RALF BORNDÖRFER[1], STEPHAN SCHWARTZ[2], WILLIAM SURAU[3]

Vertex Covering with Capacitated Trees

[1] 0000-0001-7223-9174
[2] 0000-0003-2901-5065
[3] 0000-0003-1104-3383

Zuse Institute Berlin
Takustr. 7
14195 Berlin
Germany

Telephone: +49 30 84185-0
Telefax: +49 30 84185-125

E-mail: `bibliothek@zib.de`
URL: `http://www.zib.de`

ZIB-Report (Print) ISSN 1438-0064
ZIB-Report (Internet) ISSN 2192-7782

Vertex Covering with Capacitated Trees

Ralf Borndörfer Stephan Schwartz William Surau

Zuse Institute Berlin, Takustr. 7, 14195 Berlin

Abstract

The covering of a graph with (possibly disjoint) connected subgraphs is a fundamental problem in graph theory. In this paper, we study a version to cover a graph's vertices by connected subgraphs subject to lower and upper weight bounds, and propose a column generation approach to dynamically generate feasible and promising subgraphs. Our focus is on the solution of the pricing problem which turns out to be a variant of the NP-hard Maximum Weight Connected Subgraph Problem. We compare different formulations to handle connectivity, and find that a single-commodity flow formulation performs best. This is notable since the respective literature seems to have dismissed this formulation. We improve it to a new coarse-to-fine flow formulation that is theoretically and computationally superior, especially for large instances with many vertices of degree 2 like highway networks, where it provides a speed-up factor of 10 over the non-flow-based formulations. We also propose a preprocessing method that exploits a median property of weight constrained subgraphs, a primal heuristic, and a local search heuristic. In an extensive computational study we evaluate the presented connectivity formulations on different classes of instances, and demonstrate the effectiveness of the proposed enhancements. Their speed-ups essentially multiply to an overall factor of 20. Overall, our approach allows the reliabe solution of instances with several hundreds of nodes in a few minutes. These findings are further corroborated in a comparison to existing districting models on a set of test instances from the literature.

1 Introduction

Partitioning a graph subject to constraints on the components is a basic problem in graph theory, combinatorial optimization, and computer science: Connected and homogeneous components are supposed to contain related and similar items, respectively, and capacitated components of similar size can be distributed to processors of equal capacity. The treatment of such constraints is clearly at the heart of every solution method. We will argue in this paper that single commodity flows, which seem to have been dismissed in the graph partitioning literature, can be a versatile and powerful tool to deal with connectivity constraints, and that homogeneity can be expressed –in an algorithmically advantageous way– in terms of medians, at least for a certain type of graph partitioning problems that arise from districting. While these concepts are general and should carry over to other (graph partitioning) problems involving connectivity, our exposition will be based upon and is motivated by a concrete real-world application of toll enforcement. A recent survey on homogeneous covering and partitioning problems, as well as heuristic and exact solution approaches can be found in [Sch20].

Graph partitioning problems have a broad range of applications in parallel computation, scheduling, network design, problem decomposition, image processing, etc., see [Bul+16] for a survey. We are particularly interested in geographical districting, which finds its use in political districting [Hes+65; GN70; MJN98; RSS13] or the planning of school districts [FG90], police patrol areas [Ami+02; CHQ10; CCY19], sales territories [HS71; FP88; ZS05; SRC11], and even waste collection [CGP19]. In all these applications, a geographical map is represented as a graph, which is segmented into smaller regions, the districts. These should have desirable properties including contiguity, compactness, and balancedness with respect to criteria such as the number of voters, children, or customers. While contiguity and balancedness are clear, compactness is a more fuzzy goal that is typically handled in the objective function. Informally, the districts should be as "round-shaped" as possible. In political districting, this is supposed to counteract gerrymandering, i.e., the creation of politically biased voting districts. In other applications, the objective aims at minimizing the distance between any two points in the district, or between any point and the district center, see [KR19] for an overview of different formalizations of "compactness".

The problem motivating this study concerns the optimal toll enforcement on road networks with traffic volumes on the edges. Our goal is to find a segmentation of the network into smaller, connected, and possibly overlapping parts of similar size that contain roads with similar traffic volume. The latter property is referred to as "homogeneity" and corresponds to the compactness requirement of districts. Transforming the problem to the line graph blends in with the literature on districting problems.

As for the compactness, homogeneity is a vague quantity. A number of possibilities to measure homogeneity are overviewed by Hansen and Jaumard [HJ97]. All measures are based on the dissimilarity of values d_{uv} for every node pair (u, v) and aim at either maximizing the separation between different groups, or the homogeneity within the individual group. If we define dissimilarities as bee-line or shortest path distances, the concepts of compactness and homogeneity coincide and can be handled in the same way. We propose to do that by measuring homogeneity in terms of the difference $d_v - d_c$ to a district center c, whose node value d_c is the median of all node values in its district. The district center minimizes the sum of absolute differences and can be determined efficiently, and, even more important, it will turn out that the median property gives rise to a number of very effective preprocessing techniques that reduce the problem size substantially.

Districting problems are often solved heuristically, e.g., with tabu search, simulated annealing, and genetic algorithms [RSS13]. Exact solution approaches are rare. Early formulations enumerate all possible districts [GN70; Nyg88]. Mehrotra, Johnson, and Nemhauser [MJN98] propose a column generation approach to reduce the number of considered subgraphs. However, they restrict the set of feasible districts by first demanding that any district node has to be within a (unit edge-weight) radius of 2 from the district center, and, second, by requiring any district to be a subtree of the shortest path tree rooted at the district center. The latter restriction is likewise imposed by Clautiaux, Guillot, and Pesneau [CGP19], who also employ column generation for a districting problem. This additional constraint simplifies the pricing problem drastically: On a tree, it can be solved with a dynamic programming algorithm [CS97] or a special branch-and-bound procedure [SC98]. A different exact approach was proposed by Segura-Ramiro et al. [Seg+07]. Binary variables x_{uv} indicate if node u belongs to the district centered at node v. The connectivity of the districts is enforced by node separator inequalities that are similar to subtour elimination constraints. While the implementation of the model is not pursued in [Seg+07], a cutting plane approach for the separation of violated inequalities is implemented by Salazar-Aguilar, Ríos-Mercado, and Cabrera-Ríos [SRC11].

We consider in this paper a column generation approach that prices arbitrary connected districts, i.e., we dismiss the restriction to subtrees of shortest path trees. To the best of our knowledge, this general version has not been studied before, and we will show that the expanded scope has theoretical and practical consequences. Namely, our pricing problem becomes a rooted and budgeted variant of the well-known *Maximum Weight Connected Subgraph Problem (MWCS)*. The most prominent MIP approaches to the MWCS are based on single-commodity flows, multi-commodity flows, arc separators, and node separators, which we review and compare. Our results show that the single-commodity flow formulation provides the best overall performance on all considered instances. This is particularly interesting as most of the existing computational comparisons for MWCS formulations do not include this "forgotten" formulation. We propose to revive it and suggest an improvement that is based on the coarse-to-fine (C2F) paradigm, combining one coarse and many fine single commodity flows. This new coarse-to-fine single commodity flow formulation is particularly suited for graphs with a large fraction of degree 2 vertices, which is a typical property of transit networks such as motorway networks or public transport networks. In this case, the C2F flow formulation is theoretically superior to the standard single-commodity flow formulation, and it outperforms the non-flow-based formulations by a speed-up factor of 10.

Finally, we perform an extensive computational study. We report on computational experiments on 24 real-world instances from a toll enforcement problem on the German motorway network. We compare different connectivity formulations, and measure the impact of algorithmic components such as preprocessing, primal heuristics, and local search. Remarkably, these improvements are almost orthogonal and essentially multiply to an overall speed-up factor of 20. To expand the spectrum of scenarios, we propose a method to generate synthetic transit networks from Voronoi diagrams or trees. The computational study on these instances confirms the results from the real-world scenarios. Overall, our approach allows the reliable solution of instances with several hundreds of nodes in a few minutes. We also compare our SCF based column generation approach to the integrated IP approaches described above on a set of test instances from [SRC11]. The results show that our approach can compete with the existing models. In fact, it turns out to be the only one that can solve all instances within the given time limit.

The rest of the paper is structured as follows. In Section 2 we introduce our median approach to modeling homegeneity, formalize the problem, settle the complexity, and outline our column generation approach. Section 3 is dedicated to the pricing problem, which turns out to be a variant of the Maximum Weight Connected Subgraph Problem. We review MIP formulations for connectivity, propose a new coarse-to-fine single commodity flow approach, and show that it produces tighter formulations than the standard one. We also propose a number of preprocessing techniques for the pricing problem and present a local search heuristic. The presented formulations and improvements are evaluated in our computational study in Section 4, and we conclude in Section 5.

2 The HCTCP and a Column Generation Approach

In this section, we formalize the investigated covering problem, the HCTCP, discuss the penalization of inhomogeneity, and prove that the HCTCP is NP-hard. Finally, we present an integer program (IP) for this problem and describe a column generation approach for the linear relaxation.

Definitions and Notation For variables or parameters φ defined on a finite set X and for $X' \subseteq X$, we denote by $\varphi(X')$ the sum $\sum_{x \in X'} \varphi_x$. In addition, for a graph G and a vertex set $V' \subseteq V(G)$, we denote by $G[V']$ the induced subgraph by V'. Also, we write $v \in G$ instead of $v \in V(G)$.

For the *Homogeneous Capacitated Tree Covering Problem (HCTCP)* we consider an undirected graph $G = (V, E)$ with two vertex weights $w, d \in \mathbb{R}^V_{\geq 0}$ that can be interpreted as the weight and, respectively, the demand of a vertex. In addition, we are given weight bounds $0 \leq W_L \leq W_U$. By $\mathcal{T}_{L,U}$ we denote the set of trees $T \subseteq G$ with $W_L \leq w(T) \leq W_U$. The elements of $\mathcal{T}_{L,U}$ are also referred to as templates. The inhomogeneity of a template T is penalized by

$$p(T) := \min_{u \in T} \sum_{v \in T} w_v \, |d_u - d_v|. \tag{1}$$

The HCTCP is to find any number of trees $T_1, \ldots, T_t \in \mathcal{T}_{L,U}$ with $V = \bigcup_{i=1}^t V(T_i)$ minimizing $\sum_{i=1}^t p(T_i)$.

First, we note that the restiction to trees, instead of arbitrary connected subgraphs, is not a limitation. Since we are only concerned with vertex weights, adding additional edges to a subgraph does not affect its penalty nor the vertex covering. However, for brevity, it is more convenient to speak of trees instead of connected subgraphs. Next we discuss the choice of the inhomogeneity penalty.

Penalizing Inhomogeneity A few remarks concerning the penalty function are in place. Our goal is to penalize the inhomogeneity of a template T with respect to d, i.e., the values $(d_v)_{v \in T}$ should be as similar as possible. A number of measures for (in)homogeneity are listed by Hansen and Jaumard [HJ97], a recent survey can be found in [Sch20]. Our penalty function is a weighted version of the star measure discussed in both articles, i.e., in the star objective function each w_v is replaced by 1.

While general approaches consider arbitrary dissimilarities (d_{uv}) between node pairs, we are concerned with a node weight induced measure. Note that our measure can easily be extended to multidimensional vertex weights d by applying a vector norm instead of the absolute value.

More important, however, is that the (weighted) star measure induces a special center of the template: The vertex $u \in T$ that minimizes the (weighted) sum of differences. We call this node the *root* of the template, and since it is not necessarily unique, we accept any node that minimizes the expression as root of the template. It is well known that for the star objective, the root node is the one with median demand, since the median minimizes the sum of absolute deviations. Furthermore, this property carries over to the weighted median, as we will discuss in detail in Section 3.4. The minimization of the expression with multidimensional weights d was examined in [VZ00].

The results of this paper translate to the star objective, i.e., to the unweighted median case. We implemented and tested both variants. However, we decided to present the results focussing on the more general case of the weighted median. This is particularly important for our preprocessing routine presented in Section 3.4.

Complexity The HCTCP is NP-hard. A reduction from the number partitioning problem is straight forward. Given a multiset $S = \{s_1, \dots, s_n\}$ of positive integers, the partitioning problem asks for a partition into two subsets S_1 and S_2 whose elements sum up to the same number. By considering a complete graph on S with vertex weights $w_i = s_i$ and arbitrary d, and by setting $W_L = W_U = \frac{1}{2}\sum_{i=1}^{n} s_i$, we can see that this HCTCP instance has a feasible solution iff the partitioning instance is feasible.

