February 2007

A Fast Look-ahead Heuristic for the Multi-depot Vehicle Routing Problem

Juergen Branke
Institute AIFB, University of Karlsruhe, Germany, branke@aifb.uni-karlsruhe.de

Christian Schmidt
Locom Software GmbH, Karlsruhe, Germany, christian.schmidt@locom.de

Markus Withopf
Institute AIFB, University of Karlsruhe, Germany and Locom Software GmbH, Karlsruhe, Germany

In: Oberweis, Andreas, u.a. (Hg.) 2007. *eOrganisation: Service-, Prozess-, Market-Engineering*; 8. Internationale Tagung Wirtschaftsinformatik 2007. Karlsruhe: Universitätsverlag Karlsruhe

ISBN: 978-3-86644-094-4 (Band 1)
ISBN: 978-3-86644-095-1 (Band 2)
ISBN: 978-3-86644-093-7 (set)

A Fast Look-ahead Heuristic for the Multi-depot Vehicle Routing Problem

J. Branke[1], C. Schmidt[2], M. Withopf[1,2]

[1] Institute AIFB, University of Karlsruhe, Germany
branke@aifb.uni-karlsruhe.de

[2] Locom Software GmbH, Karlsruhe, Germany
christian.schmidt@locom.de

Abstract

The multi-depot vehicle routing problem (MDVRP) is a very challenging part of supply chain optimization. We propose here a simple yet powerful heuristic for the MDVRP with an integrated look-ahead. Compared to other state-of-the-art approaches, our heuristic is significantly faster, but yields competitive results and even found several new best solutions on a set of standard benchmark problems.

1 Introduction

The share of logistics and transportation in the overall cost of a product is increasing. Subsequently, there is a growing pressure in the logistics industry for optimization and cost reduction. In this paper, we look at the multi-depot vehicle routing problem as a typical optimization problem in modern logistics: Given some customers and their demands, three questions have to be answered:

1. From which depot a customer is served
2. On which route/truck a customer is served
3. The sequence in which the customers are served on the route.

All these aspects are interdependent. Nevertheless, many approaches split the problem into two stages to reduce the complexity: first, customers are assigned to depots, and then a classical vehicle routing problem (VRP) is solved for each depot.

The heuristic we propose in this paper also works in stages. It uses a simple construction heuristic for assigning customers to depots, followed by a savings algorithm to solve the resulting VRP for each depot. The savings algorithm uses a look-ahead procedure to approximate the effect a decision has on the final solution quality. Finally, our method applies some simple local improvement heuristics to fine-tune the solution. The overall heuristic is not only competitive to other state-of-the-art heuristics and finds several new best solutions, but it is also very fast.

The paper is structured as follows. First, we provide a more detailed definition of the MDVRP in Section 2. A short overview of related work is provided in Section 3. Section 4 describes the proposed heuristic in detail. Empirical results are summarized in Section 5. The paper concludes with a summary and some ideas for future work.

2 The multi-depot vehicle routing problem

The standard VRP is described as follows: n customers must be serviced from a unique depot. Each customer i has a given demand d_i for goods. Vehicles used for delivery have a maximal capacity C and can drive a maximal distance D before they have to be back at the depot. The goal is to minimize the total transportation cost which is often equated with the total distance driven. A solution to the VRP is a collection of tours starting and ending at the depot, where each customer is assigned to exactly one tour, each tour has length at most D, and the total demand for each tour is at most C.

The multi-depot vehicle routing problem (MDVRP) considered here differs from the standard VRP by having several depots. Tours have to be assigned to depots, i.e. they can start at any depot, but have to return to the same depot they started from. This is a common scenario in logistics. The VRP as well as the MDVRP are NP-complete problems [TV01].

Note that the benchmarks most commonly used in the literature [HEC] have an additional constraint on maximal the number of vehicles per depot.

3 Related work

Since the MDVRP is NP-complete, it is unlikely that exact polynomial time algorithms exist. For this reason, we concentrate on heuristic approaches in this section. The heuristics found in the literature can be roughly categorized into three groups: simple heuristics that attempt to solve the whole problem simultaneously, and heuristics that first assign customers to depots, and then solve a classical VRP problem for each depot in a second stage. As a third group, metaheuristics are discussed.

