Lagrangian Decomposition, Metaheuristics, and Hybrid Approaches for the Design of the Last Mile in Fiber Optic Networks

Markus Leitner[1,2] and Günther R. Raidl[2]

[1] Carinthia University of Applied Sciences
School of Telematics / Network Engineering
Klagenfurt, Austria
markus.leitner@fh-kaernten.at
[2] Vienna University of Technology
Institute for Computergraphics and Algorithms
Vienna, Austria
raidl@ads.tuwien.ac.at

Abstract. We consider a generalization of the (Price Collecting) Steiner Tree Problem on a graph with special redundancy requirements for customer nodes. The problem occurs in the design of the last mile of real-world communication networks. We formulate it as an abstract integer linear program and apply Lagrangian Decomposition to obtain relatively tight lower bounds as well as feasible solutions. Furthermore, a Variable Neighborhood Search and a GRASP approach are described, utilizing a new construction heuristic and special neighborhoods. In particular, hybrids of these methods are also studied and turn out to often perform superior. By comparison to previously published exact methods we show that our approaches are applicable to larger problem instances, while providing high quality solutions together with good lower bounds.

Keywords: Network Design, Variable Neighborhood Search, Greedy Randomized Adaptive Search Procedure, Lagrangian Relaxation, Redundancy, Steiner Tree Problem, Survivable Network Design.

1 Introduction

We consider a real-world communication network design problem arising in the expansion of existing fiber optic networks. "Fiber-to-home" has recently become economically feasible for individual households. Since the coverage of larger districts with such networks requires enormous financial resources, good algorithms for finding cost-efficient network layouts are crucial.

We consider the problem of augmenting an existing network infrastructure by additional links (and switches) in order to connect potential customer nodes. Two types of customers exist: For type-1 customers, a standard, single link connection suffices, while type-2 customers require more reliable connections,

M.J. Blesa et al. (Eds.): HM 2008, LNCS 5296, pp. 158–174, 2008.

ensuring connectivity even when a single link or routing node fails. We also consider a variant of the problem in which the redundancy condition for type-2 customers is relaxed in the sense that a connection is allowed via a final non-redundant branch that does not exceed a certain length $b_{\max}$.

In previous work, summarized in Section 3, we approached this problem with integer linear programming techniques, including an extended multi-commodity flow formulation and a branch-and-cut algorithm. These techniques allow to find proven optimal solutions for relatively small instances.

Here we propose an approach that is also feasible for larger instances and nevertheless provides performance guarantees. It is based on a Lagrangian decomposition of the network flow model in order to obtain relatively tight lower bounds. The Lagrangian dual problem is hereby solved via the Volume Algorithm [1], which is known to often perform better than a standard subgradient search. Upper bounds and thus primal (feasible) solutions are identified at the same time, and they are improved by local search utilizing several neighborhoods. Furthermore, we propose two metaheuristic approaches based on Variable Neighborhood Search [2] and GRASP [3] to obtain primal solutions in relatively short time.

From a theoretical point-of-view, we are able to show that our Lagrangian decomposition represents a stronger model than the linear programming relaxation of the original multi-commodity network flow model. This observation is also clearly supported by our experimental results: The Volume Algorithm usually finds significantly better lower bounds in shorter times, and the obtained heuristic solutions are typically better or equal than those that could eventually be obtained by the previous approaches.

In a more general sense, this work is a good example on how Lagrangian relaxation can be applied in combination with local search based metaheuristics in order to solve a difficult practical problem heuristically and provide a quality guarantee, i.e. a lower bound, at the same time. The next section will formally introduce the problem. In Section 3 we give a short summary on related previous work. An abstract variant of the multi-commodity flow formulation from [4] is presented in Section 4 together with the Lagrangian decomposition approach for solving it. Section 5 presents the neighborhood structures used to improve solutions in the metaheuristics of Section 6 as well as in the hybrid Lagrangian approaches given in Section 7. Experimental results are discussed in Section 8, and Section 9 concludes this work.

2 Problem Definition

We are given a connected undirected graph $G = (V, E)$ representing the spatial topology of the surrounding area of potential customers. Edges in E correspond to possible cable routes and have associated lengths $l_e \geq 0$ and construction costs $c_e \geq 0$ for installing a corresponding fiber optic link. The node set V is the disjoint union of customer nodes C and spatial nodes S (switches, possible Steiner nodes). Set C is partitioned into subsets C_1 and C_2, whereby customers C_1 require a single connection (type-1) and customers C_2 need to be

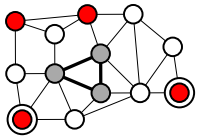


Fig. 1. Problem Instance

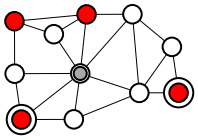


Fig. 2. Shrunken Instance

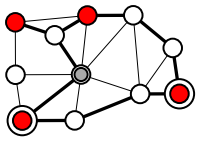


Fig. 3. Solution with $b_{\max} = 0$

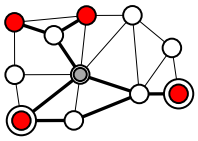
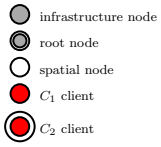


Fig. 4. Solution with $b_{\max} > 0$



redundantly connected (type-2). Each customer node $k \in C$ further has associated a prize $p_k \geq 0$, i.e. expected return of investment. The already existing network infrastructure is represented by the subgraph $I = (V_I, E_I)$ of G, see Figure 1.

In a first preprocessing step, we shrink the whole existing network infrastructure, i.e. the root and all connected infrastructure nodes, into a single node $0 \in V$. From all edges connecting a node $i \in V$ to the existing infrastructure, only the cheapest edge is kept and finally replaced by an edge $(0, i)$ with the same length and costs, see Figure 2.

Let subgraph $G' = (V', E')$ with $V' \subseteq V$ and $E' \subseteq E$ represent a solution network we seek. The following conditions specify how customer nodes are to be connected:

- *Simple connection:* A customer node k from C_1 is feasibly connected iff there exists a path from node 0 to k.
- *Redundant connection:* A customer node k from C_2 is feasibly connected iff there exist two node (and edge) disjoint paths from node 0 to k, see Figure 3.
- $b_{\max}$*-redundant connection:* Occasionally, the biconnectivity condition for the nodes in set C_2 is relaxed in the sense that such a node $k \in C_2$ may be connected to any biconnected (Steiner or customer) node $j \in V$ (the *branch-node* of k) via a single path of maximum total length $b_{\max}(k) > 0$. This (optional) single path is called *branch-line* and $b_{\max}(k)$ the *maximum branch-line length* for customer k, see Figure 4.

