BuR – Business Research
Official Open Access Journal of VHB
Verband der Hochschullehrer für Betriebswirtschaft e.V.
Volume 1 | Issue 2 | December 08 | 207–218

A Cooperative and Adaptive Variable Neighborhood Search for the Multi Depot Vehicle Routing Problem with Time Windows

Michael Polacek, Institute for Business Administration, University of Vienna, Email: michael.polacek@gmail.com
Siegfried Benkner, Institute for Scientific Computing, University of Vienna, Email: sigi@par.univie.ac.at
Karl F. Doerner, Institute for Business Administration, University of Vienna, Email: karl.doerner@univie.ac.at
Richard F. Hartl, Institute for Business Administration, University of Vienna, Email: richard.hartl@univie.ac.at

Abstract
In this paper we propose two cooperation schemes to compose new parallel variants of the Variable Neighborhood Search (VNS). On the one hand, a coarse-grained cooperation scheme is introduced which is well suited for being enhanced with a solution warehouse to store and manage the so far best found solutions and a self-adapting mechanism for the most important search parameters. This makes an a priori parameter tuning obsolete. On the other hand, a fine-grained scheme was designed to reproduce the successful properties of the sequential VNS. In combination with the use of parallel exploration threads all of the best solutions and 11 out of 20 new best solutions for the Multi Depot Vehicle Routing Problem with Time Windows were found.

Keywords: Parallelization, Cooperation, Adaptation, Variable Neighborhood Search, Multi Depot Vehicle Routing Problem with Time Windows.

Manuscript received May 22, 2008, accepted by Karl Inderfurth (Operations and Information Systems) October 1, 2008.

1 Introduction

In recent years, cluster and grid architectures have become more and more popular. These architectures enable the design and development of cooperating algorithms to solve complex problems in the field of combinatorial optimization more efficiently than their sequential counterparts. The cooperation can take place between the same metaheuristic paradigms (e.g., Alba 2005; Crainic and Toulouse 2002), between different metaheuristics (e.g., Le Bouthillier and Crainic 2005) or combinations of metaheuristics and mathematical programming (e.g., Fischetti and Lodi 2003; Hansen, Mladenović, and Urosevic 2006).

The aim of this paper is twofold: First, we propose a cooperative and adaptive algorithm based on the philosophy of the Variable Neighborhood Search (VNS). This metaheuristic described by Hansen and Mladenović (1999) is applied to solve Multi Depot Vehicle Routing Problems with Time Windows. Second, in combination with the use of parallel exploration threads new best solutions were found. The sequential algorithm was published by Polacek, Hartl, Doerner, and Reimann (2004) and also applied to a real-world routing problem (Polacek, Doerner, Hartl, Kiechle, and Reimann 2007). For the p-median problem, a cooperative implementation of the VNS was recently developed (e.g., Crainic, Gendreau, Hansen, and Mladenović 2004; Garcia Lopez, Melian Batista, Moreno Perez, and Moreno Vega 2002). Moreno Perez, Hansen, and Mladenović (2005) provide a survey of parallel VNS implementations.

In recent years, some papers on the parallelization of algorithms for solving the capacitated vehicle routing problem have been published (e.g., Jozefowiez, Semet, and Talbi 1999, 2005; Ralphs 2004). Jozefowiez, Semet, and Talbi (1999) devel-

BuR – Business Research
Official Open Access Journal of VHB
Verband der Hochschullehrer für Betriebswirtschaft e.V.
Volume 1 | Issue 2 | December 08 | 207–218

oped a parallel Pareto genetic algorithm as well as a Pareto tabu search for a bi-objective VRP whereas Ralphs (2004) developed a parallel exact procedure based on branch and cut for the problem at hand. For the Vehicle Routing Problem with Time Windows Le Bouthillier and Crainic (2005) developed a cooperative parallel metaheuristic. Many applications were developed in the last few years in the broader field of parallel computing in transportation (e.g., Florian and Gendreau 2001). In the book by Alba (2005) a recent and comprehensive overview of the different parallel metaheuristics can be found. To make the use of parallel metaheuristics accessible to a broad range of users, different libraries were developed by Alba and the MALLBA Group (2002) and Cahon, Melab, and Talbi (2004). Moreno Perez, Hansen, and Mladenović (2005) outline four different parallel VNS approaches. The first strategy analyzed by Garcia Lopez, Melian Batista, Moreno Perez, and Moreno Vega (2002) parallelizes the local search in the sequential VNS to get a balanced load among the processors and is denoted as Synchronous Parallel VNS (SPVNS). The second approach called Replicated Parallel VNS (RPVNS) and described by Crainic, Gendreau, Hansen, and Mladenović (2004) simply runs an independent VNS procedure on each processor. This non-cooperating parallelization is characterized by a multi start behavior. The same authors also report a more complex parallel variant denoted as Cooperative Neighborhood VNS (CNVNS) where several independent VNS processes cooperate by asynchronously exchanging information about the best solution identified so far. Communication takes place after a complete iteration through the set of neighborhoods. The last parallel strategy introduced is the Replicated Shaking VNS (RSVNS) proposed by Garcia Lopez, Melian Batista, Moreno Perez, and Moreno Vega (2002). RSVNS uses a synchronous cooperation mechanism where each worker processor generates one neighboring solution and applies the local search. In this paper we discuss different cooperation schemes and we propose an adaptive VNS where no a priori parameter tuning is necessary. First, from a technical point of view, it presents the first cooper- ative and adaptive implementation of a VNS for this problem and several design issues for cooperation and adaptation of the VNS algorithm are discussed. Second, from a problem oriented point of view, the computational results show that the approach is competitive with the sequential VNS implementation (Polacek, Hartl, Doerner, and Reimann 2004) and the Tabu Search (TS) algorithm published in (Cordeau, Laporte, and Mercier 2001, 2004), with respect to both solution quality and computation times. The parallelization strategy we use is an extension of the one implemented in CNVNS. The worker processes communicate exclusively with the master process which operates as the central memory. This allows an asynchronous cooperation of individual processes. In our proposed variants each worker has to search through a certain number of neighborhoods. However, compared to the CNVNS, in the fine-grained cooperation scheme this must not necessarily conclude the whole set of neighborhoods in one worker task. In the coarsegrained cooperation scheme, however, the number of iterations performed by each worker is vastly higher than the number of neighborhoods. This results in a more independent search via individual processes. The remainder of the paper is organized as follows: The routing problem is illustrated in Section 2 and the solution procedure of the sequential algorithm is discussed in Section 3. Section 4 reviews the main ideas of the cooperation and adaptation schemes and provides the details of the implementation and the design choices. Computational results are presented and discussed in Section 5. Section 6 concludes the paper with a resume of the applied approach.

