386.64	MB	8331.34	11.0	8264.60	11	2.06	1115	8251.65	11.0	**8163.24**	**11**	1.08	2021
R2_6_5	11	15640.6	MB	15637.54	11.0	15430.80	11	1.42	862	15558.66	11.0	**15418.00**	**11**	0.91	1621
R2_6_6	11	12937.47	MB	13133.25	11.0	13038.58	11	1.52	920	13026.65	11.0	**12936.28**	**11**	0.70	1766
R2_6_7	11	10536.84	MB	10487.56	11.0	10437.39	11	2.12	1000	10352.03	11.0	**10269.96**	**11**	0.80	1904
R2_6_8	11	8023.64	MB	7886.24	11.0	7849.32	11	1.72	1095	7805.77	11.0	**7752.78**	**11**	0.68	2086
R2_6_9	**11**	**13567.84**	MB	14181.45	11.0	14016.38	11	4.52	876	14000.78	11.0	13885.52	11	3.19	1627
R2_610	11	12607.09	MB	12799.15	11.0	12775.18	11	1.83	881	12706.72	11.0	**12568.79**	**11**	1.10	1690
C1_6_1	**60**	**14095.64**	GH	14095.64	60.0	**14095.64**	**60**	0.00	286	14095.64	60.0	**14095.64**	**60**	0.00	540
C1_6_2	56	14325.96	MB	14446.21	56.0	14179.06	56	1.92	495	14278.31	56.0	**14174.12**	**56**	0.74	737
C1_6_3	56	13898.99	MB	13866.54	56.0	13842.83	56	0.46	509	13842.21	56.0	**13803.50**	**56**	0.28	767
C1_6_4	56	13610.66	MB	13626.16	56.0	13615.92	56	0.35	538	13603.40	56.0	**13578.66**	**56**	0.18	812
C1_6_5	**60**	**14085.7**	BVH	14085.72	60.0	14085.72	60	0.00	306	14085.72	60.0	14085.72	60	0.00	564
C1_6_6	60	14089.7	BVH	14089.66	60.0	**14089.66**	**60**	0.00	413	14089.66	60.0	**14089.66**	**60**	0.00	674
C1_6_7	59	14659.74	GH	14832.65	58.6	**15017.03**	**58**	-1.23	473	14803.08	58.6	15032.51	58	-1.42	726
C1_6_8	57	14976.88	GH	14690.74	57.0	14409.78	57	2.42	433	14510.17	57.0	**14343.05**	**57**	1.17	675
C1_6_9	**56**	**13733.56**	MB	14265.06	56.0	14017.73	56	3.87	483	13883.26	56.0	13767.45	56	1.09	723
C1_610	56	13758.19	MB	14128.71	56.0	13906.05	56	3.22	492	13788.90	56.0	**13688.57**	**56**	0.73	742
C2_6_1	**18**	**7774.1**	MB	7789.40	18.0	7780.84	18	0.20	553	7791.82	18.0	7786.86	18	0.23	987
C2_6_2	18	7486.88	MB	7764.29	17.8	8800.94	17	-11.76	727	7763.97	17.8	**8799.38**	**17**	-11.77	1224
C2_6_3	17	8371.07	GH	7676.89	17.6	7795.66	17	0.96	762	7613.00	17.6	**7604.00**	**17**	0.12	1275
C2_6_4	17	7216.45	MB	7269.90	17.2	7054.65	17	3.95	722	7088.64	17.2	**6993.77**	**17**	1.36	1266
C2_6_5	**18**	**7576.35**	MB	7694.89	18.0	7592.79	18	1.56	581	7606.34	18.0	7578.12	18	0.40	1007
C2_6_6	**18**	**7478.63**	MB	8515.65	18.0	7984.40	18	13.87	635	7910.69	18.0	7554.61	18	5.78	1088
C2_6_7	18	7560.53	MB	8474.41	18.0	**7520.34**	**18**	12.69	727	8234.69	18.0	7610.04	18	9.50	1190
C2_6_8	18	7352.42	MB	7771.07	17.8	**8696.15**	**17**	-10.64	672	7734.91	17.8	8782.31	17	-11.05	1159
C2_6_9	**18**	**7350.94**	MB	7609.44	18.0	7356.19	18	3.52	669	7384.14	18.0	7364.93	18	0.45	1148
C2_610	**17**	**7523.34**	MB	7781.30	17.6	8334.99	17	3.43	656	7697.89	17.6	7938.94	17	2.32	1136
RC1_6_1	**55**	**17454.39**	MB	18210.19	55.0	17987.59	55	4.33	275	17928.76	55.0	17751.33	55	2.72	494
RC1_6_2	**55**	**16208.24**	MB	16883.37	55.0	16718.63	55	4.17	436	16686.63	55.0	16548.43	55	2.95	671
RC1_6_3	55	15524.33	MB	15968.19	55.0	15907.78	55	3.03	333	15642.26	55.0	**15499.02**	**55**	0.92	584
RC1_6_4	55	15180.72	MB	15295.11	55.0	15214.81	55	1.47	358	15192.70	55.0	**15072.90**	**55**	0.79	621
RC1_6_5	55	17468.57	MB	17981.80	55.0	17879.49	55	3.34	329	17543.75	55.0	**17401.34**	**55**	0.82	551
RC1_6_6	**55**	**17248.87**	MB	17913.64	55.0	17646.26	55	3.85	329	17466.21	55.0	17355.10	55	1.26	548
RC1_6_7	**55**	**16454.79**	MB	17484.20	55.0	17159.31	55	6.26	410	17143.21	55.0	17058.40	55	4.18	636
RC1_6_8	**55**	**16462.49**	MB	17043.31	55.0	16955.52	55	3.53	336	16705.09	55.0	16510.65	55	1.47	568
RC1_6_9	**55**	**16153**	MB	16806.32	55.0	16609.24	55	4.04	378	16525.18	55.0	16435.71	55	2.30	604
RC1_610	**55**	**16030.86**	MB	16483.29	55.0	16388.47	55	2.82	265	16391.34	55.0	16316.51	55	2.25	498
RC2_6_1	15	13275.93	GH	13415.25	15.0	13314.03	15	1.92	1020	13322.77	15.0	**13163.03**	**15**	1.21	1573
RC2_6_2	12	12071.4	GH	11652.71	12.8	12039.89	12	-1.70	1250	11539.21	12.8	**11853.72**	**12**	-2.65	1970
RC2_6_3	11	9978.25	MB	10220.80	11.0	10032.99	11	3.62	1006	10066.43	11.0	**9863.35**	**11**	2.06	1889
RC2_6_4	11	7349.88	MB	7409.35	11.0	7344.31	11	2.46	1069	7274.39	11.0	**7231.64**	**11**	0.59	1993
RC2_6_5	13	11919.72	MB	12224.60	12.8	**12560.43**	**12**	-2.67	1286	12188.40	12.8	12612.91	12	-2.96	1954
RC2_6_6	12	10700.42	LC	12498.61	11.2	12464.98	11	1.76	1242	12405.28	11.2	**12282.52**	**11**	1.00	1963
RC2_6_7	11	11687.04	MB	11510.79	11.0	11347.57	11	4.15	927	11309.89	11.0	**11052.49**	**11**	2.33	1706
RC2_6_8	**11**	**10474.95**	MB	10744.43	11.0	10627.04	11	2.57	894	10617.44	11.0	10488.75	11	1.36	1658
RC2_6_9	11	10113.82	MB	10094.97	11.0	9982.66	11	2.15	895	10060.58	11.0	**9882.71**	**11**	1.80	1661
RC2_610	**11**	**9339.41**	MB	9611.45	11.0	9510.51	11	2.91	900	9500.07	11.0	9340.06	11	1.72	1660
Tot.	2076	798645		823814.02	2076.40	818863.38	2071		37731	811014.16	2076.40	807470.21	2071		65718
Avg.								2.83	629					1.28	1095
< PB						22						30			
#B		29				6						28			

