

A Parallel Two-phase Metaheuristic for Routing Problems with Time Windows

Hermann Gehring and Jörg Homberger
University of Hagen, Profilstraße 8, D-58084 Hagen, Germany

Abstract:

This paper describes the parallelization of a two-phase metaheuristic for the vehicle routing problem with time windows and a central depot (VRPTW). The underlying objective function combines the minimization of the number of vehicles in the first search phase of the metaheuristic and the minimization of the total travel distance in the second search phase. The parallelization of the metaheuristic is based on the concept of cooperative autonomy, i.e., several autonomous two-phase metaheuristics cooperate through the exchange of solutions. The parallelized two-phase metaheuristic was subjected a comparative test on the basis of 356 problems from the literature with sizes varying from 100 to 1000 customers. The derived results seem to justify the proposed parallelization concept.

Key words:

Evolution strategy, metaheuristic, tabu search, hybridization, parallelization, vehicle routing, time windows.

1 Introduction and problem formulation

The vehicle routing problem with time windows (VRPTW) is an extension of the well-known vehicle routing problem (VRP). It can be described as follows (cf. Homberger and Gehring 1999): A number of n customers are to be serviced from a central depot with vehicles of the same capacity. The demand of each customer is to be covered by exactly one service within a given time window. The objective is to determine a feasible route schedule which primarily minimizes the number of vehicles (primary criterion) and secondarily the total travel distance (secondary criterion).

In the recent past, quite good results have been achieved for the VRPTW with metaheuristics such as tabu search, simulated annealing, genetic algorithms and evolution strategies. Only few authors report on the solution of the VRPTW by means of hybrid and parallel approaches.

According to Toulouse et al. (1996) three types of parallelization strategies seem to be appropriate for the methods most used in combinatorial optimization: (1) parallelization of operations within an iteration of the solution method (see e.g. Garcia et al. 1994), (2) decomposition of problem domain or search space (see e.g. Badeau et al. 1997), and (3) multi-search threads with various degrees of synchronization and cooperation (see e.g. Schulze and Fahle 1999). In this paper a type 3 parallelization approach for solving the VRPTW is presented and evaluated. The multi-search threads are realized by a differently configured two-phase metaheuristic, which is a modification of the method from Gehring and Homberger (1999). In contrast to this method, a (μ, λ) -evolution strategy – instead of a $(1, \lambda)$ -evolution strategy – is now used to minimize the vehicle number in the first search phase. Further differences in the first search phase concern the evaluation of individuals passed over to the next generation and the criterion for terminating the first search phase. In the second search phase the total travel distance is again minimized with the same tabu search algorithm. However, another termination criterion for the second search phase is used here. In each of the search phases a process cooperation by means of solution exchange takes place. Finally, the evaluation reported here is based on several runs per test instance instead of one run only.

2 A two-phase hybrid metaheuristic for the VRPTW

The proposed metaheuristic combines a (μ, λ) -evolution strategy and a subsequently executed tabu search. During the first phase, when the (μ, λ) -evolution strategy is applied, the search process is dedicated to the reduction of the vehicle number. The best solution found in the first search phase, which always comprises a minimal vehicle number, is passed over to the second phase. There arise no problems with the encoding of transferred solutions, since an identical encoding is used in both phases: solutions to the problem are represented by sequence vectors indicating the order in which the customers are served. In the second phase a tabu search is carried out. The pursued goal is the mini-

mization of the total travel distance, i.e., no attempts are made to reduce the vehicle number further. The overall two-phase search process is roughly outlined in Figure 1.