Note that the reduction exploits that we can set $W_L = W_U$. While the general case remains open, we can prove NP-hardness under slightly modified conditions. If the cardinality t of a covering would be part of the input, we could reduce the bin packing problem to this HCTCP version without the use of a lower weight bound W_L (cf. [CGP19]). This still does not take the special objective function into account. Let us therefore assume that the covering cardinality is not specified and that the dissimilarities are given as a matrix (d_{uv}) instead of the node induced dissimilarity. In this case, a reduction from bin packing is still possible. Given a bin packing instance with item weights $w_1, \dots, w_n$ and bin capacity W, we consider the accordingly weighted complete graph and set $W_L = 0$ and $W_U = W$. By defining the dissimilarity matrix with values $d_{vv} = \frac{1}{w_v}$ and 0 off the diagonal, we attain that every subgraph has a penalty of 1. Hence, the objective of this HCTCP is to minimize the number of covering trees, which is equivalent to the bin packing problem.

A Column Generation Approach We start with a very basic IP formulation for the HCTCP. To this end, we consider the inhomogeneity penalty for template T as a parameter $p_T \geq 0$. Since the set $\mathcal{T}_{L,U}$ consists of all feasible templates, we can use the following classical covering IP formulation:

$$\min_{x} \quad \sum_{T \in \mathcal{T}_{L,U}} p_T \, x_T \tag{2a}$$

$$\text{s.t.} \quad \sum_{T \ni v} x_T \geq 1 \qquad \forall v \in V \tag{2b}$$

$$x_T \in \{0,1\} \qquad \forall T \in \mathcal{T}_{L,U} \tag{2c}$$

We have a binary variable for every feasible template (2c), and minimize the cumulated penalty (2a) while covering all vertices (2b). While this formulation is simple and self-evident, it is impracticable to enumerate all feasible templates or determine all associated penalties. Therefore, we propose a column generation scheme to iteratively generate only promising templates.

Using the linear programming relaxation and introducing dual variables (π_v) for the constraints (2b) we obtain the following dual LP.

$$\max_{\pi} \quad \sum_{v \in V} \pi_v \tag{3a}$$

$$\text{s.t.} \quad \sum_{v \in T} \pi_v \leq p_T \qquad \forall T \in \mathcal{T}_{L,U} \tag{3b}$$

$$\pi_v \geq 0 \qquad \forall v \in V \tag{3c}$$

Given the dual values corresponding to an optimal primal LP solution, the reduced costs for a template $T \in \mathcal{T}_{L,U}$ are therefore $p_T - \sum_{v \in T} \pi_v$. Hence, the pricing problem in a column generation approach is

$$\min_{T \in \mathcal{T}_{L,U}} p_T - \sum_{v \in T} \pi_v. \tag{4}$$

3 Solving the Pricing Problem

Let us take a closer look at the pricing problem. We are now looking for a single feasible template that minimizes a combination of its penalty and the dual values on the vertices. Resubstituting for p_T yields

$$\min_{T \in \mathcal{T}_{L,U}} \left(\min_{u \in T} \sum_{v \in T} w_v \, |d_u - d_v| \right) - \sum_{v \in T} \pi_v.$$

We can tackle this problem in two different ways. First, we can setup a single IP to find a template T with minimum reduced costs in G. Therefore, we introduce a binary variable y_v for each $v \in V$ that indicates if $v \in T$. For determining the penalty of the constructed template, we precompute all differences $d_{uv} = |d_u - d_v|$, and introduce a variable δ_v^u for every pair of vertices with $\delta_v^u = 1$ iff $v \in T$ and u is the root of T. Note that δ_u^u indicates if u is the root of T. The main challenge in the formulation, however, is to ensure the connectivity of the template. Let us ignore for the moment the specifics of this constraint and, istead, use the set $\mathcal{Y} := \{\chi(V') \ : \ V' \subseteq V \text{ such that } G[V'] \text{ is connected}\}$. The convex hull of $\mathcal{Y}$ is known as the connected subgraph polytope and we will discuss the modelling of $\mathcal{Y}$ in Section 3.1. A resulting IP formulation for the pricing problem can then be stated as follows.

$$\begin{aligned}
\min_{y,\,\delta} \quad & \sum_{u,v\,\in V} w_v\, d_{uv}\, \delta_u^v \;-\; \sum_{v\in V} \pi_v\, y_v && && \text{(5a)}\\
\text{s.t.} \quad & W_L \;\le\; \sum_{v\in V} w_v\, y_v \;\le\; W_U && && \text{(5b)}\\
& \sum_{u\in V} \delta_u^u \;=\; 1 && && \text{(5c)}\\
& \delta_u^u \;\le\; y_u && \forall v \in V && \text{(5d)}\\
& \delta_u^u + y_v - 1 \;\le\; \delta_v^u && \forall u, v \in V && \text{(5e)}\\
& y \;\in\; \mathcal{Y} && && \text{(5f)}\\
& \delta_v^u \;\in\; \{0,1\} && \forall u, v \in V && \text{(5g)}
\end{aligned}$$

The objective (5a) is straight forward, and (5b) enforce the weight constraints. Then, (5c) guarantees that exactly one root is chosen, and (5d) that the root is indeed part of the template. Constraints (5e) couple the root vertex with the template vertices, and, finally, we have the unspecific connectivity constraint (5f).

A second variant to solve the pricing problem is to consider the problem for fixed roots. As we will see, this significantly simplifies the formulation. The question is, if it is better to solve one large optimization problem or many smaller problems, i.e., one for every vertex. If we fix a vertex r to be the (potential) root of the template, we can eliminate the δ variables and obtain the following simpler IP formulation.

$$\begin{aligned}
\min_{y} \quad & \sum_{v\,\in V} (w_v\, d_{rv} \;-\; \pi_v)\, y_v && \text{(6a)}\\
\text{s.t.} \quad & W_L \;\le\; \sum_{v\in V} w_v\, y_v \;\le\; W_U && \text{(6b)}\\
& y_r \;=\; 1 && \text{(6c)}\\
& y \;\in\; \mathcal{Y} && \text{(6d)}
\end{aligned}$$

The objective (6a) is much simpler than in the preceding integrated approach. Furthermore, we can not only drop quadratically many δ variables, but also lose the complicated coupling constraints. This makes the problem much more amenable. Indeed, we found that it is far superior to solve a rooted IP (6) for every vertex, instead of solving the single, more complex IP (5). Therefore, let us further investigate the rooted formulation (6).

By defining $c_v := \pi_v - w_v\, d_{rv}$ and by replacing the objective (6a) with $\max_y \sum_{v\in V} c_v\, y_v$, we can transform the problem (6) into a maximization problem. Now, we are seeking a connected subgraph of maximum weight c that contains some root node r and that is capacitated from below and above with respect to vertex weights w. This problem is particularly close to the well-studied Maximum Weight Connected Subgraph Problem which we will outline in the following.

Maximum Weight Connected Subgraph (MWCS)

Instance: $G = (V, E)$, $c \in \mathbb{R}^V$.

Problem: Find $V' \subseteq V$ such that $G[V']$ is connected and $c(V')$ is maximal.

In the rooted version of MWCS, a given set $R \subseteq V$ has to be a subset of the selected nodes V'. In the budgeted variant, we are given additional weights $w \in \mathbb{R}^V_{\ge 0}$ and numbers $0 \le W_L \le W_U$, and demand that

$w(V') \in [W_L, W_U]$. Since the restriction to trees of maximum weight is again without loss of generality, our pricing problem is a single-rooted and budgeted MWCS.

A plethora of applications has driven the research on the MWCS and its variants. This includes applications in oil-drilling [HP94], communication network design [LD98; KLT15], systems biology [Ide+02; Dit+08; Yam+09; Bac+12], environmental conservation [Con+07; DG10], video activity detection [CG12], and forest planning [Car+13].

Despite these numerous studies, and to the best of our knowledge, the single-rooted, general budgeted MWCS has not been considered before. The only works that consider a lower bound $W_L > 0$ seem to be [HP94; LD96] and [LD98], where $w \equiv 1$ and $W_L = W_U = k$, i.e., a connected subgraph of maximum weight with exactly k nodes is sought. All three papers settle for suboptimal solutions. While Hochbaum and Pathria [HP94] present a dynamic program that is optimal on trees, but only provides a $\frac{1}{k}$-approximation for general graphs, Lee and Dooly [LD98] propose a heuristic that reduces the problem to the single-rooted case. The single-rooted case of k-MWCS is discussed in [LD96]. The authors present a single-commodity flow formulation, but dismiss the approach because the LP relaxation is not optimal. Instead, they propose heuristic algorithms based on enumerating possible vertex choices.

The works of Dilkina and Gomes [DG10] and Álvarez-Miranda, Ljubić, and Mutzel [ÁMLM13b] are closest to our pricing problem. Both consider a rooted and budgeted MWCS where only an upper weight bound is imposed, i.e., $W_L = 0$. The connectivity models and results therein are detailed in the following section.

3.1 Enforcing Connectivity in MIPs

In this section, we discuss various ways to model connectivity within a (mixed) integer program ((M)IP). A decent number of papers are dedicated to this problem and the related study of the connected subgraph polytope. These are mostly motivated by the Steiner tree problem and its variants concerning Steiner arborescences, prize-collecting Steiner tree, and maximum weight connected subgraphs. Similar connectivity formulations, apart from the vast literature concerning travelling salesman problems, originate for instance from the generalized minimum spanning tree problem [MLT95; Pop09; Pop20], a connected network design problem [MR05] or the minimum arborescence problem [Duh+08].

The first connectivity models for Steiner problems by Aneja [Ane80] and Wong [Won84] date back to the 1980s and propose a "row generation scheme" using minimum cuts, and, respectively, a multi-commodity flow. Since then, other formulations as well as new separation and preprocessing techniques have been developed. A review of different formulations and comparisons of the respective LP relaxations can be found in [GM93; KPH93; MW95; PD01; RFK20]. In addition, computational comparisons of distinct models for variants of the Steiner tree problem are carried out in [DG10; ÁMLM13a; ÁMLM13b; Fis+17]. An excellent survey article by [Lju20] covers all relevant topics concerning Steiner trees, including variants, MIP formulations, preprocessing techniques, and applications.

Complementing the variety of IP formulations, the problem has also been studied from a polyhedral perspective. The connected subgraph polytope is the convex hull of all node incidence vectors inducing a connected subgraph. A full description is known when the graph is a tree [KLS91], a cycle [Goe94b], series-parallel [Goe94b], or complete bipartite [Lüt18]. Other important facets and valid inequalities for the mentioned and related polytopes are presented in [CR94; KM98; ÁMLM13a; ÁMLM13b; WBB17]. The edge-induced connected subgraph polytope was studied in [Goe94a; Goe94b; KZ14; BKN15].

Computational studies show that preprocessing methods and primal heuristics generally have a huge impact on the solution time. Reduction tests can prove that certain nodes or edges must belong to every solution, or that they cannot be part of any solution. This helps to drastically reduce the problem size. A number of techniques for different Steiner problems are proposed in [CGR92; KM98; PD01; CCL06; Lju+06; GVHS08; EKK14; Lei+18; RK19; RKM19]. Unfortunately, these algorithms are highly problem specific, and most of the approaches do not translate to our pricing problem, since they conflict with our capacity constraints $W_L \leq w(T) \leq W_U$ or make use of Steiner terminals that we do not have. However, we will confirm the strong impact of our proposed preprocessing routine (cf. Section 3.4) as well as a local search procedure (cf. Section 3.5).

Let us now focus on the two works concerning the rooted and budgeted MWCS that are closest to our pricing problem. Dilkina and Gomes [DG10] compare three connectivity models: A single-commodity flow (SCF), a multi-commodity flow (MCF), and a Steiner arborescence (SA) formulation based on cuts. On 100 synthetic 10×10 grid instances with 3 roots, the computational comparison shows that the SCF LP relaxation is fastest but provides the worst integrality gap. The SA LP relaxation is also relatively fast and gives the best gap. The MCF LP relaxation is the slowest of the three models and the gap is between the SCF and SA relaxations. With respect to optimal integer solutions, the results in [DG10] indicate that SCF is best if the upper weight bound W_U is so large that it almost can be ignored. In the other case, however, SA performs better on the considered instances.

Álvarez-Miranda, Ljubić, and Mutzel [ÁMLM13b] refrain from including the SCF formulation into their computational comparison, and only evaluate the SA formulation against a node separator (NS) formulation. They find that the performance of the two formulations is complementary, and depends on the instance. It seems as the NS formulation with fewer variables performs better on dense graphs, whereas the SA formulation seems better suited for sparser graphs.

