Variable Neighborhood Search and GRASP for Three-Layer Hierarchical Ring Network Design*

Christian Schauer and Günther R. Raidl

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

Abstract. We introduce the Three-Layer Hierarchical Ring Network Design Problem, which arises especially in the design of large telecommunication networks. The aim is to connect nodes that are assigned to three different layers using rings of bounded length. We present tailored Variable Neighborhood Search (VNS) and GRASP approaches to solve large instances of this problem heuristically, and discuss computational results indicating the VNS' superiority.

1 Introduction

In this paper we present the *Three-Layer Hierarchical Ring Network Design* (3-LHRND) problem, which belongs to the research field of network design and has not been considered in the literature before. 3-LHRND finds applications in the planning of larger hierarchical communication networks where survivability in case of failures plays a major role. The nodes of the network are assigned to different layers—in our case three. The aim is to connect these nodes within each layer and, additionally, the layers among each other hierarchically using rings of bounded length for the matter of survivability. As a possible application one might imagine a telecommunication network, where different technologies are interconnected, e.g., high-speed fiber for layer 1, low-speed fiber for layer 2 and copper for layer 3.

In the context of survivability fault tolerance is an important issue especially in the case of wide area networks to guarantee high reliability. A common way to achieve these demands is the use of so-called *self healing rings* (often referred to as rings for short) that ensure connectivity of two nodes in case of the failure of a third node due to the possible routing in two different directions. Rings are, in fact, the simplest node-biconnected structures.

With the increasing size of networks a single ring connecting all nodes would, however, not be efficient anymore. Due to the large diameter of such a network, capacity requirements of single links as well as communication delays would soon be too high. Furthermore, simultaneous failures in more than one node would in general disconnect large parts of the network. This issues can be addressed by

* This work has been funded by the Vienna Science and Technology Fund (WWTF) through project ICT10-027.

C.A. Coello Coello et al. (Eds.): PPSN 2012, Part I, LNCS 7491, pp. 458–467, 2012.

dividing the network into subnetworks of limited size, i.e., the network consists of smaller rings that are hierarchically interconnected. Interconnection nodes are often referred to as *hub nodes*. When the interconnection is realized by ring structures the network is called a *hierarchical ring network*. To ensure survivability also in case of failures in hub nodes, the rings are interconnected via two different nodes—a so-called *dual homing* approach. In our case we consider a hierarchy spanning nodes on three different layers using dual homing, and consequently we refer to this problem as the *Three-Layer Hierarchical Ring Network Design problem* (3-LHRND). While much work already exists for related problems, 3-LHRND has so far not been addressed. As we will argue in Section 3, 3-LHRND is NP-hard and also difficult to solve in practice. In order to approximately solve larger instances, we propose a *Variable Neighborhood Search* (VNS) and a *Greedy Randomized Adaptive Search Procedure* (GRASP). Both exploit the special structure of the problem in various ways.

The rest of the paper is organized as follows. In Section 2 we summarize related work. Section 3 defines the 3-LHRND, and Section 4 presents our VNS and GRASP. Computational results are discussed in Section 5. Finally, Section 6 concludes this contribution, also pointing out suggestions for future research.

2 Related Work

While network design is a fast growing field of research, 3-LHRND with its strict definition and constraints has not been considered yet. Nevertheless, a lot of work has been contributed to similar problems, which contain some (but never all) of the aspects of 3-LHRND.

In the field of network design, the hierarchical/layered aspect has first been explicitly considered in 1986, when Current *et al.* [5] introduced the *Hierarchical Network Design Problem* (HNDP). In the HNDP the network consists of two primary nodes that are connected to a path and secondary nodes that connect to this path such that the whole network comprises to a tree. The authors present an Integer Linear Programming (ILP) formulation. Additionally, they designed a heuristic for the HNDP based on a K-shortest path algorithm, to find the best path connecting the primary nodes, and a minimum spanning tree heuristic, to connect the remaining nodes to the best path found.

Balakrishnan *et al.* introduced the *Multi-Level Network Design* (MLND) in [3], which is a generalization of the well-known Steiner network problem [6]. By definition nodes are assigned to L different levels in MLND. In the core of their work, the authors focus on the $L = 2$ case and first present an ILP formulation, which they later extend to a multi-commodity flow formulation.