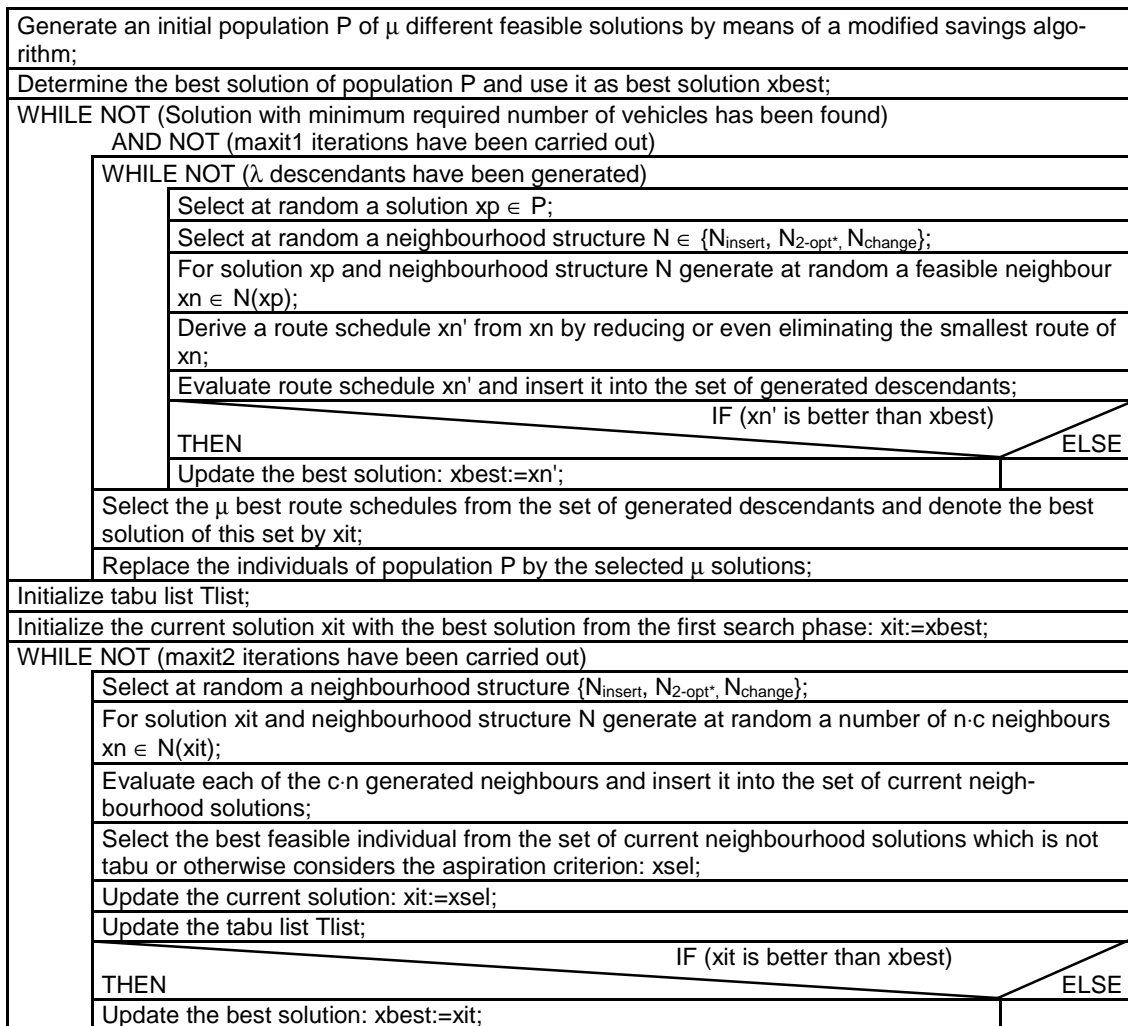


Figure 1. Structure chart for the two-phase hybrid metaheuristic.

The first search phase starts with the generation of an initial population P of μ different feasible solutions. For this purpose a modification of the savings algorithm from Clarke and Wright (1964) is used. The modification concerns the consideration of the imposed time window constraints and the inclusion of a stochastic component in the iterated selection of the next savings element. In each iteration of the (μ, λ) -evolution strategy λ descendants of the actual population are generated and evaluated. The μ best descendants form the next population. Further, the best solution is – eventually – updated in each iteration. The first phase terminates, if a solution with an ad minimum required number of vehicles has been found or if a given number $maxit1$ of iterations has been carried out.