Now that we have reviewed the literature on MIP formulations for connected subgraphs, with an emphasis on the MWCS, we shift our focus to the single-rooted case. For a graph $G = (V, E)$ and a node $r \in V$, an r-tree is a tree of G containing r. Consequently, we study the r-tree polytope, i.e., the convex hull of

$$\mathcal{Y}_r := \{\chi(V') \;:\; V' \subseteq V \text{ is the vertex set of an } r\text{-tree}\}.$$

Note that $\mathcal{Y}_r$ is, in fact, a facet of the connected subgraph polytope. Goemans [Goe94b] studies the related r-tree polytope $conv(\mathcal{Y}_r^+)$ in the dimension $\{0,1\}^{|E|+|V|}$, i.e., he considers a concatenation of edge and node incidence vectors of r-trees. Clearly, $conv(\mathcal{Y}_r)$ is a projection of $conv(\mathcal{Y}_r^+)$. Since Goemans [Goe94b] gives a full description of $conv(\mathcal{Y}_r^+)$ if G is a cycle or series-parallel, the optimization over $conv(\mathcal{Y}_r)$ is also polynomial in these cases. We note that in [Goe94b], the empty graph is also considered an r-tree to simplify the polyhedral study. Since this edge case is not practically relevant, and for ease of exposition, we exclude the empty graph from $\mathcal{Y}_r$.

The presence of a single root simplifies the (M)IP formulations. Hence, we present four important formulations that describe $\mathcal{Y}_r$. For some of the formulations we consider the bidirected version $D = (V, A)$ of the undirected graph $G = (V, E)$.

Single-Commodity Flow (SCF) A single-commodity flow to ensure connectivity was proposed by Gavish and Graves [GG81] in the context of the travelling salesman problem. Maculan [Mac87] transforms the formulation to the Steiner tree problem. Similar SCF models are used in [LD96; Shi05; Con+07] and [DG10]. The latter two papers consider a rooted and budgeted (only with upper bounds) version of MWCS. Our formulation is adapted for the single-root case. This spares us from introducing an artificial super source.

$$\begin{aligned} y_r &= 1 && && \text{(7a)} \\ \sum_{a \in \delta^-(v)} x_a - \sum_{a \in \delta^+(v)} x_a &= y_v && \forall v \in V \setminus \{r\} && \text{(7b)} \\ x_{uv} &\leq M\, y_v && \forall (u,v) \in A && \text{(7c)} \\ x_a &\geq 0 && \forall a \in A && \text{(7d)} \\ y_v &\in \{0,1\} && \forall v \in V && \text{(7e)} \end{aligned}$$

The idea is to construct a flow that emerges at the root node and where each node of the chosen subgraph consumes one unit of flow while all other nodes satisfy flow conservation. This is ensured by equalities (7b). Inequalities (7c) are necessary to activate every node that is used by the flow. The use of a big M parameter is bad for the LP relaxation, but necessary. It should be chosen as small as possible, and $M = |V| - 1$ is an upper bound.

Multi-Commodity Flow (MCF) The first multi-commodity flow formulation for connectivity is due to Beasley [Bea84] and, independently, to Wong [Won84]. Maculan [Mac87] shows that the LP relaxation of MCF is stronger than the SCF relaxation. As stated above, Dilkina and Gomes [DG10] empirically confirm this result on synthetic grid instances, but find that SCF performs much better. Our results on real-world and fabricated instances substantiate this conclusion.

$$
\begin{aligned}
y_r &= 1 && && \text{(8a)}\\
\sum_{a \in \delta^-(w)} x_a^v - \sum_{a \in \delta^+(w)} x_a^v &= 0 && \forall v, w \in V \setminus \{r\}, w \neq v && \text{(8b)}\\
\sum_{a \in \delta^-(v)} x_a^v &= y_v && \forall v \in V \setminus \{r\} && \text{(8c)}\\
\sum_{a \in \delta^+(v)} x_a^v &= 0 && \forall v \in V \setminus \{r\} && \text{(8d)}\\
x_a^w &\leq y_v && \forall v, w \in V \setminus \{r\},\, a \in \delta^-(v) && \text{(8e)}\\
x_a^v &\geq 0 && \forall a \in A,\, \forall v \in V \setminus \{r\} && \text{(8f)}\\
y_v &\in \{0,1\} && \forall v \in V && \text{(8g)}
\end{aligned}
$$

In the MCF formulation, we consider an r-v flow x^v for every node $v \neq r$. Flow conservation is stated in (8b), and the flow value is determined by (8c) and (8d): It is set to 1, if node v is chosen, and otherwise set to 0. Finally, (8e) guarantees, that a node is chosen if any flow uses it.

The advantage of the MCF formulation is that the arc flow is binary and hence, we have no big M. This benefits the LP relaxation of the MCF. On the other hand, the number of variables and constraints is much larger than in the SCF formulation.

Rooted Arc Separators (RAS) A second way to enforce connectivity is with separators. First, we consider a formulation using edge cuts. As our problem is stated on an undirected graph, we can stick with the natural undirected formulation, first proposed by Aneja [Ane80], or consider the corresponding formulation in the bidirected graph. Chopra and Rao [CR94] show that the LP relaxation of the bidirected formulation is tighter than for the undirected case. Goemans and Myung [GM93] further investigate the relation between the two and give an extended undirected relaxation that is equivalent to the bidirected arc cut relaxation. However, we present the in the literature predominantly used arc formulation, that goes back to a Steiner arborescence formulation by Wong [Won84]. Similar models are applied in [KM98; Lju+06; Duh+08; DG10; ÁMLM13b; Fis+17; RK19].

$$\begin{aligned} y_r &= 1 && && \text{(9a)}\\ \sum_{a\in\delta^-(v)} x_a &= y_v && \forall v \in V\setminus\{r\} && \text{(9b)}\\ \sum_{a\in\delta^-(S)} x_a &\geq y_v && \forall v \in S,\, \forall S \subseteq V\setminus\{r\} && \text{(9c)}\\ x_a &\in \{0,1\} && \forall a \in A && \text{(9d)}\\ y_v &\in \{0,1\} && \forall v \in V && \text{(9e)} \end{aligned}$$

If the x variables span an arborescence rooted at r, then the nodes of the arborescence form a connected subgraph in G. Therefore, we have (9b) to activate the nodes of the arborescence, and to ensure an in-degree of 1. In addition, we have (9c) to ensure that there is a directed r-v-path, if v is chosen.

The set $\delta^-(S)$ in (9c) is an arc separator for r and v, i.e., there is no r-v-path in $D \setminus S$. Since there is an exponential number of these constraints, the LP relaxation is solved with a cutting plane approach. The crucial ingredient is the separation routine, where violated constraints are identified. If the maximum flow value from r to v in D with arc capacities x is smaller than y_v, the inequality of any corresponding minimum cut is violated. Hence, the separation can be solved efficiently with $n-1$ maximum flow computations.

There are a number of possibilities to strengthen the LP relaxation of the model (see [KM98; Lju+06; Fis+17]). For instance, the initial model should be extended with 2-cycle inequalities $x_{uv} + x_{vu} \leq y_v$ for every $(u,v) \in A$, allowing at most one of each bidirected arc pair to be part of the arborescence. Also, the valid inequalities $y_v \geq x_{vw}$ for every $(v,w) \in A$, $v \neq r$ should be included at the beginning. Ljubić et al. [Lju+06] note that for any separator S and for $v \in S$ we can add the equations (9b) for nodes in S and subtract the inequality (9c) to obtain the valid inequality

$$\sum_{a\in A\cap S^2} x_a \leq \sum_{u\in S\setminus\{v\}} y_u.$$

These *generalized subtour elimination constraints* are already studied by Goemans [Goe94b], and found to be facet-defining under certain circumstances.

Other tricks are concerned with improving the separation routine. First of all, we have to decide when to cut off fractional solutions of the LP relaxation. While Ljubić et al. [Lju+06] decide to check for a violated inequality for each v with $y_v > 0$, Fischetti et al. [Fis+17] choose a threshold of $y_v \geq 0.5$ for a fractional separation, and Dilkina and Gomes [DG10] separate only for $y_v > 1 - \varepsilon$. Further improvements like back cuts, nested cuts, and the use of minimum cardinality separators were proposed by Koch and Martin [KM98] and empirically confirmed in [Lju+06] and [Fis+17].

The idea of back cuts, that was already described by Chopra, Gorres, and Rao [CGR92], is to find a maximum flow (minimum cut) not only in the described network but also in the reversed network where every arc is flipped. As the resulting cuts tend to be different, more cuts are generated and the number of cutting plane iterations empirically decreases. Ljubić et al. [Lju+06] note that both cuts (forward and backward) can be found with one maximum flow calculation using a specific implementation from [CG95].

Nested cuts are another way to generate more cuts in a single cutting plane iteration. The idea is to set all capacities of already found minimum cut arcs to 1, and to iterate until the flow value is 1. This ensures that arc sets of found violated cuts are disjoint.

The third approach for enhancing the separation is to use minimum cuts of minimum cardinality. Therefore, a small $\varepsilon > 0$ is added to every capacity before computing a maximum flow. While this might lead to increased running times of flow calculations, the use of minimum cardinality cuts can have a great impact on the overall performance. Indeed, the results are split. While [KM98] and [Fis+17] confirm a very positive effect, [Lju+06] report extended running times.

Rooted Node Separators (RNS) The rooted node separators formulation is similar to the previously described RAS, but can be formulated on the original undirected graph and uses only node variables. Such a formulation was first used by Fügenschuh and Fügenschuh [FF08] and later applied in [Bac+12; ÁMLM13b; Car+13; ÁMS17; Fis+17].

Let us denote by $\mathcal{N}(r,v)$ the set of r-v-node separators, i.e., all sets $S \subseteq V$ such that there is no r-v-path in $G \setminus S$.

$$\begin{aligned} y_r &= 1 && && \text{(10a)} \\ \sum_{w\in S} y_w &\geq y_v && \forall v \in V \setminus \{r\},\ \forall S \in \mathcal{N}(r,v) && \text{(10b)} \\ y_v &\in \{0,1\} && \forall v \in V && \text{(10c)} \end{aligned}$$

The formulation is straight forward. Inequalities (10b) ensure that if node v is chosen, then any r-v-node separator must also contain a chosen node.

For the separation routine we describe two possibilities. The first one is in accordance with the RAS separation and described, for instance, in [FF08] and [ÁMLM13b]. A directed auxiliary graph D is created by splitting each node v into an arc (v_{in}, v_{out}) with capacity y_v. Every edge $vw \in E$ is replaced by the arcs (v_{out}, w_{in}) and (w_{out}, v_{in}), each with capacity 1. Now, all procedures from the RAS separation carry over, when comparing the maximum r_{out}-v_{in}-flow value to y_v.

Since the described separation procedure is rather time consuming, Fischetti et al. [Fis+17] propose a different variant that ignores violations of (10b) for fractional solutions. Instead, their separation routine is only called when the branching produced an integer solution. A great advantage of this approach is that the authors of [Fis+17] show that minimal separators for integer solutions can be found in linear time.

Spanning Tree Heuristic (STH) The idea of the spanning tree heuristic is to determine a suitable spanning tree T of G and to solve the HCTCP on T instead of G. Magnanti and Wolsey [MW95] give a full description of $\mathcal{Y}_r$ if G is a tree. Let us denote by $\pi(v)$ the predecessor of $v \neq r$ on the unique r-v-path. Then, the following inequalities are a complete description of $\mathcal{Y}_r$.

$$\begin{aligned} y_r &= 1 && && \text{(11a)} \\ y_v &\leq y_{\pi(v)} && \forall v \in V \setminus \{r\} && \text{(11b)} \\ y_v &\geq 0 && \forall v \in V && \text{(11c)} \end{aligned}$$

A maximum weight r-subtree of an r-tree is usually determined by a dynamic program, see [MW95]. The authors also propose a dynamic programming approach for the capacitated version, but only for the case $W_L = 0$, i.e., without a lower weight bound. We refrain from developing an adjusted dynamic program for the general capacitated case. Instead, we use the LP formulation (11), which shows to be solved very fast.

While this approach works with any spanning tree, we choose one that is particularly suited for the median objective. In particular, we consider a bidirected version of G and set arc weights $\varphi_{u,v} := |d_v - d_r|$ for arc (u,v). The chosen spanning tree is now a shortest-path tree from r with respect to φ.

3.2 Coarse-to-Fine Flow Formulation

In this section, we discuss how to improve the SCF for graphs with a large fraction of degree 2 vertices. These graphs usually contain long induced paths where every vertex has degree 2. We exploit the fact that with any chosen vertex of such a path (not containing the root), at least one of the endpoints of the path has to be chosen as well. Following this idea, the coarse network is formed by contracting the specified paths. Moreover, the connectivity on each of the paths can be modeled as a separate SCF with two possible roots, the fine flow. In doing so, we are able to set smaller big M values on the fine flow problems which leads to a better LP relaxation than the ordinary SCF.