2 Problem Description

The parallel VNS is applied to the Multi Depot Vehicle Routing Problem with Time Windows (MDVRPTW). It is a generalization of the well-known Vehicle Routing Problem with Time Windows (VR- PTW) where instead of one depot, several depots with different locations and associated fleets have to be considered. The number of customers is denoted by n and the number of depots is denoted by m. Thus, the problem is defined on a complete graph $G = (V, A)$, where $V = \{v_1, ..., v_m, v_{m+1}, ..., v_{m+n}\}$ is the vertex set and $A = \{(v_i, v_j) : v_i, v_j \in V, i \neq j\}$ is the arc set. Vertices v_1 to v_m correspond to m depots, while the vertices v_{m+1} to v_{m+n} represent n customers. Each vertex $v_i \in V$ has several non-negative weights associated with it, namely, a demand d_i, a service time s_i, as well as an earliest e_i and latest l_i possi-

BuR – Business Research
Official Open Access Journal of VHB
Verband der Hochschullehrer für Betriebswirtschaft e.V.
Volume 1 | Issue 2 | December 08 | 207–218

ble start time for the service, which define a time window $[e_i, l_i]$. For the depots these time windows correspond to the opening hours. Furthermore, the depot vertices v_1 to v_m feature no demands and service times, i.e. $d_i = s_i = 0, \forall i \in \{1, ..., m\}$. Associated to each arc (v_i, v_j) is a non-negative travel time or cost c_{ij}. Finally, a fleet of K vehicles is located at m depots. Each depot has t vehicles. Each vehicle k has associated a non-negative capacity D_k and a non-negative maximum route duration T_k. Note, that the distribution of vehicles over the depots is fixed a priori and is given as input data.
Based on this graph, the MDVRPTW consists of building K vehicle routes such that each vehicle starts and ends at its home depot, each customer is served by one and only one vehicle, the total load and duration of vehicle k does not exceed D_k and T_k respectively, the service at each customer i begins within the associated time window $[e_i, l_i]$ and each vehicle route starts and ends within the time window of its depot. The objective is to minimize the total distance travelled by all vehicles.

3 A VNS for the MDVRPTW

VNS is a metaheuristic for solving combinatorial and global optimization problems proposed by Hansen and Mladenović (1999, 2001). The paper at hand deals with the parallelization of the VNS for the MDVRPTW published by Polacek, Hartl, Doerner, and Reimann (2004). For convenience of the reader we repeat the approach and describe the required modifications for the cooperation and adaptation schemes. The steps of the basic VNS are shown in Figure 1. Here, $N_\kappa(\kappa = 1, \ldots, \kappa_{max})$ is a finite set of pre-selected neighborhood structures. The stopping condition may be, e.g., maximum CPU time allowed, maximum number of iterations or maximum number of iterations between two improvements.
The basic VNS consists of both a stochastic component, i.e., the randomized selection of a neighbor in the shaking phase, and a deterministic component, that is the application of an iterative improvement procedure in each iteration. Finally, the solution obtained is compared to the incumbent one and will be accepted as a new starting point if an improvement was made, otherwise it will be rejected. Note, that following Polacek, Hartl, Doerner, and Reimann (2004), also ascending moves are permitted. Below, the implementation of each part of the VNS to solve the MDVRPTW is described. The description consists of the building of an initial solution, the shaking phase including the neighborhood structure definition with the necessary exchange operators, the local search method, and the acceptance decision.

Figure 1: Steps of the basic VNS (cf., Hansen and Mladenović, 2001)

Initialization. Select the set of neighborhood structures $N_\kappa(\kappa = 1, \ldots, \kappa_{max})$, that will be used in the search; find an initial solution x; choose a stopping condition;
Repeat the following until the stopping condition is met:

1. Set $\kappa \leftarrow 1$;
2. Repeat the following steps until $\kappa = \kappa_{max}$:
   (a) *Shaking*. Generate a point x' at random from κ^{th} neighborhood of x ($x' \in N_\kappa(x)$);
   (b) *Iterative improvement*. Apply some local search method with x' as initial solution; denote with x'' the so-obtained local optimum;
   (c) *Acceptance decision*. If this local optimum x'' is better than the incumbent, move there ($x \leftarrow x''$), and continue the search with N_1 ($\kappa \leftarrow 1$); otherwise, set $\kappa \leftarrow \kappa + 1$;

3.1 Initial Solution

To construct an initial solution for the MDVRPTW, each customer i is first assigned to the nearest depot. Then all customers associated with a depot are ranked with respect to increasing centers of their time windows. Finally, routes are constructed by sequentially appending the pre-ordered customers at the end of a route in a cyclic manner for all routes. The initial solution is not necessarily feasible, the following iterative process of the VNS needs to overcome this and must come up with a feasible solution.

3.2 Shaking

The set of neighborhood structures used for shaking is the core of the VNS. The main difficulty is to find a balance between effectiveness and the chance to get out of local optima.

