Multilevel Variable Neighborhood Search for Periodic Routing Problems*

Sandro Pirkwieser and Günther R. Raidl

Institute of Computer Graphics and Algorithms
Vienna University of Technology, Vienna, Austria
{pirkwieser,raidl}@ads.tuwien.ac.at

Abstract. In this work we present the extension of a variable neighborhood search (VNS) with the multilevel refinement strategy for periodic routing problems. The underlying VNS was recently proposed and performs already well on these problems. We apply a path based coarsening scheme by building fixed (route) segments of customers accounting for the periodicity. Starting at the coarsest level the problem is iteratively refined until the original problem is reached again. This refinement is smoothly integrated into the VNS. Further a suitable solution-based recoarsening is proposed. Results on available benchmark test data as well as on newly generated larger instances show the advantage of the multilevel VNS compared to the standard VNS, yielding better results in usually less CPU time. This new approach is especially appealing for large instances.

1 Introduction

Periodic routing problems are a generalized variant of the classical routing problem where customers must be served several times in a given planning period instead of only once on a single day. Applications exist in many real-world scenarios as in courier services, grocery distribution, waste collection, or for various sorts of suppliers. The *Periodic Vehicle Routing Problem* (PVRP) is defined on a directed graph $G = (V, A)$, where $V = \{v_0, v_1, \dots v_n\}$ is the vertex set and $A = \{(v_i, v_j) : v_i, v_j \in V, i \neq j\}$ is the arc set. A planning horizon of t days, referred to by $T = \{1, \dots, t\}$, is considered. Vertex v_0 represents the depot at which are based m vehicles with positive capacities Q_k and maximum route durations D_k, $k = 1, \dots, m$. Each vertex of $V \setminus \{v_0\}$ corresponds to a city or a customer and has a nonnegative demand q_i, a nonnegative service duration d_i, a service frequency f_i, and a non-empty set $C_i \subseteq \{T' \mid T' \subseteq T,\ |T'| = f_i\}$ of allowed combinations of visit days. Each arc $(i, j) \in A$ has associated a nonnegative travel time (costs) c_{ij}. The PVRP consists of selecting one visit combination per customer and designing (at most) m vehicle routes on each day on G such that

1. each route starts and ends at the depot,

* This work is supported by the Austrian Science Fund (FWF) under contract number P20342-N13.

2. each customer i belongs to f_i routes over the planning horizon corresponding to a feasible visit combination in C_i,
3. the total demand of the customers on a route followed by vehicle k does not exceed Q_k, and its total duration does not exceed D_k, and
4. the total travel times (costs) over all routes is minimized.

We further consider a special case of the PVRP, the *Periodic Traveling Salesman Problem* (PTSP), where $m = 1$ and the vehicle capacity as well as the route duration are unconstrained.

As all available benchmark instances for both problems assume a homogeneous vehicle fleet, and vehicle capacities and maximum route durations are supposed to be independent of the day, we also restrict ourselves to this situation.

Multilevel refinement strategies [1, 2] successively coarsen an initial problem instance, yielding a sequence of instances of decreasing size, representing the problem on different abstraction levels. The smallest instance is then solved by some auxiliary technique (e.g. a metaheuristic), and the obtained solution is *extended* to a solution of the previous level. This solution is eventually improved (e.g. again by some metaheuristic) and the solution extension continued until a solution for the original problem is obtained. Optionally, the whole process is iterated; hereby, a *recoarsening* is performed exploiting the last obtained solution in order to derive an eventually better hierarchy of abstraction levels. Multilevel refinement strategies have been successfully applied to several combinatorial optimization problems, including graph partitioning, the traveling salesman problem, and also the classical capacitated vehicle routing problem. When applied sensible, they seem to be able to often improve the scalability of metaheuristics, either improving running times or final solution qualities.

The subject of this work therefore is to study the extension of a variable neighborhood search (VNS) metaheuristic that was already successfully applied to periodic routing problems by a multilevel refinement strategy in order to improve the performance of the VNS especially on larger instances.

In the following section we review related work. The underlying VNS and the multilevel extension are presented in Sections 3 and 4, respectively. Experimental results, in which also new larger benchmark instances are used, are discussed in Section 5. Finally, conclusions and an outlook on future work is given in Section 6.