3.1 Single stage methods

For single-depot VRPs, the savings method is one of the best fast heuristics available[CW64]. Several authors have adapted this method to multi-depot VRPs. In this case, savings additionally depend on the depot the tour is assigned to. A particular difficulty is that the savings values may change when tours are combined. Different authors have addressed this problem either by only allowing combinations that do not change the savings values [Til69], or by re-calculating the savings values when necessary [Weu83, Mat78]. As expected, the latter yields better results but requires more computational effort.

In [TH71], the multi-depot savings method is extended by a look-ahead procedure: For every decision, all possible alternatives are considered on the first level, the 3 best (according to savings values) for the second level, and the single best alternative for the third level. Then the decision on the first level is fixed which allows to realize the highest sum of savings values on the considered three levels. The authors conclude that the improvements obtained by integrating the look-ahead "are marginal and these occur at the expense of greatly increased computer times".

3.2 Two-stage and multi-stage methods

The simplest two-stage procedure assigns each customer to the closest depot, and then solves a VRP for each depot [Mat78]. Gillet and Johnson [GJ76] supplement this idea with local improvement steps moving a customer from one depot to another. Ashour and Bhatt [AB75] construct a minimum spanning tree over all depots and customers and use

this to assign customers to depots.

Salhi and Sari [SS97] proposed a rather complex multi-stage heuristic. We discuss this heuristic in more detail, as it is the basis for the assignment phase of our approach. First, customers are divided into core customers and borderline customers by looking at the ratio between the distance to their nearest depot and the distance to the second-nearest depot. If this ratio is smaller than some parameter ϵ (i.e. the nearest depot is much closer than the second-nearest), the customer is terminally assigned to the nearest depot as core customer. Otherwise, the two closest depots have a similar distance, and the assignment is left open. Note that by varying ϵ between 0 and 1, the percentage of customers assigned to depots as core customers may be varied from 0% to 100%. In a second step, VRPs are solved for each depot with the core customers only. In Step 3, borderline customers are inserted one by one in the order of decreasing opportunity cost, where opportunity cost is the absolute difference between inserting the customer into a tour starting at the closest depot, and inserting the customer into a tour starting at the second closest depot (thereby, the customer can be inserted into an existing tour or a new tour can be created to serve the customer directly from the depot). The customer is inserted where the additional cost is minimal, and the tours are updated. Note that only insertions are allowed that keep the tour valid.

After all customers have been inserted, a multitude of local improvement heuristics are applied, including the removal of tours and reinsertion of its customers elsewhere, the move of a single customers from one tour to another, the swapping of customers between tours etc.

3.3 Metaheuristics

The probably best methods for the MDVRP to date are Tabu Search (TS) metaheuristics (for a general introduction to TS see e.g. [GL98]). The different variants mainly differ in the neighborhood used: In [RLB96], a λ-interchange neighborhood is used. [CGL97] move customers from one tour to another. This may result in infeasible solutions which are then penalized. Finally, the variant proposed in [Sch04] allows to move or swap subsets of customers from one tour to another. The last two variants are responsible for all of

the best known solutions of the 23 standard benchmark problems from [HEC] ([CGL97] found 17 best known solutions, [Sch04] found 6). Evolutionary algorithms have been used in [TS01] and [OH04].

4 Look-ahead Assignment and Routing Heuristic

In this section, we present a new fast two-stage heuristic with look-ahead. In the following, we denote this heuristic as Look-ahead Assignment and Routing (LAR).

The first stage is closely inspired by the complex multi-stage heuristic from Salhi and Sari [SS97]. We start by dividing the customers into core customers and borderline customers depending on the parameter ϵ as described in Section 3 and generate initial tours with only the core customers. The insertion of the borderline customers differs in two major ways: first, we not only consider the two closest depots as potential sources for a customer, but all depots j with $w_c/w_j > \epsilon$, where w_c is the distance to the closest depot, and w_j is the distance to depot j. Second, and more importantly, we allow tours to become infeasible with respect to tour length and/or capacity restrictions when greedily inserting customers.

After all customers have been assigned to depots, we discard the constructed tours and solve a VRP for each depot from scratch, using a savings algorithm with look-ahead. The use of look-ahead is inspired by the use of look-ahead in the multi-depot savings method proposed in [TH71].

In order to make a decision in the current iteration, the look-ahead procedure constructs a decision tree, looking several stages into the future for several alternative choices. Our look-ahead procedure has four configuration parameters:

1. a_1, the number of alternatives considered on the first level,
2. a_2, the number of alternatives considered on the second level,
3. r, the decrease in the number of alternatives considered from one level to the next after stage two. Together with a_2, this determines the number of alternatives considered in level i, as $a_i = \max\{1, a_2 - (i-2)r\}$.