Regarding the objective, we distinguish between two alternative goals:

- In the *Operative Planning Task* (OPT) we focus on finding a minimum-cost subgraph G' feasibly connecting all customers C, with the total costs being

$$c_{\mathrm{OPT}}(G') = \sum_{e \in E'} c_e. \tag{1}$$

This case can be considered a generalization of the classical *Steiner tree problem on a graph* (STP) where a special form of redundancy is required for the nodes in C_2.

– In the *Strategic Simulation Task* (SST) customers' prizes are also considered, and the objective is to only connect a subset $C' \subseteq C$ of customers so that the costs for building the network minus the earned prizes are minimized. In order to always have positive total costs, which eases some parts of our algorithms and notations, we perform a simple transformation by adding the constant $\sum_{k \in C} p_k$ to the objective function, yielding

$$c_{\mathrm{SST}}(G') = \sum_{e \in E'} c_e - \sum_{k \in C'} p_k + \sum_{k \in C} p_k = \sum_{e \in E'} c_e + \sum_{k \in C \setminus C'} p_k. \qquad (2)$$

This problem variant is generalization of the *price-collecting Steiner tree problem* (PCSTP).

As already the classical Steiner tree problem on a graph is NP-hard [5], this obviously also holds for both of our problem variants. In the following presentation of our solution approaches, we primarily consider the more complex SST case if not explicitly stated and assume $p_k = \infty,\ \forall k \in C$ for the OPT case.

3 Previous Work

The Steiner Tree Problem (STP) has been considered by a lot of authors, see e.g. [6] for a survey. The Price Collecting Steiner Tree Problem (PCSTP) was introduced by Segev [7] who considered the Node Weighted STP, which is a special version of the PCSTP. The term "price collecting" has been introduced by Balas [8] for the Price Collecting Traveling Salesman Problem. A survey on methods for Survivable Network Design which can be seen as a more general version of our problem can be found in [9].

In our first attempt described in [4], we modeled this problem as an integer linear program (ILP) by means of an extended multi-commodity network flow (MCF) formulation. With the general purpose ILP-solver CPLEX [10], instances with up to 190 total nodes, 377 edges but only 6 customer nodes could be solved to proven optimality, and instances up to 2804 nodes, 3082 edges and 12 customer nodes could be solved with a final gap of about 7%. Unfortunately, this approach turned out to be unsuitable for larger instances and/or in particular instances with larger number of customer nodes, as already solving the linear programming (LP) relaxation of the ILP requires too much time due to the huge number of variables involved.

In [11], we approached this problem with a different formulation based on directed connectivity constraints. While this formulation involves only a reasonable number of variables, the number of inequalities is exponentially large. By using a branch-and-cut algorithm, however, this model could be solved relatively well, and we were able to find proven optimal solutions for instances with up to 190 nodes, 377 edges, and 13 customer nodes. For larger, practical instances

this approach unfortunately still is not applicable at all or finds quite poor solutions with huge LP-gaps only. Finally, another even stronger model based on directed connectivity constraints which does not consider $b_{\max}$ redundancy has been presented in [12].

4 Abstract ILP Model and Lagrangian Decomposition

To formulate this problem as an abstract ILP, we utilize decision variables $x_e \in \{0,1\}$, $\forall e \in E$, indicating whether or not edge e is part of the solution, i.e. $x_e = 1 \leftrightarrow e \in E'$. For customer nodes $k \in C$ variables $y_k \in \{0,1\}$ denote whether or not feasible connections according to the customers' types and $b_{\max}(k)$ exist. Our model is based on the MCF formulation from [4], but all the different types of flow variables for each customer $k \in C$ on directed arcs are replaced by simple variables $f_e^k \in \{0,1\}$, $\forall e \in E$, indicating whether or not edge e is part of the single path (type-1) or pair of disjoint paths plus the eventual branch-line (type-2) for connecting customer k; f^k denotes the vector of all these variables for a customer k.

Let F_k, $\forall k \in C$, be the set of all incidence vectors on E corresponding to feasible connections for customer k. We can now formulate the SST-variant of our problem in the following abstract way:

$$\text{minimize} \quad \sum_{e \in E} c_e x_e + \sum_{k \in C} p_k (1 - y_k) \tag{3}$$

$$\text{s.t.} \quad f_e^k \leq x_e \qquad \forall k \in C,\ \forall e \in E \tag{4}$$

$$f^k \in F_k \text{ if } y_k = 1 \qquad \forall k \in C \tag{5}$$

$$f_e^k \in \{0,1\} \qquad \forall k \in C,\ \forall e \in E \tag{6}$$

$$x_e \in \{0,1\} \qquad \forall e \in E \tag{7}$$

The objective function (3) uses variables x_e and y_k but otherwise corresponds to (2). Inequalities (4) are called *coupling constraints* and enforce an edge to appear in the solution when it is used for connecting at least one customer. Conditions (5) ensure feasible connections for all selected customers ($y_k = 1$). The OPT-variant of the model is obtained by simply ignoring the second term in the objective function and the conditions on y_k in (5).

Note that in this form, the model is not yet a concrete ILP, as conditions (5) are not expressed by means of linear inequalities. Ideally, we would substitute them by a set of linear inequalities describing the convex hull $\text{conv}(F_k)$ of all incidence vectors of feasible connections for each customer k in dependence of variables y_k. Unfortunately, finding a (compact) set of such inequalities is not trivial. While this can be achieved for simple (type-1) connections via a network flow formulation, this task is quite difficult for the biconnected case involving branch-lines (type-2).

Our MCF model from [4] represents a concrete instantiation of this abstract model. As can be easily shown, however, it does not contain a complete description of $\text{conv}(F_k)$ but just a formulation that is valid for integer solutions. We

conclude that the MCF-formulation from [4] therefore is not as strong as an "ideal" instantiation of the abstract model.

4.1 Lagrangian Decomposition

For a general introduction to Lagrangian relaxation and decomposition see e.g. [13]. We relax the coupling constraints (4) of our abstract model in a classical Lagrangian fashion, i.e., by substituting them with corresponding penalty terms in the objective function. This yields model LR(λ):

$$\text{minimize} \quad \sum_{e\in E} c_e x_e + \sum_{k\in C} p_k(1-y_k) + \sum_{k\in C}\sum_{e\in E} \lambda_{k,e}\cdot(f_e^k - x_e) = \tag{8}$$

$$= \sum_{k\in C} p_k + \sum_{e\in E}\left(c_e - \sum_{k\in C}\lambda_{k,e}\right)x_e + \sum_{k\in C}\left(\sum_{e\in E}\lambda_{k,e}f_e^k - p_k y_k\right) \tag{9}$$

$$\text{s.t.} \quad f^k \in F_k \text{ if } y_k = 1 \qquad \forall k \in C \tag{10}$$

$$f_e^k \in \{0,1\} \qquad \forall k \in C,\ \forall e \in E \tag{11}$$

$$x_e \in \{0,1\} \qquad \forall e \in E \tag{12}$$

Parameters $\lambda_{k,e} \geq 0$, $\forall k \in C$, $\forall e \in E$, are the Lagrangian multipliers, and for any feasible instantiation of them the optimal solution of LR(λ) yields a lower bound on the optimal solution value of our original abstract model [13]

For a specific selection of λ, this relaxation can be efficiently solved as it decomposes into $|C|$ independent problems of determining individual cheapest connections for each $k \in C$ on a graph whose edge costs are $\lambda_{k,e}$ (see Section 4.2). A node k is finally connected ($y_k = 1$) and the variables f_e^k corresponding to the identified connection are set to one iff the connection pays off, i.e. $\sum_{e\in E}\lambda_{k,e}f_e^k \leq p_k$. Otherwise, the connection is discarded by setting $y_k = 0$ and $f_e^k = 0$, $\forall e \in E$. Optimal values for variables x_e, $e \in E$, are independently determined by simple inspection, i.e. $x_e = 1$ iff $c_e < \sum_{k\in C}\lambda_{k,e}$, $\forall e \in E$.