Construction Given is a graph $G = (V, E)$ together with a root $r \in V$ and the vertices of the coarse network $V_c \subseteq V$ with $r \in V_c$. Let $V_{>2} \subseteq V$ be the vertices with degree greater than 2, and we demand $V_{>2} \subseteq V_c$. Note that with this condition, each connected component of the induced subgraph $G[V \setminus V_c]$ is a path. The arcs of the coarse network $D_c = (V_c, A_c)$ are constructed as follows: Whenever there exists a path between two coarse vertices $u, v \in V_c$, only containing vertices from $V \setminus V_c$ (except for u and v), the path is replaced by two coarse arcs (u, v) and (v, u). Observe that this construction also replaces an edge between two coarse vertices with two arcs. For ease of exposition and w.l.o.g. we assume that the vertex set V_c is chosen in a way such that A_c is a simple set: When $(u, v) \in A_c$ is a multiarc then at least one arc originates from a path that contains vertices from $V \setminus V_c$. By lifting one of these vertices to V_c the resulting coarse network does not contain the corresponding arc anymore. For an arc $(u, v) \in A_c$, we denote by $P_f(u, v)$ the set of vertices from $V \setminus V_c$ on the unique path between u and v.

Recall that the bidirectional graph of G is denoted by $D = (V, A)$. The fine network is then given by $D_f = (V, A_f)$, where $A_f = A \setminus A_c$.

MIP Model For each $v \in V$ we introduce the variable $y_v \in \{0, 1\}$ which is equal to 1 iff v is part of the subgraph rooted in r. Every arc $a \in A_c$ in the coarse network is associated with a flow variable $\hat{x}_a \in \mathbb{R}_{\geq 0}$. Similarly, we introduce for every arc $a \in A_f$ the flow variable $x_a \in \mathbb{R}_{\geq 0}$. The flow variables in the coarse network are bounded from above by $M = |V| - 1$. Let $(u, v) \in A_f$ and $(i, j) \in A_c$ such that $v \in P_f(i, j)$. We set the upper bound M_{uv} to the maximum of the two shortest path distances from u to i and to j, respectively. For simplicity, we assume that all leaves in G are part of the coarse vertices V_c. Note that M_{uv} is significantly smaller than the parameter M for the coarse flow. Now we can state the MIP formulation for the coarse-to-fine flow (C2F):

$$y_r = 1 \tag{12a}$$

$$\sum_{a \in \delta^-_{D_c}(v)} \hat{x}_a - \sum_{a \in \delta^+_{D_c}(v)} \hat{x}_a - \sum_{a \in \delta^+_{D_f}(v)} x_a = y_v \qquad \forall v \in V_c \setminus \{r\} \tag{12b}$$

$$\hat{x}_{uv} \leq y_v \cdot M \qquad \forall (u, v) \in A_c \tag{12c}$$

$$\hat{x}_{\hat{a}} + x_a \leq y_w \cdot M \qquad \forall \hat{a} \in A_c \ \forall w \in P_f(\hat{a}) \ \forall a \in \delta^-_{D_f}(w) \tag{12d}$$

$$\sum_{a \in \delta^-_{D_f}(v)} x_a - \sum_{a \in \delta^+_{D_f}(v)} x_a = y_v \qquad \forall v \in V \setminus V_c \tag{12e}$$

$$x_{uv} \leq y_u \cdot M_{uv} \qquad \forall (u, v) \in A_f : u \in V_c \tag{12f}$$

$$x_{uv} \leq y_v \cdot M_{uv} \qquad \forall (u, v) \in A_f : v \notin V_c \tag{12g}$$

$$y_v \in \{0, 1\} \qquad \forall v \in V \tag{12h}$$

$$\hat{x}_a \geq 0 \qquad \forall a \in A_c \tag{12i}$$

$$x_a \geq 0 \qquad \forall a \in A_f \tag{12j}$$

Observe that the C2F formulation has similarities to the SCF formulation (7). The coarse flow is handled in the constraints (12b) and (12c). In the former constraints, every chosen coarse vertex consumes not only one unit of flow, but also the cumulative fine flow that emerges from the vertex. The latter constraints (12c) force the head of a coarse arc to be chosen if it has flow. Constraints (12d) ensure that with a coarse arc $a \in A_c$ all fine vertices of the set $P_f(a)$ have to be chosen as well. The addition of the fine flow variable in (12d) is not necessary but strengthens the LP relaxation. Constraints (12e) - (12g) handle the fine flow, which is basically a SCF that can emerge at any coarse vertex. In order to activate a coarse vertex that serves as a source for the fine flow, constraints (12f) are necessary.

Let us now study the C2F and SCF formulations from a polyhedral perspective. We show that the LP relaxation for the C2F model is tighter than for the SCF. To this end, we denote by $\mathcal{C}$ and $\mathcal{F}$ the feasible solutions of the LP relaxation of the C2F and SCF model, respectively, projected onto the y variables, i.e.,

$$\begin{aligned}\mathcal{C} &:= \left\{ y \in [0,1]^V \;\middle|\; \exists \hat{x} \in \mathbb{R}^{A_c}_{\geq 0}\; \exists x \in \mathbb{R}^{A_f}_{\geq 0} : (y, \hat{x}, x) \text{ satisfy (12a) - (12g)} \right\},\\ \mathcal{F} &:= \left\{ y \in [0,1]^V \;\middle|\; \exists x \in \mathbb{R}^{A}_{\geq 0} : (y, x) \text{ satisfy (7a) - (7c)} \right\}.\end{aligned}$$

Proposition 1. *Let $G = (V, E)$ be a graph together with a root $r \in V$, and let $V_c \subseteq V$ with $r \in V_c$ and $V_{>2} \subseteq V_c$ be a set of coarse vertices, then*

$$\mathcal{C} \subseteq \mathcal{F}.$$

Proof. The coarse-to-fine formulation is characterized by the set of coarse vertices, to avoid ambiguities we call the polytope corresponding to the LP relaxation $\mathcal{C}_{V_c}$. Let $y \in \mathcal{C}_{V_c}$ be a feasible solution and let $\hat{x} \in \mathbb{R}^{A_c}_{\geq 0}$ and $x \in \mathbb{R}^{A_f}_{\geq 0}$ be flow values such that $(y, \hat{x}, x)$ satisfy (12a) - (12g). If $V_c = V$ the coarse network D_c is equal to D and no flow on fine vertices exists. Therefore, $(y, \hat{x})$ satisfies (7a) - (7c) and $y \in \mathcal{F}$.

Now we show that if $V_c \subset V$ we can always lift a non-empty vertex set $P_f(a)$ for $a \in A_c$ to the coarse network such that $y \in \mathcal{C}_{V_c \cup P_f(a)}$. Let $(u, v) \in A_c$ be any coarse arc with $P_f(u, v) \neq \varnothing$. The new coarse network is $D'_c = (V'_c, A'_c)$ where $V'_c = V_c \cup P_f(u, v)$ and A'_c are the resulting coarse arcs from the procedure described above. The resulting fine graph is $D'_f = (V, A'_f)$ where $A'_f \subset A_f$ contains no arc incident to vertices in $P_f(u, v)$. For the new flow vector of the fine graph $x' \in \mathbb{R}^{A'_f}_{\geq 0}$, we simply set $x'_a = x_a$ for all $a \in A'_f$. In the new flow vector for the coarse graph $\hat{x}' \in \mathbb{R}^{A'_c}_{\geq 0}$, we set any arc $a \in A'_c \cap A_c$ to $\hat{x}'_a = \hat{x}_a$. Now let $(i, j) \in A'_c \cap A_f$ be an arc on the corresponding directed path P_{uv} from u to v. W.l.o.g. we assume $\hat{x}_{uv} \geq \hat{x}_{vu}$, and we set $\hat{x}'_{ij} = \hat{x}_{uv} + x_{ij} - x_{ji}$ and $\hat{x}'_{ji} = 0$. In order to prove that $y \in \mathcal{C}_{V_c \cup P_f(a)}$ we have to show that $(y, \hat{x}', x')$ satisfies (12a) - (12g). Trivially, constraints (12a) and (12d) - (12g) are fulfilled. For (12b) let $w \in P_f(u, v)$, and $i \in V'_c$ be the predecessor and $j \in V'_c$ be the successor of w on P_{uv}. Then

$$\begin{aligned}&\sum_{a \in \delta^-_{D'_c}(w)} \hat{x}'_a - \sum_{a \in \delta^+_{D'_c}(w)} \hat{x}'_a - \sum_{a \in \delta^+_{D'_f}(w)} x'_a\\ &= \hat{x}'_{iw} + \hat{x}'_{jw} - \hat{x}'_{wi} - \hat{x}'_{wj}\\ &= \hat{x}'_{iw} - \hat{x}'_{wj}\\ &= \hat{x}_{uv} + x_{iw} - x_{wi} - \hat{x}_{uv} - x_{wj} + x_{jw}\\ &= x_{iw} - x_{wi} - x_{wj} + x_{jw}\\ &= \sum_{a \in \delta^-_{D_f}(w)} x_a - \sum_{a \in \delta^+_{D_f}(w)} x_a\\ &= y_v,\end{aligned}$$

and (12b) is satisfied. Furthermore, constraint (12c) is satisfied as well because

$$\begin{aligned}\hat{x}'_{iw} &= \hat{x}_{uv} + x_{iw} - x_{jw}\\ &\leq \hat{x}_{uv} + x_{iw}\\ &\leq y_w \cdot M.\end{aligned}$$

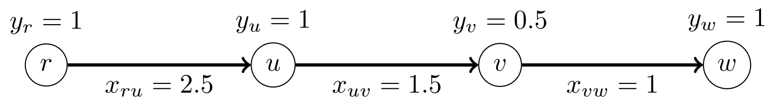


Figure 1: Feasible solution of the LP relaxation of SCF (7). The y values, however, do not allow for a feasible solution of C2F (12) with $V_c = \{r\}$.

We conclude that $y \in \mathcal{C}_{V_c \cup P_f(a)}$, and we iterate this process until $V_c = V$. □

The preceding result established that the C2F formulation is at least as tight as the SCF formulation. Next we show that it is indeed tighter.

Proposition 2. *In general, we do not have $\mathcal{C} = \mathcal{F}$.*

Proof. Consider the fractional flow depicted in Figure 1. Since we have $M = 3$ for the SCF formulation (7), $y_v = 0.5$ is feasible. For the C2F model with $V_c = \{r\}$, on the other hand, we have $M_{uv} = 2$, and hence $y_v = 0.5$ is not feasible anymore. □

Our implementation of the C2F model differs from (12) in order to obtain a tighter bound on the big M value for the coarse flow. By ignoring the fine flow in (12b) we achieve smaller values on the coarse flow and can therefore set $M_c = |V_c| - 1$. More specifically, we replace the constraints (12b) - (12d) with the following constraints:

$$\sum_{a \in \delta^-_{D_c}(v)} \hat{x}_a - \sum_{a \in \delta^+_{D_c}(v)} \hat{x}_a = y_v \qquad \forall v \in V_c \setminus \{r\} \tag{13b}$$

$$\hat{x}_{uv} \leq y_v \cdot M_c \qquad \forall (u,v) \in A_c \tag{13c}$$

$$\hat{x}_a \leq y_w \cdot M_c \qquad \forall a \in A_c \ \forall w \in P_f(a) \tag{13d}$$

3.3 Reduction Techniques for the Pricing Problem

When we consider a fixed root r, we might be able to eliminate certain parts of the network, or even exclude r from the set of possible roots. In this section we investigate possibilities to reduce the size of the pricing problem by applying different techniques during a preprocessing phase.

$\mathbf{W_U}$-radius A simple and yet effective method is to exclude all vertices that are not within distance W_U to r w.r.t. the node weights w. This can be done with Dijkstra's algorithm on the bidirected version of G where arc (u,v) has weight $w_{uv} := w_v$. The resulting graph is called the root graph for root r.

Minimum Separators For very sparse graphs, it might be useful to compute a minimum r-v-separator $S \subseteq V$ for every $v \in V$ in advance, and to add the valid inequality

$$y_v \leq y(S)$$

to the respective IP. A minimum r-v-separator can be computed with a maximum flow algorithm. Hao and Orlin [HO92] propose an implementation, where the computation time to compute minimum cuts from r to all other nodes is comparable with the time to find a single minimum r-v-cut.

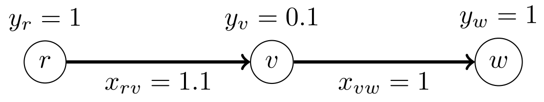


Figure 2: Feasible solution of the LP relaxation of (7) (single-commodity flow). The solution is not feasible if inequalities (14) are added to (7).