In [12] Thomadsen and Stidsen give a detailed overview about the advantages of rings in survivable networks and present a Branch-and-Price approach for the Hierarchical Ring Network Problem with two levels using single homing, i.e., each ring connects to one node of the interconnection ring. The authors assume that node-to-layer assignments are not prespecified but are to be found during the design process.

In the past the attention in hierarchical network design has been primarily drawn to two level problems. However, in recent years the interest in more general multi (> 2) level problems has increased. Trampont, Destré and Faye [13] describe a three level problem, like our 3-LHRND but with tree structure, where the authors study a variant of the multi-source Weber problem. In this case terminals (layer 3) must be connected to a central equipment (layer 1) via concentrators (layer 2), which are not fixed at the beginning. The objective is to find the best location for the concentrators, i.e., determine the layer 2, such that the overall costs of the network are minimized. To solve this problem the authors present two stabilized column generation approaches.

The *Capacitated m-Ring-Star Problem* was introduced by Baldacci *et al.* in [4]. There the nodes are partitioned into three subsets, one depot node, customer nodes, and transit points (Steiner nodes). The network consists of m rings of bounded length and edges connected to the ring. All rings must contain the depot node and connect the customers directly or via a transit point (i.e., the additional edges). The authors compare two mathematical programming formulations, namely a two-indexed and a two-commodity flow formulation.

In [2] Aringhieri *et al.* apply different Metaheuristics to solve two ring network problems. One is comparable to the previously mentioned [12], the other does not use an interconnection ring but a set of paths to connect the level two rings. The authors designed a construction heuristic and two neighborhood structures embedded in a Tabu Search. Moreover, they consider Path Relinking, "eXploring Tabu Search", and Scatter Search approaches.

From another field of research originates the *Two-Echelon Location-Routing Problem* [11], which is a combination of the Facility Location Problem (FLP) and the Vehicle Routing Problem (VRP) but, nevertheless, shares similarities with 3-LHRND. Here the node set is partitioned in platforms (layer 1), satellites (layer 2) and customers (layer 3). In this case the locations for the platforms and satellites to be opened must be determined (i.e., the FLP aspect). Additionally, two different vehicle fleets transport goods either from the platforms to the satellites or from the satellites to the costumers (i.e., the VRP aspect). The main differences to 3-LHRND are that the platforms need not be connected and not all platforms and satellites need to be opened but only the beneficial (or sometimes mandatory) ones. The authors present a VNS with a total of 21 specific neighborhood structures.

3 Three-Layer Hierarchical Ring Network Design

This section introduces the 3-LHRND with all its details. We are given an undirected, complete graph $G = (V, E)$ with vertex set V, edge set E, and a cost function that assigns costs $c_{ij} \geq 0$ to each edge $(i, j) \in E$. Each vertex is assigned to one of three layers, i.e., the vertex set is partitioned into three disjoint subsets V_1, V_2 and V_3 with $V_1 \cup V_2 \cup V_3 = V$ and $V_i \cap V_j = \emptyset,\ \forall i, j \in \{1, 2, 3\},\ i \neq j$. A feasible solution to 3-LHRND is a subgraph $G_L = (V, E_L)$ connecting all the nodes V and satisfying the following constraints; see Figure 1 for an example.

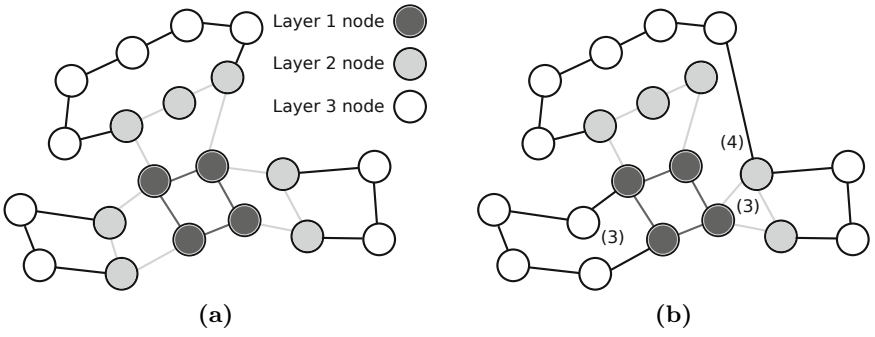

Fig. 1. Schematic representations of (a) a feasible solution and (b) an infeasible solution (the numbers in parentheses indicate the violated constraints)