The generation of a descendant involves measures aiming at the reduction of the vehicle number. A descendant is calculated in three steps (for details see Homberger and Gehring 1999): In the first step a solution x_p is selected at random from the actual population P . The selection probability is the same for each element of P . The second step comprises the generation of a descendant candidate. First, a neighbourhood structure N is selected at random from the set of neighbourhood structures $\{N_{insert}, N_{2-opt^*}, N_{change}\}$. The change of the neighbourhood structure during the search achieved in this way has, according to Hansen and Mladenovic (1999), a positive impact on the search process. The neighbourhood structure N_{insert} is based on an exchange concept from Or (1976), while N_{2-opt^*} was introduced by Potvin and Rousseau (1995) and N_{change} by Osman (1993). Then, a feasible move is determined at random for neighbourhood structure N and applied to solution x_p . If the resulting descendant candidate x_n is unfeasible, then the candidate generation is repeated. Each of the available move operators comprises a stochastic element. The repetition of a move leads therefore usually to a

different descendant candidate. The third step serves the transformation of candidate xn to the final neighbourhood solution xn' by means of a modified Or-opt operator (cf. Homberger and Gehring 1999). This operator aims at reducing the customer number of the smallest route of route schedule xn or even at eliminating the smallest route. For this purpose, attempts are made to insert subsequently all customers of the smallest route in a feasible way into other routes. If no customer can be inserted in another route, then route schedule xn is transferred unchanged to xn' .

The evaluation of a route schedule underlying the selection of the μ best descendants is, on the one hand and according to Homberger and Gehring (1999), based on the four criteria "total vehicle number", "customer number of the smallest route", "minimal delay" and "total travel distance". These criteria are applied in lexicographic order as given above. The so-called "minimal delay" refers to the smallest route of a solution and estimates how easily this route can be eliminated in the further search process (for details see Homberger and Gehring 1999). The criterion "customer number of the smallest route" pursues the same purpose. On the other hand, a second lexicographic evaluation function is used here alternatively. It is derived from the given first evaluation function by replacing the criterion "minimal delay" by the so-called "caused overload". Since it is possible that in certain cases the dissolution of a tour is prevented by the vehicle capacity, the latter criterion tries to estimate the impact of this limitation. The two evaluation functions, the one with "minimal delay" and the other with "caused overload", are used to configurate the concurrently executed search processes differently (see Section 4).

In the second search phase, the tabu search concept from Glover (1986) is used. At the beginning, the tabu list $Tlist$ is initialized as empty list and the actual solution is initialized with the best solution from the first search phase. In each of the following iterations a number of $c \cdot n$ neighbours of the actual solution, where c , $c > 0$, is a constant, is generated at random and evaluated. The underlying neighbourhood structure is selected at random from the same set of neighbourhood structures as used in the first search phase. The evaluation of each of the $c \cdot n$ neighbours is based on the vehicle number as primary criterion and the total travel distance as secondary criterion. The best feasible evaluated neighbour is selected from the set of $c \cdot n$ generated neighbours and used as new actual solution if the selected feasible neighbour is not tabu and respects the aspiration criterion. However, according to the aspiration criterion, the tabu status of one or more connections of a route schedule is ignored if the respective neighbour represents a new best solution. At the end of an iteration, the tabu list $Tlist$ and the best solution $xbest$ are updated. The update of the tabu list consists of those connections between customers or between customers and the depot which are eliminated when the transition from the (old) actual solution to its best evaluated neighbour is performed (cf. Potvin et al. 1996). The second search phase is terminated, if a number of $maxit2$ iterations has been carried out.

3 Parallelization of the two-phase metaheuristic

Since a PC-LAN is available only, a coarse grained parallelization of type 3 is chosen. It comprises assignment and interaction components. The assignment components concern the subdivision of the total task into subtasks, the definition of processes which concurrently carry out the subtasks, and the distribution of the processes over the computing resources. The interaction components serve the control of the process execution and the cooperation of the processes. Here, the assignment components are chosen as follows. The total task, i.e., the solution of a problem instance of the VRPTW, is not subdivided into subtasks. A number of np differently configured processes is rather defined, each running on a LAN-workstation and solving a complete problem instance of the VRPTW. As to the interaction components, the process control is based on a master slave-model, and the process cooperation on a weak concept of cooperative autonomy. In the following, the process control and especially the process cooperation, which has a direct impact on the course of the search, are described in more detail.

Process control concerns the control of the process interaction and the synchronization of the cooperating processes, especially with respect to the coordination of concurrent resource requirements. According to the master slave-model, a master process acts as superordinate control instance and sends control signals to the parallelly executed autonomous, but subordinate, processes, the so-called slave processes. If a slave has carried out the required number of iterations, it sends back a defined control signal to the master and continues the search process. As soon as the master has received this

signal from each slave, the master terminates the concurrent search process by sending out a stop signal to each slave.