The Lagrangian dual problem is the challenge of finding an optimal vector of Lagrange multipliers λ^* so that the lower bound obtained by $LR(\lambda^*)$ becomes as large as possible. As this maximization problem is convex and piecewise linear, subgradient algorithms are well suited for this purpose [13]. While different variants of such methods exist, the *Volume Algorithm* [1] has proven to be more effective than several alternatives on various occasions [14,15], and we therefore apply it here. Also, our preliminary comparisons indicate the superiority of this algorithm over the standard subgradient strategy as described in [13]. Due to space limitations, we unfortunately cannot describe the Volume Algorithm here.

4.2 Determining an Individual Optimal Connection

In each iteration of the Volume Algorithm, we need to determine for each customer $k \in C$ the cheapest feasible connection on the graph in which each edge $e \in E$ has costs $\lambda_{k,e} \geq 0$. While a simple shortest path calculation from 0 to k

returns this connection for type-1 customers $k \in C_1$, we need to determine the cheapest pair of node-disjoint paths from 0 to k for type-2 customers $k \in C_2$. Suurballe and Tarjan [16] presented an algorithm to efficiently compute a shortest arc-disjoint pair of paths between two nodes s and t on a directed graph $G_{\mathrm{D}} = (V, A)$ in time $O(|A| + |V| \log |V|)$, see also [17]. Initially a shortest path tree from s as well as the shortest path P_1 from s to t are determined and the costs of each arc (i, j) are replaced with $c_{i,j} - d_j + d_i$, with d_i and d_j representing the costs of the shortest paths from s to i and j, respectively. After reversing all arcs on P_1, a shortest path P_2 from s to t is determined on this new (residual) graph using these adapted arc costs. Finally, the cheapest arc-disjoint pair of paths is given by $P_1 \triangle P_2$.

We apply this algorithm on the *split graph* of the original graph to compute node-disjoint paths. The split graph is obtained by replacing each node $v \in V$ by a pair of nodes v', v'' connected by an arc (v', v'') with zero costs. Furthermore, for each (undirected) edge $(u, v) \in E$ arcs (u, v') and (v'', u) with the same costs (and lengths) are created. Since each node v' has only one outgoing arc and each node v'' has only one ingoing arc, any pair of edge-disjoint paths is also node-disjoint.

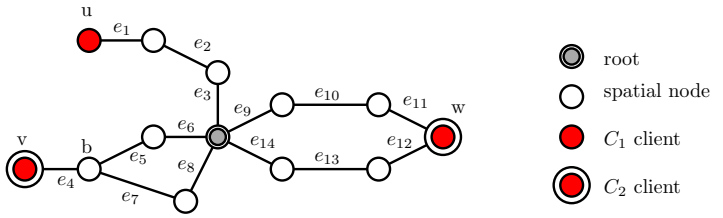
A simple extension of the algorithm above for the case of $b_{\max}$-redundancy (see Fig. 3) is to determine the overall cheapest combination of a shortest pair of paths to a node in the $b_{\max}$-neighborhood of a customer k and a simple path from this node to k. Although we believe that a more efficient algorithm, at least with respect to average time complexity, can be found, we currently use this extension, which increases worst time complexity by a factor proportional to the size of the $b_{\max}$ neighborhood. Unnecessary calculations can be avoided by only considering possible branch-nodes j for which $d_j < \frac{1}{2} c_{\mathrm{curr}}$ with c_{curr} being the costs of the so far cheapest connection.

4.3 Theoretical Comparison of LR(λ^*) and the MCF Formulation

For each concrete instantiation of λ, all subproblems obtained by the Lagrangian decomposition are always solved to optimality and integrality. Therefore, $f^k \in \mathrm{conv}(F_k)$ holds for all $k \in C$, and the abstract constraints (5) of our model (3) to (7) can be regarded as "ideally instantiated". Assuming we would be able to identify an optimal Lagrange vector λ^*, the lower bound obtained by LR(λ^*)) is at least as good as the lower bound determined by an LP-relaxation of the model. As already argued before, the MCF-formulation from [4] is weaker than an "ideal" instantiation of the abstract model. We therefore conclude that LR(λ^*)) is stronger than the LP-relaxation of the MCF-formulation. Our experimental results in Section 8 also clearly support this fact.

5 Neighborhoods for Improving Primal Solutions

Our algorithms make use of three types of neighborhoods. While the first two aim at reducing the cost of a given solution, the last type consisting of two



$E'_{\mathrm{u}} = \{e_1, e_2, e_3\}$ | $E'_{\mathrm{v}} = \{e_4, e_5, e_6, e_7, e_8\}$ | $E'_{\mathrm{w}} = \{e_9, e_{10}, e_{11}, e_{12}, e_{13}, e_{14}\}$

$B(E'_{\mathrm{u}}) = u$ | $B(E'_{\mathrm{v}}) = b$ | $B(E'_{\mathrm{w}}) = w$

$P(E'_{\mathrm{u}}) = \{e_1, e_2, e_3\}$ | $P(E'_{\mathrm{v}}) = \{e_5, e_6\}$ | $P(E'_{\mathrm{w}}) = \{e_9, e_{10}, e_{11}\}$

$Q(E'_{\mathrm{u}}) = \emptyset$ | $Q(E'_{\mathrm{v}}) = \{e_7, e_8\}$ | $Q(E'_{\mathrm{w}}) = \{e_{12}, e_{13}, e_{14}\}$

$L(E'_{\mathrm{u}}) = \emptyset$ | $L(E'_{\mathrm{v}}) = \{e_4\}$ | $L(E'_{\mathrm{w}}) = \emptyset$

Fig. 5. An exemplary candidate solution and the representation of its connections

concrete neighborhoods tries to improve a solution by removing customers from a candidate solution. Therefore, the latter is only applicable to the SST variant.