Table 16: Gehring/Homberger VRPTW instances, 600 customers..

	Best known			ALNS 25K						ALNS 50K					
	veh.	cost	References	avg. sol.	avg. #veh.	best sol.	best #veh.	avg. gap (%)	avg. time (s)	avg. sol.	avg. #veh.	best sol.	best #veh.	avg. gap (%)	avg. time (s)
R1_8_1	**79**	**39612.2**	BVH	37859.00	80.0	37631.40	80	-4.43	684	37756.07	80.0	37492.04	80	-4.69	1010
R1_8_2	**72**	**33548.54**	MB	34705.63	72.0	34435.01	72	3.45	613	34273.25	72.0	33816.69	72	2.16	917
R1_8_3	**72**	**30151.9**	MB	31065.56	72.0	30746.68	72	3.03	645	30593.64	72.0	30317.49	72	1.47	969
R1_8_4	**72**	**26838.04**	MB	29002.34	72.0	28831.80	72	8.06	678	28672.14	72.0	28568.78	72	6.83	1025
R1_8_5	**72**	**34741.53**	MB	36198.65	72.0	36038.57	72	4.19	524	35739.41	72.0	35503.63	72	2.87	809
R1_8_6	**72**	**31737.47**	MB	32820.77	72.0	32757.13	72	3.41	610	32487.44	72.0	32360.07	72	2.36	913
R1_8_7	**72**	**29538.4**	MB	30493.16	72.0	30393.12	72	3.23	644	30089.26	72.0	29979.63	72	1.86	967
R1_8_8	72	28342.64	MB	28803.77	72.0	28622.63	72	1.63	676	28509.27	72.0	**28341.21**	**72**	0.59	1020
R1_8_9	72	34231.38	MB	34961.84	72.0	34856.18	72	2.17	576	34437.83	72.0	**34218.41**	**72**	0.64	864
R1_810	**72**	**31730.45**	MB	33144.45	72.0	32665.95	72	4.46	586	32729.29	72.0	32569.97	72	3.15	879
R2_8_1	**15**	**28440.28**	MB	29209.61	15.0	28923.27	15	2.71	951	29086.28	15.0	28822.48	15	2.27	1811
R2_8_2	15	23335.67	MB	23655.16	15.0	23524.65	15	1.64	1070	23492.15	15.0	**23274.22**	**15**	0.94	1964
R2_8_3	**15**	**17992.25**	MB	18188.06	15.0	18103.52	15	1.09	1127	18137.61	15.0	18078.82	15	0.81	2091
R2_8_4	15	13625.25	MB	13658.48	15.0	13584.57	15	1.82	1213	13525.52	15.0	**13413.79**	**15**	0.83	2322
R2_8_5	**15**	**24611.39**	MB	25479.70	15.0	25260.54	15	3.53	978	25255.01	15.0	25077.09	15	2.62	1803
R2_8_6	**15**	**20697.06**	MB	21104.29	15.0	20969.81	15	1.97	1032	21014.57	15.0	20973.12	15	1.53	1962
R2_8_7	15	17058.3	MB	17114.71	15.0	**16977.49**	**15**	0.81	1119	17128.01	15.0	16980.58	15	0.89	2134
R2_8_8	15	13053.31	MB	13187.89	15.0	13054.95	15	1.87	1254	13063.15	15.0	**12945.52**	**15**	0.91	2365
R2_8_9	**15**	**22588.02**	MB	23303.95	15.0	23138.51	15	3.17	982	23061.61	15.0	22877.21	15	2.10	1849
R2_810	15	21551.26	MB	21372.15	15.0	21240.42	15	1.33	979	21233.28	15.0	**21092.27**	**15**	0.67	1841
C1_8_1	**80**	**25030.36**	M	25184.38	80.0	25184.38	80	0.62	397	25184.38	80.0	25184.38	80	0.62	741
C1_8_2	75	25518.17	GH	25711.25	74.2	25667.72	74	0.68	664	25634.80	74.2	**25536.76**	**74**	0.38	993
C1_8_3	72	25438.6	BVH	25359.87	72.0	24756.97	72	2.96	373	24728.90	72.0	**24629.86**	**72**	0.40	682
C1_8_4	72	24040.47	MB	24256.32	72.0	24118.80	72	1.33	378	24005.77	72.0	**23938.33**	**72**	0.28	706
C1_8_5	80	25166.3	BVH	25166.28	80.0	**25166.28**	**80**	0.00	417	25166.28	80.0	**25166.28**	**80**	0.00	762
C1_8_6	80	25160.9	BVH	25162.17	80.0	**25160.85**	**80**	0.01	560	25162.21	80.0	**25160.85**	**80**	0.01	913
C1_8_7	79	25518.85	GH	25481.02	79.0	**25425.92**	**79**	0.22	623	25449.95	79.0	25428.67	79	0.09	972
C1_8_8	76	25379.85	MB	25740.77	75.2	25622.69	75	1.14	608	25538.76	75.2	**25450.99**	**75**	0.34	930
C1_8_9	73	24713.38	MB	26318.36	72.2	26169.29	72	2.26	575	25673.55	72.2	**25737.46**	**72**	-0.25	868
C1_810	72	29536.81	GH	27097.82	72.0	26382.98	72	5.45	473	26151.75	72.0	**25697.68**	**72**	1.77	770