The weak concept of cooperative autonomy (for the concept of cooperative autonomy see Enslow 1978) applied for process cooperation is characterized by four properties: (1) concurrent execution of completely autonomous processes, (2) process cooperation through the exchange of solutions, (3) hierarchical process control according to the master slave-model, and (4) process communication by means of a shared memory. As to the process communication, the respective communication object comprises two data items within a common data area:

- The first item keeps a solution *xcomit*, which is the best solution found in an iteration of a first search phase. Solution *xcomit* is from time to time accessed and updated by the cooperating processes during the execution of their first search phase.
- The second item keeps the global best solution, i.e., the best solution over all cooperating processes, designated *xglob*. This item is occasionally accessed and updated by the cooperating processes during the execution of their first and their second search phase.

In order to read and update the two data items, the cooperating processes call the communication procedure *procom* specified in Figure 2. A procedure call takes place after each iteration in the first and in the second search phase. A procedure call passes over four parameters to procedure *procom*: (1) the actual search phase *sphase* („phase1“ or „phase2“), (2) the current number of iterations *iter*, (3) the best solution found in the respective iteration *xit*, and (4) the local best solution *xbest*. In the first search phase *xit* denotes the best of the μ selected descendants with respect to the first lexicographic evaluation function including the four introduced criteria. In the second search phase, however, *xit* is the best of the *c-n* generated neighbours with respect to the vehicle number and the total travel distance. The communication procedure consists of three subsequent blocks. While the first block is executed in any case, the second block is carried out if the calling procedure executes the first search phase, and the third block if the second search phase is executed.