Articulation Points A special case of these inequalities arises when S has only one element: an articulation point. Lüthen [Lüt18] also followed this direction but found that there are too many cuts to add. We concentrate on a subset of these that proves to be very useful. If $\{u\}$ is an r-v-separator and $uv \in E$, then the following inequality, which we call *Articulation Point Neighborhood Cut (APNC)*, is valid:

$$y_v \;\leq\; y_u. \tag{14}$$

Note that if G is a tree, every node is an articulation point, and, as we have seen in formulation (11), these inequalities then essentially describe the r-tree polytope (cf. [MW95]). However, we can also use them to strengthen the LP relaxation of other connectivity formulations, as the example in Figure 2 shows. Moreover, all APNC can be computed efficiently, since articulation points can be enumerated by a depth-first search [HT73].

Goemans [Goe94b] also exploits articulation points to facilitate connectivity requirements in rooted tree problems: If v is an articulation point in a graph G with root r, then a (maximum weight) rooted tree problem can be solved independently for any connected component of $G \setminus v$ not containing r, by defining v as the new root of this component. Since our templates are subject to weight bounds, however, this approach does not carry over to the HCTCP.

Restricting Connectivity Formulations to the Core Graph If all APNC were added, we can further simplify the graph. In particular, we create the *core graph* by removing all outer branches of the original graph, i.e., we successively remove nodes of degree 1 unless the node is r. The APNC guarantee that any chosen nodes in the removed part are connected to the core graph and that the respective articulation node is chosen as well. Consequently, our IP connectivity constraints can be restricted to the core graph. For very sparse, tree-like instances, which we mostly consider here, the restriction to the core graph proves to have an enormous impact.

3.4 Exploiting the Median Property

The remainder of this section is dedicated to reductions under the assumption that r has (weighted) median demand of the vertices of selected r-trees. All presented reductions translate to the median case, if we consider the star penalty function instead of the weighted version (1). The presentation for the weighted version, however, is more general.

Definition 3 (weighted median)**.** *Given elements $d_1 < \cdots < d_n$ with positive weights $w_1, \ldots, w_n$ and $\Sigma_w := \sum_{i=1}^n w_i$, element d_k is a weighted median of (d, w) if*

$$\sum_{i=1}^{k-1} w_i \;\leq\; \Sigma_w/2 \quad \textit{and} \quad \sum_{i=k+1}^{n} w_i \;\leq\; \Sigma_w/2.$$

As for the median of an even number of values, the weighted median can be not unique. If one of the inequalities is tight, there are two consecutive elements that satisfy the condition. In this case we accept both elements as a weighted median. However, we do not accept any value inbetween the two. In particular, we do not take the arithmetic mean of both values as the weighted median.

The next result implies that the root nodes have a weighted median demand in the corresponding subgraph.

Proposition 4. *Let $d_1 < \cdots < d_n$ and $w_1, \ldots, w_n > 0$.*
Then d_u is a weighted median of (d, w) iff $u \in \arg\min_{u\in[n]} \sum_{v\in[n]} w_v \, |d_u - d_v|$.

Proof. Suppose that d_u is a weighted median of (d, w) and consider some $j \in [n]$, w.l.o.g. $j < u$. Define $\Delta := d_u - d_j$ and note that $\Delta > 0$. Now,

$$\begin{aligned}
&\sum_{v\in[n]} w_v \, |d_j - d_v| - \sum_{v\in[n]} w_v \, |d_u - d_v| \\
&= \sum_{v=1}^{j} w_v \, (d_j - d_u) + \sum_{v=j+1}^{u-1} w_v \, (2d_v - d_j - d_u) + \sum_{v=u}^{n} w_v \, (d_u - d_j) \\
&\geq \, - \Delta \sum_{v=1}^{j} w_v - \Delta \sum_{v=j+1}^{u-1} w_v + \Delta \sum_{v=u}^{n} w_v \\
&= \Delta \left(\sum_{v=u}^{n} w_v - \sum_{v=1}^{u-1} w_v \right) \, \geq \, 0.
\end{aligned}$$

Note that if the weighted median is unique, then the last inequality is strict. In the other case and if d_j is not a weighted median of (d, w), the first inequality is strict. ☐

For our purpose it is not necessary that the elements are distinct. If multiple vertices have the same demand, however, each of the vertices with weighted median demand minimizes the penalty function and can serve as root of the subgraph at hand. Let us now formally define the problem of eliminating potential root nodes.

Median Feasibility Problem

Instance: An HCTCP instance (G, w, d, W_L, W_U) and a node $r \in V(G)$.

Problem: Is there a tree $T \in \mathcal{T}_{L,U}$, $r \in T$, for which d_r is the weighted median of (d, w)?

The Median Feasibility Problem is NP-hard. The reduction from the Partition Problem in Section 2 also translates to this case. However, here we can also prove NP-hardness without exploiting the upper weight bound. Therefore, we consider the following related problem.

Balanced Connected Subgraph

Instance: Graph $G = (V, E)$, $V = V_R \,\dot\cup\, V_B$, $k \in 2\mathbb{N}$.

Problem: Is there a $V' \subseteq V$, $|V'| \geq k$, with $|V' \cap V_R| = |V' \cap V_B|$ such that $G[V']$ is connected.

Bhore et al. [Bho+19] show that the Balanced Connected Subgraph problem is NP-hard by a reduction from Exact-Cover-by-3-Sets. The problem remains NP-hard on simpler graph classes like planar or bipartite graphs, and also if a specified vertex r is required to be chosen (Balanced Connected r-Subgraph).

Proposition 5. *The Median Feasibility Problem is NP-hard, even without an upper weight bound W_U.*

Proof. The reduction is from Balanced Connected r-Subgraph. Given an instance (G, V_R, V_B, r, k), we construct a Median Feasibility instance as follows. We add a node v^* that serves as root node for the Median Feasibility Problem, and the edge $\{rv^*\}$ to G. Furthermore, we set $w \equiv 1$, $d_{v^*} = 0$, $d_v = 1$ if

$v \in V_R$, and $d_v = -1$ if $v \in V_B$, as well as $W_L = k+1$ which is an odd number. Note that any subgraph with $k+1$ nodes, for which v^* has the (unit weighted) median demand, has the same number of V_R and V_B vertices. Thus, the Median Feasibility instance has a solution if and only if the Balanced Connected r-Subgraph instance has a solution. □

Despite this complexity result, we can still simplify the instance or exclude potential roots. Our preprocessing algorithm is based on two routines: one that discards nodes from the graph, and one that fixes nodes. If we find that a node cannot belong to any feasible template rooted at r, we can remove this node. Accordingly, if a node has to belong to any feasible template, we fix it. In order to detail both routines, we define $V_G^< := \{v \in G : d_v < d_r\}$, and analogously, $V_G^=$ and $V_G^>$ for a graph G with fixed root r. If the considered graph G cannot be confused, we omit the subscript and write $V^<, V^=$, and $V^>$ instead. We may also refer to the three sets as blue, white, and red nodes, respectively.

Lemma 6. *Given an HCTCP instance (G, w, d, W_L, W_U) and a fixed root r, let*

$$w_{\min}^< := \frac{W_L}{2} - w(V^=), \qquad w_{\max}^< := \min\left(\frac{W_U}{2},\, w(V^>) + w(V^=)\right),$$
$$w_{\min}^> := \frac{W_L}{2} - w(V^=), \qquad w_{\max}^> := \min\left(\frac{W_U}{2},\, w(V^<) + w(V^=)\right).$$

Then, any graph $S \in \mathcal{T}_{L,U}$ for which r has the weighted median demand, satisfies

$$w(V_S^<) \in [w_{\min}^<,\, w_{\max}^<] \qquad \text{and} \qquad w(V_S^>) \in [w_{\min}^>,\, w_{\max}^>].$$

Proof. The bounds follow directly from the definition of the weighted median and the weight bounds on feasible templates. □

The preprocessing algorithm that exploits the median property is detailed in Algorithm 1. Our method to discard nodes checks for vertices whose inclusion violates the upper weight bound on $V^<$ or $V^>$. To this end, we employ Dijkstra's algorithm on the bidirected version of the current graph with special arc weights, adjusted for the blue and red color class, respectively. Note that we ignore the weight of all fixed nodes (cf. line 2). In return, we subtract the weight of fixed nodes of the color at hand from the upper weight bound (cf. line 5).

In order to fix a node v, we consider the connected component C of r in $G \setminus v$. If C violates a lower bound, this component cannot contain a feasible template and thus, we can fix node v to be part of any solution. While this concept can be generalized to arbitrary (node separating) sets S, the implication that at least one node from S has to be chosen is not as strong as fixing a single node. In our implementation we precompute the articulation points of G and consider the nodes in a depth-first search order. Therefore, if v is an articulation point we can efficiently deduce the color weights of the root component from the parent node. If, in addition, the root component of $G \setminus v$ is feasible, we do not have to consider any descendants of v in the DFS tree.

Since the weight bounds may change when we remove nodes, and the set of fixed nodes impacts the removal, it makes sense to repeat the procedures until a stable state is reached. Whenever the resulting graph does not meet a lower weight bound, or when we removed a fixed node, we can deduce that node r cannot have a (weighted) median demand in a feasible template (cf. line 18). Hence, we eliminate r from the set of potential roots. For any of the remaining root nodes we can add the valid inequalities from Lemma 6 to the respective MIP, i.e.,

$$w_{\min}^< \leq w(V^<) \leq w_{\max}^<, \qquad w_{\min}^> \leq w(V^>) \leq w_{\max}^>. \tag{15}$$

Algorithm 1: MedianInducedPreprocessing

```
Input: HCTCP instance $G$, $w$, $d$, $W_L$, $W_U$,
       potential root $r \in V(G)$
Output: relevant subgraph $G' \subseteq G$,
        set $V^f$ of fixed nodes
1  Procedure discard_nodes():
2      Consider bidirected version $D$ of $G$ with arc weights
           $w^<_{(u,v)} = \begin{cases} w_v & \text{, if } v \in V^< \setminus V^f \\ 0 & \text{, else} \end{cases}$ and $w^>_{(u,v)} = \begin{cases} w_v & \text{, if } v \in V^> \setminus V^f \\ 0 & \text{, else} \end{cases}$ ;
3      $\ell^< \leftarrow$ node distance labels to $r$ from single source Dijkstra on $(D, w^<)$ ;
4      $\ell^> \leftarrow$ node distance labels to $r$ from single source Dijkstra on $(D, w^>)$ ;
5      Remove nodes from $G$ with $w^<(v) > w^<_{\max} - w(V^f \cap V^<)$ or $w^>(v) > w^>_{\max} - w(V^f \cap V^>)$ ;
6  Procedure fix_nodes():
7      for $v \in G \setminus r$ do
8          $C \leftarrow$ connected component of $r$ in $G \setminus v$ ;
9          if $w(V^<_C) < w^<_{\min}$ or $w(V^>_C) < w^>_{\min}$ then
10             $V^f \leftarrow V^f \cup \{v\}$ ;
11 $V^f \leftarrow \{r\}$ ;
12 repeat
13     repeat
14         Determine $V^<$, $V^=$, $V^>$ and $w^<_{\min}$, $w^<_{\max}$, $w^>_{\min}$, $w^>_{\max}$ ;
15         discard_nodes() ;
16     until No nodes were removed;
17     if $w(V^<) < w^<_{\min}$ or $w(V^>) < w^>_{\min}$ or $V^f \not\subseteq V(G)$ then
18         return $(\varnothing, \varnothing)$ ;
19     fix_nodes() ;
20 until No nodes were fixed;
21 return $(G, V^f)$ ;
```

3.5 Local Search

Recall that the reduced costs of subgraph S are $\overline{p}(S) := p(S) - \pi(S)$. We present a local search method that aims at quickly identifying subgraphs of negative reduced costs by adding or removing single nodes from an already promising subgraph. As such promising subgraphs we use all graphs that were found in the current pricing optimization, since these already have negative reduced costs. In addition, we perform the local search on all graphs that were used in the current solution of the restricted master problem, i.e., trees T with $x_T > 0$ in the optimal RMP solution. Note that due to complementary slackness, these subgraphs have reduced costs of 0.

In order to find good modifications of a subgraph S, we have to recompute reduced costs of slightly changed graphs, and are therefore particularly interested in the change of penalty. More specifically, given a tree $T \in \mathcal{T}_{L,U}$ with penalty $p(T)$ and a node v, we need to efficiently determine $p(T + v)$. Recomputing the weighted median and recalculating the weighted sum of differences is inefficient. With more stored information and suitable data structures, however, one can efficiently compute $p(T + v)$ from $p(T)$. We refrain from the technical details of this method, and instead focus on a different approach. For a tree T with weighted median demand $\hat{d}$ and $v \notin T$, we define the potential of v as $\varphi(v) := w_v \, |d_v - \hat{d}|$, and note that

$$p(T+v) \;\leq\; p(T)+\varphi(v).$$

Thus, we overestimate the reduced costs as $\tilde{p}(T+v) := p(T)+\varphi(v)-\pi(T+v)$. The error of this estimation can be as large as $w(T)\,|d_v-\hat{d}|$, but shows to be small in practice if d_v is close to $\hat{d}$.