BuR – Business Research
Official Open Access Journal of VHB
Verband der Hochschullehrer für Betriebswirtschaft e.V.
Volume 1 | Issue 2 | December 08 | 207–218

To define a neighborhood for the incumbent solution an appropriate function or operator must be specified. The main issue is that the neighborhood operator should allow to sufficiently perturb the incumbent solution while still making sure that the new solution keeps important parts of the incumbent.

The operator we use in the shaking phase is the CROSS exchange operator developed by Taillard, Badeau, Gendreau, Guertin, and Potvin (1997). The main idea of this exchange is to take two segments of different routes and exchange them. Compared with the VNS by Polacek, Hartl, Doerner, and Reimann (2004) the selection criterion is slightly changed. Now it is possible to select the same route twice. This allows to explore more customer visit combinations within one route.

An extension to the CROSS exchange operator is introduced by Braysy (2003). Here the sequences get inverted, i.e., the orientation of the selected route parts changes. Consequently, this operator is called inverted CROSS exchange - iCROSS exchange for short. Both operators are used to define a set of neighborhood structures for the VNS.

The set of neighborhood structures used is divided into two parts: the first half considers only routes belonging to a given depot, whereas the second half selects routes from two different depots. In the first six neighborhood structures a sequence of up to the number of customers on a route can be exchanged. In detail, in the first neighborhood structure one customer is exchanged. In the second neighborhood structure a sequence length of up to two customers is exchanged. The potential sequence length of the customers is extended to five (within neighborhood structure five). In neighborhood structure six the customers of the whole route can be exchanged with customers of another route within the same depot. In neighborhood structure six to twelve, routes of two depots are considered.

Note that the maximum sequence length just acts as an upper bound for the sequence length removed in a given neighborhood. Thus, while in each neighborhood all possible sequence lengths are equally likely to be chosen, overall, there is a strong bias towards smaller sequence lengths to focus the search rather close to the incumbent solution. However, significant changes may occur.

In addition, in each neighborhood the iCROSS exchange operator is applied with a probability $1/(2 \cdot K)$ to both routes to further increase the extent of the perturbation. This is also a modification to the VNS by Polacek, Hartl, Doerner, and Reimann (2004) where the probability for the iCROSS exchange operator applied to only one route was $1/\kappa_{max}$. By including the fleet size K and, therefore, the possible number of routes, the new probability function is now correlated to the problem complexity and not determined by a preset search parameter, which leads to a slightly better performance of the search.

The neighborhood parameter κ_{max} is the elementary parameter in the VNS. In the standard VNS for the MDVRPTW, κ_{max} is set to 12 whereas in our adaptive parallel approach this parameter is adjusted within the search.

3.3 Iterative improvement

A solution obtained through shaking is afterwards submitted to an iterative improvement procedure to come up with a local optimal solution. Here the local search is a restricted version of the 3-opt where the length of the sequences to be exchanged is bounded by an upper limit of three. The customers have time windows; therefore, a restricted version of the 3-opt is beneficial with respect to runtime and possible time window violations.

After each shaking, only the two routes that have changed need to be re-optimized. In the iterative improvement phase the first improvement strategy is realized.

3.4 Acceptance decision

After the shaking and the iterative improvement procedures have been performed, the solution thus obtained has to be compared to the incumbent solution to be able to decide whether or not to accept it. We use a modified acceptance decision which is based on threshold-accepting ideas (cf., Dueck and Scheuer 1990) to allow non-improving (ascending) moves. A solution with an improved solution quality is always accepted, while deteriorating solutions are accepted as long as their objective value does not exceed a fixed threshold. This threshold is given by $\theta\%$ of the so far best found solution value. As proposed by Polacek, Hartl, Doerner, and Reimann (2004) ascending moves are only performed after a minimum number of σ iterations counted from the last accepted move.

Because of the fact that a reasonable presetting of the threshold parameter θ strongly depends on

BuR – Business Research
Official Open Access Journal of VHB
Verband der Hochschullehrer für Betriebswirtschaft e.V.
Volume 1 | Issue 2 | December 08 | 207–218

the given problem instance we integrated this parameter into our self-adapting mechanism for the parallel VNS variant.

4 Cooperation Schemes

We propose two parallel cooperation schemes both of which are based on a central memory mechanism provided by the master process which stores and manages the best found solutions. Hence, the communication is done exclusively between this master process and the individual worker processes which act as search threads. This allows an asynchronous exchange of information where the whole solution data is only transmitted when a new best solution was discovered. If the worker process improves the obtained solution it sends only the value of the new solution to the master process. If this value improves the best solution value found so far, the worker sends the complete solution data to the master, which in turn sends it to all other working processes.
The parallelization takes place at a level where each search thread executes all three main components of the VNS several times. This includes the shaking phase, the local search, as well as the comparison of the new obtained solution with the incumbent one. The point in time where the working process communicates its so far best solution to the master is triggered by an iteration counter.
In our cooperative architecture each processor executes exactly one process. In the context of this paper a search thread is defined as a single process which runs exclusively on one processor and does not share any resources with other search threads.