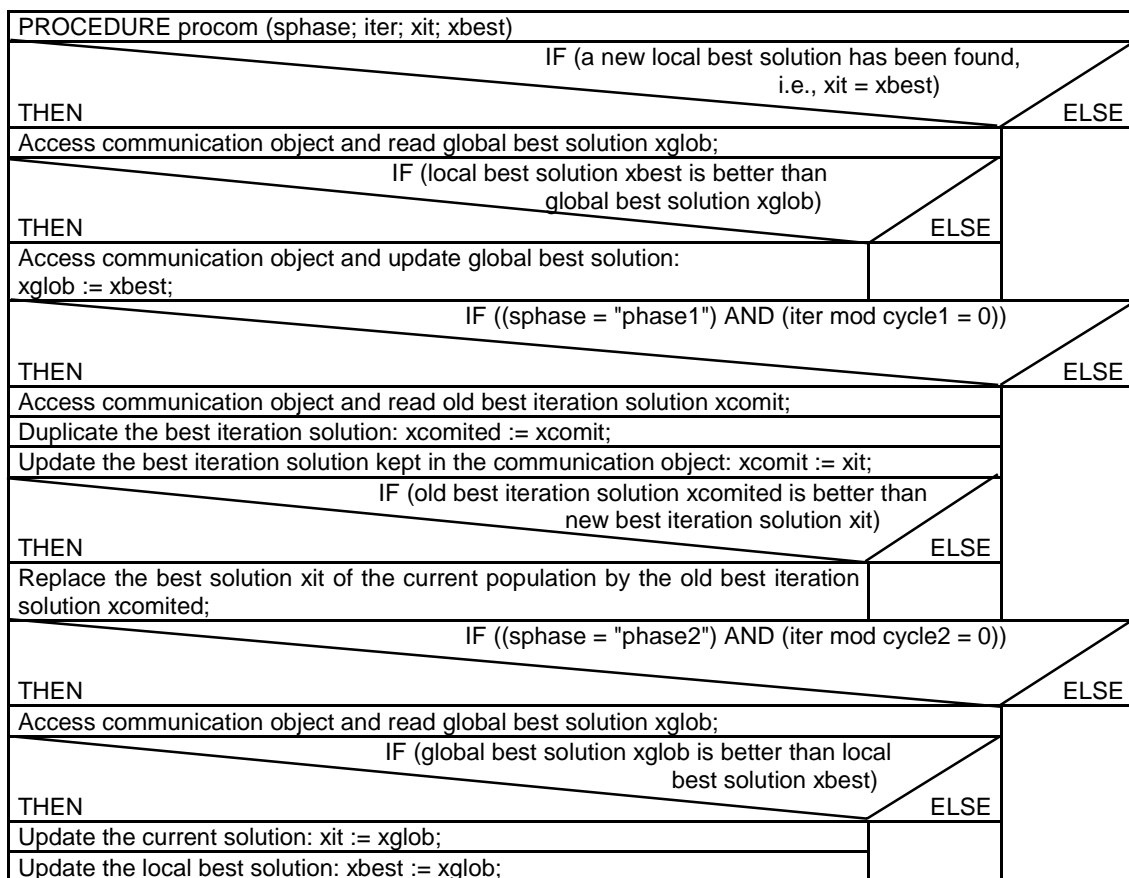


Figure 2. Structure chart for the process communication procedure.

The first block serves the update of the global best solution x_{glob} if the local best solution x_{best} of the calling process is better than the global best solution x_{glob} . Better with respect to the objective function. Since the first block is executed in each iteration of a slave process, the variable x_{glob} contains finally the best calculated solution over all concurrently executed processes.

The second block is carried out repeatedly, i.e., always after a number of $cycle1$ iterations. The aim is to enhance the chances for finding a solution with a smaller vehicle number. For this purpose, the best solution x_{comit} of a preceding population is taken over in the population of the calling process if the old best iteration solution x_{comit} is better than the best solution x_{it} generated by the calling process. But now best with respect to the vehicle number and, in the case of equal vehicle numbers, to the customer number in the smallest tour. In this case, the calling process performs a jump in the solution space and evolves a more promising search path. The best solution x_{it} generated by the calling process is used to update the old iteration solution x_{comit} .

The third block is also carried out repeatedly, but always after a number of $cycle2$ iterations. If the global best solution x_{glob} kept in the shared memory is – with respect to the objective function – better than the actual solution x_{it} of the calling process, then, on the one hand, solution x_{glob} is used as starting point for the further search of the calling process. This jump directs the search to a more promising area of the solution space. On the other hand, solution x_{glob} is used to update the local best solution x_{best} . Hence, the same global best solution cannot be used a further time as starting-point for the further search. In other words, it is prevented that the exchange of solutions causes cycling.

The jumps mentioned above cause an intensification of the search in certain parts of the solution space. In the first search phase, the intention is to explore wide parts of the solution space. Therefore, it is tolerated that the solutions which serve as starting points for an intensification of the search often will fall behind the global best solution. In the second search phase, however, the aim is to minimize the total distance for a given minimum vehicle number. Therefore the global best solution is always used as the starting point for an intensification of the search. The cooperating slave processes are configured in different ways. This is achieved by using different seed values for the generation of random numbers. Given the same starting point for an intensification of the search, the cooperating processes will therefore evolve different search paths.

4 Evaluation of the parallel two-phase metaheuristic

In order to demonstrate the performance of the developed parallel method, a comparative test for different problem sizes has been carried out. In the following, the test problems used, the configuration of the developed method, and the numerical results obtained are described.

The test covers six groups of problem instances. These groups differ in the number of customers per instance. The first group consists of the 56 problem instances introduced by Solomon (1987). Each instance of this group comprises 100 customers. The problem instances of the residual 5 groups G02, G04, G06, G08, and G10 have been generated by the authors. The size of these instances ranges from 200 up to 1000 customers and enables a test which also is based on realistically sized problem instances. For comparative tests these instances are available in the OR-Library from Beasley (1990) (see: <http://www.fernuni-hagen.de/WINF/touren/menuefrm/probinst.htm>).

The 56 problem instances of the first group are subdivided into 6 problem classes (C1, C2, R1, R2, RC1, RC2) with specific properties. A similar subdivision is chosen for the generated 5 problem groups. Each of these groups consists of 60 problem instances which comprise 200, 400, 600, 800 and 1000 customers per instance of the second, third, fourth, fifth and sixth group. The 60 problem instances of each of these 5 groups are always subdivided into 6 problem classes which correspond to the Solomon classes. While the number of problem instances per class varies between 8 and 12 for the 56 Solomon problems, each of the other classes consists of 10 instances.