2 Related Work

From a problem oriented view recent successful approaches for one or both of the described problems are [3], [4], [5], and [6]. Cordeau et al. [3] describe a Tabu search for periodic and multi-depot routing problems, achieving for all previously known benchmark instances new best results at that time. The algorithm is based on rather simple standard neighborhoods that move single customers to different routes or change single visit combinations. New random test instances have further been introduced in this work. A well performing construction type algorithm with an embedded improvement procedure for the PTSP is presented by Bertazzi et al. [4]. Alegre et al. [5] apply a Scatter search heuristic tailored

especially to solve PVRP instances having a long planning horizon. Nevertheless they also obtained improved results for many standard benchmark instances. Most recently Hemmelmayr et al. [6] tackled the PVRP and PTSP with a sophisticated VNS that again yielded many new best results. As neighborhoods for diversification they utilized moving or exchanging route segments of different maximal size as well as changing several visit combinations. As improvement procedure they applied the well-known 2-opt and a restricted 3-opt. Similar to [3] they also allow infeasible solutions during the search but additionally apply an acceptance criterion similar to Simulated Annealing [7]. Francis et al. [8] gave a recent survey on periodic routing problems considering several variants of it.

A VNS based on the one from [6] has further been described for the PVRP with time windows in [9]. The concept of multilevel refinement including an overview on applications is covered in [1, 2]. We point out the work by Rodney et al. [10] which introduces a multilevel refinement strategy for the capacitated vehicle routing problem. They coarsen the problem by building paths (segments) of customers through fixing the corresponding edges and consider such a path as an atomic unit in the following. Due to the capacity restrictions they propose to balance these paths according to the accumulated demand and systematically allow a gradually decreasing violation of the limiting constraint. Further the multilevel strategy has been previously applied to the traveling salesman problem [11, 12] in a similar way. Yet we are not aware of an approach that was designed to handle the periodicity as well as to accomplishing such a smooth integration into a metaheuristic. Further, to our knowledge VNS has not been combined with multilevel refinement so far.

3 Underlying Variable Neighborhood Search

Variable neighborhood search (VNS) [13] in general utilizes random moves in a series of neighborhoods of growing size for diversification, referred to as shaking, and usually an embedded local search component for intensification. The core of the VNS is parameter free (after deciding about the neighborhoods), making it relatively easy to use. In the last decade this metaheuristic has been successfully applied to a wide range of combinatorial optimization problems. In the following, we give a rather short overview on our underlying VNS as most parts of it have already been described in more detail in [6, 9].

To smooth the search space and help escaping local optima, the VNS relaxes the restrictions on vehicle load and route duration and adds penalties corresponding to the excess of these constraints to the cost function; similar to [9] a constant penalty factor of 1000 turned out to work reasonably well for the benchmark instances from literature and is therefore used, whereas an adaptive penalty was applied in [6]. Initially a possible visit combination is selected for each customer at random. Afterwards we apply the Clarke and Wright savings algorithm [14] in case of multiple routes. If we end up with too many routes, the customers of those routes holding the least customers are relocated in a greedy way (for details we refer to [6]). By doing so the constructed routes might ex-

ceed maximal vehicle load or allowed tour duration. For instances with a single vehicle, i.e. especially for all PTSP instances, we make use of best insertion.

In the shaking phase we utilize three different neighborhood structures with increasing perturbation size per type: (i) randomly changing up to six visit combinations with greedy insertion for the new visit days, whereas for the PVRP we also allow reassigning the same visit combination, (ii) moving a random segment of up to three customers of a route to another one on the same day, and (iii) exchanging two random segments of up to three customers between two routes on the same day. We thus have a total of 12 shaking neighborhoods (i.e. $k_{\max} = 12$) which are always considered in a fixed order. These settings reflect previous experience and showed a good performance in preliminary tests. Contrary to [6] only segments of up to three instead of six customers are exchanged, we recognized no performance gain when doing so.

For intensification we apply the well-known 3-opt intra-route exchange procedure in a first improvement fashion for the PVRP and 2-opt for PTSP (restricting the latter to segments of length 10 in case of $n > 300$), only considering routes changed during shaking. Both are applied as long as an improvement is achieved, i.e. until a local optimum is reached. Additionally each new PVRP incumbent solution is subject to a 2-opt* inter-route exchange heuristic [15]. Hereby for each pair of routes of the same day all possible exchanges of the routes' end segments are considered. In case of the PTSP we additionally apply 3-opt on all routes. This is an addition to the work of [6].