4.1 Coarse-Grained Cooperation

In the coarse-grained cooperation scheme exactly one ascending move is performed per thread, i.e. at least $2 \cdot \sigma$ iterations are made between communications. As described in Section 3.4, if there is no improvement of the solution after σ iterations, also non-improving solutions will be accepted if the solution value is below the threshold of θ% of the best found solution. If improving solutions are found, the iteration counter for allowing an ascending move gets reset each time.
Instead of making the second ascending move the worker communicates the so far best solution value to the master process. If no improving solution was found by one of the other search threads, the working process continues the search with its own best found solution.
On the one hand, the coarse granularity enables an independent search of the working processes which consequently reduces the communication with the master process. On the other hand, this form of cooperation enables the application of a self-adapting mechanism for the most influential parameters of the applied VNS: the neighborhood parameter κ_{max} and the ascending move parameter θ.
Within the adaptation process the values for both parameters have lower and upper bounds. So the range of θ goes from 4 to 10 whereas all even numbers from 4 to 20 can be assigned to κ_{max}. Furthermore, there is an iterator x_i for every possible value i of the two parameters. If a search thread obtains an improving solution with the current parameter setting the associated iterators are incremented by 1. The parameter values for each search thread are selected by the roulette wheel method. Here, for every parameter value i the probability $P(i)$ to get chosen is calculated by the following function:

$$(1) \qquad P(i) = \frac{\ln(x_i)}{\sum_{j \in \Omega} \ln(x_j)} \forall i \in \Omega$$

where Ω denotes the set of all possible parameter values. The natural logarithm is applied to avoid the dominance of a specific parameter setting. So the VNS has the possibility to adjust its parameter values to find an appropriate setting for the different phases of the search process.
Furthermore, a solution warehouse to store and manage the so far best found solution can be used instead of accepting only the best solution. The solution warehouse emphasizes the diversification of the search. In our case it stores the 10 best solutions found so far and the starting point for each search thread is randomly chosen from these solutions.

4.2 Fine-Grained Cooperation

The fundamental idea behind the fine-grained cooperation scheme was to develop a parallel VNS which retains the successful properties of the sequential one. Hence, a fine granularity was realized and the decision about accepting ascending moves was assigned to the master process.
More precisely, every κ_{max} iterations a working process sends its new best solution value to the

BuR – Business Research
Official Open Access Journal of VHB
Verband der Hochschullehrer für Betriebswirtschaft e.V.
Volume 1 | Issue 2 | December 08 | 207–218

master process. This is also done in case the new best solution generated by the worker does not improve the starting solution of the subtask. If a search thread does not reach the neighborhood of κ_{max} the value of κ is stored and serves as starting value for the next subtask.
Because of the fact that the master process also accepts ascending moves two solutions have to be stored: the current solution which is the starting point for the search threads and the so far best found solution. If σ iterations have passed without an improvement the master accepts an non-improving solution if the value of the objective function does not exceed θ percent of the value of the best found solution. An important issue here is that after accepting an ascending move the master process has to reject all improving solutions from the other search threads until all of them have retrieved the non-improving solution as starting point. Otherwise the principle of the ascending move is not effective. For the sake of completeness, note that new best solutions are always accepted by the master process.

5 Computational Results

5.1 Performance Measurement

The experiments with the parallel variants of our algorithms were run on the cluster IBM 1350 of the Technical University of Vienna with 144 Pentium IV Nocona 3.6 GHz processors (2 processors per node) connected via InfiniBand low latency node interconnect.
The VNS is implemented in ANSI C++ using elements of the Standard Template Library (STL). For programming parallel processors the Message-Passing Interface (MPI) is used. MPI is a library of functions and macros that can be implemented in C, FORTRAN, and C++ programs. The definition of MPI is documented in the MPI standard (MPI, 1995).
For analyzing the performance of our parallel algorithm we measured the speed-up as well as the efficiency obtained on varying numbers of processors. The efficiency and speed-up measures we used are based on the definitions recommended by Alba and Luque (2005) and defined by Alba and the MALLBA Group (2002). The original definitions are introduced by Barr and Hickman (1993) and Karp and Flatt (1990). In this article the speedup is defined as

$$(2) \qquad S_p = T_{seq}/T_p$$

where S_p is the speed-up obtained on p processors, T_{seq} is the measured sequential execution time and T_p is the parallel execution time on p processors. Efficiency is defined as

$$(3) \qquad E_p = S_p/p$$

As our algorithm is non-deterministic we use the average parallel execution time and the average sequential execution time. All our reported results are averaged over 20 instances. In our implementation we need one processor which serves as central memory frontend. As this processor is only a communication node we introduced a measure for the working processors. We denote these measures as *worker speed-up* and *worker efficiency*.
Worker speed-up is defined as

$$(4) \qquad S_q^w = T_{seq}/T_q^w$$

where S_q^w is the speed-up obtained on q workers, T_{seq} is the measured sequential execution time and T_q^w is the parallel execution time on $q = p - 1$ processors.
Worker efficiency is defined as

$$(5) \qquad E_q^w = S_q^w/(p-1)$$

Note that for the complete algorithm p processors are required. The p-th processor stores and manages the best found solutions and is not considered in the computation of the worker speed-up and the worker efficiency.

5.2 Numerical Results

The VNS described in this paper originates from the VNS introduced by Polacek, Hartl, Doerner, and Reimann (2004) which is denoted as VNS$_{prev}$. Compared to the VNS$_{prev}$ the current VNS includes two modifications. On the one hand, the CROSS exchange operator in the shaking phase can also be applied within one route. On the other hand, the probability for the application of the iCROSS exchange operator has slightly changed.
The problem instances used for the analysis originate from Cordeau, Laporte, and Mercier (2001) and are available on the internet at http://www.hec.ca/chairedistributique/data. The

BuR – Business Research
Official Open Access Journal of VHB
Verband der Hochschullehrer für Betriebswirtschaft e.V.
Volume 1 | Issue 2 | December 08 | 207–218