For our neighborhood structures, a candidate solution $S' = (V', E', C', X')$ is represented, by its node set V', total edge set E', feasible connected customers $C' = \{k \in C \mid y_k = 1\}$ and individual connections $X' = \{E'_k \mid k \in C'\}$ with $E'_k = \{e \in E' \mid f_e^k = 1\}$. In other words, E'_k is the set of edges used to eventually connect customer k. Note that there may exist multiple connections to a single customer node in a solution S' in which case we store only one of them.

Furthermore, for each connection E'_k we maintain its internal structure consisting of its branch-node $B(E'_k) \in V'$, edge sets $P(E'_k), Q(E'_k) \subseteq E'$ of its two paths between 0 and $B(E'_k)$ and finally the edge set of its branch-line $L(E'_k) \subseteq E'$. Note that we assume $B(E'_k) = k$ if $b_{\max}(k) = 0$ or $k \in C_1$ as well as internally define $P(E'_k)$ to be the "first" path of a connection, i.e. $P(E'_k)$ is used for type-1 customers while $Q(E'_k) = L(E'_k) = \emptyset$ for type-1 customers, see Figure 5. Finally, to allow for efficient updates of a solution with respect to connections, we maintain for each edge $e \in E'$ a list of the customers that are connected via this edge: $M_e = \{k \in C' \mid e \in E'_k\}$.

5.1 Connection Exchange Neighborhood

The *Connection Exchange Neighborhood* (CEN) consists of all solutions differing from the current solution S' by exactly one connection E'_k, see Algorithm 1. To determine the best neighboring solution for a fixed customer $k \in C'$, CEN calculates the saving due to removing the corresponding connection E'_k (which is the sum of all edge costs exclusively used to connect k). The connection to k leading to minimum additional costs is then determined by calculating the cheapest feasible connection to k in a graph with edge costs $c'_e = 0$, $\forall e \in E'' = E' \setminus \{e \in E'_k \mid M_e = \{k\}\}$ and $c'_e = c_e$, $\forall e \in E \setminus E''$. For type-1 nodes and type-2 nodes with $b_{\max}(k) = 0$, the computational complexity of finding this new connection for one specific client node k is bounded by $O(|E| + |V| \log |V|)$. For type-2 customers with $b_{\max}(k) > 0$, we iteratively consider each possible

Algorithm 1. Connection Exchange (Solution S')

$c'_e = 0 \ \forall e \in E'$
$c'_e = c_e \ \forall e \in e \setminus E'$
$d_{\text{opt}} = 0$
forall $k \in C'$ **do**
  $E'' = \{e \in E'_k \mid M_e = \{k\}\}$
  $c'_e = c_e, \ \forall e \in E''$
  $d = \sum_{e \in E''} c_e$
  E''_k = shortest connection to k using edge costs c'
  $d = \sum_{e \in E''} c_e - \sum_{e \in E''_k} c'_e$
  if $d > d_{\text{opt}}$ **then**
    $d_{\text{opt}} = d - \sum_{e \in E''_k} c'_e$
    store solution S' with E''_k replacing E'_k as best solution
  $c'_e = 0, \ \forall e \in E''$
return best solution

branch-nodes, yielding an upper bound of $O(b(|E|+|V|\log|V|))$, with b denoting the maximum number of possible branch-nodes. Therefore, the whole CEN which consists of exponentially many feasible connections can be efficiently searched for the best neighbor in $O(|C|\, b\, (|E| + |V| \log |V|))$.

5.2 Key-Path Exchange Neighborhood

A *key-node* of a solution S' is a node $v \in V' \setminus C'$ with node degree $\deg_{S'}(v) \geq 3$, while a *key-path* is a path $K_{\text{P}} = (V_{\text{P}}, E_{\text{P}})$ whose end nodes are either key-nodes or customer nodes $k \in C$, while all other nodes are Steiner nodes $v \in V' \setminus (C \cup 0)$ of degree two, i.e. $\deg_{S'}(v) = 2$. This concept of key-paths is well known for the STP and several metaheuristic methods utilizing a key-path exchange neighborhood have been proposed, see e.g. [18]. The *Key-Path Exchange Neighborhood* (KPEN) given in Algorithm 2 extends this concept by exchanging key-paths while respecting node- as well as $b_{\max}$ redundancy. KPEN of a candidate solution S' consists of all feasible solutions that differ from S' by at most one key-path. To ensure feasibility, after exchanging a key-path K_{P}, three relevant cases need to be considered. If K_{P} is used to connect type-1 customer only, it may simply be replaced by any other path, while if it is used in a branch-line $L(E'_k)$ of a type-2 customer $k \in C_2$, the maximum length of the new path may be at most $b_{\max}(k) - \sum_{e \in (L(E'_k) \setminus E_{\text{P}})} l_e$. Finally, if K_{P} is used in the first path $P(E'_k)$ of a C_2 customer k, all edges incident to "internal" nodes of its second path $Q(E'_k)$ may not be used by the new key-path to guarantee node redundancy (and vice versa for the alternate path $Q(E'_k)$). All other edges e of S' are treated as pseudo-infrastructure, i.e. $c'_e = 0$.

5.3 Connection Remove Neighborhood

Instead of exchanging a customer's connection as in CEN, the *Connection Remove Neighborhood* (CRN) removes the connection to a single customer node

Algorithm 2. Key Path Exchange (Solution S')

determine key-paths W
$d_{\text{opt}} = 0$
forall key-paths $(V_{\text{P}}, E_{\text{P}}) \in W$ **do**
  // actual key-path connects its end nodes m, n
  $c'_e = 0 \ \forall e \in E' \setminus E_{\text{P}}$
  $c'_e = c_e \ \forall e \in E_{\text{P}} \cup (E \setminus E')$
  choose $e \in E_{\text{P}}$ randomly
  $l_{\max} = \infty$
  forall $k \in M'_e$ **do**
    if $e \in P(E'_k)$ **then**
      $c'_e = \infty, \ \forall e \in E$ incident to a inner node of $Q(E'_k)$
    else if $e \in Q(E'_k)$ **then**
      $c'_e = \infty, \ \forall e \in E$ incident to a inner node of $P(E'_k)$
    else if $e \in L(E'_k)$ **then**
      $l_{\max} = b_{\max}(k) - \sum_{e \in (L(E'_k) \setminus E_{\text{P}})} l_e$
  $(V'_{\text{P}}, E'_{\text{P}})$ = shortest path from m to n using c'_e with max. length $l_{\max}$
  $d = \sum_{e \in E_{\text{P}}} c_e - \sum_{e \in E_{\text{P}'}} c'_e$
  if $d > d_{\text{opt}}$ **then**
    $d_{\text{opt}} = d$
    store solution S' with $(V_{\text{P}}, E_{\text{P}})$ replacing $(V_{\text{P}}, E_{\text{P}})$ as best solution
return best solution

$k \in C'$. CRN of a current solution S' therefore consists of all solutions S'', where exactly one customer connected in C' is not connected anymore, i.e. $C'' \subset C' \wedge |C'' \setminus C'| = 1$. As a customer's connection may consist of $O(|V|)$ edges only, CRN consisting of $|C'|$ neighboring solutions can be searched in $O(|C'||V|)$ time.