1. The nodes in V_1 are connected by a single independent ring containing no other node.
2. The nodes from layers 2 and 3 are connected by two respective sets of paths, containing no nodes from other layers. Each node appears in exactly one path.
3. The end nodes of each path at layers 2 and 3 are further connected to two different nodes (hubs) in the directly preceding layer; i.e., dual homing is realized. We refer to the edges connecting paths to hubs as *uplinks*. Consequently, there are no edges directly connecting a node from layer 1 to a node from layer 3.
4. The two hub nodes a path is connected to must themselves be connected by a simple path at their layer; i.e., the connection to a ring may not be established via more than two layers.
5. The lengths of layer $k \in \{2, 3\}$ paths in terms of the number of nodes is bounded below and above by specified limits $b_k^{\mathrm{l}} \geq 2$ and $b_k^{\mathrm{u}} \geq 2$, respectively.

The objective is to find a feasible solution with minimum total costs $c(E_L) = \sum_{(i,j)\in E_L} c_{ij}$.

Considering these definitions, we observe that finding the layer 1 ring resembles the classical Traveling Salesman Problem (TSP), which can be solved independently. In contrast, optimal structures of layers 2 and 3 strongly depend on each other. Only a suitable, concerted choice of layer 2 and layer 3 paths allow for efficient uplinks and overall connectivity. Thus, the layers 2 and 3 cannot be treated separately. The rings connecting layer 2 with layer 1 consist each of at most $\lfloor |V_1|/2 \rfloor + b_2^{\mathrm{u}} + 1$ nodes and edges, while the rings connecting layer 3 and layer 2 have up to $b_2^{\mathrm{u}} + b_3^{\mathrm{u}}$ nodes/edges.

3-LHRND obviously is NP-hard even when looking at each layer independently: As mentioned, for layer 1 the subproblem directly corresponds to the TSP. Concerning layers 2 and 3, the classical capacitated vehicle routing problem (CVRP) can be reduced to each by assuming all nodes of the preceding layer are connected via a single ring and together represent the CVRP's depot.

4 A VNS and GRASP for 3-LHRND

As already mentioned, the problem to connect all layer 1 nodes to a single ring resembles the TSP and can be solved independently. We apply the Concorde TSP solver [1] and focus in the following only on the more interesting layers 2 and 3.

Due to the 3-LHRND's complexity and relations to other network design problems and CVRP variants, we decided to solve it approximately using VNS and GRASP, as these techniques are known to work well in these domains, see, e.g., [11]. At first we designed a simple construction heuristic, which we used to create initial solutions for the VNS and then randomized for the GRASP. For general introductions to VNS and GRASP, we refer to [8,10].

4.1 Construction Heuristic

Our construction heuristic is inspired by the simple nearest-neighbor heuristic for the TSP. To create the layer 2 paths we start with an unvisited layer 2 node i, search for the nearest hub node $h \in V_1$ according to the edge costs c_{ih}, and add the uplink e_{ih} to our solution. Then we determine in layer 2 the closest unvisited node j to our current node i, add the edge e_{ij} to our solution, mark i as visited, and make j our current node. We repeat this procedure until the given upper bound for layer 2 paths b_2^{u} is reached. Then we search for the nearest hub node different from h back to layer 1 and add this uplink to our solution. Following this approach we add layer 2 paths to our solution until all layer 2 nodes are marked as visited. In case that the length of the last path would be less than the lower bound for layer 2 paths b_2^{l} we make the second to last path shorter to satisfy all constraints.

To create the layer 3 paths we apply the same procedure, but now we further have to ensure that the second uplink connects to the same layer 2 path as the first uplink.