C2_8_1	**24**	**11654.72**	MB	11678.08	24.0	11665.21	24	0.20	730	11672.47	24.0	11664.00	24	0.15	1238
C2_8_2	**24**	**11422.34**	MB	11456.70	24.0	11428.07	24	0.30	807	11440.98	24.0	11433.46	24	0.16	1397
C2_8_3	**23**	**11554.18**	MB	11312.58	24.0	11184.67	24	-2.09	839	11212.69	24.0	11188.30	24	-2.96	1468
C2_8_4	**23**	**10963.49**	MB	11511.87	23.2	11440.25	23	5.00	955	11180.00	23.2	10999.42	23	1.97	1627
C2_8_5	**24**	**11432.92**	MB	12110.19	24.0	11902.99	24	5.92	896	11565.06	24.0	11451.57	24	1.16	1441
C2_8_6	**24**	**11357.86**	MB	12282.80	24.4	12342.70	24	8.14	812	11909.95	24.2	11403.57	24	4.86	1360
C2_8_7	**24**	**11397.54**	MB	12058.86	24.6	11540.25	24	5.80	881	11871.66	24.4	11412.08	24	4.16	1443
C2_8_8	24	11206.32	MB	12728.62	23.8	13892.26	23	-8.28	860	12371.35	23.8	**13878.40**	**23**	-10.86	1414
C2_8_9	**24**	**11249**	MB	13015.41	24.0	12358.05	24	15.70	897	12446.59	24.0	11650.10	24	10.65	1469
C2_810	**23**	**11284.46**	MB	11837.70	23.8	12103.56	23	4.90	786	11746.59	23.8	12173.74	23	4.10	1358
RC1_8_1	73	31590.23	MB	31990.65	73.0	31851.54	73	2.29	438	31396.64	73.0	**31275.38**	**73**	0.39	720
RC1_8_2	**72**	**39696.2**	GH	29762.99	73.0	29537.14	73	-25.02	608	29377.34	73.0	29172.08	73	-25.99	912
RC1_8_3	**72**	**35577.87**	GH	28634.08	73.0	28466.83	73	-19.52	646	28301.03	73.0	28164.66	73	-20.45	970
RC1_8_4	**72**	**32654.1**	GH	27481.23	73.0	27393.06	73	-15.84	685	27303.22	73.0	27201.39	73	-16.39	1029
RC1_8_5	**73**	**30454.15**	MB	31228.63	73.0	31067.35	73	2.54	578	30742.88	73.0	30548.23	73	0.95	865
RC1_8_6	**73**	**29674.68**	MB	31019.63	73.0	30863.25	73	4.53	573	30749.36	73.0	30511.07	73	3.62	858
RC1_8_7	**72**	**43829.43**	GH	30600.17	73.0	30455.56	73	-30.18	575	30135.52	73.0	30007.82	73	-31.24	868
RC1_8_8	**72**	**43694.6**	GH	30006.93	73.0	29820.15	73	-31.33	580	29603.68	73.0	29547.96	73	-32.25	872
RC1_8_9	**72**	**41816.7**	GH	29918.07	73.0	29812.35	73	-28.45	581	29493.38	73.0	29360.93	73	-29.47	871
RC1_810	**72**	**41182.44**	GH	29518.16	73.0	29373.39	73	-28.32	586	29147.32	73.0	28993.52	73	-29.22	884
RC2_8_1	20	19989.12	MB	20734.14	19.8	21005.11	19	-1.05	1385	20605.53	19.8	**20954.95**	**19**	-1.67	2046
RC2_8_2	17	18099.68	MB	18369.12	17.0	18184.31	17	1.86	1728	18208.97	17.0	**18032.89**	**17**	0.98	2585
RC2_8_3	15	15116.26	MB	15033.15	15.0	**14800.78**	**15**	1.57	1212	14920.27	15.0	14810.81	15	0.81	2172
RC2_8_4	15	11392.25	MB	11592.05	15.0	11402.27	15	1.97	1196	11440.47	15.0	**11368.19**	**15**	0.64	2263
RC2_8_5	**16**	**19105.75**	MB	19293.34	16.4	19214.57	16	0.98	1529	19181.34	16.4	19180.13	16	0.40	2328
RC2_8_6	**15**	**18882.3**	MB	19560.50	15.0	19173.09	15	3.59	1063	19210.08	15.0	19075.89	15	1.74	1918
RC2_8_7	15	17461.44	MB	17798.95	15.0	17519.63	15	2.71	990	17643.28	15.0	**17329.32**	**15**	1.81	1858
RC2_8_8	15	16529.24	MB	16756.53	15.0	16485.06	15	3.26	994	16368.61	15.0	**16226.78**	**15**	0.87	1859
RC2_8_9	15	15823.5	MB	16071.87	15.0	15979.71	15	2.45	989	15902.97	15.0	**15687.20**	**15**	1.38	1825
RC2_810	**15**	**14892.29**	MB	15013.68	15.0	14944.14	15	0.82	1001	15048.01	15.0	14953.29	15	1.05	1873
Tot.	2754	1429914		1381184.08	2762.60	1372619.40	2758		48411	1365178.33	2762.20	1358291.43	2758		81648
Avg.								-0.86	807					-2.07	1361
< PB						15						25			
#B		35				5						22			

Table 17: Gehring/Homberger VRPTW instances, 800 customers..