5.4 Restricted Two Connection Remove Neighborhood

CRN can be easily generalized to simultaneously remove multiple customer nodes. However, removing the connections to $l > 1$ customers at once will result in $|C|^l$ neighboring solutions and the computational effort of searching such a neighborhood would be $O(|C|^l|V|)$. We therefore concentrate on simultaneously removing pairs of customers $i, j \in C', i \neq j$ which share at least one edge exclusively used by them, i.e. $\exists e \in E' \mid M_e = \{i, j\}$. The *Restricted two Connection Remove Neighborhood* (R2CRN) can be searched in $O(|V| \min(|E'|, |C'|^2))$.

6 Metaheuristics

In this section we present metaheuristic approaches utilizing the neighborhoods explained in Section 5 to compute feasible solutions. After describing a construction heuristic in Section 6.1, we present a Variable Neighborhood Search (VNS) with embedded Variable Neighborhood Descent (VND) in Section 6.2 and – as an alternative – a GRASP/VND hybrid in Section 6.3.

6.1 Minimum Spanning Tree Augmentation Heuristic

We use a three-phase approach called *Minimum Spanning Tree Augmentation Heuristic* (MSTAH) to construct a feasible solution for a given selection of customers $C' \subseteq C$ to be connected. Initially, a Steiner tree G_{T} is computed using the *Minimum Spanning Tree* (MST) heuristic from [19]. This procedure determines a MST T_D on the *distance network*, which is the complete graph $D = (C', C' \times C')$ with node set C' and edge costs $d(u, v)$ corresponding to the costs of the cheapest paths between any $u, v \in C'$ in G. A feasible solution S'' to the Steiner Tree Problem is derived by further computing a MST on $G(T_D)$ which is the subgraph of G induced by all edges part of any cheapest path corresponding to an edge in T_D. In its second phase, MSTAH augments $S'' = (V'', E'')$ by feasible connections to C_2 customers. Such connections are determined by individually calculating the cheapest feasible connection (compare Section 4.2) for all customers $k \in C_2$. All so far selected edges $e \in E''$ are considered as pseudo-infrastructure, i.e. having zero costs. Finally, an edge minimal solution is extracted (i.e. no further edges can be deleted without violating feasibility) by greedily removing unnecessary key-paths in decreasing cost order. A similar heuristic which does not consider $b_{\max}$ redundancy has been presented in [12]. Similar to MSTAH the heuristic from [12] uses the MST heuristic [19] to compute a Steiner tree. As opposed to MSTAH redundancy for C_2 customers is ensured by adding a redundant route to each type-2 customer avoiding any inner node of the existing primary path using so far selected edge as pseudo-infrastructure.

6.2 Variable Neighborhood Search

We use the general VNS scheme with VND as embedded local improvement [2]. In VND, we alternate between CEN, KPEN, CRN, R2CRN in this order, with the latter two considered only in the SST variant.

Our shaking algorithm used to escape local optima modifies a solution S' by excluding a subset of its Steiner nodes as well as changing the set of connected customers C' in the SST variant: A set of $l = 1, \ldots, l_{\max} = |C|$ Steiner nodes $V_{\mathrm{F}} \subset V' \setminus C$ of the current solution S' is randomly chosen for removal. Furthermore, we select a set of $m = \lfloor \frac{l}{3} \rfloor$ customer nodes $C_{\mathrm{C}} \subset V'_i \in C$ at random. The set of customers C'' connected in the new solution S'' is $C'' = C' \triangle V_{\mathrm{C}}$, i.e. we add those customers of V_{C} that are currently unconnected while removing the so far connected ones. Finally, we apply MSTAH using the following adapted edge costs c' with a sufficiently large value for M ($M \gg \max_{e \in E} c_e$).

$$c'_e = \begin{cases} M & \text{if } e \text{ is incident to a nodes } v \in V_{\mathrm{F}}, \\ 0 & \text{if } e \in E' \text{ and } e \text{ not incident to a node } v \in V_{\mathrm{F}}, \\ c_e & \text{else.} \end{cases}$$

Edge costs c' ensure the creation of a new solution S'' that is in general similar to S' while those Steiner nodes selected for exclusion will not be used unless there is no other option to obtain a feasible solution S''.

6.3 Greedy Randomized Adaptive Search Procedure

As an alternative to the general VNS, we also consider a GRASP in which local search is again performed by the above mentioned VND. A similar approach utilizing node- and path-based neighborhoods has been already proposed for the classical STP by Martins et al. [18]. They used a modified version of the MST heuristic [19] in the construction phase. Similarly, we modify our construction heuristic MSTAH by randomizing Kruskal's algorithm for computing the MST on the distance network D. Let $d_{\max} = \max\{d(u,v) \mid \forall (u,v) \in C' \times C'\}$ and $d_{\min} = \min\{d(u,v) \mid \forall (u,v) \in C' \times C'\}$ be the maximum and minimum distances, respectively. Instead of always adding the cheapest feasible edge that connects two yet unconnected components, the randomized spanning tree construction selects the edge to be included next randomly from a restricted candidate list consisting of all feasible edges $(u,v) \in C' \times C'$ with $d(u,v) \leq d_{\min} + \alpha(d_{\max} - d_{\min})$ with $0 < \alpha \leq 1$.

7 Combining Lagrangian Decomposition and Variable Neighborhood Descent

As described in Section 4 we solve the Lagrangian dual problem of determining optimal λ^* by the Volume Algorithm. In each iteration we need to determine optimal x_e variables as well as f_e^k variables for the current set of Lagrangian multipliers λ. The latter are computed by calculating individual cheapest connections for each customer $k \in C$ and eventually choosing to connect k in case it pays off. Obviously, the graph $S' = (V', E')$ induced by the set of edges $E' = \{e \in E \mid \exists k \text{ s.t. } f_e^k = 1\}$ is a primal feasible solution. This offers multiple ways of hybridizing the Lagrangian decomposition approach with metaheuristics in order to obtain better primal solutions and reduce the gap between lower and upper bounds.

Here, we pursue two alternatives: Either we immediately try to improve promising solutions gained by the iterations of the Volume Algorithm, or we store the N best solutions obtained by the Volume Algorithm and try to improve them after termination of the Volume Algorithm. In both cases, we use VND with CEN, KPEN, CRN, and R2CRN in this order to generate a local optimum for a given candidate solution (CRN and R2CRN are again only considered in the SST variant). According to the classification of hybrid metaheuristics given in [20] the former approach is a sequential hybridization with respect to the order of execution, while the latter falls into the category of interleaved hybridization.