4. m, the maximum number of levels considered.

So, while the standard savings algorithm in each iteration connects the two tours with the largest savings value, the look-ahead savings algorithm considers the a_1 alternatives with the largest savings value. For each of these, it tries the a_2 alternatives with then largest savings value, and so on for m levels. Finally, out of the a_1 alternatives, it selects the one that allowed to achieve the largest total savings. An example is given in Figure 1.

After the VRPs are solved, we use three simple local improvement heuristics to generate the final solution. These improvement heuristics are 2-opt and 3-opt applied to individual tours, and an inter-tour swap operator that attempts to swap some customers between two tours. Thereby, all possible sub-tours of length at most 3 are examined and it is tested, whether an insertion into any other tour at any possible location would be feasible and better. These local improvement steps are iterated three times.

As mentioned above, the most commonly used MDVRP benchmark problems [HEC] additional restrict the number of vehicles per depot. LAR can not handle such a constraint explicitly, because the assignment may assign more customers to a depot than can be handled by the vehicles, and because the savings heuristic does not allow to restrict the resulting number of vehicles. However, a simple twist allowed us to generate feasible solutions for all cases: We simply limit the capacity of a depot in the assignment step to a fraction p of its maximal capacity (i.e. the maximal number of vehicles times vehicle capacity). Once the assigned customers reach the depot's capacity, it is no longer considered as a potential source for the remaining customers. The fraction p is an additional parameter, but a default setting of $p = 0.975$ seemed to work well in most cases.

5 Empirical results

5.1 Experimental setup

In order to evaluate the quality of our LAR procedure, we tested it on 33 standard benchmarks from the literature, all publicly available at [HEC].

The first 23 test instances are classical benchmarks from the literature. Instances 1-7 are based on some single-depot VRPs described in [CE69], instances 8-11 are proposed in