	Best known			ALNS 25K						ALNS 50K					
	veh.	cost	References	avg. sol.	avg. #veh.	best sol.	best #veh.	avg. gap (%)	avg. time (s)	avg. sol.	avg. #veh.	best sol.	best #veh.	avg. gap (%)	avg. time (s)
R110_1	**100**	**54145.31**	MB	55493.78	100.0	55108.89	100	2.49	825	55029.87	100.0	54720.19	100	1.63	1229
R110_2	91	56367.45	GH	54167.93	91.6	57478.64	91	-2.27	698	52844.31	91.6	**55428.79**	**91**	-4.66	1066
R110_3	**91**	**46621.19**	MB	52196.91	91.0	51840.30	91	11.96	435	50296.23	91.0	49634.84	91	7.88	807
R110_4	**91**	**43461.84**	MB	46878.36	91.0	46645.90	91	7.86	435	45626.47	91.0	45303.47	91	4.98	829
R110_5	**91**	**70838.01**	GH	54671.65	92.0	54270.08	92	-22.82	705	53259.01	92.0	53089.15	92	-24.82	1061
R110_6	**91**	**49059.8**	MB	54109.35	91.4	55826.83	91	10.29	541	52485.80	91.4	54555.32	91	6.98	905
R110_7	**91**	**45847.84**	MB	50656.58	91.0	49880.51	91	10.49	429	48869.74	91.0	48141.47	91	6.59	801
R110_8	**91**	**42767.77**	MB	46752.56	91.0	46512.13	91	9.32	452	45286.98	91.0	44853.70	91	5.89	847
R110_9	**91**	**51391.8**	MB	53216.59	92.0	53163.89	92	3.55	706	52139.44	92.0	52015.72	92	1.45	1067
R11010	**91**	**49348.36**	MB	50861.54	92.0	50592.40	92	3.07	674	50007.62	92.0	49769.85	92	1.34	1038
R210_1	**19**	**42922.56**	BSJ	44524.99	19.0	44213.65	19	3.73	1192	43904.40	19.0	43264.68	19	2.29	2083
R210_2	19	34918.49	BSJ	34969.52	19.0	34698.44	19	1.60	1800	34564.64	19.0	**34417.47**	**19**	0.43	2795
R210_3	19	25689.62	BSJ	26067.93	19.0	25964.09	19	2.63	2088	25807.15	19.0	**25400.16**	**19**	1.60	3230
R210_4	19	18858.24	BSJ	18594.33	19.0	18425.77	19	1.43	2289	18477.00	19.0	**18332.77**	**19**	0.79	3480
R210_5	**19**	**37265.32**	BSJ	38149.38	19.0	37773.72	19	2.37	1328	37833.57	19.0	37746.01	19	1.52	2247
R210_6	**19**	**30725.2**	BSJ	31253.33	19.0	30975.00	19	1.72	1453	31007.89	19.0	30778.85	19	0.92	2500
R210_7	19	24363.83	BSJ	24340.48	19.0	24243.39	19	1.45	1923	24228.74	19.0	**23991.71**	**19**	0.99	3009
R210_8	19	18185.38	BSJ	18361.52	19.0	18139.74	19	2.90	2313	18037.86	19.0	**17844.36**	**19**	1.08	3540
R210_9	**19**	**33777.76**	BSJ	35005.84	19.0	34872.05	19	3.64	1345	34496.05	19.0	34349.70	19	2.13	2265
R21010	**19**	**31599.84**	BSJ	32006.08	19.0	31782.57	19	1.29	1645	31803.51	19.0	31682.52	19	0.64	2586
C110_1	**100**	**42478.95**	GH	42478.95	100.0	**42478.95**	**100**	0.00	499	42478.95	100.0	**42478.95**	**100**	0.00	915
C110_2	92	42920.7	BVH	42339.69	91.6	42667.84	91	0.21	798	42222.18	91.6	**42249.60**	**91**	-0.06	1188
C110_3	90	40934.87	MB	41395.50	90.0	40915.89	90	2.52	506	40904.59	90.0	**40376.43**	**90**	1.31	884
C110_4	90	40410.58	MB	40681.78	90.0	40441.12	90	1.76	515	40222.27	90.0	**39980.07**	**90**	0.61	902
C110_5	100	42469.2	BVH	42469.50	100.0	**42469.18**	**100**	0.00	542	42469.18	100.0	**42469.18**	**100**	0.00	968
C110_6	100	42471.3	BVH	42472.69	100.0	**42471.29**	**100**	0.00	678	42471.57	100.0	**42471.29**	**100**	0.00	1103
C110_7	99	42711.39	GH	42726.27	99.0	**42673.51**	**99**	0.12	739	42708.94	99.0	42688.64	99	0.08	1159
C110_8	96	42170.31	MB	42641.48	95.4	42402.12	95	0.67	757	42539.98	95.4	**42359.27**	**95**	0.43	1150
C110_9	91	45386.93	GH	42048.67	91.2	41586.54	91	1.37	645	41774.68	91.2	**41482.00**	**91**	0.71	1005
C11010	**90**	**40894.38**	MB	43409.67	90.0	43132.22	90	6.15	612	42554.17	90.0	42214.60	90	4.06	962
C210_1	**30**	**16879.24**	LL	16905.00	30.0	**16879.24**	**30**	0.15	888	16893.15	30.0	**16879.24**	**30**	0.08	1514
C210_2	**29**	**17228.82**	MB	17446.99	29.4	17677.61	29	1.27	1066	17314.77	29.4	17563.06	29	0.50	1719
C210_3	**29**	**16367.59**	MB	16938.59	30.0	16253.60	30	3.49	971	16446.58	30.0	16109.71	30	0.48	1690
C210_4	29	17153.19	MB	16845.74	29.0	16712.08	29	5.21	1151	16063.32	29.0	**16011.30**	**29**	0.32	1905
C210_5	**30**	**16586.46**	GH	17613.87	30.6	16825.34	30	6.19	964	16888.66	30.4	16596.69	30	1.82	1575
C210_6	30	16371.65	MB	17393.97	30.4	17596.06	30	6.26	1070	16696.06	30.2	**16369.10**	**30**	2.00	1697
C210_7	**31**	**16578.42**	MB	17348.99	31.0	16878.12	31	4.65	978	17057.54	31.0	16590.48	31	2.89	1617
C210_8	**29**	**17219.59**	LC	18921.39	29.6	19122.58	29	9.88	1047	17790.97	29.6	18407.27	29	3.32	1700
C210_9	30	16651.96	MB	17626.12	30.0	16679.15	30	8.17	1104	16999.89	30.0	**16294.72**	**30**	4.33	1771
C21010	**29**	**16178.26**	MB	18856.35	29.0	18447.85	29	16.55	1103	18375.30	29.0	17582.15	29	13.58	1759
RC110_1	**90**	**47143.9**	MB	51246.49	90.0	50976.00	90	8.70	517	49693.36	90.0	48933.68	90	5.41	863
RC110_2	**90**	**44906.58**	MB	47283.88	90.0	46913.77	90	5.29	539	46647.41	90.0	46165.33	90	3.88	904
RC110_3	**90**	**43782.57**	MB	45167.52	90.0	44833.81	90	3.16	562	44408.40	90.0	44014.81	90	1.43	938
RC110_4	**90**	**41917.14**	MB	43355.81	90.0	43144.87	90	3.43	668	42844.52	90.0	42607.34	90	2.21	1071
RC110_5	**90**	**47632.31**	MB	50533.91	90.0	50226.31	90	6.09	431	49082.31	90.0	48934.53	90	3.04	772
RC110_6	**90**	**46391.6**	MB	50436.65	90.0	49703.43	90	8.72	402	49131.04	90.0	48766.98	90	5.91	745
RC110_7	**90**	**46157.71**	MB	49716.92	90.0	49238.95	90	7.71	460	48308.95	90.0	48005.94	90	4.66	806
RC110_8	**90**	**45585.08**	MB	48391.77	90.0	47670.50	90	6.16	396	47416.90	90.0	47122.61	90	4.02	743
RC110_9	**90**	**45405.54**	MB	48343.65	90.0	47930.01	90	6.47	513	46998.60	90.0	46889.79	90	3.51	864
RC11010	**90**	**45041.64**	MB	47210.76	90.0	46716.69	90	4.82	466	46284.90	90.0	46080.51	90	2.76	822
RC210_1	22	30320.41	BSJ	30930.47	21.2	30478.44	21	1.76	1429	30618.08	21.2	**30396.13**	**21**	0.73	2316
RC210_2	19	26592.4	BSJ	26301.14	19.4	**27552.05**	**18**	-4.54	1955	26412.31	19.4	27681.62	18	-4.14	2953
RC210_3	**18**	**20588.38**	BSJ	21313.73	18.0	20983.66	18	3.52	1324	21060.93	18.0	20811.18	18	2.30	2443
RC210_4	18	16480.17	BSJ	16617.79	18.0	16254.55	18	3.81	1345	16499.16	18.0	**16007.59**	**18**	3.07	2544
RC210_5	18	29352.08	LC	29008.22	18.0	28647.57	18	2.26	1249	28610.45	18.0	**28368.48**	**18**	0.85	2198
RC210_6	**18**	**27003.3**	MB	29267.17	18.0	28825.98	18	8.38	1136	29005.97	18.0	28746.61	18	7.42	2035
RC210_7	**18**	**26161.91**	BSJ	27503.47	18.0	27110.84	18	5.13	1106	26958.52	18.0	26765.43	18	3.04	2067
RC210_8	18	24995	BSJ	25445.17	18.0	25211.63	18	1.94	1103	25128.20	18.0	**24961.29**	**18**	0.67	2097
RC210_9	**18**	**23582.89**	MB	24729.65	18.0	24420.99	18	4.86	1129	24417.63	18.0	24113.72	18	3.54	2079
RC21010	**18**	**22481.03**	BSJ	23544.52	18.0	23193.63	18	4.73	1101	23143.27	18.0	23056.75	18	2.95	2066
Tot.	3438	2099741		2157188.54	3443.80	2146751.97	3438		57741	2119549.93	3443.40	2110924.81	3438		95895
Avg.								3.73	962					1.89	1598
< PB						16						22			
#B		38				6						22			