4.2 Variable Neighborhood Descent

Variable Neighborhood Descent (VND) [8] extends simple local search by systematically considering a set of different neighborhood structures in a deterministic way until a solution is reached that is locally optimal w.r.t. all of them. Thus, the improvement potential strongly relies on the neighborhood structures and the order of their application.

Our VND, which is used within VNS as well as GRASP, searches eight neighborhood structures that are induced by the following operators in the given order.

Two-Edge-Exchange (2EE): Based on the well known operator for the TSP this operator investigates all feasible candidate solutions that differ in at most two edges. In this case 2EE is applied to each layer 2 and layer 3 path separately starting with the first uplink and ending with the second. The number of neighbors and, consequently, the runtime for completely searching this neighborhood is bounded by $O((b_k^{\mathrm{u}})^2 \cdot \lceil V_k / b_k^{\mathrm{l}} \rceil)$ for $k \in \{2, 3\}$.

Three-Edge-Exchange (3EE): After the use of 2EE a reduced form of three-edge-exchange is applied to all layer 2 and layer 3 paths. This reduced form, as presented in [9], only deals with the cases that are not covered by 2EE, i.e., when indeed three edges are replaced by three new edges. This limitation and the fact that the number of neighbors lies in $O((b_k^{\mathrm{u}})^3 \cdot \lceil V_k / b_k^{\mathrm{l}} \rceil)$, $k \in \{2, 3\}$, makes searching 3EE relatively fast in relation to the standard three-edge-exchange for the TSP. Like 2EE this operator is applied to all edges of a path including the uplinks, and only feasible solutions are considered.

Split-Rings (SR): This operator splits one path into two subpaths and relinks the end of the first and the beginning of the second subpath back to the preceding layer. Care must be taken as some layer 2 nodes along a split path might serve as hub nodes for layer 3 paths. If the split is performed between two hub nodes of an attached layer 3 path, this layer 3 path would not be connected to the same layer 2 path anymore. Each layer 3 path affected in this way is therefore relinked by determinig its cheapest feasible pair of hub node connections. Additionally, the splitting point must be chosen so that both subpaths exceed b_k^{l}, $k \in \{2, 3\}$.

Two-Node-Exchange (TNE): As the name indicates this operator considers all solutions in which two nodes from different paths on the same layer are exchanged. If an exchange on layer 2 affects a hub node, then the attached layer 3 paths must be relinked as described above. This is achieved be relinking either within the old or within the new path of the exchanged hub node.

One-Node-Move (ONM): This operator moves a single node from one path to another, i.e., the node is removed from its old path and inserted in another path from the same layer when the corresponding path length bounds b_k^{l} and b_k^{u} are satisfied. Again, if a layer 2 hub node is moved then relinking is necessary for the attached layer 3 paths. All nodes as well as all feasible insertion points are considered.

Append-Rings (AR): This operator represents the complement to SR. It connects two paths on the same layer. The length of the new path must not exceed the upper bound b_k^{u}. For layer 2 the existing uplinks can be used but relinking of layer 3 paths might be necessary, when two layer 3 paths are merged, which are connected to different layer 2 paths.

Change-Uplinks (CU): The CU operator optimizes the uplinks for all layer 2 and layer 3 paths. For each path, the overall cheapest pair of hubs is determined, taking care that the two hub nodes must be different and the hub nodes of each layer 3 must appear in the same layer 2 path.

Merge-Rings (MR): This operator provides an additional opportunity to merge short paths into a new one. While AR only considers their appendance, MR tries to insert one path between any two nodes of another path from the same layer, providing the path length limit is not exceeded.

We implemented the VND with its eight neighborhood structures with three different step functions, namely classical next and best improvement and, additionally, we used a function that applies a move immediately, when an improvement

was found (like next improvement) but then (in contrast to next improvement) stick with this operator until no further improvement can be found within this neighborhood.

4.3 Variable Neighborhood Search (VNS)

A general VNS, as used for this paper, is a stochastic algorithm that combines VND as local search procedure with an outer search mechanism performing random moves in typically larger neighborhoods in order to escape the local optima of the VND. The outer random moves are also called *shaking* [8].