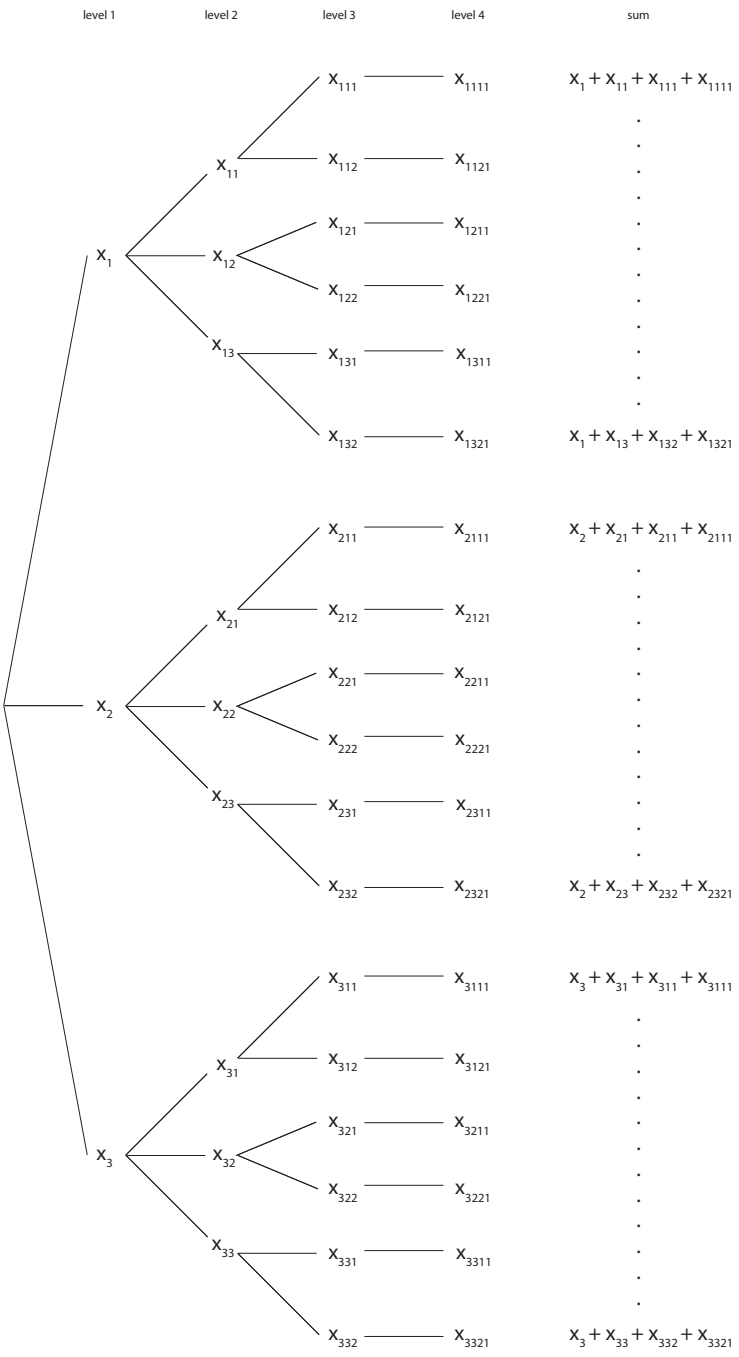

Figure 1: Decision tree for parameters $a_1 = 3, a_2 = 3, r = 1, m = 3$. The variable x denotes the savings realized by certain choices, e.g., x_{ij} denotes the savings realized if the ith highest savings value is selected in the first step, and the jth highest savings value is selected in the second step.

[GJ76], and instances 12-23 are from [CGW93]. The last ten instances have been proposed more recently in [CGL97]. Some characteristics of the test problems are summarized in Table 1. All instances are based on Euclidean distances.

Our parameter settings for the algorithm were as follows: For parameter ϵ used during assignment, all values in $\{0.4, 0.5, 0.6, \ldots 1.0\}$ were used (i.e. we ran the algorithm 7 times and report on the best found value here). For look-ahead, we set $a_1 = 20, a_2 = 1, r = 0, m = 50$. To keep the number of vehicles within the allowed limits, we set $p = 0.975$.

5.2 Results on standard benchmarks

The results for the first 23 instances are summarized in Table 2, reporting on the running time (T) and the deviation from the best known solution (Δ) for different algorithms from the literature. The last column ("Best") reports on the quality of the best known solution, and where this solution has been published. Note that [CGL97] report on two types of results: The runs with default parameter setting listed in columns 4/5 of the table, and results obtained by "playing around" with algorithm parameters, e.g. increasing the number of iterations, often resulting in the best known solution listed in the last column. Of all the approaches reported, only those from [CGL97, Sch04, TS01] take into account the constraint on the number of vehicles per depot. In [RLB96, SS97, OH04], this constraint is simply ignored.

As can be seen, the solution found by LAR is generally very close to the best known solution. For problem instance 8, we even found a new best solution, which is quite remarkable given that these problems have been thoroughly tested by many researchers. Although a direct comparison of the runtimes of the different heuristics is difficult due to differences in hardware (see footnote in Table 2 for underlying hardware specifications), it is obvious that LAR is very fast (5 seconds runtime on average).

Table 1: Characteristics of test instances.

			Vehicle restrictions			
Problem	# Customers	# Depots	Capacity	Distance[a]	#Vehicles[b]	Utilization[c]
1	50	4	80	∞	4	60,70%
2	50	4	160	∞	2	60,70%
3	75	2	140	∞	3	64,95%
4	100	2	100	∞	8	91,13%
5	100	2	200	∞	5	72,90%
6	100	3	100	∞	6	81,00%
7	100	4	100	∞	4	91,13%
8	249	2	500	310	14	86,47%
9	249	3	500	310	12	67,26%
10	249	4	500	310	8	75,66%
11	249	5	500	310	6	80,71%
12	80	2	60	∞	5	72,00%
13	80	2	60	200	5	72,00%
14	80	2	60	180	5	72,00%
15	160	4	60	∞	5	72,00%
16	160	4	60	200	5	72,00%
17	160	4	60	180	5	72,00%
18	240	6	60	∞	5	72,00%
19	240	6	60	200	5	72,00%
20	240	6	60	180	5	72,00%
21	360	9	60	∞	5	72,00%
22	360	9	60	200	5	72,00%
23	360	9	60	180	5	72,00%
24	48	4	200	500	1	82,13%
25	96	4	195	480	2	78,21%
26	144	4	190	460	3	78,42%
27	192	4	185	440	4	83,68%
28	240	4	180	420	5	93,08%
29	288	4	175	400	6	87,40%
30	72	6	200	500	1	79,00%
31	144	6	190	475	2	87,98%
32	216	6	180	450	3	84,44%
33	288	6	170	425	4	94,36%