Table 18: Gehring/Homberger VRPTW instances, 1000 customers..

	Optimal		ALNS 25K			
	cost	ref	avg. sol.	best sol.	avg. gap (%)	avg. time (s)
R101***25	**617.1**	KDMSS99	617.1	**617.1**	0.00	3
R102***25	**547.1**	KDMSS99	547.1	**547.1**	0.00	3
R103***25	**454.6**	KDMSS99	454.6	**454.6**	0.00	4
R104***25	**416.9**	KDMSS99	416.9	**416.9**	0.00	4
R105***25	**530.5**	KDMSS99	530.5	**530.5**	0.00	3
R106***25	**465.4**	KDMSS99	465.4	**465.4**	0.00	3
R107***25	**424.3**	KDMSS99	424.3	**424.3**	0.00	4
R108***25	**397.3**	KDMSS99	397.3	**397.3**	0.00	4
R109***25	**441.3**	KDMSS99	441.3	**441.3**	0.00	3
R110***25	**444.1**	KDMSS99	444.1	**444.1**	0.00	4
R111***25	**428.8**	KDMSS99	428.8	**428.8**	0.00	4
R112***25	**393**	KDMSS99	393.0	**393.0**	0.00	4
C101***25	**191.3**	KDMSS99	191.3	**191.3**	0.00	4
C102***25	**190.3**	KDMSS99	190.3	**190.3**	0.00	4
C103***25	**190.3**	KDMSS99	190.3	**190.3**	0.00	4
C104***25	**186.9**	KDMSS99	186.9	**186.9**	0.00	4
C105***25	**191.3**	KDMSS99	191.3	**191.3**	0.00	4
C106***25	**191.3**	KDMSS99	191.3	**191.3**	0.00	4
C107***25	**191.3**	KDMSS99	191.3	**191.3**	0.00	4
C108***25	**191.3**	KDMSS99	191.3	**191.3**	0.00	5
C109***25	**191.3**	KDMSS99	191.3	**191.3**	0.00	4
RC101***25	**461.1**	KDMSS99	461.1	**461.1**	0.00	4
RC102***25	**351.8**	KDMSS99	351.8	**351.8**	0.00	4
RC103***25	**332.8**	KDMSS99	332.8	**332.8**	0.00	4
RC104***25	**306.6**	KDMSS99	306.6	**306.6**	0.00	4
RC105***25	**411.3**	KDMSS99	411.3	**411.3**	0.00	4
RC106***25	**345.5**	KDMSS99	345.5	**345.5**	0.00	4
RC107***25	**298.3**	KDMSS99	298.3	**298.3**	0.00	4
RC108***25	**294.5**	KDMSS99	294.5	**294.5**	0.00	4
R201***25	**463.3**	L99	463.3	**463.3**	0.00	4
R202***25	**410.5**	L99	410.5	**410.5**	0.00	4
R203***25	**391.4**	L99	391.4	**391.4**	0.00	4
R204***25	**355**	C03	355.2	**355.0**	0.06	6
R205***25	**393**	L99	393.0	**393.0**	0.00	5
R206***25	**374.4**	CR99	374.4	**374.4**	0.00	5
R207***25	**361.6**	KLM01	361.6	**361.6**	0.00	5
R208***25	**328.2**	FDGG04	328.2	**328.2**	0.00	11
R209***25	**370.7**	KLM01	371.5	**370.7**	0.21	6
R210***25	**404.6**	CR99	404.6	**404.6**	0.00	5
R211***25	**350.9**	KLM01	350.9	**350.9**	0.00	6
C201***25	**214.7**	L99	214.7	**214.7**	0.00	7
C202***25	**214.7**	L99	214.7	**214.7**	0.00	8
C203***25	**214.7**	L99	214.7	**214.7**	0.00	8
C204***25	**213.1**	CR99	214.4	**213.1**	0.59	8
C205***25	**214.7**	L99	214.7	**214.7**	0.00	8
C206***25	**214.7**	L99	214.7	**214.7**	0.00	7
C207***25	**214.5**	L99	214.5	**214.5**	0.00	9
C208***25	**214.5**	L99	214.5	**214.5**	0.00	7
RC201***25	**360.2**	L99	360.2	**360.2**	0.00	4
RC202***25	**338**	CR99	338.0	**338.0**	0.00	4
RC203***25	**326.9**	FDGG04	326.9	**326.9**	0.00	4
RC204***25	**299.7**	FDGG04	299.7	**299.7**	0.00	5
RC205***25	**338**	L99	338.0	**338.0**	0.00	4
RC206***25	**324**	KLM01	324.0	**324.0**	0.00	4
RC207***25	**298.3**	KLM01	298.3	**298.3**	0.00	5
RC208***25	**269.1**	C03	269.1	**269.1**	0.00	6
Tot.	18551.0		18553.2	18551.0		276
Avg.					0.02	5
< PB				0		
#B		56		56		