Table 1: Modified VNS compared to TS and previous VNS

					TS				VNS (Avg. 32 Runs)	
Nr.	n	m	t	Time	Value	VNS_{prev}	$Time_{total}$	$Time_{best}$	Value	RPD
01	48	4	2	28	1074.12	1074.12	67.86	9.49	1074.12	0.00%
02	96	4	3	79	1762.48	1762.36	115.09	27.63	1763.66	0.07%
03	144	4	4	115	2397.06	2385.94	143.27	75.88	2388.73	0.12%
04	192	4	5	144	2865.71	2840.59	153.23	93.53	2847.56	0.25%
05	240	4	6	181	3050.80	3018.38	136.40	89.36	3015.27	-0.10%
06	288	4	7	221	3670.13	3675.61	157.41	96.63	3674.60	-0.03%
07	72	6	2	53	1418.22	1418.22	70.48	7.66	1418.22	0.00%
08	144	6	3	102	2118.50	2099.49	106.35	54.92	2103.21	0.18%
09	216	6	4	160	2760.46	2752.61	135.71	69.11	2753.61	0.04%
10	288	6	5	227	3507.26	3540.60	124.66	65.70	3541.01	0.01%
11	48	4	1	32	1016.59	1021.61	183.22	18.30	1011.65	-0.97%
12	96	4	2	81	1486.26	1488.28	193.38	89.40	1488.32	0.00%
13	144	4	3	143	2028.85	2014.06	199.94	82.30	2012.37	-0.08%
14	192	4	4	188	2228.64	2242.45	180.74	106.51	2239.02	-0.15%
15	240	4	5	227	2527.60	2525.20	135.41	89.39	2498.85	-1.04%
16	288	4	6	261	2960.93	2940.73	153.76	99.69	2909.45	-1.06%
17	72	6	1	61	1241.25	1249.45	203.22	62.15	1247.51	-0.16%
18	144	6	2	146	1823.24	1831.03	163.55	99.24	1809.25	-1.19%
19	216	6	3	262	2288.38	2314.70	165.31	90.84	2294.19	-0.89%
20	288	6	4	263	3120.32	3109.78	132.89	86.38	3093.51	-0.52%
				2974	45346.8	45305.21	2921.88	1414.13	45184.09	-0.28%

data set consists of 20 instances which differ with respect to their size as well as their time window tightness.

The first four columns of Table 1 describe the benchmark instances with a consecutive instance number, the number of customers which have to be served, the number of depots and the number of available vehicles at each depot denoted by Nr., n, m and t, respectively. Note that the same number of vehicles t is assigned to each of the m depots. The next two columns show the runtime in minutes and the corresponding solution value of the TS introduced by Cordeau, Laporte, and Mercier (2001) and improved by the some group of authors (Cordeau, Laporte, and Mercier 2004). These results were obtained after 10^6 iterations on a 2 GHz Pentium 4 computer. Furthermore, the column VNS_{prev} provides the average results of 10 runs of the original VNS for the MDVRPTW which are compared with the results of the modified VNS. The relative percentage deviation (RPD) is stated in the last row. $Time_{total}$ and $Time_{best}$ show the total runtime for 10^8 iterations and the time when the best solution was found, respectively. The new VNS calculations were performed on a single processor of a Pentium 3.6 GHz dual processor computer.

The modified VNS outperforms VNS_{prev} with an average improvement of 0.28%. So the new shaking phase was qualified to be used for the parallel VNS variants. Furthermore, a remarkable fact is that on average all results were found in half of the total runtime.

To obtain comparable data for the parallel VNS variants we implemented the RPVNS introduced by Garcia Lopez, Melian Batista, Moreno Perez, and Moreno Vega (2002) to report the *contribution of cooperation*. The RPVNS is one of the simplest parallel VNS approaches because there is no form of cooperation between the individual search threads. Every thread performs a complete VNS run. The average results of the VNS on the 32 working processes are illustrated in Table 1. Furthermore, we report the best results and the average runtimes for the first 2, 4, 8 and 16 search threads as well as for the total of 32 processes in Table 2(a).

Table 2 presents 4 different series of exploration runs where each series consists of 5 runs with

BuR – Business Research
Official Open Access Journal of VHB
Verband der Hochschullehrer für Betriebswirtschaft e.V.
Volume 1 | Issue 2 | December 08 | 207–218

Table 2: Exploration runs

(a) VNS (Avg. 32 Runs)

	Time	E_p	Total	RPD
	2921.88	100.00%	45184.09	0.00%

(b) RPVNS

Worker	Time	E_p	Total	RPD
2	2929.52	100.26%	45130.68	-0.12%
4	2926.06	100.14%	45015.01	-0.37%
8	2928.32	100.22%	44891.39	-0.65%
16	2930.02	100.28%	44846.79	-0.75%
32	2921.87	100.00%	44780.63	-0.89%

(c) coarse-grained non-adaptive VNS

Worker	Time	E_q^w	Total	RPD
2	3032.13	96.36%	45316.92	0.29%
4	3008.39	97.12%	45225.19	0.09%
8	3060.36	95.47%	45138.47	-0.10%
16	2997.76	97.47%	45023.94	-0.35%
32	3036.72	96.22%	44951.25	-0.52%

(d) coarse-grained adaptive VNS

Worker	Time	E_q^w	Total	RPD
2	2853.21	102.41%	45240.28	0.12%
4	2848.66	102.57%	45092.96	-0.20%
8	2854.37	102.36%	45053.06	-0.29%
16	2849.54	102.54%	45008.18	-0.39%
32	2829.68	103.26%	44894.48	-0.64%

(e) fine-grained VNS

Worker	Time	E_q^w	Total	RPD
2	3105.19	94.10%	45125.14	-0.13%
4	3121.39	93.61%	44999.13	-0.41%
8	3148.21	92.81%	44877.13	-0.68%
16	3132.59	93.27%	44813.75	-0.82%
32	3175.49	92.01%	44679.83	-1.12%

Table 3: Speed-up runs

(a) coarse-grained non-adaptive VNS

Worker	Time	S_q^w	E_q^w	Total	RPD
2	1517.38	1.93	96.28%	45635.55	1.00%
4	763.08	3.83	95.73%	45425.72	0.53%
8	386.52	7.56	94.49%	45368.97	0.41%
16	190.85	15.31	95.69%	45614.37	0.95%
32	95.60	30.56	95.51%	45482.93	0.66%
Avg.			95.54%	45505.51	0.71%