[a] ∞: no tour length restriction

[b] max. number of vehicles per depot

[c] $Utilisation = \frac{total\ demand\ of\ all\ customers}{\#Depots \cdot \#Vehicles \cdot Capacity}$

Table 2: Performance comparison of several methods

Problem	[RLB96]		[CGL97]		[SS97]		[TS01]		[OH04]	LAR		Best[a]	
	Δ	T[b]	Δ	T[b]	Δ	T[b]	Δ	T[b]	Δ	Δ	T[b]	c(s*)	
1	**0.00%**	03:12	**0.00%**	03:14	1.89%	00:06	2.58%	05:36	4.77%	0.87%	00:01	576.87	C
2	**0.00%**	04:48	0.07%	03:28	2.34%	00:06	1.79%	03:06	1.37%	1.37%	00:01	473.533	C
3	**0.00%**	05:48	0.62%	05:40	0.74%	00:06	8.31%	04:24	10.25%	1.61%	00:01	641.186	C
4	**0.24%**	11:24	0.52%	07:47	4.64%	01:18	6.08%	08:24	2.33%	4.28%	00:03	1001.47	S
5	**0.03%**	12:48	0.44%	08:13	3.62%	00:18	0.64%	10:18	4.68%	2.33%	00:03	750.029	C
6	**0.00%**	08:24	0.15%	07:39	1.38%	00:18	11.35%	09:42	3.69%	1.38%	00:01	876.5	C
7	0.89%	06:48	**0.82%**	07:43	3.87%	01:54	10.38%	07:24	3.77%	4.25%	00:01	884.664	S
8	1.17%	09:24	1.11%	25:26	1.80%	14:06	8.55%	41:48	5.79%	**-0.14%**	00:21	4433.31	S
9	1.56%	41:12	**1.12%**	26:44	3.30%	07:06	10.50%	31:24	9.35%	1.64%	00:15	3877.37	S
10	**0.39%**	43:00	1.63%	25:30	4.64%	04:30	17.41%	45:30	9.02%	2.78%	00:12	3655.18	S
11	2.67%	36:24	**0.75%**	25:55	4.51%	04:06	15.15%	37:24	9.19%	3.09%	00:07	3554.18	C
12	**0.00%**	05:24	**0.00%**	05:34	0.60%	00:18	7.81%	04:48	**0.00%**	**0.00%**	00:02	1318.95	C
13	**0.00%**	04:48	**0.00%**	05:35	**0.00%**	00:18	**0.00%**	05:18	**0.00%**	**0.00%**	00:01	1318.95	C
14	0.41%	02:36	**0.00%**	05:26	**0.00%**	00:18	**0.00%**	05:48	0.41%	0.41%	00:02	1360.12	C
15	1.84%	15:30	**1.15%**	14:04	3.24%	01:06	22.10%	07:54	2.95%	2.52%	00:03	2505.42	C
16	**0.00%**	11:06	**0.00%**	14:03	0.48%	01:06	5.74%	09:54	0.61%	0.19%	00:03	2572.23	C
17	0.82%	05:48	**0.41%**	13:42	**0.41%**	00:42	6.85%	11:48	0.82%	0.82%	00:03	2709.09	C
18	2.11%	23:12	**0.21%**	24:51	4.06%	01:54	47.53%	20:18	5.43%	3.11%	00:05	3702.84	S
19	**0.00%**	22:00	**0.00%**	25:12	1.07%	01:54	3.39%	28:18	1.92%	0.21%	00:05	3827.06	C
20	0.96%	10:00	**0.00%**	24:43	0.41%	01:24	7.07%	31:48	0.96%	0.96%	00:05	4058.07	C
21	3.32%	48:42	**1.12%**	48:10	5.73%	03:36	25.52%	24:42	8.25%	3.73%	00:08	5474.83	C
22	0.28%	33:30	0.24%	48:54	0.71%	03:42	4.97%	27:24	3.71%	**0.14%**	00:08	5702.15	C
23	0.82%	17:18	0.73%	47:52	**0.18%**	03:18	3.16%	29:42	0.82%	0.82%	00:08	6095.46	C
Avg.	**0.76%**	**19:16**	**0.48%**	**18:30**	**2.16%**	**02:20**	**9.86%**	**17:57**	**3.92%**	**1.58%**	**00:05**		

[a] Origin of best known solution: C [CGL97] with tuned parameters and longer runtime, S [Sch04]

[b] Runtime in minutes on the following hardware: Sun 4/370 [RLB96], Sun Sparcstation 10 [CGL97], VAX 4000-500 [SS97], DEC Alpha [TS01], and Pentium4/2.8 GHz for LAR

Table 3: Performance comparison of LAR with the Tabu search algorithm proposed by Cordeau et al. [CGL97] on the benchmarks proposed there.