Table 19: Solomon VRPTW instances with 25 customers, comparison to exact solutions (distances and travel times are truncated to one decimal and traveled distance is minimized). The table should be read as the preceding tables. The abbreviations in the *ref* column refers to the following papers: *C03* – Chabrier [10], *CR99* – Cook and Rich [14], *DP03* – Danna and Le Pape [20], *FDGG04* – Feillet [24], *IV03* – Irnich and Villeneuve [34], *KLM01* – Kallehauge et al. [35], *KDMSS99* – Kohl et al. [36] and *L99* – Larsen [39].

	Optimal		ALNS 25K			
	cost	ref	avg. sol.	best sol.	avg. gap (%)	avg. time (s)
R101***50	**1044**	KDMSS99	1044.0	**1044.0**	0.00	9
R102***50	**909**	KDMSS99	909.0	**909.0**	0.00	10
R103***50	**772.9**	KDMSS99	772.9	**772.9**	0.00	10
R104***50	**625.4**	KDMSS99	626.1	**625.4**	0.12	11
R105***50	**899.3**	KDMSS99	899.8	**899.3**	0.05	9
R106***50	**793**	KDMSS99	793.0	**793.0**	0.00	10
R107***50	**711.1**	KDMSS99	711.1	**711.1**	0.00	10
R108***50	**617.7**	CR99	617.7	**617.7**	0.00	11
R109***50	**786.8**	KDMSS99	786.8	**786.8**	0.00	10
R110***50	**697**	KDMSS99	697.0	**697.0**	0.00	10
R111***50	**707.2**	L99	707.2	**707.2**	0.00	10
R112***50	**630.2**	L99	635.1	635.0	0.77	11
C101***50	**362.4**	KDMSS99	362.4	**362.4**	0.00	9
C102***50	**361.4**	KDMSS99	361.4	**361.4**	0.00	11
C103***50	**361.4**	KDMSS99	361.4	**361.4**	0.00	11
C104***50	**358**	KDMSS99	358.0	**358.0**	0.00	12
C105***50	**362.4**	KDMSS99	362.4	**362.4**	0.00	10
C106***50	**362.4**	KDMSS99	362.4	**362.4**	0.00	10
C107***50	**362.4**	KDMSS99	362.4	**362.4**	0.00	10
C108***50	**362.4**	KDMSS99	362.4	**362.4**	0.00	11
C109***50	**362.4**	KDMSS99	362.4	**362.4**	0.00	12
RC101***50	**944**	KDMSS99	944.0	**944.0**	0.00	9
RC102***50	**822.5**	KDMSS99	822.8	**822.5**	0.04	10
RC103***50	**710.9**	KDMSS99	710.9	**710.9**	0.00	10
RC104***50	**545.8**	KDMSS99	545.8	**545.8**	0.00	10
RC105***50	**855.3**	KDMSS99	855.3	**855.3**	0.00	10
RC106***50	**723.2**	KDMSS99	723.2	**723.2**	0.00	9
RC107***50	**642.7**	KDMSS99	643.7	**642.7**	0.16	10
RC108***50	**598.1**	KDMSS99	598.1	**598.1**	0.00	10
R201***50	**791.9**	L99	795.8	**791.9**	0.49	13
R202***50	**698.5**	L99	698.5	**698.5**	0.00	14
R203***50	**605.3**	C03	608.2	605.9	0.49	15
R204***50	**506.4**	IV03	506.4	**506.4**	0.00	24
R205***50	**690.1**	C03	698.2	696.7	1.17	15
R206***50	**632.4**	C03	634.0	**632.4**	0.25	16
R207***50	-	-	576.1	**576.1**	0.01	22
R208***50	-	-	489.6	**487.7**	0.39	29
R209***50	**600.6**	C03	602.5	**600.6**	0.32	15
R210***50	**645.6**	C03	648.3	**645.6**	0.42	16
R211***50	**535.5**	IV03	549.8	543.3	2.67	25
C201***50	**360.2**	L99	360.2	**360.2**	0.00	25
C202***50	**360.2**	CR99	360.2	**360.2**	0.00	27
C203***50	**359.8**	CR99	359.8	**359.8**	0.00	27
C204***50	**350.1**	KLM01	350.1	**350.1**	0.00	29
C205***50	**359.8**	CR99	359.8	**359.8**	0.00	30
C206***50	**359.8**	CR99	359.8	**359.8**	0.00	26
C207***50	**359.6**	CR99	359.6	**359.6**	0.00	27
C208***50	**350.5**	CR99	350.5	**350.5**	0.00	28
RC201***50	**684.4**	L99	684.8	684.8	0.06	12
RC202***50	**613.6**	FDGG04	613.6	**613.6**	0.00	12
RC203***50	**555.3**	C03	555.3	**555.3**	0.00	15
RC204***50	**444.2**	DP03	444.2	**444.2**	0.00	19
RC205***50	**630.2**	FDGG04	630.2	**630.2**	0.00	12
RC206***50	**610**	FDGG04	610.3	**610.0**	0.05	13
RC207***50	**558.6**	FDGG04	558.6	**558.6**	0.00	16
RC208***50	-	-	497.9	**481.8**	3.33	24
Tot.			32560.9	32519.7		841
Avg.					0.19	15

Table 20: Solomon VRPTW instances with 50 customers, comparison to exact solutions .