As the time for performing VND on a candidate solution is not negligible, it is critical to apply it wisely on a well-chosen subset of candidate solutions only. In the interleaved approach, we found the following self-adaptive strategy with the exogenous parameters δ, γ, and $\beta_{\max}$ to work well. Let S' and S'_{best} be the current and so far best solutions obtained by the Volume algorithm, respectively. VND is applied to S' iff $c(S') \leq (1+\beta)\, c(S'_{\text{best}})$. Preliminary tests indicated that a good value for β is not easy to find as it depends on the problem instance, and so we automatically adapt it each δ iterations as follows. Let r be the ratio of how

Table 1. Instance set characteristics

Set	#	$\|V\|$	$\|E\|$	$\|C\|$	$\overline{\|C\|}$	$\|C_1\|$	$\overline{\|C_1\|}$	$\|C_2\|$	$\overline{\|C_2\|}$	$b_{\max}$	$\overline{\|V(b_{\max})\|}$
ClgSE-I1	25	190	377	5-8	5.9	3-5	3.8	2-3	2.1	30	3.79
ClgSE-I2	15	190	377	11-17	13.8	7-12	8.9	4-7	4.9	30	8.97
ClgSE-I3	15	190	377	8-12	9.6	5-8	6.0	3-6	3.6	30	6.04
ClgME-I1	25	1757	3877	6-10	7.2	4-7	5.0	2-3	2.3	100	4.96
ClgME-I2	15	1523	3290	11-14	12.2	8-11	8.7	3-4	3.5	100	8.71
ClgN1B-I1	20	2804	3082	11-14	11.8	8-11	8.5	3-4	3.3	100	8.49
ClgN1B-I2	19	2804	3082	7-11	9.0	3-6	4.1	4-6	5.0	100	3.99
ClgN1E-I1	20	3867	8477	8-14	11	3-6	4.1	5-9	6.9	150	4.12
ClgN1E-I2	20	3867	8477	10-12	10.6	6-8	6.4	4-5	4.2	150	6.39

often VND has been applied during the last δ iterations of the Volume algorithm. If $r < \gamma$ we set $\beta = \min(2\beta, \beta_{\max})$ while $\beta = \max(\beta/2, \beta_{\max})$ if $r > \gamma$. We chose $N = 50$, $\beta_{\min} = 0.01$, $\beta_{\min} = 0.4$, $\gamma = 0.05$ and $\delta = 100$ and initially set $\beta = 0.1$.

Furthermore, we memorize hash-values of candidate solution which have already been used as starting solutions to avoid unnecessary runs of VND. These hash-values are also used to ensure that the N solutions stored in the sequential approach are pairwise different.

We initialize Lagrangian multipliers by $\lambda_{k,e} = c_e/|C|$ ensuring a positive lower bound in the first iteration of the Volume Algorithm. Referring to the description of the Volume algorithm in [15], we further configured it as follows: The target value T is set to $T = 1.1 z_{\text{UB}}$ with z_{UB} being the actual upper bound unless the actual lower bound $z_{\text{LB}} > 0.9\,T$ in which case T is multiplied by 1.1. We initially set $f = 0.1$ and $\alpha = 0.01$. After 20 consecutive non-improving iterations, f is multiplied by 0.67 in case it is greater than 10^{-4} and by 1.1 in an improving iteration if $f < 1$. If z_{LB} did not improve by more than 1% within the last 100 iterations and if $\alpha > 10^{-5}$, we multiply α by 0.85. The Volume Algorithm is terminated if $\lceil z_{\text{LB}} \rceil = z_{\text{UB}}$, after 250 consecutive non improving iterations, or if the maximum time limit is reached.

8 Computational Results

We used real-world instances from a German city [21] to test our approaches, see Table 1. All experiments have been performed on a single core of an Intel Xeon 5150 with 2.66GHz and 8GB RAM; ILOG CPLEX 10.0 has been used to solve the ILP for the MCF formulation from [4]. For GRASP we chose $\alpha = 0.25$ and generated 100 initial solutions, and the VNS was terminated after 100 iterations of the outermost, largest shaking move. An absolute time limit of 7200 seconds has been used for all experiments.

Table 2 compares lower bounds generated by our Lagrangian Decomposition (LD) approach to the LP-relaxation values of the MCF formulation from [4]. RED refers to the problem variant with standard redundancy constraints for C_2 customers while BMAX describes those experiments using $b_{\max}$ redundancy. As the sequential Lagrangian Decomposition approach (SEQ) as well as the interleaved approach (INT) yield similar bounds we only report the relative improvement of

Table 2. Improvement of lower bounds comp. to the LP-relaxation of MCF [4] in %.

Set	OPT+RED	SST+RED	OPT+BMAX	SST+BMAX
ClgS-I1	0.00	0.05	6.83	6.98
ClgS-I2	0.00	0.14	5.98	5.96
ClgS-I3	0.00	0.51	5.53	4.95
ClgM-I1	0.00	0.00	2.04	2.04
ClgM-I2	0.00	0.15	4.54	3.71
ClgN1B-I1	0.00	3.07	-	-
ClgN1B-I2	0.00	2.12	-	-
ClgN1E-I1	0.00	0.14	-	-
ClgN1E-I2	0.00	0.02	-	-

Table 3. Relative gaps and corresponding standard deviations in %

		OPT			SST		
	Set	LD	SEQ	INT	LD	SEQ	INT
RED	ClgS-I1	1.77 (2.45)	1.65 (2.39)	**1.63** (2.38)	1.76 (2.45)	1.65 (2.39)	**1.63** (2.38)
	ClgS-I2	12.80 (6.16)	9.12 (4.05)	**8.84** (4.08)	13.45 (7.07)	9.98 (6.18)	**9.13** (4.65)
	ClgS-I3	7.49 (6.07)	5.73 (4.81)	**5.54** (4.55)	8.89 (6.19)	7.28 (5.03)	**7.09** (4.84)
	ClgM-I1	4.29 (2.61)	2.80 (2.17)	**2.70** (2.10)	4.22 (2.62)	2.80 (2.17)	**2.61** (2.10)
	ClgM-I2	9.88 (7.10)	6.58 (4.75)	**5.89** (4.43)	11.60 (6.70)	8.50 (5.77)	**7.67** (5.65)
	ClgN1B-I1	4.12 (3.50)	2.82 (2.82)	**2.50** (2.19)	4.17 (3.45)	2.88 (2.80)	**2.58** (2.20)
	ClgN1B-I2	1.96 (1.81)	1.32 (1.43)	**1.27** (1.44)	1.84 (1.73)	1.34 (1.46)	**1.29** (1.46)
	ClgN1E-I1	3.13 (3.33)	1.51 (1.57)	**1.23** (1.24)	3.08 (3.23)	1.65 (1.81)	**1.23** (1.24)
	ClgN1E-I2	5.62 (4.67)	3.55 (2.51)	**3.21** (2.09)	5.36 (4.04)	3.53 (2.52)	**3.20** (2.08)
BMAX	ClgS-I1	2.26 (3.19)	2.13 (3.00)	**1.74** (2.40)	2.26 (3.19)	2.13 (3.00)	**1.74** (2.40)
	ClgS-I2	19.49 (7.36)	14.41 (4.46)	**12.87** (4.34)	19.53 (7.11)	14.60 (4.91)	**13.15** (4.89)
	ClgS-I3	9.05 (7.44)	6.47 (4.47)	**6.23** (4.30)	10.26 (7.67)	7.31 (4.32)	**7.14** (4.21)
	ClgM-I1	5.27 (3.22)	3.41 (2.14)	**3.09** (1.96)	5.27 (3.23)	3.34 (2.10)	**3.09** (1.96)
	ClgM-I2	15.19 (9.49)	9.29 (5.66)	**8.27** (4.53)	15.89 (9.37)	9.85 (5.86)	**9.02** (5.16)