	[CGL97]	LAR		
Problem	c(s)	c(s)	Δ	T
24	861,32	861,319	0,00%	00:00
25	1307,61	1349,92	3,24%	00:01
26	1806,60	1805,7	**-0,05%**	00:02
27	2072,52	2172,05	4,80%	00:05
28	2385,77	2428,23	1,78%	00:09
29	2723,27	2710,39	**-0,47%**	00:13
30	1089,56	1116,19	2,44%	00:00
31	1666,60	1804,08	8,25%	00:01
32	2153,10	2217,94	3,01%	00:04
33	2921,85	3228,52	10,50%	00:08
Avg.			**3,35%**	**00:04**

The results for the 10 instances proposed in [CGL97] can be found in Table 3. On these test problems, we improved on the best known solutions for 2 out of the 10 problem instances. On average, LAR performed 3.35% worse than the Tabu search algorithm used in [CGL97], which is mainly due to the bad performance on Problem 31 and 33.

Over all 33 test instances, LAR has a deviation from the best known solution of 2.11%. While this is worse than the 0.33% obtained by the TS used in [CGL97], LAR is significantly faster (based on the running times observed in Table 2 – the running time for Tabu search on problem instances 24-33 was not reported in [CGL97]).

5.3 Design choices

In this section, we examine for some algorithm design choices their effect on solution quality. For this purpose, we compare three algorithmic variants:

1. Our proposed LAR

2. LAR where we require feasibility of tours when assigning borderline customers to depots (as in the heuristic from Salhi and Sari [SS97])

3. A method which requires feasibility of tours when assigning borderline customers to depots, and then takes the resulting tours and applies local improvement. This is similar to the heuristic proposed by Salhi and Sari [SS97], although they had some more local improvement heuristics.

All these algorithms are followed by the usual three iterations of our local improvement heuristics 2-opt, 3-opt and Swap.

As summarized above, in these tests, LAR achieved an average deviation from the best known solution of 2.11%. If we insist on feasible tours during the assignment stage (as in [SS97]), the deviation increases to 2.77%. If we additionally omit the re-planning of tours with the look-ahead savings algorithm and only apply local improvement (as in [SS97]), the deviation increases further to 5.27%. On the other hand, omitting the re-planning of tours reduces the running time by approximately 50%.

5.4 Parameter sensitivity

In this subsection, we have a closer look at the influence of the parameter settings. Figure 2 looks at the influence of the number of first-level alternatives in the look-ahead procedure on the solution quality. As can be seen, considering only slightly more than just the best alternative improves performance significantly. After about 5 considered alternatives, additional alternatives improve the results only slightly, while the running time keeps increasing linearly.

Similarly, looking some stages ahead can help a lot, while after about 50 stages, the additional effect becomes negligible, see Figure 3

Overall, the heuristic seems to be fairly robust with respect to parameter settings. Also, note that we always used the same parameter settings for all 33 problems, which differ significantly in size and structure.

We also tested what solution quality we could obtain by tuning the parameters for each problem separately. To this end, we tested all combinations of the following parameter settings: $a_1 \in \{1, 2, 5, 10, 20, 30, 40, 50, 60, 70, 80, 100\}$, $a_2 \in \{1, 2, 3, 4, 5\}$, $m \in \{1, 5, 10, 20, 30, 40, 50\}$, $p \in \{0.8, 0.9, 0.95, 0.975, 1.0\}$. The best parameter settings for

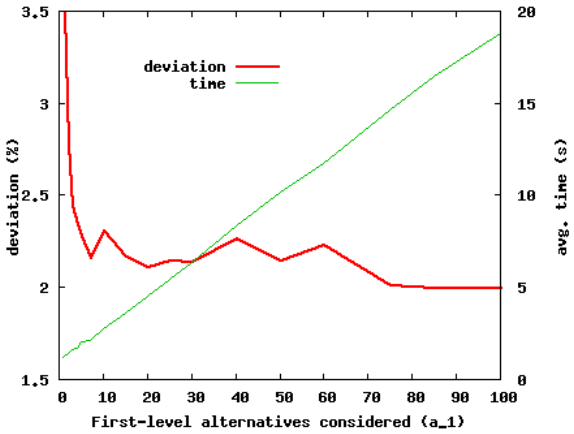

Figure 2: Average deviation over all 33 test problems depending on maximal number of alternatives considered by look-ahead procedure on first level, a_1. $m = 50$.