To enhance the overall VNS performance Hemmelmayr et al. [6] propose to not only accept better solutions, but sometimes also solutions having a worse objective value. This is done in a systematic way using the Metropolis criterion like in simulated annealing [7]. A linear cooling scheme is used in a way such that the acceptance rate of worse solutions is nearly zero in the last iterations.

4 Multilevel Variable Neighborhood Search

In this section we extend the described VNS with ideas from the multilevel refinement strategy, hence we call it the *Multilevel VNS* (MLVNS). The basic idea of multilevel refinement [1, 2] is to suitably coarsen the problem which has two effects: (i) to concentrate more on the costly parts of the problem during optimization, and (ii) at the same time to (temporarily) reduce its size, making it more tractable at the coarser levels. The problem is then (approximately) solved at the highest level and the obtained solution refined in an iterative way until a solution to the original problem is reached.

Contrary to most existing multilevel refinement approaches, our method

- does not automatically obtain one specific feasible solution at the coarsest level to start with, which is due to the periodicity in our case, and
- does not have several subsequent (and even independent) applications of the same method on different levels, but smoothly integrates the transitions from the most abstract level to the original problem in the VNS; i.e. many levels with small changes (in fact always only one) are used.

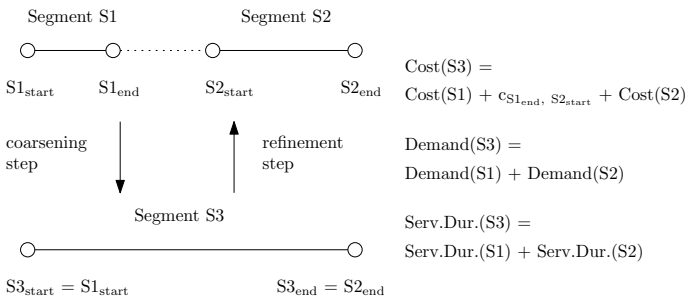


Fig. 1. Coarsen: merge two segments into one, refine: split a segment into two.

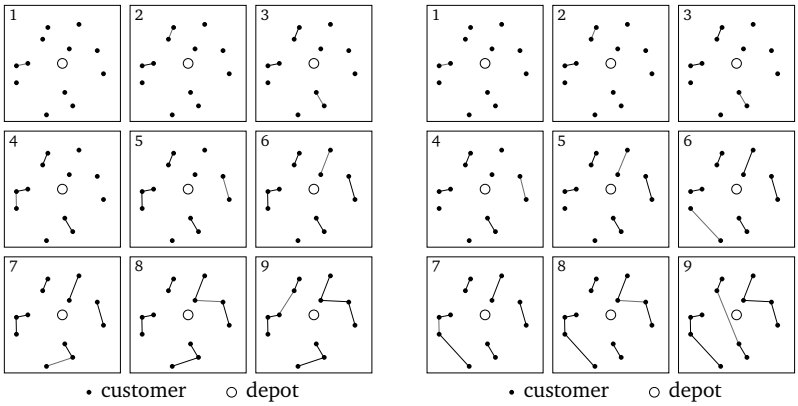
In the following we propose an initial coarsening process operating at a given problem instance, as well as a recoarsening based on an incumbent solution.

4.1 Initial Problem Coarsening

Two common coarsening schemes on graphs are based on either merging nodes (into "super-nodes") or building segments (paths). Since we have to determine sequences of customer visits it is rather natural to follow the latter scheme here, keeping in line with previous work [11, 10]. We opt for an exact coarsening, i.e. the objective value of a coarsened solution always equals the one of the corresponding fully refined solution. Basically during coarsening the nodes of the graph (i.e. single customers) are merged to segments via fixing a specific edge between them. One such coarsening step of merging two segments into a longer segment and the corresponding reversed refinement step are shown in Figure 1. Note that a single customer can be regarded a segment with equal start and end point, too. The accumulated cost, demand, and service duration of the newly created segment must be set accordingly.