For the local search, we consider a tree T and two parameters k^+ and $k^- \in \mathbb{N}$. Now we iteratively determine the best k^+ neighbors of T w.r.t. φ. For any of the 2^{k^+} subsets S^+, we check if $T+S^+$ is feasible, and after checking with a prefix tree that the same set of nodes is not already part of the template pool, we include it. Furthermore, we determine the best k^- nodes in T that can be feasibly removed from $G[V(T)]$ (independently). Again, for every subset S^-, we include any feasible, new graph $T+S^+-S^-$ with negative reduced costs.

4 Computational Study

The goal of our computational study is threefold: We compare different formulations for connectivity in the context of the presented covering problem, and we measure the impact of three algorithmic enhancements. Finally, we compare the column generation approach to a compact districting model from [Seg+07] that was implemented in [SRC11].

The computations for the HCTCP show that most of the time is spent solving the pricing problem. Therefore, our methods aim to facilitate the pricing process. The optimization procedure with algorithmic enhancements is depicted in Figure 3. The setting and the test instances are specified in Section 4.1. The main results of our computational study are as follows:

Connectivity Formulations In Section 4.2 we find that among the four existing connectivity formulations, single-commodity flow, multi-commodity flow, rooted arc separators, and rooted node separators, the single-commodity flow offers the best overall performance. For most of the instances it performs best by far, and for the other instances it is close to the best performance. This is particularly interesting since the SCF formulation for connectivity seems to have been dismissed in the literature. The new coarse-to-fine single commodity flow formulation, introduced in Section 3.2, performs even better on the instance set, particularly on large instances.

Algorithmic Enhancements The impact of our proposed tuning methods is described in Section 4.3. In particular, we examine the impact of the primal pricing heuristic based on spanning trees (cf. Section 3.1),

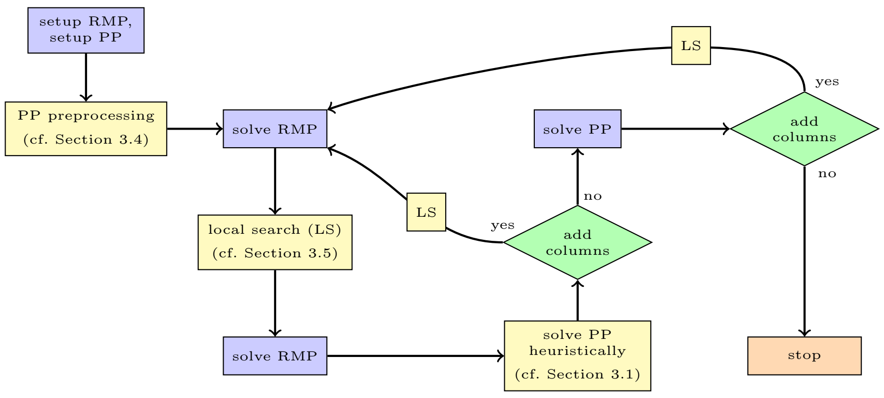

setup RMP, setup PP
PP preprocessing (cf. Section 3.4)
solve RMP
local search (LS) (cf. Section 3.5)
solve RMP
solve PP heuristically (cf. Section 3.1)
add columns
yes
LS
no
solve PP
add columns
yes
LS
no
stop

Figure 3: Optimization procedure with all algorithmic enhancements (in yellow) integrated.

the center elimination preprocessing routine described in Section 3.4, and the local search outlined in Section 3.5.

Our results show that each procedure is able to significantly reduce the computation time. The largest effect is due to the heuristic solution of the pricing problem. Moreover, the improvements prove to be essentially independent, leading to a multiplicative effect in the overall solution time.

Application to Districting We discuss in Section 4.4 how to adjust our column generation approach to classical districting problems, and integrate it into a branch-and-price algorithm. We compare our new algorithm to an exact and a heuristic model on 10 instances found in [SRC11]. While these instances have unusual features that make it hard to find any feasible solution, our computational results show that our approach is competitive.

4.1 Setting and Test Instances

We ran the experiments on machines equipped with Intel Xeon E3-1234 CPUs with 3.7GHz and 32GB RAM. Our code is written in Python 3.6 and we used Gurobi 9.1 as LP and IP solver. We introduce a time limit of 24 hours, and instances that terminate due to memory errors are set to the time limit. For all experiments in Sections 4.2 and 4.3, we use the following setting:

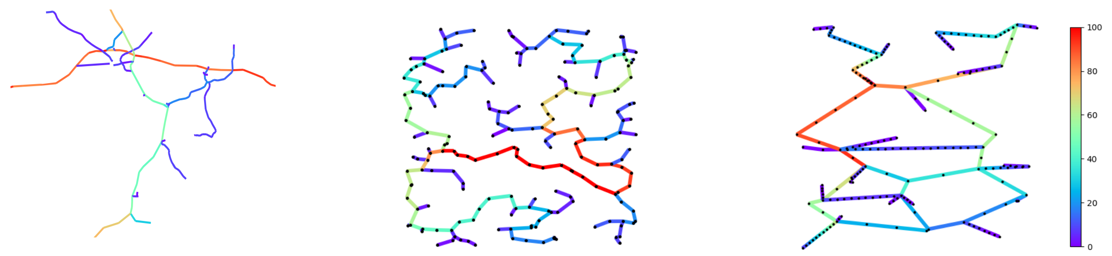
We apply all reduction techniques from Section 3.3 except for the minimum separators. We found that the articulation point neighborhood cuts (14) in combination with the restriction to the core graph are especially helpful for our instances. In contrast to Section 4.4 we do not run a branch and price algorithm, but only solve the root LP relaxation with column generation, and then solve the respective IP. Since we are concerned with a covering problem, we do not encounter any feasibility issues. In addition, the objective value of the root LP relaxation provides a lower bound for the optimal integer objective value. Considering the 1857 successful runs, the average optimality gap was 0.7%. 1366 of these runs show a gap of less than 1% and 876 runs finished with a provable optimal solution.

Regarding the test instances, we are originally concerned with network instances that constitute the German motorway network. These instances stem from a research project on optimal truck toll control with the German Federal Office for Goods Transport (cf. [BSS16; Bor+17]). In order to have more instances, that can also be made publicly available, we generate two types of random transit networks with artificial traffic volume that resemble the real-world instances: random trees, and random networks based on Voronoi diagrams. The code for our transit network generator can be found in a GitHub repository[1], and the instances and instance-wise computational results are available in an online supplement[2] to this article. Typical instances for the three types are displayed in Figure 4.

In accordance with the districting literature, the HCTCP is formulated as a covering problem of weighted vertices. The application, on the other hand, is focussed on the covering of weighted edges. Therefore, we consider the line graph of every single transit network and, for clarity, we label these instances with `_lg`.

German motorways The German motorway network is divided into 24 districts where local mobile control teams patrol the network on predefined control sections. Homogeneous traffic within each control section is an operative goal. Furthermore, the sections are subject to lower and upper length bounds. When considering the line graph, the HCTCP models the problem of optimally designing the control sections. The subnetworks are sparse and often tree-like. Denser networks can be found in the Ruhr area, but are still planar and have a maximum degree of 4. The instances are grouped with respect to the magnitude of the computation time for the single-commodity flow formulation without any heuristics into 4 groups: tiny, small, medium, and large.

[1]https://github.com/stephanschwartz/transit_network_generator
[2]https://github.com/stephanschwartz/vertex_covering_with_capacitated_trees



(a) German motorway, region around Hanover

(b) Random tree

(c) Voronoi network

Figure 4: Exemplary transit network instances for the three instance classes.

Table 1: Selected properties of considered instance sets: # denotes the number of instances in the respective group, *tw* the treewidth, b [%] the average fraction of edges that are bridges, and p [%] the fraction of instances that are planar. The values are associated with the actual instances, i.e., with the line graphs of the described transit networks.

	#	$\lvert V \rvert$	$\lvert E \rvert$	*tw*	b [%]	p [%]
`ger_lg_tiny`	7	117.9	138.1	3.0	53.4	100.0
`ger_lg_small`	7	162.3	207.0	3.6	33.6	57.1
`ger_lg_medium`	6	215.8	276.0	3.8	37.7	33.3
`ger_lg_large`	4	224.2	311.0	4.5	27.0	0.0
`tree_lg`	25	199.0	242.6	2.8	46.9	100.0
`voronoi_lg_medium`	33	204.9	255.6	4.4	30.8	12.1
`voronoi_lg_large`	17	206.2	265.5	4.8	23.5	0.0

Random trees To mimic smaller instances of the German motorways, we consider 25 random trees. We build a tree from random points in a rectangle by using Kruskal's algorithm on the complete graph with Euclidean edge weights. The assigned traffic is a combination of random origin-destination traffic and traffic obtained by a gravity model.

Voronoi networks To mimic a denser transit network, we create a Voronoi diagram and consider the graph that is spanned by the ridges between the Voronoi cells. This is our basic Voronoi graph with Euclidean edge lengths. Now, we add additional random leaves, split the edges to obtain a specified number of nodes, and stretch each edge by a random factor. The artificial traffic is generated as for the random trees. We consider 50 of these instances that are split into two groups based on the computation time of the SCF without any heuristics.

4.2 Connectivity Formulations

In order to compare the connectivity formulations, we implemented all described models from Section 3.1. The single-commodity flow (SCF) and multi-commodity flow (MCF) implementations are straight forward, and the coarse-to-fine flow (C2F) follows the presentation in Section 3.2.

For the rooted arc separators (RAS) we add all 2-cycle inequalities to the initial model. We tested different threshold values to start the separation routine and found that separating only for $y_v > 0.999$ led to the best performance. Also, the adding of nested cuts and generalized subtour elimination constraints

Table 2: Average computation times (in seconds) for different connectivity formulations. Usage of the pricing heuristic was disabled, the median preprocessing and the local search were enabled. Superscripts denote the number of timeouts.

Instance set	SCF	MCF	RAS	RNS	C2F
ger_lg_tiny	1.8	10.3	4.5	1.4	1.9
ger_lg_small	14.1	87.0	86.2	15.0	12.4
ger_lg_medium	77.4	684.2	407.3	116.8	54.0
ger_lg_large	1127.5	25525.2^{1}	23556.2^{1}	16211.6	760.4
tree_lg	20.8	374.2	174.0	14.7	27.1
voronoi_lg_medium	2353.8	11652.7	8775.6^{1}	29095.9^{5}	900.5
voronoi_lg_large	8694.9	25419.4	36263.4^{6}	71480.9^{10}	2006.9

during the separation proved to have a significant impact. We did not include back cuts in the separation, but tried to obtain minimum cuts of minimum cardinality by adding a small $\varepsilon > 0$ to every capacity before computing a maximum flow. However, this proved to have a negative effect on the running time.

Concerning the rooted node separators (RNS), the separation of integer solutions (cf. [Fis+17]) was better than the separation of fractional solutions (cf. [FF08; ÁMLM13b]). In addition, we use back cuts to generate more valid inequalities.

Since we want to study the performance of the connectivity formulations, we disable the pricing heuristic which would otherwise dominate the solution process. However, we enable the median induced preprocessing and add further templates via local search. Various tests indicate that this does not impact the validity of the results, while drastically reducing computation times.

Table 2 illustrates the performance of the different connectivity formulations associated with the HCTCP. Among the four existing formulations, the single-commodity flow performs best on the whole instance set. Concerning the ger_lg and voronoi_lg instances, the advantage over the other formulations is significant, especially for larger instances. On the tree_lg instances, the rooted node separators perform best, but the SCF formulation is still competitive, in contrast to MCF and RAS. On the voronoi_lg instances, the RAS and MCF formulations are better than the RNS, but all three fall clearly short of the SCF. The RNS formulations suffer the most timeouts (but no memory errors). All timeouts for the MCF and RAS, on the other hand, were due to memory errors. For almost all instances that were not suspended, the RAS is much faster than the MCF.

The newly introduced coarse-to-fine flow formulation performs even better than the SCF, especially on larger instances. We developed the C2F model with regard to the real-world ger_lg instances; the voronoi_lg instances were created later. It is all the more surprising that the impact on these instance sets is even stronger. For one voronoi_lg_large instance, depicted in Figure 5, the solution time drops from 18806 seconds (SCF) to 1888 seconds (C2F).