In the literature a multitude of approaches for solving the VRPTW has been proposed. Here, only some of the best methods published in the recent past can be considered. The methods included in the comparative test are: The sequential metaheuristics from Chiang and Russell (1997), Liu and Shen (1999) and Homberger and Gehring (1999); the type 3 parallelization approaches from Schulze and Fahle (1999) and Gehring and Homberger (1999); the parallelized two-phase hybrid metaheuristic as introduced here with communication, designated HM4C, and without communication, designated HM4.

Method HM4 is derived from method HM4C by means of a simple modification: the reduction of the process communication procedure *procom* to the first block. The result is that in the case of method HM4 four search paths are evolved independently of each other and the best calculated solution is determined as solution to the problem. A comparison of the solutions obtained by the methods HM4 and HM4C will reveal the synergetic effect caused by the process cooperation.

The configuration of the methods HM4 and HM4C concerns the population size (μ), the size of the neighbourhood in the first and the second search phase (λ and $c \cdot n$), the length of the tabu list (*tabul*), the number of iterations in the first and the second search phase (*maxit1* and *maxit2*), the lexicographic function *eval* for evaluating a route schedule xn' in phase 1 (*mdel* for inclusion of criterion "minimal delay" and *ovld* for "caused overload"), and the number of iterations defining the communication frequency in the first and the second search phase (*cycle1* and *cycle2*). Some of these configuration parameters are constant, some vary with the problem size, and some vary between the four slave processes which constitute the parallel methods HM4 and HM4C. Details are given in Table 1.

Table 1. Configuration of the methods HM4 and HM4C.

Parameters	Problem groups					
	Solomon	G02	G04	G06	G08	G10
μ	10	10	10	10	10	10
λ	20	20	20	20	20	20
$c \cdot n$	20·100 = 2000	20·200 = 4000	20·400 = 8000	20·600 = 12000	20·800 = 16000	20·100 = 20000
<i>tabul</i>	10	10	10	10	10	10
<i>maxit1</i> (slave 1)	10000	200	400	600	800	1000
<i>maxit1</i> (slave 2,3,4)	100000	2000	4000	6000	8000	10000
<i>maxit2</i>	100000	2000	4000	6000	8000	10000
<i>eval</i> (slave 1,2,3)	<i>mdel</i>	<i>mdel</i>	<i>mdel</i>	<i>mdel</i>	<i>mdel</i>	<i>mdel</i>
<i>eval</i> (slave 4)	<i>ovld</i>	<i>ovld</i>	<i>ovld</i>	<i>ovld</i>	<i>ovld</i>	<i>ovld</i>
<i>cycle1</i>	3	3	3	3	3	3
<i>cycle2</i>	50	100	200	300	400	500

Table 2. Comparison of results for the problem instances from Solomon (1987).

Problem class	Mean values	Sequential methods			Parallel methods		
		Chiang & Russell (1997)	Liu & Shen (1999)	Homberger & Gehring (1999)	Gehring & Homberger (1999)	HM4	HM4C
C1	MNV	10.00	10.00	10.00	10.00	10.00	10.00
	MTD	828.38	830.06	828.38	829	828.50	828.63
C2	MNV	3.00	3.00	3.00	3.00	3.00	3.00
	MTD	591.42	591.03	589.86	590	589.88	590.33
R1	MNV	12.17	12.17	11.92	12.41	12.08	12.00
	MTD	1204.19	1249.57	1228.06	1201	1208.14	1217.57
R2	MNV	2.73	2.82	2.73	2.91	2.73	2.73
	MTD	986.32	1016.58	969.95	945	965.46	961.29
RC1	MNV	11.88	11.88	11.63	12.00	11.50	11.50
	MTD	1397.44	1412.87	1392.57	1356	1389.65	1395.13
RC2	MNV	3.25	3.25	3.25	3.25	3.25	3.25
	MTD	1229.54	1204.87	1144.43	1140	1126.22	1139.37

Since optimal solutions are known only for some problem instances, the methods are evaluated on the basis of best solutions. In the following, the achieved results are presented as averages over each class of problems. Calculated are always the mean number of vehicles *MNV* and the mean total travel distance *MTD*. The calculation of these means is always based on the best solutions that were achieved for the problem instances of the respective problem class.