Due to the periodicity, building customer sequences on a daily basis as for the single day of the classical vehicle routing problem would not make much sense. On the one hand this would require to choose a visit day combination per customer in advance (a bad choice would potentially result in a bad coarsening) and on the other hand such segments would most likely not allow to change its visit day combination without the need of breaking it apart, thus creating only short-lived segments being inconsistent with the general idea of the multilevel refinement strategy. Therefore it is necessary to apply a coarsening process respecting the periodicity and building segments spanning the whole planning horizon. This is achieved by incorporating the customers' sets of available visit day combinations—the service frequencies alone might not be sufficient—and allow customers and subsequently segments to be merged only if their sets are equal.

At this point several initial problem coarsenings could be obtained by using different cost criteria as well as processing sequences. For example, in [10] per level they randomly choose an unmatched segment and fix an edge between it and the nearest unmatched segment according to the savings measure, whereas the



(a) Setting I (b) Setting II

Fig. 2. Exemplary (deterministic) initial problem coarsenings for customers having the same visit day combinations, iteratively building longer (route) segments.

actual costs of connecting the segments' free end nodes only considering segments in a defined surrounding for matching are used in [11]. We investigated several possibilities and after preliminary tests came up with these findings/settings:

- In general we observed the tendency that the greedier the selection for matching is, the better are the results on average; hence we always select the cheapest matching among all possible ones.
- Regarding the size (length) of the segments, i.e. how many customers are contained in the segment, we used two options:
  - *Setting I*: Enforce no limit at all, thus coarsen in a pure greedy fashion.
  - *Setting II*: Start with an allowed maximal length of two and gradually increase this limit by one whenever no more matching could be found. The maximum number of occurrences of one set of visit day combinations is used as an upper bound for this maximal length.
- We settled on using the actual costs of connecting the segments' free end nodes, thereby trying all four possible ways.
- Optionally we set a limit for the connection costs by multiplying the average inter-customer travel costs by a given factor δ_c, hence having an instance independent measure
- Optionally we respect given limits on vehicle capacity and route duration when building the segments.

A small schematic coarsening example for both settings I and II without limiting the connection costs is shown in Figure 2. The common practice of coarsening with a random factor yielded (so far) clearly worse results.

Having the coarsest problem, we generate the initial solution as described in Section 3 (there is no need to change this procedures). Then we interweave the iterative refinement with the VNS' execution. First, we select a fraction of the total VNS iterations in which this initial coarsening/refinement phase takes place.

Then these iterations are divided by the number of present refinements (plus one to also execute some iterations on the fully refined problem) to determine the iterations per (mini-)level.

Refinement is exactly the reversed process as coarsening (in a "last in–first out" fashion), thus splitting the usually most costly matchings first and keeping the good ones accordingly longer. Hence in a refinement step (refer to Figure 1) we have to locate the segment in a route at all days of the actual visit day combination and split it in two segments again, thereby preserving the direction.

At the end of this phase we may either continue with the fully refined problem or apply a recoarsening which will be the subject of the next section.

4.2 Solution-Based Recoarsening

Solution-based recoarsening is a very common practice to extend the concept of multilevel refinement beyond the initial coarsening/refinement phase. For this, the problem is recoarsened on the basis of a current incumbent solution in such a way as to preserve its structure, hence neglecting/destroying no obtained information. Despite this the coarsening principle stays the same. This obviously leads to considerably less degrees of freedom during the coarsening and in our case automatically to rather short segments.

This time we restrict our attention to adjacent segments in the current solution's routes, whereas again, they must have equal sets of visit day combinations. Further, such a segment pair must be adjacent on all visit days, only a whole reversed occurrence (s.t. the same end points are connected) is acceptable. As a cost criterion we directly use the present connection costs.

Regarding the recoarsening procedure we basically adhere again to the greedy approach as used in the initial problem based coarsening, but try to prevent obtaining the same recoarsening in case we use the same solution multiple times: Using a parameter x, we always select one of the x cheapest possible matchings per step (or level) at random. Initially, x is set to one, but when we encounter the same solution to be used for recoarsening again, x is incremented. Contrary, if a different solution than in the previous iteration is used we reset x to one and turn the procedure into a pure greedy selection again. This extension further reduces the risk of getting stuck in local optima.