(b) fine-grained VNS

Worker	Time	S_q^w	E_q^w	Total	RPD
2	1569.78	1.86	93.07%	45264.99	0.18%
4	783.60	3.73	93.22%	45169.03	-0.03%
8	392.13	7.45	93.14%	45319.84	0.30%
16	198.14	14.75	92.17%	45121.79	-0.14%
32	99.34	29.41	91.92%	45393.73	0.46%
Avg.			92.70%	45253.88	0.15%

an exponentially increasing number of workers. Compared to the sequential run the runtime is approximately the same. However, the total number of iterations is determined by multiplying the number of search threads with the standard value of 10^8 iterations. All tables contain the total runtime, the worker efficiency E_w, the total objective value over all instances and the RPD comparing this value with the average value of the 32 VNS processes presented in Table 2(a). Table 2(b) contains the RPVNS results described in the previous paragraph. Here the use of 2 search threads achieves an improvement of 0.12% while the use of 32 threads brings an improvement of 0.89% in solution quality.

More significant are Table 2(c) and Table 2(d) which represent the results of the coarse-grained cooperation scheme with and without adaption and a solution warehouse. While the non-adapting variant already has a high worker efficiency with an average value of 96.53% which ascribes to the minimal communication effort, the average efficiency value of the adapting search lies with 102.63% above the 100% mark. The reason for this is the fact, that the self-adapting mechanism allows a strong bias towards the smaller sequence lengths during the exchange operations in the shaking phase of the VNS. This leads to a lower computational effort and is therefore noticeable in the runtimes of the self-adapting parallel variant. Furthermore, the parameterless version achieves on average 0.16% better results than the parallel VNS with fixed parameters. A graphical representation of all exploration runs is given in Figure 5.2 (see on page 10). The x-axis denotes the number of workers and the y-axis denotes the RPD to the average

BuR – Business Research
Official Open Access Journal of VHB
Verband der Hochschullehrer für Betriebswirtschaft e.V.
Volume 1 | Issue 2 | December 08 | 207–218

RPVNS results.
Table 2(e) contains the results of the fine-grained cooperation scheme. It can be shown that this variant successfully replicates the effective properties of the sequential VNS. Moreover, with the use of several parallel search processes the solution quality can be remarkably improved in nearly the same runtime compared to the standard VNS. In the case of 32 workers the improvement is more than 1%. Although this scheme is defined by an extremely fine granularity, the worker efficiency has a more than satisfying average value of 93.16%.
Table 3 shows the behavior of the coarse- and fine-grained parallel VNS in two series of speed-up runs. Here, the total number of iterations is constantly set to 10^8 in all runs. The self-adapting mechanism and the solution warehouse were omitted in the speed-up runs for the coarse-grained variant. The two tables contain data about the total time, the worker speed-up S_q^w, the worker efficiency E_q^w, the total value over all instances and the RPD to the average RPVNS value. Again, the worker efficiency and consequently the worker speed-up are relatively high in both series. In the coarse-grained runs E_q^w is 95.54% on average and in the fine-grained runs it is 92.70%. In both cases the efficiency value is lower than in the exploration runs because of the fact that improving solutions were found mostly in the beginning of a search. Therefore, more solution data has to be transferred between the processes. The fine-grained cooperation scheme convinces with its marginal average RPD while still possessing an excellent runtime scalability.
Finally, Table 4 gives an overview of the best results for the MDVRPTW. First, the best found results so far by the VNS$_{prev}$ and the TS are presented along with the minimum of both methods. Then the VNS column states all the new best found solutions obtained by the parallel variants of the VNS. The last column shows the RPD between the incumbent values and the new VNS values. Hence, on average the results for the MDVRPTW were again improved by 0.37%.

6 Conclusion

In this paper two parallel VNS approaches were introduced. The advantage of the coarse-grained cooperation scheme is the independency of its search threads. Each process can be configured with an individual parameter setup. A self-adapting mechanism was developed to control these settings and to evaluate the results obtained by the individual search processes. The main contribution of this technique is that no a priori parameter tuning for different problem instances is required. Therefore, the coarse-grained cooperation scheme is well suited for problems which cannot be studied in detail before applying the search like it may occur in the real-world.
The fine-grained cooperation scheme allows a successful replication of the effective properties of the sequential VNS. With the use of 32 search threads the intensified exploration of the solution space improves the solution quality by 1.12% compared to the results of a single run of the sequential procedure. Here, the impact of the cooperation becomes apparent by comparing the results of this scheme with the best results obtained by 32 independent RPVNS runs. The fine-grained cooperation scheme is best suited for cases where the characteristics of the problem instances are known in advance and appropriate parameter settings can be made. Furthermore, at a constant number of iterations the runtime of a complete search applied to all instances was reduced from 48.7 to 1.7 hours.
Both cooperation schemes show extremely high efficiency values which results in an excellent runtime scalability. Finally, for all 20 MDVRPTW instances the best known solution was found and in 11 cases a new best solution was obtained.

Acknowledgements

Karl Doerner is grateful to Enrique Alba for helpful comments and fruitful discussions during his stay at the NEO Lab in Malaga. Financial support of this research from the Fonds zur Förderung der wissenschaftlichen Forschung (FWF) under grant #P20342-N13 and from the Special Research Program SFB F011 "AURORA" is gratefully acknowledged.