LD in Table 2. LD generally generates equal bounds for the OPT case when $b_{\max}$-redundancy is not considered, while the achieved lower bounds are better when dealing with the SST variant or when considering $b_{\max}$-redundancy. The LP relaxation of the MCF formulation from [4] could not be solved for one instance of set ClgN1E-I1 (OPT variant) within 2 hours. Therefore, Table 2 reports the relative improvements for the remaining 19 instances of this set.

Table 3 compares relative gaps between upper and lower bounds generated by LD, SEQ, and INT and corresponding standard deviations (in parentheses). In general, one can observe the expected behavior that the gap increases with increasing number of customers.

SEQ and INT consistently yield for all problem variants and instances the smallest gaps, which are usually significantly better than those of LD. Table 4 depicts relative improvements of the generated upper bounds compared to LD. Without considering $b_{\max}$-redundancy, INT generally finds solutions equally good or even better than those that could be obtained by the MCF formulation [4] within 2 hours. As the MCF formulation from [4] could not identify a feasible solution for several instances of set ClgN1E-I1 (4 instances in the OPT variant and 7 instances in the SST variant) we do not report the average improvement of MCF for this set. Average values for GRASP and VNS have been computed using 10 runs per instance.

Table 4. Relative improvement of upper bounds compared to LD in %

	Set	MCF	SEQ	INT	GRASP	VNS
OPT+RED	ClgS-I1	**0.14** (0.19)	0.12 (0.19)	**0.14** (0.19)	-0.13 (1.02)	0.12 (0.21)
	ClgS-I2	**3.40** (2.85)	3.15 (2.73)	**3.40** (2.85)	3.03 (3.23)	3.38 (2.85)
	ClgS-I3	**1.74** (2.17)	1.57 (2.16)	**1.74** (2.17)	1.48 (2.33)	1.63 (2.30)
	ClgM-I1	1.53 (1.01)	1.41 (1.12)	**1.61** (1.11)	1.54 (1.13)	1.22 (1.67)
	ClgM-I2	3.18 (2.88)	2.87 (2.74)	**3.51** (2.85)	3.23 (2.78)	2.73 (4.15)
	ClgN1B-I1	1.50 (1.82)	1.22 (1.57)	**1.51** (1.83)	1.47 (1.87)	1.41 (1.93)
	ClgN1B-I2	0.66 (1.05)	0.62 (1.02)	**0.67** (1.05)	0.53 (1.09)	**0.67** (1.05)
	ClgN1E-I1	- (-)	1.52 (1.77)	**1.78** (2.00)	1.65 (2.11)	1.14 (2.04)
	ClgN1E-I2	1.07 (3.32)	2.07 (2.41)	**2.64** (2.74)	2.56 (2.75)	2.36 (2.78)
SST+RED	ClgS-I1	**0.13** (0.19)	0.11 (0.19)	**0.13** (0.19)	-0.14 (1.02)	0.00 (0.43)
	ClgS-I2	**3.67** (2.92)	2.98 (2.90)	**3.67** (2.92)	3.30 (3.33)	2.87 (3.76)
	ClgS-I3	**1.57** (2.38)	1.40 (2.34)	**1.57** (2.38)	1.21 (2.57)	1.30 (2.43)
	ClgM-I1	1.49 (0.99)	1.35 (1.05)	**1.55** (1.04)	1.48 (1.06)	0.95 (1.89)
	ClgM-I2	3.44 (2.67)	2.71 (2.60)	**3.45** (2.62)	3.01 (2.60)	2.14 (4.14)
	ClgN1B-I1	**1.50** (1.80)	1.21 (1.51)	1.49 (1.82)	-0.86 (7.66)	0.81 (2.25)
	ClgN1B-I2	**0.54** (0.89)	0.49 (0.86)	**0.54** (0.89)	-2.68 (6.94)	-0.12 (1.83)
	ClgN1E-I1	- (-)	1.35 (1.56)	**1.75** (1.98)	1.60 (2.05)	0.36 (2.04)
	ClgN1E-I2	1.21 (2.53)	1.88 (1.95)	**2.43** (2.27)	2.10 (2.56)	1.92 (2.31)
OPT+BMAX	ClgS-I1	**0.50** (1.33)	0.12 (0.24)	0.48 (1.32)	0.23 (1.82)	0.48 (1.32)
	ClgS-I2	**5.71** (3.90)	4.08 (3.68)	5.36 (4.04)	4.97 (4.36)	5.22 (3.97)
	ClgS-I3	**2.60** (3.32)	2.18 (3.31)	2.40 (3.38)	1.68 (3.95)	2.15 (3.66)
	ClgM-I1	1.73 (2.13)	1.74 (1.69)	**2.05** (1.87)	1.84 (1.83)	1.94 (1.90)
	ClgM-I2	4.02 (6.17)	4.82 (4.99)	**5.67** (5.17)	5.51 (5.11)	5.61 (5.11)
SST+BMAX	ClgS-I1	**0.50** (1.33)	0.12 (0.24)	0.48 (1.32)	0.23 (1.82)	0.47 (1.32)
	ClgS-I2	**5.52** (4.10)	4.00 (3.23)	5.17 (4.15)	4.78 (4.41)	3.97 (4.28)
	ClgS-I3	**2.80** (3.70)	2.46 (3.78)	2.61 (3.77)	1.89 (4.26)	2.16 (4.03)
	ClgM-I1	1.62 (1.78)	1.81 (1.67)	**2.05** (1.83)	1.84 (1.83)	1.85 (1.93)
	ClgM-I2	3.59 (6.94)	4.95 (4.84)	**5.63** (5.08)	5.42 (5.09)	4.63 (4.87)