For recoarsening we have so far fixed the connection cost limit to $\delta_c/2$ of the average inter-customer travel costs (in case the factor δ_c is used at all). Since we utilize a feasible solution for guiding the recoarsening a violation of the vehicle capacity or route duration is impossible and must not be checked. Finally, the subsequent refinement phase is handled in the same way as the initial refinement in the preceding section. Therefore we also choose a fraction of the VNS' iterations to be allotted to a recoarsening/refinement phase.

4.3 Handling Segments in the VNS

Due to incorporating the multilevel refinement strategy into the VNS there are a few more things to consider when dealing with segments of merged customers:

Table 1. Summarized average results on available benchmark test data.

test data		instances	VNS		MLVNS			
			%-gap BKS	%-gap HDH	setting	%-gap BKS	%-gap HDH	%-time VNS
PVRP	old	32	2.67 (3.63)	-0.30 (1.01)	$II_{0.33}$	2.27 (3.18)	-0.67 (1.43)	105.0
	new	10	2.47 (1.58)	-0.80 (0.86)	$II_{0.5}$	2.45 (1.52)	-0.82 (0.88)	87.4
PTSP	old	23	0.80 (1.03)	0.01 (0.32)	$II_{0.5}$	0.32 (0.41)	-0.47 (0.74)	90.0
	new	10	0.28 (0.16)	-0.01 (0.03)	$II_{0.5}$	0.22 (0.14)	-0.08 (0.07)	79.0

- As already mentioned the solution construction heuristics need not to be changed, the same applies to moving or exchanging route segments and all local search procedures.
- After each refinement step we immediately apply the intra-route local search in use on the routes that contain refined segments.
- Although the segments have different start and end points, i.e. are asymmetric, when changing visit combinations during shaking we always reinsert them in the direction at the time of creation instead of the one in the previous route. We observed no gain doing otherwise nor when trying both directions.

5 Experimental Results

Currently our experiments are aimed at investigating the different performance of the standard VNS and the multilevel VNS, hence we are not yet hunting for new best solutions. Nevertheless, we already did find a few ones but we will not elaborate on this. The algorithms have been implemented in C++, compiled with GCC 4.3 and executed on a single core of a 2.83 GHz Intel Core2 Quad Q9550 with 8 GB RAM.

For each chosen setting we performed 20 runs per instance with a limit of 10^6 iterations, i.e. solution evaluations. Preliminary tests on all considered instances suggested to ignore the given limits on vehicle capacity and tour duration during coarsening (which would only affect PVRP instances), probably because the VNS was built to cope with infeasibility. If not stated otherwise 20% of all iterations are devoted to the initial coarsening/refinement phase and recoarsening is applied four times also devoting 20% of the iterations each. The two remaining options we varied are the type of problem coarsening (setting I or II) and the cost limit factor δ_c; these will be denoted by $[\text{I,II}]_{\delta_c}$. We experienced that usually it is better to enforce a merging cost limit.

Due to space limitations results on available "old" and "new" benchmark test data[1] both for the PVRP and the PTSP are presented in concise form in Table 1 (see [3] for further details and origins of these instances). Here we state the percentage gap to the so far best known solutions (%-gap BKS) as well as to the average results of the VNS described by Hemmelmayr et al. [6] using the same

[1] Available on `http://neumann.hec.ca/chairedistributique/data/`

Table 2. New larger PVRP and PTSP (setting $m = 1$ and ignoring D and Q) instances using the generation method introduced in [3], and our best found solutions' values.

Id	n	m	t	D	Q	service frequencies					best found solutions	
						f_1	f_2	f_3	f_4	f_6	PVRP	PTSP
pr11	336	14	4	480	185	112	112		112		11611.19	6618.53
pr12	384	16	4	475	195	128	128		128		11428.81	7062.96
pr13	432	18	4	450	185	144	144		144		10686.30	6757.86
pr14	480	20	4	475	185	160	160		160		13733.77	7740.23
pr15	528	22	4	470	190	176	176		176		15430.77	8377.39
pr16	576	24	4	455	185	192	192		192		13465.62	7640.57
pr17	360	15	6	445	165	90	90	90		90	16166.45	9459.08
pr18	432	18	6	450	170	108	108	108		108	19891.65	11329.53
pr19	504	21	6	440	160	126	126	126		126	23825.45	11344.27
pr20	576	24	6	450	165	144	144	144		144	24285.09	11660.07