BuR – Business Research
Official Open Access Journal of VHB
Verband der Hochschullehrer für Betriebswirtschaft e.V.
Volume 1 | Issue 2 | December 08 | 207–218

Figure 2: Results of the exploration runs (x-axis: number of workers, y-axis: RPD)

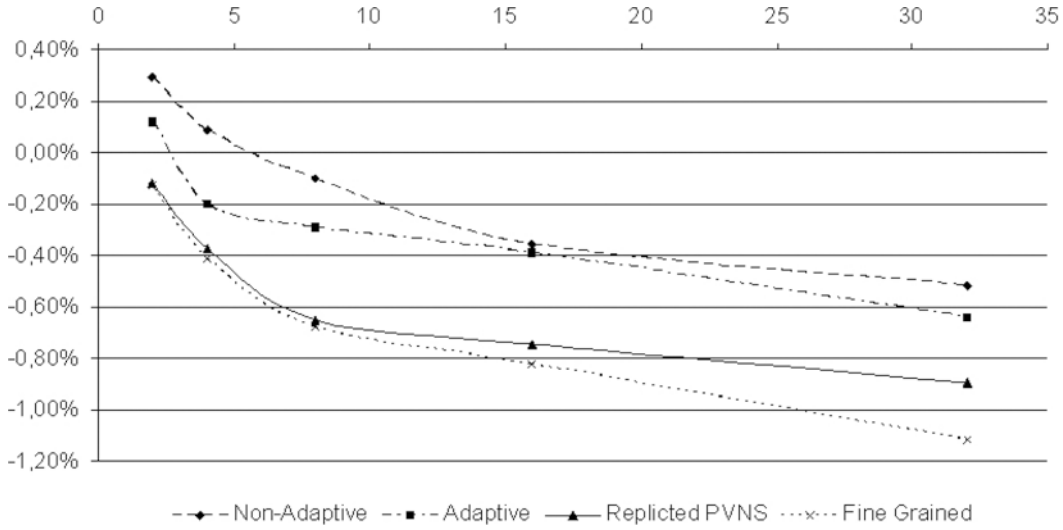

Table 4: New best results for the MDVRPTW

Nr.	VNS best	TS best	Previous Best	New Best	RPD
01	**1074.12**	**1074.12**	**1074.12**	**1074.12**	0.00%
02	**1762.21**	**1762.21**	**1762.21**	**1762.21**	0.00%
03	**2373.65**	**2373.65**	**2373.65**	**2373.65**	0.00%
04	**2815.48**	2852.29	**2815.48**	**2815.48**	0.00%
05	2993.94	3029.65	2993.94	**2965.18**	-0.96%
06	3629.72	3627.18	3627.18	**3612.72**	-0.40%
07	**1418.22**	**1418.22**	**1418.22**	**1418.22**	0.00%
08	**2096.73**	2102.61	**2096.73**	**2096.73**	0.00%
09	2730.54	2737.82	2730.54	**2727.42**	-0.11%
10	3499.56	3505.27	3499.56	**3483.22**	-0.47%
11	**1005.73**	**1005.73**	**1005.73**	**1005.73**	0.00%
12	1472.76	1478.51	1472.76	**1467.72**	-0.34%
13	**2001.83**	2011.24	**2001.83**	**2001.83**	0.00%
14	2215.51	2202.08	2202.08	**2196.28**	-0.26%
15	2465.25	2494.57	2465.25	**2456.52**	-0.35%
16	2896.03	2901.02	2896.03	**2853.32**	-1.47%
17	**1236.24**	**1236.24**	**1236.24**	**1236.24**	0.00%
18	1796.21	1792.61	1792.61	**1788.18**	-0.25%
19	2292.45	2285.10	2285.10	**2269.33**	-0.69%
20	3076.37	3079.16	3076.37	**3013.71**	-2.04%
	44852.55	44969.28	44825.63	44617.81	-0.37%

BuR – Business Research
Official Open Access Journal of VHB
Verband der Hochschullehrer für Betriebswirtschaft e.V.
Volume 1 | Issue 2 | December 08 | 207–218

References

Alba, Enrique (2005): Parallel Metaheuristics: A New Class of Algorithms, Wiley: Hoboken, New Jersey.

Alba, Enrique and Gabriel Luque (2005): Measuring the Performance of Parallel Metaheuristics, Wiley: Hoboken, New Jersey.

Alba, Enrique and the MALLBA Group (2002): MALLBA: A library of skeletons for combinatorial optimization (Proceedings of the Euro-Par, LNCS 2400), Springer, London, 927–932.

Barr, Richard S. and Betty L. Hickman (1993): Reporting computational experiments with parallel algorithms: issues, measures, and experts' opinions, *ORSA Journal of Computing*, 5 (1): 2–18.

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

Cahon, Sebastien, Nordine Melab and El-Ghazali Talbi (2004): ParadisEO: A framework for the reusable design of parallel and distributed metaheuristics, *Journal of Heuristics*, 10 (3): 357–380.

Cordeau, Jean-Francois, Gilbert Laporte and Anne Mercier (2001): A unified tabu search heuristic for vehicle routing problems with time windows, *Journal of the Operational Research Society*, 52 (8): 928–936.

Cordeau, Jean-Francois, Gilbert Laporte and Anne Mercier (2004): An improved tabu search algorithm for the handling of route duration constraints in vehicle routing problems with time windows, *Journal of the Operational Research Society*, 55 (5): 542–546.

Crainic, Teodor G., Michel Gendreau, Pierre Hansen and Nenad Mladenovic (2004): Cooperative parallel variable neighborhood search for the p-median, *Journal of Heuristics*, 10 (3): 293–314.

Crainic, Teodor G. and Michel Toulouse (2002): Parallel strategies for meta-heuristics, in: Fred W. Glovers (ed.): *Handbook of Metaheuristics*, Kluwer, Dordrecht, 251–285.

Dueck, Gunter and Tobias Scheuer (1990): Threshold accepting: A general purpose optimization algorithm appearing superior to simulated annealing, *Journal of Computational Physics*, 90 (1): 161–175.