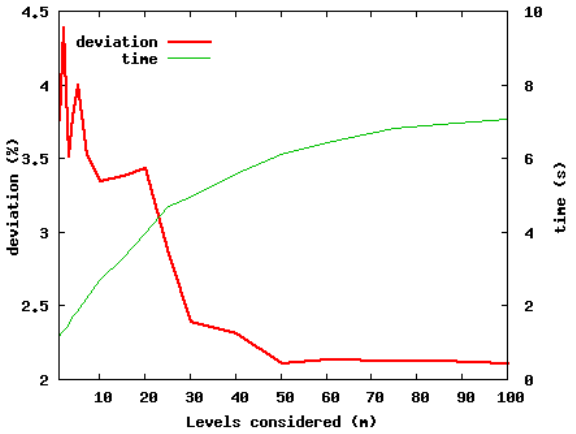

Figure 3: Average deviation over all 33 test problems depending on maximal number of levels for look-ahead, m. $a_1 = 20$.

each problem, together with the best solution found, are summarized in Table 4. Parameter fine-tuning further improved performance from an average error of 2.11% with standard parameter settings to an average error of 1.15%. Furthermore, in addition two of the three new best solutions reported above could be further improved, and for two more problems new best solutions were found (making a total of 5 new best solutions).

Table 4: Best results obtained with different parameter settings

	Best		LAR								
Problem	c(s*)		ϵ	a_1	a_2	r	m	p	c(s)	Δ	
1	576,87	C	0,5	10	1	1	20	0,80	577,46	0,10%	
2	473,533	C	1,0	20	1	1	50	1,00	480,04	1,37%	
3	641,186	C	0,9	20	1	1	50	1,00	651,479	1,61%	
4	1001,47	S	0,4	100	5	1	50	1,00	1023,35	2,18%	
5	750,029	C	0,3	100	5	1	50	1,00	762,58	1,67%	
6	876,5	C	0,5	70	4	1	50	1,00	885,91	1,07%	
7	884,664	S	0,8	100	1	1	100	0,98	912,198	3,11%	
8	4433,31	S	1,0	20	1	1	50	1,00	4427,16	**-0,14%**	
9	3877,37	S	0,8	100	1	1	100	0,98	3922,21	1,16%	
10	3655,18	S	0,8	20	1	1	50	1,00	3749,17	2,57%	
11	3554,18	C	0,8	5	3	1	50	1,00	3627,00	2,05%	
12	1318,95	C	1,0	2	1	1	5	1,00	1318,95	0,00%	
13	1318,95	C	1,0	2	1	1	5	1,00	1318,95	0,00%	
14	1360,12	C	1,0	1	1	1	1	1,00	1365,69	0,41%	
15	2505,42	C	1,0	2	1	1	5	1,00	2568,50	2,52%	
16	2572,23	C	1,0	20	1	1	5	1,00	2577,12	0,19%	
17	2709,09	C	1,0	1	1	1	1	1,00	2731,37	0,82%	
18	3702,84	S	0,1	30	1	1	50	0,90	3818,05	3,11%	
19	3827,06	C	1,0	2	1	1	2	1,00	3835,28	0,21%	
20	4058,07	C	1,0	1	1	1	1	1,00	4097,05	0,96%	
21	5474,83	C	0,4	20	1	1	50	1,00	5681,03	3,77%	
22	5702,15	C	1,0	2	1	1	5	1,00	5710,38	0,14%	
23	6095,46	C	1,0	1	1	1	1	1,00	6145,58	0,82%	**1,29%**[a]
24	861,32	C	0,4	100	1	1	100	0,98	861,32	0,00%	
25	1307,61	C	0,1	150	1	1	150	0,98	1302,77	**-0,37%**	
26	1806,6	C	0,7	100	1	1	100	0,98	1792,62	**-0,77%**	
27	2072,52	C	0,6	100	1	1	100	0,95	2091,01	0,89%	
28	2385,77	C	0,7	150	1	1	150	0,98	2374,41	**-0,48%**	
29	2723,27	C	0,6	100	1	1	50	0,95	2697,85	**-0,93%**	
30	1089,56	C	0,5	100	1	1	100	0,85	1102,45	1,18%	
31	1666,6	C	0,6	100	1	1	100	0,98	1731,77	3,91%	
32	2153,1	C	0,8	150	1	1	150	0,98	2165,64	0,58%	
33	2921,85	C	0,8	150	1	1	150	0,98	3048,79	4,34%	
										1,15%[b]	