The results obtained for the 56 problem instances from Solomon are presented in Table 2. The results may be summarized as follows:

- The results achieved by the parallel method HM4C are comparable to those generated by the best sequential method and significantly better than those calculated with the parallel method from Gehring and Homberger (1999). It has to be considered, however, that for the latter method only one run per problem instance has been carried out.
- The comparison of the results achieved by the methods HM4 and HM4C indicates a small synergetic effect.

The results, i.e., averages *MNV* and *MTD*, for the last five problem groups are given in Table 3. The results may be summarized as follows:

- For problem groups G02 and G04 the cumulated vehicle number CNV is slightly higher for the parallel method HM4C than for the method from Gehring and Homberger (1999), but for the remaining groups G06, G08 and G10 significantly smaller. The reduction of the cumulated vehicle number increases with the problem size.
- The cumulated total travel distance CTD for the parallel method HM4C is, for all problem groups, higher than for the method from Gehring and Homberger (1999). The reason is, that the computing times for the latter method are significantly higher than for method HM4C.
- A synergetic effect cannot be observed for problem group G02. For the remaining groups, however, a synergetic effect increasing with the problem size can be observed. In the case of group G10, the parallelization causes a reduction of the vehicle number by 1 unit for 9 of 60 problem instances.

Table 3. Comparison of results for the problem instances of groups G02, G04, G06, G08 and G10.

Problem class	Mean values	Problem group G02			Problem group G04			Problem group G06		
		Gehring & Homberger (1999)	HM4	HM4C	Gehring & Homberger (1999)	HM4	HM4C	Gehring & Homberger (1999)	HM4	HM4C
C1	MNV	18.90	18.90	18.90	38.00	38.00	38.00	57.90	57.90	57.70
	MTD	2782	2838.93	2842.08	7584	7853.09	7855.82	14792	14802.68	14817.25
C2	MNV	6.00	6.00	6.00	12.00	12.00	12.00	17.90	17.90	17.80
	MTD	1846	1852.63	1856.99	3935	3969.36	3940.19	7787	7758.30	7889.96
R1	MNV	18.20	18.20	18.20	36.30	36.30	36.30	54.50	54.50	54.50
	MTD	3705	3808.27	3855.03	8925	9396.99	9478.22	20854	21441.67	21864.47
R2	MNV	4.00	4.00	4.00	8.00	8.00	8.00	11.00	11.00	11.00
	MTD	3055	3095.33	3032.49	6502	6603.38	6650.28	13335	13520.26	13656.15
RC1	MNV	18.00	18.00	18.10	36.10	36.10	36.10	55.10	55.10	55.00
	MTD	3555	3717.96	3674.91	8763	9139.82	9294.99	18411	18930.50	19114.02
RC2	MNV	4.30	4.40	4.40	8.60	8.90	8.80	11.80	12.10	11.90
	MTD	2675	2651.35	2671.34	5518	5614.49	5629.43	11522	11597.51	11670.29
All	CNV	694	695	696	1390	1393	1392	2082	2085	2079
	CTD	176180	179645	179328	412270	425771	428489	867010	880509	890121

Problem class	Mean values	Problem group G08			Problem group G10		
		Gehring & Homberger (1999)	HM4	HM4C	Gehring & Homberger (1999)	HM4	HM4C
C1	MNV	76.70	76.60	76.10	96.00	96.00	95.40
	MTD	26528	26500.48	26936.68	43273	43409.82	43392.59
C2	MNV	24.00	23.90	23.70	30.20	29.80	29.70
	MTD	12451	11947.43	11847.92	17570	17408.71	17574.72
R1	MNV	72.80	72.80	72.80	91.90	91.90	91.90
	MTD	34586	34646.69	34653.88	57186	56445.13	58069.61
R2	MNV	15.00	15.00	15.00	19.00	19.00	19.00
	MTD	21697	21741.75	21672.85	31930	32004.02	31873.62
RC1	MNV	72.40	72.30	72.30	90.00	90.10	90.10
	MTD	38509	39740.69	40532.35	50668	50904.62	50950.14
RC2	MNV	16.10	16.10	16.10	19.00	18.70	18.50
	MTD	17741	17837.64	17941.23	27012	27039.33	27175.98
All	CNV	2770	2767	2760	3461	3455	3446
	CTD	1515120	1524147	1535849	2276390	2272116	2290367

In summary, it can be stated that the developed parallel method HM4C is undoubtedly on a par with other sequential and parallel methods for solving smaller instances of the VRPTW. For more realistically sized problems with up to 1000 customers results generated with methods from other authors are not available. With respect to the vehicle number, the proposed parallel method HM4C outperforms the parallel method from Gehring and Homberger (1999) for larger problems. The parallelization concept underlying method HM4C causes further a significant synergetic effect which, absolutely, increases with the problem size. It can be expected, that the chosen way of parallelization leads to similar effects in the case of other combinatorial optimization problems.