Fischetti, Matteo and Andrea Lodi (2003): Local branching, *Mathematical Programming Series B*, 98 (1-3): 23–47.

Florian, Michael and Michel Gendreau (2001): Applications of parallel computing in transportation, *Parallel Computing*, 27 (12): 1521–1522.

Garcia Lopez, Felix, Belen Melian-Batista, Jose A. Moreno-perez and J. Marcos Moreno-Vega (2002): The parallel variable neighbourhood search for the p-median problem, *Journal of Heuristics*, 8 (3): 375–388.

Hansen, Pierre and Nenad Mladenovic (1999): An introduction to variable neighborhood search, *Meta-Heuristics: Advances and Trends in Local Search Paradigms for Optimization*, Kluwer Academic Publishers, Boston: 433–458.

Hansen, Pierre and Nenad Mladenovic (2001): Variable neighborhood search: Principles and applications, *European Journal of Operational Research*, 130 (3): 449–467.

Hansen, Pierre, Nenad Mladenovic and Dragan Urosevic (2006): Variable neighborhood search and local branching, *Computers & Operations Research*, 33 (10): 3034–3045.

Jozefowiez, Nicolas, Frederic Semet and El-Ghazali Talbi (2002): Parallel and hybrid models for multiobjective optimization: application to the vehicle routing problem, in: Juan Guervós, Panagiotis Adamidis, Hans-Georg Beyer, José-Luis Fernández-Villacanas and Hans-Paul Schwefel (eds.): *Parallel Problem Solving from Nature - PPSN VII*, Lecture Notes in Computer Science, Vol. 2439, Springer, Berlin et al., 271–280.

Jozefowiez, Nicolas, Frederic Semet and El-Ghazali Talbi (2006): Enhancements of NSGA II and its applications to the vehicle routing problem with route balancing, in: El-Ghazali Talbi, Pierre Liardet, Pierre Collet, Evelyne Lutton, and Marc Schoenauer (eds.): *Artificial Evolution: 7th International Conference - EA 2005*, Lecture Notes in Computer Science, Vol. 3871, Springer, Berlin et al., 131–142.

Karp, Alan and Horace P. Flatt (1990): Measuring parallel processor performance, *Communications of the ACM*, 33 (5): 539–543.

Le Bouthillier, Alexandre and Teodor G. Crainic (2005): A cooperative parallel meta-heuristic for the vehicle routing problem with time windows, *Computers & Operations Research*, 32 (7): 1685–1708.

Moreno Perez, Jose A., Pierre Hansen and Nadan Mladenovic (2005): Parallel variable neighborhood search, in: Enrique Alba (ed.): *Parallel Metaheuristics: A New Class of Algorithms*, Wiley, Hoboken, New Jersey, 131–142.

MPI (1995) A message-passing interface standard version 1.1, *Message Passing Interface Forum. http://www.mpi-forum.org/docs/docs.html.*

Polacek, Michael, Richard F. Hartl, Karl Doerner and Marc Reimann (2004): A variable neighborhood search for the multi depot vehicle routing problem with time windows, *Journal of Heuristics*, 10 (6): 613–627.

BuR – Business Research
Official Open Access Journal of VHB
Verband der Hochschullehrer für Betriebswirtschaft e.V.
Volume 1 | Issue 2 | December 08 | 207–218

Polacek, Michael, Karl Doerner, Richard F. Hartl and Guenter Kiechle (2007): Scheduling periodic customer visits for a traveling salesperson, *European Journal of Operational Research*, 179 (3): 823–837.

Ralphs, Ted K. (2004): Parallel Branch and Cut for Capacitated Vehicle Routing, *Parallel Computing*, 29 (5): 607–629.

Taillard, Eric, Philippe Badeau, Michel Gendreau, Francois Guertin and Jean-Yves Potvin (1997): A tabu search heuristic for the vehicle routing problem with soft time windows, *Transportation Science*, 31 (2): 170–186.

Biographies

Michael Polacek has authored several articles dealing with stochastic local search procedures for routing problems in recent years. His studies began with a degree in Business Informatics at the Vienna University of Technology, and he has completed his master's thesis at the University of Vienna. Based on this initial work on VNS for an enriched vehicle routing problem, he has continued his studies in this field and completed his doctorate thesis at the University of Vienna which involved comprehensive work on a broad range of routing problems.

Siegfried Benkner is an Associate Professor of Computer Science and the Head of the Institute of Scientific Computing at the University of Vienna. He received his Ph.D. from the TU Vienna. Siegfried Benkner was a research fellow at the NEC Europe Research Labs, St. Augustin, Germany (1998-1999). His research interests include programming languages, compilation and runtime systems for parallel and distributed systems. He was involved in numerous international research projects and was the Technical Director of the EU project HPF+. He has published more than 90 peer-reviewed scientific articles and is a member of the ACM and the IEEE.

Karl F. Doerner is Assistant Professor of Business Administration at the University of Vienna. He received his Ph.D. in Computer Science and Business Administration from the University of Vienna. Karl Doerner is scientific advisor of Salzburg Research Forschungsgesellschaft where he was employed as senior researcher for the period 2007-2008. His research interests include heuristic and hybrid optimization techniques in transportation and logistics. He has been involved in numerous national research projects. He has published more than 50 peer-reviewed scientific articles.

Richard F. Hartl received his Ph.D. in Mathematics from the TU Vienna. He was Post Doc in Toronto (1982), Associate Professor at the TU Vienna (until 1993), and Full Professor in Magdeburg (1993-1995). Currently he is Full Professor of Production and Logistics at the University of Vienna and Extramural Fellow of CentER, Tilburg. His research addresses various topics in production, operations management and transportation. More than 150 publications in leading journals like Management Science, Transportation Science, International Journal of Production Research, Production and Operations Management, etc.; more than 400 ISI-citations in the SCI; various scientific grants and consulting projects.