Table 5. Median run times

	Set	MCF^{LP}	MCF	LD	SEQ	INT	GRASP	VNS
OPT+RED	ClgS-*	0.2	0.9	2.0	1.7	3.7	1.7	1.2
	ClgM-*	58.2	3490.4	77.4	99.7	234.0	59.6	34.5
	ClgN1B-*	91.5	739.0	72.3	93.2	216.5	118.6	87.2
	ClgN1E-*	1103.9	7220.9	371.9	659.5	2684.9	351.6	211.9
SST+RED	ClgS-*	0.2	1.0	2.0	1.8	3.9	1.8	1.1
	ClgM-*	71.1	3052.2	90.2	109.4	226.9	58.0	30.2
	ClgN1B-*	96.1	603.5	68.2	101.1	203.2	115.3	77.2
	ClgN1E-*	824.9	7220.9	365.4	583.7	2241.2	365.7	206.8
OPT+BMAX	ClgS-*	0.3	3.2	7.7	8.4	10.5	2.0	1.5
	ClgM-*	403.6	7205.9	2865.5	3604.6	7200.0	409.8	200.5
SST+BMAX	ClgS-*	0.3	3.1	8.3	8.3	10.7	2.1	1.3
	ClgM-*	380.6	7205.9	2260.4	3401.4	6214.3	400.0	181.9

Both, GRASP and VNS also produce high quality solutions with small advantages for VNS which seem to be more stable with respect to solution quality, i.e. it almost always produces slightly better average solutions than LD. Median run times of all approaches are given in Table 5, where MCF^{LP} denotes the LP-relaxation of MCF. The CPU-times of all our approaches are in the same order of magnitude as the times for solving the LP-relaxations of the MCF formulation, but high quality feasible solutions are identified in addition to the often better lower bounds. In further tests we observed that VNS and GRASP typically

produce quite good solutions very early in the search process. In that way they might be a fast alternative to solve practical instances when no performance guarantee is wanted.

9 Conclusions and Future Work

In this article we considered a generalized version of the (Price Collecting) Steiner Tree Problem where some customers have redundancy requirements. Based on an abstract version of a previously published multi-commodity flow formulation we proposed an approach based on Lagrangian decomposition which is stronger than the LP-relaxation of this MCF formulation from a theoretical point of view. Promising primal solutions are directly obtained and improved by a VND utilizing several types of neighborhoods. Furthermore, VNS and GRASP metaheuristics have been considered, making also use of the VND. Results indicate that combining Lagrangian decomposition with local search based metaheuristics produces near-optimal solutions with good performance guarantees, i.e. with relatively small gaps. In future we want accomplish a more detailed computational study with additional larger instances as well as consider an exact approach based on branch-and-price.

Acknowledgments. This work is supported by the Austrian Research Promotion Agency (FFG) under grant 811378, by the Austrian Science Fund (FWF) under grant 811378 and by the Austrian Exchange Service (Acciones Integradas, grant 13/2006). The authors further want to thank Ulrich Pferschy for fruitful discussions.

References

1. Barahona, F., Anbil, R.: The volume algorithm: producing primal solutions with a subgradient method. Mathematical Programming 87(3), 385–399 (2000)
2. Hansen, P., Mladenovic, N.: An introduction to variable neighborhood search. In: Voss, S., Martello, S., Osman, I.H., Roucairol, C. (eds.) Meta-heuristics, Advances and trends in local search paradigms for optimization, pp. 433–458. Kluwer Academic Publishers, Dordrecht (1999)
3. Feo, T., Resende, M.: Greedy randomized adaptive search procedures. Journal of Global Optimization 6(2), 109–133 (1995)
4. Wagner, D., Raidl, G.R., Pferschy, U., Mutzel, P., Bachhiesl, P.: A multicommodity flow approach for the design of the last mile in real-world fiber optic networks. In: Waldmann, K.H., Stocker, U.M. (eds.) Operations Research Proceedings 2006, pp. 197–202. Springer, Heidelberg (2007)
5. Karp, R.M.: Reducibility among combinatorial problems. In: Miller, E., Thatcher, J.W. (eds.) Complexity of Computer Computations, pp. 85–103. Plenum Press (1972)
6. Winter, P.: Steiner problem in networks: a survey. Networks 17(2), 129–167 (1987)
7. Segev, A.: The node-weighted Steiner tree problem. Networks 17(1), 1–17 (1987)

8. Balas, E.: The prize collecting traveling salesman problem. Networks 19, 621–636 (1989)
9. Kerivin, H., Mahjoub, A.R.: Design of survivable networks: A survey. Networks 46(1), 1–21 (2005)
10. ILOG: CPLEX 10.0. (2006), http://www.ilog.com
11. Wagner, D., Pferschy, U., Mutzel, P., Raidl, G.R., Bachhiesl, P.: A directed cut model for the design of the last mile in real-world fiber optic networks. In: Fortz, B. (ed.) Proceedings of the International Network Optimization Conference 2007, Spa, Belgium, pp. 1–6, 103 (2007)
12. Chimani, M., Kandyba, M., Mutzel, P.: A new ILP formulation for 2-root-connected prize-collecting Steiner networks. In: Arge, L., Hoffmann, M., Welzl, E. (eds.) ESA 2007. LNCS, vol. 4698, pp. 681–692. Springer, Heidelberg (2007)
13. Beasley, J.E.: Lagrangean relaxation. In: Reeves, C. (ed.) Modern heuristic techniques in combinatorial problems, pp. 243–303. Blackwell Scientific Publications, Malden (1993)
14. Bahiense, L., Barahona, F., Porto, O.: Solving Steiner tree problems in graphs with Lagrangian relaxation. Journal of Combinatorial Optimization 7(3), 259–282 (2003)
15. Haouari, M., Siala, J.C.: A hybrid Lagrangian genetic algorithm for the prize collecting Steiner tree problem. Computers and Operations Research 33(5), 1274–1288 (2006)
16. Suurballe, J.W., Tarjan, R.E.: A quick method for finding shortest pairs of disjoint paths. Networks 14, 325–335 (1984)
17. Kar, K., Kodialam, M., Lakshman, T.V.: Routing restorable bandwidth guaranteed connections using maximum 2-route flows. IEEE/ACM Transactions on Networking 11(5), 772–781 (2003)
18. Martins, S.L., Resende, M.G.C., Ribeiro, C.C., Pardalos, P.M.: A parallel GRASP for the Steiner tree problem in graphs using a hybrid local search strategy. Journal of Global Optimization 17(1-4), 267–283 (2000)
19. Mehlhorn, K.: A faster approximation algorithm for the Steiner problem in graphs. Information Processing Letters 27(3), 125–128 (1988)
20. Raidl, G.R.: A unified view on hybrid metaheuristics. In: Almeida, F., et al. (eds.) HM 2006. LNCS, vol. 4030, pp. 1–12. Springer, Heidelberg (2006)
21. Bachhiesl, P.: The OPT- and the SST-problems for real world access network design – basic definitions and test instances. Working Report 01/2005, Carinthia Tech Institue, Department of Telematics and Network Engineering, Klagenfurt, Austria (2005)