5 References

- Badeau, P.; Guertin, F.; Gendreau, M.; Potvin, J.-Y. and E.D. Taillard (1997), A parallel tabu search heuristic for the vehicle routing problem with time windows, *Transportation Research* 5, 109-122
- Beasley, J.E. (1990), OR-Library: Distributing test problems by electronic mail, *Journal of the Operational Research Society* 41, 1069-1072
- Chiang, W.-C. and R.A. Russell (1997), A reactive tabu search metaheuristic for the vehicle routing problem with time windows, *INFORMS Journal on Computing* 9, 417-430
- Clarke, G. and J.W. Wright (1964), Scheduling of vehicles from a central depot to a number of delivery points, *Operations Research* 12, 568-581
- Enslow, H. P. (1978), What is a 'distributed' data processing system?, *Computer* 11, 13-21
- Garcia, B.-L.; Potvin, J.-Y. and J.-M. Rousseau (1994), A parallel implementation of the tabu search heuristic for vehicle routing problems with time window constraints, *Computers & Operations Research* 21, 1025-1033
- Gehring, H. and J. Homberger (1999), A parallel hybrid evolutionary metaheuristic for the vehicle routing problem with time windows, in Miettinen, K.; Mäkelä, M.M. and J. Toivanen, eds.: *Proceedings of EUROGEN99 – Short Course on Evolutionary Algorithms in Engineering and Computer Science*, Reports of the Department of Mathematical Information Technology, No. A 2/1999, University of Jyväskylä, Finland, 57-64
- Glover, F. (1986), Future paths for integer programming and links to artificial intelligence, *Computers & Operations Research* 13, 533-549
- Hansen, P. and N. Mladenovic (1999), An introduction to variable neighbourhood search, in Voss, S.; Martello, S.; Osman, I.H. and C. Roucairol, eds.: *Meta-Heuristics: Advances and trends in local search paradigms for optimization*, Boston et al., Kluwer, 431-458
- Homberger, J. and H. Gehring (1999), Two evolutionary metaheuristics for the vehicle routing problem with time windows, *Information Systems and Operational Research, special issue: Metaheuristics for location and routing problems*, 297-318
- Liu, F. and S. Shen (1999), A route-neighbourhood-based metaheuristic for vehicle routing problem with time windows, *European Journal of Operational Research* 118, 485-504
- Or, I. (1976), Traveling salesman-type combinatorial problems and their relation to the logistics of blood banking, Ph.D. thesis, Department of Industrial Engineering and Management Science, Northwestern University, Evanston, IL
- Osman, I.H. (1993), Metastrategy simulated annealing and tabu search algorithms for the vehicle routing problem, *Annals of Operations Research* 41, 421-451
- Potvin, J.-Y. and J.M. Rousseau (1995), An exchange heuristic for routing problems with time windows, *Journal of the Operational Research Society* 46, 1433-1446
- Potvin, J.-Y., Kervahut, T., Garcia, B.-L. and J.-M. Rousseau (1996), The vehicle routing problem with time windows - part I: Tabu search, *INFORMS Journal on Computing* 8, 158-164
- Schulze, J. and T. Fahle (1999), A parallel algorithm for the vehicle routing problem with time window constraints, in Beasley, J.E. and Y.M. Sharaiha, eds.: *Combinatorial optimization: Recent advances in theory and praxis. Special issue of Annals of Operations Research* 86, 585-607
- Solomon, M.M. (1987), Algorithms for the vehicle routing and scheduling problems with time window constraints, *Operations Research* 35, 254-265
- Toulouse, M.; Cranic, T.G. and M. Gendreau (1996), Issues in designing parallel and distributed search algorithms for discrete optimization problems, Publication CRT-96-36, Centre de recherche sur les transports, Université de Montréal, Canada