iteration limit of 10^6 (%-gap HDH), whereas all these values are given in [6]. The corresponding standard deviations are written in parentheses. For the MLVNS we further state the amount of CPU time spent given in percentage of the VNS' CPU time (%-time VNS). We chose following settings for the initial temperature utilized in the Metropolis criterion: 10 for PVRP_new, 7 for PVRP_old (except 70 for instances p27–p32 having large average travel costs), as well as also 7 for PTSP_old and PTSP_new. 1000 iterations are applied on each temperature level. As can be seen the MLVNS generally performs better than the standard VNS, achieving especially on the old data sets a significant improvement. Contrary, on the new data sets no clear improvement can be noticed. Yet for all but PVRP_old there is a CPU time reduction of 15% on average. For these data sets the coarsening setting II gave consistently better results, whereas especially for PVRP/PTSP_new differences are negligible.

Since the multilevel refinement strategy is in general especially appealing for large(r) instances, but the available test data lack them, we created some on our own by applying the generation method described in [3][2]. They can be regarded a continuation of the instances introduced in this latter work, except that we evenly distributed the visit frequencies among the customers for all instances; more details, including our best found solutions' values, are given in Table 2.

When doing preliminary tests we recognized that the VNS' acceptance decision using the Metropolis criterion in some way weakens the potential gain of the multilevel extension. Hence we also performed tests with only accepting improved solutions, whose results are given in Table 3 and 5 for the PVRP and the PTSP, respectively. The corresponding results using the default acceptance decision are given in Table 4 and 6, using an initial temperature of 7 for the PVRP and 10 for the PTSP instances again with a temperature decrease every 1000 iterations. For both variants and problems we tested coarsening setting I and II with a cost limit factor δ_c of 0.5 and 0.33, as well as devoting all iterations to the initial coarsening/refinement phase and doing no recoarsening at all, whereas we always state the results of the setting yielding the best solutions on average.

[2] These instances are available on `http://www.ads.tuwien.ac.at/sandro/routing/`

Table 3. Average results of standard and multilevel VNS on new larger PVRP instances only accepting improved solutions.

Id	VNS			MLVNS $I_{0.5}$				
	avg.	sdv.	t[s]	avg.	sdv.	t[s]	%-gap	%-time
pr11	12182.87	162.55	43.9	**12091.85**	161.61	34.2	-0.75	77.9
pr12	12115.62	94.96	48.6	**11957.43**	100.46	37.5	-1.31	77.2
pr13	11490.63	97.32	57.1	**11286.32**	157.02	44.2	-1.78	77.4
pr14	14553.84	90.33	69.6	**14351.15**	126.11	52.0	-1.39	74.7
pr15	16542.60	124.83	87.1	**16028.83**	59.82	61.2	-3.11	70.3
pr16	14764.22	119.59	107.9	**14109.85**	173.60	73.2	-4.43	67.8
pr17	17127.64	109.75	47.1	**16794.86**	105.07	38.3	-1.94	81.3
pr18	20978.79	104.37	60.8	**20582.41**	115.18	48.3	-1.89	79.4
pr19	25191.24	257.97	85.3	**24811.23**	324.06	62.8	-1.51	73.6
pr20	25803.80	140.15	116.1	**25310.04**	153.18	83.4	-1.91	71.8
avg.							-2.00	75.2

Table 4. Average results of standard and multilevel VNS variants on new larger PVRP instances using the acceptance decision with the Metropolis criterion.

Id	VNS			MLVNS $I_{0.5}$				
	avg.	sdv.	t[s]	avg.	sdv.	t[s]	%-gap	%-time
pr11	12018.84	205.47	42.1	**11815.43**	127.86	33.9	-1.69	80.5
pr12	11627.45	69.89	46.0	11639.32	87.91	37.1	0.10	80.7
pr13	10859.17	48.62	54.7	**10808.92**	86.46	43.7	-0.46	79.9
pr14	13895.99	47.52	67.3	13884.29	77.81	52.7	-0.08	78.3
pr15	15661.02	67.56	84.2	**15595.55**	58.51	62.5	-0.42	74.2
pr16	13808.28	73.71	103.7	**13714.56**	118.58	75.2	-0.68	72.5
pr17	16378.20	106.68	47.5	16340.21	86.95	40.1	-0.23	84.4
pr18	20107.76	107.32	62.3	**20043.43**	102.61	51.6	-0.32	82.8
pr19	24411.55	121.89	90.3	**24207.79**	291.51	67.1	-0.83	74.3
pr20	24686.82	87.57	117.7	**24540.46**	107.78	87.9	-0.59	74.7
avg.							-0.52	78.2