The coarse-to-fine approach is apparently particularly suited for hierarchical networks with a large fraction of degree 2 vertices. This structure is typical for transit networks such as motorway networks or public transport networks. Additional experiments suggest that the advantage of the C2F method rather depends on the graph than on the traffic demand. In experiments with random demand on each vertex, the C2F formulation was still far superior to the SCF.

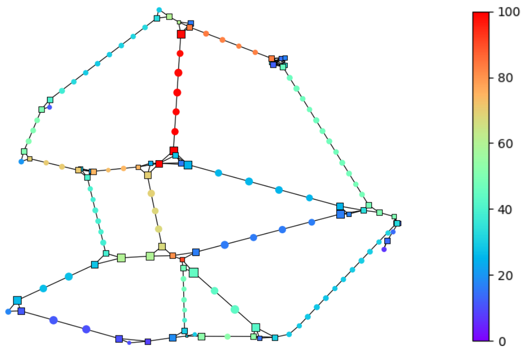


Figure 5: The core graph of a `voronoi_lg_large` instance. Squares represent coarse nodes, i.e., vertices of degree ≥ 3.

4.3 Algorithmic Enhancements

All of the applied techniques aim to reduce the time spent for the pricing optimization. In Table 3, we report the impact of the different tuning methods with the SCF and C2F formulation, while the analysis below is with respect to the SCF. However, we point out that the effects are similar for all connectivity variants.

Most remarkable is that the three improvements are essentially independent, i.e., the reduction effects multiply if combined. For instance, for the `ger_lg_large` instance set and the SCF formulation, all improvements lead to a speed-up factor of 24.6, while the individual speed-up factors are 7.88 (pricing heuristic), 1.59 (center elimination), and 2.27 (local search).

Pricing Heuristic The exact optimization takes most of the computation time. Therefore, we try to find templates of negative reduced costs heuristically, and only call the exact optimization routine if none are found. To this end, we solve the pricing problem as described in the last connectivity formulation in Section 3.1, i.e., we use LP formulation (11) on a predefined spanning tree.

This approach significantly drops the number of calls to the exact optimization routine. Averaging over all instances, we find a reduction by 85% compared to the version without the primal heuristic. In 17% of the `ger_lg` instances, in 100% of the `tree_lg` instances and in 22% of the `voronoi_lg` instances, the exact optimization was only called once to prove optimality, i.e., the primal heuristic already produced the optimal solution. Our comparison shows that the integration of the pricing heuristic has the largest impact of the three improvements.

Median Induced Preprocessing Recall that during the pricing problem, we solve a single-rooted and budgeted Maximum Weight Connected Subgraph problem for every potential root. Our median induced preprocessing routine, described in Algorithm 1, has three potential benefits: First, it can eliminate potential roots, which can be ignored in every pricing problem. Second, for the remaining roots, it can also shrink the relevant subgraph by discarding irrelevant nodes, leading to smaller pricing problems. And third, it can fix nodes to further simplify the pricing problems. Table 4 shows the effect of the preprocessing on these three aspects.

On average, our algorithm is able to exclude over 40% of all vertices as potential roots for the `ger_lg` and `tree_lg` instances, and over 20% for the `voronoi_lg` instances. While this alone leads to a significant reduction in computation time, the routine discards a considerable number of nodes in the relevant subgraph for the `ger_lg` and `tree_lg` instances, and is also able to fix a fair amount of the remaining nodes. For the `voronoi_lg` instances these effects are much weaker, and we can also observe in Table 3 that for these instances the preprocessing has a smaller effect.

Table 3: Average computation times (in seconds) for the single-commodity flow and coarse-to-fine flow formulation when including combinations of the pricing heuristic (H), median induced preprocessing (P), or the local search (L).

SCF

Instance set	---	--L	-P-	H--	-PL	H-L	HP-	HPL
ger_lg_tiny	5.2	3.2	2.6	3.4	1.8	2.3	1.9	1.5
ger_lg_small	50.4	25.9	23.9	21.1	14.1	11.8	10.6	7.4
ger_lg_medium	341.3	188.1	176.3	81.3	77.4	49.5	42.8	25.7
ger_lg_large	4642.4	2043.6	2923.6	588.8	1127.5	287.9	410.7	188.7
tree_lg	127.3	59.5	38.4	54.7	20.8	31.4	23.0	15.5
voronoi_lg_medium	5344.5	2930.6	4667.5	479.4	2353.8	313.5	394.4	235.7
voronoi_lg_large	18516.9	9956.4	17332.5	1403.5	8694.9	963.1	1140.5	701.7

C2F

Instance set	---	--L	-P-	H--	-PL	H-L	HP-	HPL
ger_lg_tiny	5.9	4.0	2.8	3.8	1.9	2.5	1.9	1.6
ger_lg_small	53.3	28.4	22.0	21.7	12.4	11.9	12.1	6.9
ger_lg_medium	212.3	132.2	117.4	95.4	54.0	41.8	36.3	22.5
ger_lg_large	3129.7	1248.3	1980.3	589.6	760.4	257.6	334.1	144.8
tree_lg	150.7	74.1	51.6	54.4	27.1	32.0	22.5	15.0
voronoi_lg_medium	2208.3	1082.8	1895.6	300.9	900.5	197.0	241.9	143.1
voronoi_lg_large	4712.1	2027.5	4418.6	609.0	2006.9	412.4	549.9	309.0

Table 4: Effects of the median induced preprocessing.

Instance set	roots eliminated [%]	discarded [%]	fixed [%]
ger_lg_tiny	44.2	38.2	24.1
ger_lg_small	44.0	36.5	15.5
ger_lg_medium	38.7	29.0	8.5
ger_lg_large	29.3	17.3	3.5
tree_lg	42.0	39.1	10.9
voronoi_lg_medium	21.4	4.8	1.9
voronoi_lg_large	19.1	3.2	1.3

The overall impact of this preprocessing routine is still remarkable. For the `tree_lg` instances it is even the single improvement with the strongest impact. Adding the valid inequalties (15), however, did not prove to have a substantial effect on the solution times.

Local Search The local search aims at generating promising template graphs in order to reduce the number of iterations in the column generation. For the model introduced in Section 3.5 we opt for $k^+ = 6$ and $k^- = 1$, even though, for large problems, the choice of a larger k^+, e.g., $k^+ = 10$, was even better. Furthermore, our computations show that it is most beneficial to apply the local search after solving the restricted master problem (for each subgraph used in the optimal solution) and after the pricing (for each subgraph with negative reduced costs). With this setup, the number of iterations drops by 50% on average over all considered instances.

4.4 Application to Districting

In classical districting, we search for an optimal partitioning of a graph into a fixed number of districts. Our model is designed for covering problems, and the number of districts is not fixed. Nevertheless, we can adapt the IP model (2) for the HCTCP to the former setting: The covering constraints (2b) become partitioning constraints by replacing the inequality sign with an equality sign. This leads to dual variables without sign restriction, but the sign of the dual variables is irrelevant in the further course. Furthermore, we add the additional constraint

$$\sum_{T \in \mathcal{T}_{L,U}} x_T = k$$

in order to fix the number of templates. This constraint changes the objective function of the pricing problem, but it remains a budgeted and rooted version of the MWCS. Therefore, the previously presented methods are still applicable.

The only publicly available districting instances seem to be the commercial territory design instances by Salazar-Aguilar, Ríos-Mercado, and Cabrera-Ríos [SRC11]. They consider the problem of partitioning a graph, representing a geographic area, into a fixed number of business districts with certain properties. The authors implement an exact and a heuristic model. The exact IP formulation, called Median-Based Territory Design Problem (MTDP), was already proposed but not implemented in [Seg+07]. The connectivity is ensured with a special set of node separator inequalities. As there are exponentially many of these inequalities, the MTDP is solved with branch-and-cut. The heuristic approach is a reformulation of the IP with quadratic terms. The resulting integer quadratic programming model, denoted by QMTDP, is solved heuristically using DICOPT.

We apply the adapted HCTCP to all large instances with 200 vertices from [SRC11]. Apart from the partitioning and the fixed number of templates, the main difference to our model is that the vertices have three weights (instead of one) with tight lower and upper bounds on the cumulative weight. The tight bounds and the partitioning make it much harder to find a feasible solution as a base for the restricted master problem. To circumvent this problem, we make use of an artificial base by constructing an arbitrary partitioning of the vertices, which may lead to templates that do not meet the requirements of a district. However, we assign prohibitively large penalties to these templates, so that none of them is part of an optimal solution. A similar method is used in [CGP19].

Since the integer version of the root LP has infeasibility issues, we implemented a branch-and-price algorithm. The branching is done on the edges such that both endpoints are forced to be either in the same or in different districts. After solving the LP relaxation of a branching node, we also solve the IP version of the restricted master problem with all previously generated districts. If we obtain an integer solution without templates from the artificial base we terminate.

Table 5: Comparison of the exact (MTDP) and heuristic (QTDP) solutions from [SRC11] to the first feasible solution of the presented branch-and-price approach on instances from [SRC11].

Inst	Objective value			Gap [%]		Time [s]		
	MTDP	QMTDP	B&P	QMTDP	B&P	MTDP	QMTDP	B&P
1	10422.0	11523	10815.4	10.56	3.77	1116	28	185
2	10646.1	11425	11188.0	7.32	5.09	7200	966	1586
3	10846.8	11443	11200.6	5.50	3.26	1468	7200	4070
4	11122.0	11443	11747.9	2.89	5.63	7200	3618	2068
5	10878.1	11097	11492.8	2.01	5.65	7200	1193	678
6	10499.3	10746	10955.4	2.35	4.34	7200	1871	1981
7	11061.0	11686	11333.5	5.65	2.46	7200	1088	285
8	10659.5	11205	11015.7	5.12	3.34	2641	592	925
9	11470.3	11648	11792.6	1.55	2.81	7200	1263	1290
10	11043.8	11780	11158.7	6.67	1.04	1211	2349	896

The results of our comparison are shown in Table 5. We did not reimplement the MTDP and QMTDP, but take the values from [SRC11]. Even though our approach is not designed for these problem types or instances, we can see that our adaption of the HCTCP can compete with the existing models. In particular, our B&P is the only approach that can solve all instances within the time limit (which was specified in [SRC11]). With more running time, the branch-and-price approach can further narrow the optimality gap. However, the benefit with the current implementation is questionable. The development of specific heuristics to generate feasible templates would certainly lead to improvements, and is a potential topic for further research.

5 Conclusions

In this paper, we studied a vertex covering problem with connected subgraphs that are subject to upper and lower weight bounds. The objective of the covering is to use subgraphs that are as homogeneous as possible, i.e., their vertices should be similar w.r.t. a given vertex weight.

We formulated the problem as an IP that is well suited for a column generation approach. The main challenge is the pricing problem which turned out to be a variant of the Maximum Weight Connected Subgraph Problem. We compared different connectivity formulations in terms of MIP, and found that the single-commodity formulation offers the best overall performance. In addition, we proposed a new flow formulation that is based on the coarse-to-fine paradigm, and that works particularly well for transit networks (where many vertices have a degree of 2). We proved that the new formulation has a tighter LP relaxation than the single-commodity flow. Our computational experiments on real-world and synthetic transit networks also confirmed the practical advantage of the new model.

Complementary to the study of connectivity formulations, we proposed a number of preprocessing techniques and a local search to facilitate the pricing problem. A newly introduced family of cuts, the Articulation Point Neighborhood Cuts, proved to be very effective. At the core of the preprocessing is a method that exploits the measuring of homogeneity in terms of a median vertex weight. The proposed algorithm proved infeasibility for a significant number of variants, while simplifying most of the remaining problems. It can also reliably solve the districting problems of [SRC11]; these are distinct in our test set in the sense that they have a partitioning flavor. This provides evidence that our methods carry over to other graph theoretic problems with connectivity restrictions.

References

[Ami+02] Steven J d'Amico, Shoou-Jiun Wang, Rajan Batta, and Christopher M Rump. "A simulated annealing approach to police district design". In: *Computers & Operations Research* 29.6 (2002), pp. 667–684.

[Ane80] Yash P Aneja. "An integer linear programming approach to the Steiner problem in graphs". In: *Networks* 10.2 (1980), pp. 167–178.

[BKN15] Mohamed Didi Biha, Hervé LM Kerivin, and Peh H Ng. "Polyhedral study of the connected subgraph problem". In: *Discrete Mathematics* 338.1 (2015), pp. 80–92.

[BSS16] Ralf Borndörfer, Guillaume Sagnol, and Stephan Schwartz. "An extended network interdiction problem for optimal toll control". In: *Electronic Notes in Discrete Mathematics* 52 (2016), pp. 301–308.