[a] avg. deviation on problem instances 1 to 23

[b] avg. deviation over all 33 problem instances

6 Conclusion

In this paper, we have considered the multi-depot vehicle routing problem (MDVRP), which extends the standard vehicle routing problem (VRP) by allowing servicing a customer from one of several depots, which is a common scenario in practical transportation problems. To solve this problem, we have proposed a new heuristic which first assigns customers to depots, and then uses a savings heuristic with integrated look-ahead to solve the VRP for each depot. An extensive empirical comparison on 33 standard benchmark problems showed that our method is not only competitive with other state-of-the-art procedures, but also much faster. For 5 out of the 33 benchmark problems, new best solutions have been found.

Overall, we conclude that our proposed method is a very fast and simple yet very effective heuristic for the MDVRP. As a next step, we are working on hybridizing this heuristic with an evolutionary algorithm to obtain even better results.

References

[AB75] S. Ashour and P. Bhatt. A graph-theoretical approach for delivery problems. Technical report, unpublished, 1975.

[CE69] N. Christofides and S. Eilon. An algorithm for the vehicle-dispatching problem. *Operational Research Quarterly*, 20(3):309–318, 1969.

[CGL97] J.F. Cordeau, M. Gendreau, and G. Laporte. A tabu search heuristic for periodic and multi-depot vehicle routing problems. *Networks*, 30:105–119, 1997.

[CGW93] I-M. Chao, B. Golden, and E. Wasil. A new heuristic for the multi-depot vehicle routing problem that improves upon best-known solutions. *American Journal of Mathematical and Management Sciences*, 13(3):371–406, 1993.

[CW64] G. Clarke and J.W. Wright. Scheduling of vehicles from a central depot to a number of delivery points. *Operations Research*, 12:568–581, 1964.

[GJ76] B.E. Gillet and J.G. Johnson. Multi-terminal vehicle-dispatch algorithm. *Omega*, 4:711–718, 1976.

[GL98] F. Glover and M. Laguna. *Tabu Search*. Kluwer, 1998.

[HEC] `http://www.hec.ca/chairedistributique/data/`.

[Mat78] F. Matthus. *Tourenplanung - Verfahren zur Einsatzdisposition von Fuhrparks*. S.Toeche-Mittler Verlag, Darmstadt, 1978.

[OH04] B. Ombuki and F. Hanshar. An effective genetic algorithm for the multi-depot vehicle routing problem. Technical report, Brock University Ontario, 2004.

[RLB96] J. Renaud, G. Laporte, and F. F. Boctor. A tabu search heuristic for the multi-depot vehicle routing problem. *Computers & Operations Research*, 23:229–235, 1996.

[Sch04] S. Scheuerer. *Neue Tabusuche-Heuristiken für die logistische Tourenplanung bei restringierendem Anhängereinsatz, mehreren Depots und Planungsperioden*. PhD thesis, Universität Regensburg, 2004.

[SS97] S. Salhi and M. Sari. A multi-level composite heuristic for the multi-depot vehicle fleet mix problem. *European Journal of Operational Research*, 103:95–112, 1997.

[TH71] F. A. Tillman and R. W. Hering. A study of a look-ahead procedure for solving the multiterminal delivery problem. *Transportation Research*, 5:225–229, 1971.

[Til69] F.A. Tillman. The multiple terminal delivery problem with probabilistic demands. *Transportation Science*, 3:192–204, 1969.

[TS01] S.R. Thanghia and S. Salhi. Genetic clustering: An adaptive heuristic for the multidepot vehicle routing problem. *Applied Artificial Intelligence*, 15:361–383, 2001.

[TV01] P. Toth and D. Vigo. *The vehicle routing problem.* Society for Industrial and Applied Mathematics, 2001.

[Weu83] H.-K. Weuthen. *Tourenplanung - Lösungsverfahren für Mehrdepot - Probleme.* PhD thesis, Universität Karlsruhe TH, 1983.