The tables show average travel costs (avg.), corresponding standard deviations (sdv.), CPU times in seconds (t[s]), as well as the percentage gaps to the VNS (%-gap) and the times in percent of the VNS (%-time) for the MLVNS. In Tables 3–6 average values of the MLVNS are printed bold whenever a statistically significant improvement compared to the VNS has been achieved, according to a Wilcoxon rank sum test with an error level of 5%. For these new larger instances the greedy coarsening setting I turned out to achieve consistently better results than setting II. Comparing Table 3 and 4, as well as Table 5 and 6 it can be clearly seen that the MLVNS yields a higher relative improvement when only accepting improved solutions, nevertheless also when using the default acceptance decision the improvement is notable, especially in case of the PTSP. Interestingly, for the PTSP using no recoarsening and extending the initial problem coarsening/refinement phase to all iterations turned out to be usually better (which only holds for the new larger instances, we checked it for the available test data, too). This has the additional positive effect that the required runtime is more than halved. Contrary, for the PVRP the decrease in runtime is again about 25%. In summary, for almost all instances and both acceptance decisions the MLVNS yields on average significantly better results than the VNS.

Table 5. Average results of standard and multilevel VNS variants on new larger PTSP instances only accepting improved solutions.

Id	VNS			MLVNS $I_{0.5}$ (no recoarsening)				
	avg.	sdv.	t[s]	avg.	sdv.	t[s]	%-gap	%-time
pr11	6856.76	28.09	188.6	**6775.87**	25.99	87.7	-1.18	46.5
pr12	7314.39	32.13	241.0	**7178.50**	29.23	109.1	-1.86	45.3
pr13	6969.09	30.63	320.7	**6861.12**	28.31	140.5	-1.55	43.8
pr14	8055.41	38.49	397.9	**7860.15**	33.72	175.9	-2.42	44.2
pr15	8678.21	33.31	503.1	**8525.48**	20.00	193.9	-1.76	38.5
pr16	7974.06	40.39	631.3	**7741.01**	25.78	246.8	-2.92	39.1
pr17	9854.09	36.76	205.9	**9639.06**	49.01	100.6	-2.18	48.9
pr18	11705.94	57.41	295.6	**11536.59**	41.65	141.6	-1.45	47.9
pr19	11795.08	61.93	424.5	**11591.77**	39.54	180.5	-1.72	42.5
pr20	12101.01	61.23	578.1	**11878.36**	31.65	244.1	-1.84	42.2
avg.							-1.89	43.9

Table 6. Average results of standard and multilevel VNS variants on new larger PTSP instances using the acceptance decision with the Metropolis criterion.

Id	VNS			MLVNS $I_{0.5}$ (no recoarsening)				
	avg.	sdv.	t[s]	avg.	sdv.	t[s]	%-gap	%-time
pr11	6736.19	32.29	182.1	**6670.51**	21.08	87.1	-0.98	47.8
pr12	7177.48	24.15	232.3	**7110.25**	25.11	106.5	-0.94	45.8
pr13	6881.71	30.23	295.1	**6789.76**	17.15	137.6	-1.34	46.6
pr14	7883.37	38.56	370.7	**7792.40**	25.88	171.2	-1.15	46.2
pr15	8518.06	35.33	471.4	**8424.37**	31.92	190.2	-1.10	40.3
pr16	7827.16	27.17	575.9	**7672.95**	22.59	238.4	-1.97	41.4
pr17	9596.32	30.20	215.9	**9534.96**	24.31	103.0	-0.64	47.7
pr18	11443.24	38.55	305.2	**11370.88**	29.46	143.1	-0.63	46.9
pr19	11551.89	39.00	421.8	**11435.78**	35.91	184.7	-1.01	43.8
pr20	11806.66	38.64	569.5	**11706.07**	28.03	257.8	-0.85	45.3
avg.							-1.06	45.2