In preliminary tests we searched for the best combination of shaking neighborhood structures. It appeared that only slight changes in the solution lead to a better and smooth improvement during the VNS.

Larger moves perturbated the solution in a way so that the VND could not steadily improve this solution. We finally ended up with the following four shaking operators based on TNE and ONM: (1) exchange two random nodes between layer 3 paths, (2) exchange two random nodes between layer 2 paths, (3) move one random layer 3 node to another path, and (4) move one random layer 2 node to another path. In all cases, only moves respecting all the constraints are performed.

Since, TNE and ONM are used later within the VND, the previous operators (2EE, 3EE, SR) typically already found improvements for the affected paths so that TNE or ONM will not simply undo the changes performed within the shaking.

4.4 Greedy Randomized Adaptive Search Procedure (GRASP)

GRASP [10] marks another metaheuristic approach to avoid getting trapped in local optima. A construction heuristic is randomized in order to create diverse initial solutions, and each of which is successively optimized by local search. The randomization of a construction heuristic is usually performed by turning from a pure greedy selection of the next component by which a partial solution is extended towards using a *Restricted Candidate List* (RCL) of promising candidate components and choosing from it at random.

For our purpose we randomized the construction heuristic from Section 4.1. In each iteration, when a new edge is added to a path from the current node i to the next node j, this randomized heuristic creates an RCL that contains the r-nearest neighbors of i, with r being a strategy parameter. Node j is then chosen randomly from the RCL, based on a uniform distribution. For r, i.e., the length of the RCL, we set $b_k^{\mathrm{u}}/2, k \in \{2, 3\}$. However, we still choose for each path the best uplinks deterministically. Due to the fact that the choice of the uplinks is a local decision with respect to each path we know that in the end the CU neighborhood structure will find the best uplinks for all paths. Therefore, a randomized choice of the uplinks would not meaningfully increase our search space but cause needless effort. After the randomized heuristic created an initial solution, it is improved by the previously described VND. This procedure is repeated until a given time-limit is reached.

Table 1. The underlying TSPLIB instances of the test set with the number of generated test cases, upper bounds, time limits and the average objective values for initial and final solutions and corresponding standard deviations obtained from VNS and GRASP (30 runs per test case).

				VNS				GRASP			
TSPLIB	#	b_k^u	t[s]	init	dev	final	dev	init	dev	final	dev
ulysses22	1	5-7	150	166.82	0.00	128.28	1.70	166.82	0.00	129.56	0.00
att48	4	5-7	150	76560.30	4406.66	61721.55	1053.39	82084.04	4146.29	63527.21	2120.45
eil51	4	5-7	150	1043.27	36.85	756.75	21.60	1075.17	50.44	779.55	36.27
berlin52	4	5-7	150	19078.36	1233.71	13709.26	428.23	19800.12	939.20	14309.64	769.90
eil76	12	5-12	150	1227.28	59.19	933.07	40.52	1462.40	130.43	971.26	37.23
gr96	18	5-12	300	1220.23	76.19	964.20	43.04	1514.36	172.12	1045.29	43.34
kroA100	18	5-12	300	59707.86	5263.22	41742.56	1584.33	74798.61	6745.93	46171.74	2974.85
kroB100	18	5-12	300	59259.01	6148.42	41796.04	1644.53	73918.06	8229.28	46204.49	2656.25
bier127	12	8-15	300	263995.65	8274.02	209636.01	3386.62	385901.48	24744.77	228707.75	5359.88
ch150	18	8-15	300	15589.56	575.06	12286.63	440.28	24454.02	1784.15	13988.46	436.43
kroA200	18	8-15	300	73801.91	3111.26	55314.71	1371.25	113770.39	8222.59	65670.67	1904.39
kroB200	18	8-15	300	75040.20	3855.24	58406.59	1665.32	118601.24	9128.76	66267.41	2340.33
gr229	12	12-20	600	3858.00	166.33	3008.36	59.19	7559.52	619.88	3588.33	141.86
pr299	12	12-20	600	116878.18	4526.67	96944.53	1507.31	246754.55	19372.46	119237.01	4592.66
lin318	18	12-20	600	100782.81	3423.72	83170.33	1789.88	204233.35	15512.07	104900.96	4322.02
gr431	18	12-20	900	4544.14	163.13	3651.96	105.69	9230.29	829.13	4525.99	258.50
pr439	18	12-20	900	267944.92	11269.23	222507.49	10831.91	560935.73	46642.47	287415.51	17304.18