	Optimal		ALNS 25K			
	cost	ref	avg. sol.	best sol.	avg. gap (%)	avg. time (s)
R101	**1637.7**	KDMSS99	1638.6	**1637.7**	0.05	30
R102	**1466.6**	KDMSS99	1467.7	1467.6	0.08	33
R103	**1208.7**	CR99	1208.9	**1208.7**	0.01	34
R104	**971.5**	IV03	977.1	976.0	0.58	34
R105	**1355.3**	KDMSS99	1355.8	**1355.3**	0.03	31
R106	**1234.6**	L99	1234.6	**1234.6**	0.00	33
R107	**1064.6**	L99	1068.2	**1064.6**	0.34	33
R108	-	-	943.5	**933.7**	1.05	36
R109	**1146.9**	CR99	1150.2	**1146.9**	0.29	31
R110	**1068**	CR99	1083.1	1075.6	1.41	33
R111	**1048.7**	CR99	1049.2	**1048.7**	0.05	33
R112	-	-	952.2	**948.6**	0.38	35
C101	**827.3**	KDMSS99	827.3	**827.3**	0.00	29
C102	**827.3**	KDMSS99	827.3	**827.3**	0.00	32
C103	**826.3**	KDMSS99	826.3	**826.3**	0.00	34
C104	**822.9**	KDMSS99	822.9	**822.9**	0.00	36
C105	**827.3**	KDMSS99	827.3	**827.3**	0.00	30
C106	**827.3**	KDMSS99	827.3	**827.3**	0.00	31
C107	**827.3**	KDMSS99	827.3	**827.3**	0.00	31
C108	**827.3**	KDMSS99	827.3	**827.3**	0.00	32
C109	**827.3**	KDMSS99	827.3	**827.3**	0.00	34
RC101	**1619.8**	KDMSS99	1629.8	**1619.8**	0.61	28
RC102	**1457.4**	CR99	1475.1	1463.5	1.22	30
RC103	**1258**	CR99	1272.2	1267.0	1.13	31
RC104	-	-	1132.8	**1132.6**	0.01	33
RC105	**1513.7**	KDMSS99	1514.2	1513.8	0.04	30
RC106	-	-	1376.1	**1373.9**	0.16	29
RC107	**1207.8**	IV03	1213.0	1209.3	0.43	30
RC108	**1114.2**	IV03	1124.6	**1114.2**	0.94	31
R201	**1143.2**	KLM01	1153.9	1148.5	0.94	45
R202	-	-	1041.0	**1036.9**	0.40	54
R203	-	-	876.5	**872.4**	0.47	60
R204	-	-	731.5	**731.3**	0.03	67
R205	-	-	952.4	**949.8**	0.27	58
R206	-	-	880.6	**880.6**	0.00	61
R207	-	-	796.4	**794.0**	0.30	72
R208	-	-	703.1	**701.2**	0.27	86
R209	-	-	860.2	**855.8**	0.52	60
R210	-	-	914.0	**908.4**	0.61	59
R211	-	-	758.3	**752.3**	0.80	67
C201	**589.1**	CR99	589.1	**589.1**	0.00	69
C202	**589.1**	CR99	589.1	**589.1**	0.00	74
C203	**588.7**	KLM01	588.7	**588.7**	0.00	80
C204	**588.1**	IV03	588.1	**588.1**	0.00	84
C205	**586.4**	CR99	586.4	**586.4**	0.00	76
C206	**586**	CR99	586.0	**586.0**	0.00	72
C207	**585.8**	CR99	585.8	**585.8**	0.00	74
C208	**585.8**	KLM01	585.8	**585.8**	0.00	74
RC201	**1261.8**	KLM01	1272.3	1262.6	0.84	42
RC202	**1092.3**	C03	1097.4	1095.8	0.47	46
RC203	-	-	937.6	**923.7**	1.50	56
RC204	-	-	788.1	**785.8**	0.29	68
RC205	**1154**	C03	1154.0	**1154.0**	0.00	45
RC206	-	-	1062.5	**1051.1**	1.08	52
RC207	-	-	976.2	**966.6**	0.99	55
RC208	-	-	790.5	**777.3**	1.70	65
Tot.			54752.7	54579.5		2649
Avg.					0.36	47

Table 21: Solomon VRPTW instances with 100 customers, comparison to exact solutions .

	Best known			ALNS 25K					ALNS 50K				
	n	type	cost	avg. sol.	best sol.	avg. gap (%)	best above B.K	avg. time (s)	avg. sol.	best sol.	avg. gap (%)	best above B.K	avg. time (s)
P01	50	C	**524.61**	524.61	**524.61**	0.00	0.00	12	524.61	**524.61**	0.00	0.00	21
P02	75	C	**835.26**	841.81	838.87	0.78	0.43	20	839.62	**835.26**	0.52	0.00	38
P03	100	C	**826.14**	828.18	**826.14**	0.25	0.00	46	826.99	**826.14**	0.10	0.00	85
P04	150	C	**1028.42**	1037.43	1031.23	0.88	0.27	96	1034.20	1029.56	0.56	0.11	176
P05	199	C	**1291.29**	1309.36	1298.92	1.40	0.59	124	1306.63	1297.12	1.19	0.45	233
P06	50	CD	**555.43**	555.43	**555.43**	0.00	0.00	12	555.43	**555.43**	0.00	0.00	21
P07	75	CD	**909.68**	913.03	**909.68**	0.37	0.00	19	911.78	**909.68**	0.23	0.00	36
P08	100	CD	**865.94**	867.65	**865.94**	0.20	0.00	42	866.97	**865.94**	0.12	0.00	78
P09	150	CD	**1162.55**	1169.06	1164.24	0.56	0.15	86	1167.68	1163.68	0.44	0.10	160
P10	199	CD	**1395.85**	1408.19	1404.17	0.88	0.60	116	1410.27	1405.88	1.03	0.72	219
P11	120	C	**1042.11**	1042.37	1042.12	0.03	0.00	73	1042.46	1042.12	0.03	0.00	132
P12	100	C	**819.56**	819.56	**819.56**	0.00	0.00	43	819.56	**819.56**	0.00	0.00	79
P13	120	CD	**1541.14**	1543.77	1542.86	0.17	0.11	61	1543.54	1542.86	0.16	0.11	113
P14	150	CD	**866.37**	866.37	**866.37**	0.00	0.00	40	866.37	**866.37**	0.00	0.00	73
Tot.			13664	13726.83	13690.13			789	13716.09	13684.21			1464
Avg.						0.39	0.15	56			0.31	0.11	105
< PB					0					0			
#B		14			7					8			

Table 22: Christofides et al. CVRP problems [13]. The column *type* indicates if the problem is capacity constrained (*C*) or both capacity and duration constrained (*CD*). The column *best above B.K* indicates how much the best solution found differs from the best known solution from the literature (in percent). The best known solutions where obtained from Cordeau et al. [16].