6 Conclusions

We extended a recently proposed leading variable neighborhood search (VNS) with the multilevel refinement strategy to a multilevel VNS (MLVNS) for better solving periodic routing problems. A path based coarsening scheme is used that builds fixed (route) segments of customers accounting for the periodicity. The refinement process, i.e. starting at the coarsest level and iteratively refining until the original problem is reached again, is smoothly integrated into the VNS. Furthermore a suitable solution-based recoarsening is proposed that respects the structure of a given solution during coarsening. We presented results on available benchmark test data as well as on newly generated larger instances that show the advantage of the multilevel VNS compared to the standard VNS, often yielding significantly better results in usually less CPU time. In general the performance gain on the PTSP instances is higher. This new approach is especially appealing for large instances.

In the future we want to test with longer runs, also for the available test data, to further analyze the performance of the MLVNS and hopefully to find some (more) new best solutions. It might further be interesting to create even

larger instances having different characteristics. We also think about alternative ways of interweaving the refinement with the VNS, such as refining whenever the VNS gets stuck for a while; similar thoughts apply to the recoarsening. Also the interrelation of the multilevel extension and the Metropolis criterion is worth to investigate further. Finally, this promising multilevel refinement strategy for periodic routing problems could be applied to other underlying algorithms as well.

References

1. Walshaw, C.: Multilevel refinement for combinatorial optimisation problems. Annals of Operations Research **131** (2004) 325–372
2. Walshaw, C.: Multilevel refinement for combinatorial optimisation: Boosting metaheuristic performance. In Blum, C., et al., eds.: Hybrid Metaheuristics: An Emerging Approach to Optimization. Volume 114 of SCI. Springer (2008) 261–289
3. Cordeau, J.F., Gendreau, M., Laporte, G.: A tabu search heuristic for periodic and multi-depot vehicle routing problems. Networks **30**(2) (1997) 105–119
4. Bertazzi, L., Paletta, G., Speranza, M.G.: An improved heuristic for the period traveling salesman problem. Computers & Operations Research **31**(8) (2004) 1215–1222
5. Alegre, J., Laguna, M., Pacheco, J.: Optimizing the periodic pick-up of raw materials for a manufacturer of auto parts. European Journal of Operational Research **179**(3) (2007) 736–746
6. Hemmelmayr, V.C., Doerner, K.F., Hartl, R.F.: A variable neighborhood search heuristic for periodic routing problems. European Journal of Operational Research **195**(3) (2009) 791–802
7. Kirkpatrick, S., Gelatt Jr., C.D., Vecchi, M.P.: Optimization by simulated annealing. Science **220**(4598) (1983) 671–680
8. Francis, P.M., Smilowitz, K.R., Tzur, M.: The period vehicle routing problem and its extensions. In Golden, B., et al., eds.: The Vehicle Routing Problem: Latest Advances and New Challenges. Springer US (2008) 73–102
9. Pirkwieser, S., Raidl, G.R.: A variable neighborhood search for the periodic vehicle routing problem with time windows. In Prodhon, C., et al., eds.: Proceedings of the 9th EU/MEeting on Metaheuristics for Logistics and Vehicle Routing, Troyes, France (2008)
10. Rodney, D., Soper, A., Walshaw, C.: Multilevel refinement strategies for the capacity vehicle routing problem. International Journal of Information Technology & Intelligent Computing **2**(3) (2007)
11. Walshaw, C.: A multilevel approach to the travelling salesman problem. Operations Research **50**(5) (2002) 862–877
12. Bouhmala, N.: Combining local search with the multilevel paradigm for the traveling salesman problem. In Blum, C., et al., eds.: Proc. 1st Intl Workshop on Hybrid Metaheuristics. (2004) 51–58
13. Hansen, P., Mladenović, N.: Variable neighborhood search. In Glover, F., Kochenberger, G., eds.: Handbook of Metaheuristics. Kluwer Academic Publishers, Boston MA (2003) 145–184
14. Clarke, G., Wright, J.W.: Scheduling of vehicles from a central depot to a number of delivery points. Operations Research **12**(4) (1964) 568–581
15. Potvin, J.Y., Rousseau, J.M.: An exchange heuristic for routeing problems with time windows. Journal of the Operational Research Society **46** (1995) 1433–1446