5 Experimental Results

For testing purposes we created a benchmark instance set based on TSPLIB[1]. We used 17 TSPLIB instances and performed a k-means clustering to determine the layer 1 and layer 2 nodes. By varying the number of nodes in the layers we derived 42 instances. All graphs are complete, with the exception that edges between layer 1 and layer 3 were removed as they cannot appear in feasible solutions.

Moreover, we defined combinations of useful upper bounds for the path lengths depending on the number of nodes in the graph, which resulted in 223 overall test cases. As a lower bound we assumed a minimum length of two for all paths.

For each of the test cases we performed 30 runs executed on a single core of an Intel Xeon (Nehalem) Quadcore CPU with 2.53 GHz and 3GB of RAM. As stopping criterion we used the same CPU time limits for both VNS and GRASP as indicated in Table 1. We always applied next improvement as step function, which provided the best results according to preliminary tests. These tests also indicated that creating initial solutions with more restricted path lengths of up to $b_k^u - 2,\ k \in \{2, 3\}$, only instead of b_k^u yields slightly worse solutions but with a significantly higher improvement potential. Consequently, we followed this strategy in our VNS and GRASP. In the following section we describe our results based on this test set.

5.1 Test Results

Results are summarized in Table 1. The columns have the following meaning: *TSPLIB* indicates the underlying instances our derived test cases are based on

[1] `http://comopt.ifi.uni-heidelberg.de/software/TSPLIB95/`

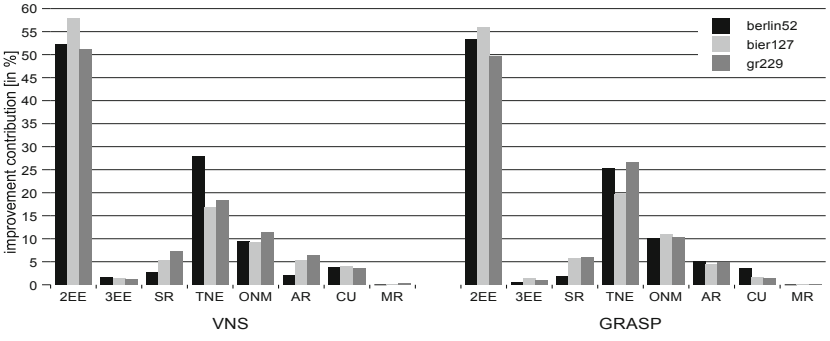

Fig. 2. Relative number of total improvements achieved by the individual neighborhood structures within VND

(the number at the end of the name denotes the number of nodes in the graph); # denotes the numbers of different test cases for each TSPLIB instance; b_k^{u} lists the range of used upper bounds for layer 2 and 3; t describes the CPU-time limits in seconds; for the *VNS* and *GRASP* sections *init* lists the average objective values of the initial solutions together with their standard deviations labeled by *dev*, while *final* denotes the average objective values of the final solutions found in each run, again together with corresponding standard deviations *dev*.

Since, in GRASP initial solutions are generated by the randomized construction heuristic, the significantly worse initial objective values and higher standard deviations are natural. As can be seen easily, the successive VND could improve these initial solutions dramatically, also leading to substantially decreased standard deviation.

Concerning VNS, it can be easily seen that it clearly outperforms GRASP. In fact, average final objective values from the VNS are always smaller than those from GRASP. Student t-tests indicated the statistical significances of these differences with error probabilities smaller than 1%. Since, there is no opportunity for GRASP to escape local optima during local search, the GRASP approach cannot easily break up the path structure of the initial solution as is done by the shaking in VNS. On the other hand, it appears that the randomized construction heuristic cannot easily provide promising path structures from the beginning.