	Best known				ALNS 25K					ALNS 50K				
	n	type	cost	ref	avg. sol.	best sol.	avg. gap (%)	best above B.K	avg. time (s)	avg. sol.	best sol.	avg. gap (%)	best above B.K	avg. time (s)
KELLY01	240	C	**5627.54**	MB	5667.04	5660.88	0.70	0.59	193	5662.57	5650.91	0.62	0.42	393
KELLY02	320	C	**8447.92**	MB	8499.27	8478.73	0.61	0.36	321	8487.94	8469.32	0.47	0.25	672
KELLY03	400	C	**11036.22**	MB	11067.48	11045.81	0.28	0.09	482	11052.72	11047.01	0.15	0.10	1015
KELLY04	480	C	**13624.52**	MB	13752.13	13635.31	0.94	0.08	654	13748.50	13635.31	0.91	0.08	1328
KELLY05	200	C	**6460.98**	TK	6479.80	6478.09	0.29	0.26	306	6482.49	6466.68	0.33	0.09	629
KELLY06	280	C	**8412.8**	MB	8507.29	8415.67	1.12	0.03	418	8543.30	8416.13	1.55	0.04	876
KELLY07	360	C	10195.56	MB	10273.80	10231.34	0.90	0.35	464	10265.15	**10181.75**	0.82	-0.14	941
KELLY08	440	C	**11663.55**	MB	11804.08	11721.35	1.20	0.50	499	11766.07	11713.62	0.88	0.43	1011
KELLY09	255	CD	**583.39**	MB	591.75	584.48	1.43	0.19	225	590.33	585.14	1.19	0.30	437
KELLY10	323	CD	**742.03**	MB	753.48	749.47	1.54	1.00	305	751.36	748.89	1.26	0.92	616
KELLY11	399	CD	**918.45**	MB	933.42	926.63	1.63	0.89	371	926.57	922.70	0.88	0.46	761
KELLY12	483	CD	**1107.19**	MB	1129.53	1125.11	2.02	1.62	458	1125.22	1119.06	1.63	1.07	911
KELLY13	252	CD	**859.11**	MB	878.22	876.01	2.22	1.97	142	874.24	864.68	1.76	0.65	285
KELLY14	320	CD	**1081.31**	MB	1107.97	1096.92	2.47	1.44	194	1103.53	1095.40	2.06	1.30	393
KELLY15	396	CD	**1345.23**	MB	1370.94	1355.91	1.91	0.79	236	1366.23	1359.94	1.56	1.09	468
KELLY16	480	CD	**1622.69**	MB	1652.00	1639.81	1.81	1.05	279	1645.67	1639.11	1.42	1.01	549
KELLY17	240	CD	**707.79**	MB	713.76	710.36	0.84	0.36	153	710.59	708.90	0.39	0.16	304
KELLY18	300	CD	**998.73**	MB	1008.11	1003.20	0.94	0.45	196	1007.84	1002.42	0.91	0.37	387
KELLY19	360	CD	**1366.86**	MB	1380.49	1377.52	1.00	0.78	227	1377.88	1374.24	0.81	0.54	449
KELLY20	420	CD	**1821.15**	MB	1843.08	1828.35	1.20	0.40	245	1834.70	1830.80	0.74	0.53	488
Tot.			88623		89413.66	88940.91			6369	89322.91	88832.02			12914
Avg.							1.25	0.66	318			1.02	0.48	646
< PB						0					1			
#B		19				0					1			

Table 23: Golden et al. CVRP problems ([30]). The best known solutions where obtained from Cordeau et al. [16], *MB* refers to the heuristic by Mester and Bräysy [42] (the results are not given in [42], but can be found in [16]), *TK* refers to the heuristic by Tarantilis and Kiranoudis [56].

	Best known			ALNS 25K					ALNS 50K				
	n	cost	ref	avg. sol.	best sol.	avg. gap (%)	best above B.K	avg. time (s)	avg. sol.	best sol.	avg. gap (%)	best above B.K	avg. time (s)
CVRP_L_21	560	**16212.83**	EST	16488.67	16296.21	1.70	0.51	869	16391.23	16224.81	1.10	0.07	1735
CVRP_L_22	600	14641.64	ORTR	14737.97	14638.37	0.73	-0.02	569	14644.06	**14631.08**	0.09	-0.07	1168
CVRP_L_23	640	**18801.13**	EST	19155.50	18925.36	1.88	0.66	1097	19112.56	18837.49	1.66	0.19	2268
CVRP_L_24	720	**21389.43**	EST	22024.22	21652.78	2.97	1.23	1259	21913.83	21522.48	2.45	0.62	2739
CVRP_L_25	760	17053.26	EST	17170.49	17082.81	1.59	0.17	650	17115.78	**16902.16**	1.26	-0.89	1320
CVRP_L_26	800	**23977.74**	EST	24577.43	24084.92	2.50	0.45	1425	24405.05	24014.09	1.78	0.15	3081
CVRP_L_27	840	17651.6	ORTR	17833.67	17749.35	1.25	0.55	723	17769.75	**17613.22**	0.89	-0.22	1504
CVRP_L_28	880	**26566.04**	EST	27315.94	26651.15	2.82	0.32	1692	27172.63	26791.72	2.28	0.85	3441
CVRP_L_29	960	**29154.34**	EST	30117.04	29487.26	3.30	1.14	1887	29976.86	29405.60	2.82	0.86	3921
CVRP_L_30	1040	**31742.64**	EST	32828.86	32133.28	3.42	1.23	2192	32607.06	31968.33	2.72	0.71	4348
CVRP_L_31	1120	**34330.94**	EST	35617.70	34962.16	3.75	1.84	2395	35472.51	34770.34	3.33	1.28	5003
CVRP_L_32	1200	**36919.24**	EST	37989.05	37401.49	2.90	1.31	2750	37818.65	37377.35	2.44	1.24	5321
Tot.		288441		295856.55	291065.13			17509	294399.98	290058.65			35849
Avg.						2.40	0.78	1459			1.90	0.40	2987
< PB					1					3			
#B		9			0					3			

Table 24: Li et al. CVRP problems [40]. *EST* refers to a solution found by hand by Li et al [40] (the instances are highly symmetrical which makes it easy to construct good solutions by hand). *ORTR* refers to a solution found by a heuristic by Li et al. [40].