In VND, all eight neighborhood structures contribute to the overall success, although the impacts vary. Figure 2 shows for three exemplary cases, which reflect the typical behavior well, how many times the application of each neighborhood structure led to an improved solution in relation to all improvements achieved. The test settings were for berlin52 $|V_1| = 4$, $|V_2| = 10$, $b_2^{\mathrm{u}} = 5$, $b_3^{\mathrm{u}} = 7$; for bier127 $|V_1| = 10$, $|V_2| = 40$, $b_2^{\mathrm{u}} = 12$, $b_3^{\mathrm{u}} = 15$; for gr229 $|V_1| = 12$, $|V_2| = 80$, $b_2^{\mathrm{u}} = 17$, $b_3^{\mathrm{u}} = 20$. One can further see here that the success of neighborhood structures only loosely depends on the upper bounds for the path lengths. The highest improvement was achieved by 2EE, which is also applied first, while 3EE had only a minor impact. TNE proves to be an important operator, followed by ONM. MR was rarely successful.

6 Conclusions and Future Work

We introduced the Three-Layer Hierarchical Ring Network Design Problem and presented a VNS and GRASP for solving it heuristically. Both strategies use a common construction strategy as well as a VND for local improvement. The VND explores eight tailored neighborhood structures, which have been shown to augment each other well. In our experiments, the VNS clearly outperformed GRASP. In the future we will investigate alternatives for the construction heuristic, e.g., based on the prominent savings heuristic from vehicle routing problems, as well as study further potentially more powerful neighborhood structures for VND/VNS. In particular, we expect further improvements by considering larger neighborhood search concepts such as cyclic exchanges over more than two paths, as well as hybrid techniques involving mixed integer linear programming for solving reasonably sized subproblems. An important practical necessity is to test the approaches also on more realistic sparse graphs.

References

1. Concorde TSP Solver, `www.tsp.gatech.edu/concorde.html`, (accessed: February 02, 2012)
2. Aringhieri, R., Dell' Amico, M.: Comparing Metaheuristic Algorithms for Sonet Network Design Problems. Journal of Heuristics 11, 35–57 (2005)
3. Balakrishnan, A., Magnanti, T.L., Mirchandani, P.: The Multi-level Network Design Problem. Tech. rep., Massachusetts Institute of Technology, Cambridge (1991)
4. Baldacci, R., Dell'Amico, M., Gonzalez, J.S.: The Capacitated m-Ring-Star Problem. Operations Research 55(6), 1147–1162 (2007)
5. Current, J.R., ReVelle, C.S., Cohon, J.L.: The hierarchical network design problem. European Journal of Operational Research 27(1), 57–66 (1986)
6. Dreyfus, S.E., Wagner, R.A.: The Steiner Problem in Graphs. Networks 1, 195–207 (1972)
7. Gendreau, M., Potvin, J.Y. (eds.): Handbook of Metaheuristics, International Series in Operations Research & Management Science, vol. 146. Springer (2010)
8. Hansen, P., Mladenović, N., Brimberg, J., Pérez, J.A.M.: Variable Neighborhood Search. In: Gendreau and Potvin [7], pp. 61–86
9. Potvin, J.Y., Rousseau, J.M.: An Exchange Heuristic for the for the Routing Problems with Time Windows. Journal of the Operational Research Society 46, 1433–1446 (1995)
10. Resende, M.G., Ribeiro, C.C.: Greedy Randomized Adaptive Search Procedures: Advances, Hybridizations, and Applications. In: Gendreau and Potvin [7], pp. 283–320
11. Schwengerer, M., Pirkwieser, S., Raidl, G.R.: A Variable Neighborhood Search Approach for the Two-Echelon Location-Routing Problem. In: Hao, J.-K., Middendorf, M. (eds.) EvoCOP 2012. LNCS, vol. 7245, pp. 13–24. Springer, Heidelberg (2012)
12. Thomadsen, T., Stidsen, T.: Hierarchical Ring Network Design Using Branch-and-Price. Telecommunication Systems 29(1), 61–76 (2005)
13. Trampont, M., Destré, C., Faye, A.: Solving a hierarchical network design problem with two stabilized column generation approaches. In: International Network Optimization Conference 2009, Pisa, Italy (2009)