[Bac+12] Christina Backes, Alexander Rurainski, Gunnar W Klau, Oliver Müller, Daniel Stöckel, Andreas Gerasch, Jan Küntzer, Daniela Maisel, Nicole Ludwig, Matthias Hein, Andreas Keller, Helmut Burtscher, Michael Kaufmann, Eckart Meese, and Hans-Peter Lenhof. "An integer linear programming approach for finding deregulated subgraphs in regulatory networks". In: *Nucleic acids research* 40.6 (2012), e43.

[Bea84] John E Beasley. "An algorithm for the Steiner problem in graphs". In: *Networks* 14.1 (1984), pp. 147–159.

[Bho+19] Sujoy Bhore, Sourav Chakraborty, Satyabrata Jana, Joseph SB Mitchell, Supantha Pandit, and Sasanka Roy. "The balanced connected subgraph problem". In: *Conference on Algorithms and Discrete Applied Mathematics.* Springer. 2019, pp. 201–215.

[Bor+17] Ralf Borndörfer, Guillaume Sagnol, Thomas Schlechte, and Elmar Swarat. "Optimal duty rostering for toll enforcement inspectors". In: *Annals of Operations Research* 252.2 (2017), pp. 383–406.

[Bul+16] Aydın Buluç, Henning Meyerhenke, Ilya Safro, Peter Sanders, and Christian Schulz. "Recent advances in graph partitioning". In: *Algorithm engineering* (2016), pp. 117–158.

[CCL06] Alysson Costa, Jean-François Cordeau, and Gilbert Laporte. "Steiner tree problems with profits". In: *INFOR: information systems and operational research* 44.2 (2006), pp. 99–115.

[CCY19] Huanfa Chen, Tao Cheng, and Xinyue Ye. "Designing efficient and balanced police patrol districts on an urban street network". In: *International Journal of Geographical Information Science* 33.2 (2019), pp. 269–290.

[CG12] Chao-Yeh Chen and Kristen Grauman. "Efficient activity detection with max-subgraph search". In: *2012 IEEE Conference on Computer Vision and Pattern Recognition.* IEEE. 2012, pp. 1274–1281.

[CG95] Boris V Cherkassky and Andrew V Goldberg. "On implementing push-relabel method for the maximum flow problem". In: *International Conference on Integer Programming and Combinatorial Optimization.* Springer. 1995, pp. 157–171.

[CGP19] François Clautiaux, Jeremy Guillot, and Pierre Pesneau. "Exact approaches for solving a covering problem with capacitated subtrees". In: *Computers & Operations Research* 105 (2019), pp. 85–101.

[CGR92] Sunil Chopra, Edgar R Gorres, and MR Rao. "Solving the Steiner tree problem on a graph using branch and cut". In: *ORSA Journal on Computing* 4.3 (1992), pp. 320–335.

[CHQ10] Kevin M Curtin, Karen Hayslett-McCall, and Fang Qiu. “Determining optimal police patrol areas with maximal covering and backup covering location models”. In: *Networks and Spatial Economics* 10.1 (2010), pp. 125–145.

[CR94] Sunil Chopra and Mendu Rammohan Rao. “The Steiner tree problem I: Formulations, compositions and extension of facets”. In: *Mathematical Programming* 64.1-3 (1994), pp. 209–229.

[CS97] Geon Cho and Dong X Shaw. “A depth-first dynamic programming algorithm for the tree knapsack problem”. In: *INFORMS Journal on Computing* 9.4 (1997), pp. 431–438.

[Car+13] Rodolfo Carvajal, Miguel Constantino, Marcos Goycoolea, Juan Pablo Vielma, and Andrés Weintraub. “Imposing connectivity constraints in forest planning models”. In: *Operations Research* 61.4 (2013), pp. 824–836.

[Con+07] Jon Conrad, Carla P Gomes, Willem-Jan Van Hoeve, Ashish Sabharwal, and Jordan Suter. “Connections in networks: Hardness of feasibility versus optimality”. In: *International Conference on Integration of Artificial Intelligence (AI) and Operations Research (OR) Techniques in Constraint Programming.* Springer. 2007, pp. 16–28.

[DG10] Bistra Dilkina and Carla P Gomes. “Solving connected subgraph problems in wildlife conservation”. In: *International Conference on Integration of Artificial Intelligence (AI) and Operations Research (OR) Techniques in Constraint Programming.* Springer. 2010, pp. 102–116.

[Dit+08] Marcus T Dittrich, Gunnar W Klau, Andreas Rosenwald, Thomas Dandekar, and Tobias Müller. “Identifying functional modules in protein–protein interaction networks: an integrated exact approach”. In: *Bioinformatics* 24.13 (2008), pp. i223–i231.

[Duh+08] Christophe Duhamel, Luis Gouveia, Pedro Moura, and Mauricio Souza. “Models and heuristics for a minimum arborescence problem”. In: *Networks* 51.1 (2008), pp. 34–47.

[EKK14] Mohammed El-Kebir and Gunnar W Klau. *Solving the maximum-weight connected subgraph problem to optimality.* 11th DIMACS Implementation Challenge Workshop. 2014.

[FF08] Armin Fügenschuh and Marzena Fügenschuh. “Integer linear programming models for topology optimization in sheet metal design”. In: *Mathematical Methods of Operations Research* 68.2 (2008), pp. 313–331.

[FG90] Jacques A Ferland and Gilles Guénette. “Decision support system for the school districting problem”. In: *Operations Research* 38.1 (1990), pp. 15–21.

[FP88] Bernhard Fleischmann and Jannis N Paraschis. “Solving a large scale districting problem: a case report”. In: *Computers & Operations Research* 15.6 (1988), pp. 521–533.

[Fis+17] Matteo Fischetti, Markus Leitner, Ivana Ljubić, Martin Luipersbeck, Michele Monaci, Max Resch, Domenico Salvagnin, and Markus Sinnl. “Thinning out Steiner trees: a node-based model for uniform edge costs”. In: *Mathematical Programming Computation* 9.2 (2017), pp. 203–229.

[GG81] Bezalel Gavish and Stephen Graves. “Scheduling and routing in transportation and distribution systems: formulations and new relaxations”. In: (1981).

[GM93] Michel X Goemans and Young-Soo Myung. “A catalog of Steiner tree formulations”. In: *Networks* 23.1 (1993), pp. 19–28.

[GN70] Robert S Garfinkel and George L Nemhauser. “Optimal political districting by implicit enumeration techniques”. In: *Management Science* 16.8 (1970), B–495.

[GVHS08] Carla P Gomes, Willem-Jan Van Hoeve, and Ashish Sabharwal. “Connections in networks: A hybrid approach”. In: *International Conference on Integration of Artificial Intelligence (AI) and Operations Research (OR) Techniques in Constraint Programming.* Springer. 2008, pp. 303–307.

[Goe94a] Michel X Goemans. “Arborescence polytopes for series-parallel graphs”. In: *Discrete Applied Mathematics* 51.3 (1994), pp. 277–289.

[Goe94b] Michel X Goemans. “The Steiner tree polytope and related polyhedra”. In: *Mathematical programming* 63.1-3 (1994), pp. 157–182.

[HJ97] Pierre Hansen and Brigitte Jaumard. “Cluster analysis and mathematical programming”. In: *Mathematical programming* 79.1-3 (1997), pp. 191–215.

[HO92] Jianxiu Hao and James B. Orlin. “A Faster Algorithm for Finding the Minimum Cut in a Graph”. In: *Proceedings of the Third Annual ACM-SIAM Symposium on Discrete Algorithms.* SODA ’92. Society for Industrial and Applied Mathematics, 1992, 165–174.

[HP94] Dorit S Hochbaum and Anu Pathria. “Node-optimal connected k-subgraphs”. In: *manuscript, UC Berkeley* (1994).

[HS71] Sidney W Hess and Stuart A Samuels. “Experiences with a sales districting model: criteria and implementation”. In: *Management Science* 18.4-part-ii (1971), P–41.

[HT73] John Hopcroft and Robert Tarjan. “Efficient Algorithms for Graph Manipulation”. In: *Communications of the ACM* 16.6 (1973), pp. 372–378.

[Hes+65] Sidney Wayne Hess, JB Weaver, HJ Siegfeldt, JN Whelan, and PA Zitlau. “Nonpartisan political redistricting by computer”. In: *Operations Research* 13.6 (1965), pp. 998–1006.

[Ide+02] Trey Ideker, Owen Ozier, Benno Schwikowski, and Andrew F Siegel. “Discovering regulatory and signalling circuits in molecular interaction networks”. In: *Bioinformatics* 18.suppl_1 (2002), S233–S240.

[KLS91] Bernhard Korte, László Lovász, and Rainer Schrader. *Greedoids.* Vol. 4. Algorithms and Combinatorics. Springer-Verlag, 1991.

[KLT15] Tung-Wei Kuo, Kate Ching-Ju Lin, and Ming-Jer Tsai. “Maximizing submodular set function with connectivity constraint: Theory and application to networks”. In: *IEEE/ACM Transactions on Networking* 23.2 (2015), pp. 533–546.

[KM98] Thorsten Koch and Alexander Martin. “Solving Steiner tree problems in graphs to optimality”. In: *Networks: An International Journal* 32.3 (1998), pp. 207–232.

[KPH93] Bassam N Khoury, Panos M Pardalos, and Donald W Hearn. “Equivalent Formulations for the Steiner Problem in Graphs”. In: *Network Optimization Problems: Algorithms, Applications And Complexity.* World Scientific, 1993, pp. 111–123.

[KR19] Jörg Kalcsics and Roger Z Ríos-Mercado. “Districting problems”. In: *Location science.* Springer, 2019, pp. 705–743.

[KZ14] Hervé Kerivin and Jinhua Zhao. “Polyhedral study for the maximum bounded r-tree problem”. In: *2014 International Conference on Control, Decision and Information Technologies (CoDIT).* IEEE. 2014, pp. 140–145.

[LD96] Heungsoon Felix Lee and Daniel R Dooly. “Algorithms for the constrained maximum-weight connected graph problem”. In: *Naval Research Logistics (NRL)* 43.7 (1996), pp. 985–1008.

[LD98] Heungsoon Felix Lee and Daniel R Dooly. “Decomposition algorithms for the maximum-weight connected graph problem”. In: *Naval Research Logistics (NRL)* 45.8 (1998), pp. 817–837.

[Lei+18] Markus Leitner, Ivana Ljubić, Martin Luipersbeck, and Markus Sinnl. “A dual ascent-based branch-and-bound framework for the prize-collecting steiner tree and related problems”. In: *INFORMS Journal on Computing* 30.2 (2018), pp. 402–420.

[Lju+06] Ivana Ljubić, René Weiskircher, Ulrich Pferschy, Gunnar W Klau, Petra Mutzel, and Matteo Fischetti. “An algorithmic framework for the exact solution of the prize-collecting Steiner tree problem”. In: *Mathematical programming* 105.2-3 (2006), pp. 427–449.

[Lju20] Ivana Ljubić. “Solving Steiner trees: Recent advances, challenges, and perspectives”. In: *Networks* (2020).

[Lüt18] Hendrik Lüthen. “Partitioning into Isomorphic or Connected Subgraphs”. PhD thesis. Technische Universität, 2018.

[MJN98] Anuj Mehrotra, Ellis L Johnson, and George L Nemhauser. “An optimization based heuristic for political districting”. In: *Management Science* 44.8 (1998), pp. 1100–1114.

[MLT95] Young-Soo Myung, Chang-Ho Lee, and Dong-Wan Tcha. “On the generalized minimum spanning tree problem”. In: *Networks* 26.4 (1995), pp. 231–241.

[MR05] Thomas L Magnanti and S Raghavan. “Strong formulations for network design problems with connectivity requirements”. In: *Networks* 45.2 (2005), pp. 61–79.

[MW95] Thomas L Magnanti and Laurence A Wolsey. “Optimal trees”. In: *Handbooks in operations research and management science* 7 (1995), pp. 503–615.

[Mac87] Nelson Maculan. “The Steiner problem in graphs”. In: *North-Holland Mathematics Studies.* Vol. 132. Elsevier, 1987, pp. 185–211.

[Nyg88] Bjørn Nygreen. “European assembly constituencies for wales-comparing of methods for solving a political districting problem”. In: *Mathematical Programming* 42.1-3 (1988), pp. 159–169.

[PD01] Tobias Polzin and Siavash Vahdati Daneshmand. “A comparison of Steiner tree relaxations”. In: *Discrete Applied Mathematics* 112.1-3 (2001), pp. 241–261.

[Pop09] PC Pop. “A survey of different integer programming formulations of the generalized minimum spanning tree problem”. In: *Carpathian Journal of Mathematics* (2009), pp. 104–118.

[Pop20] Petrică C Pop. “The generalized minimum spanning tree problem: An overview of formulations, solution procedures and latest advances”. In: *European Journal of Operational Research* 283.1 (2020), pp. 1–15.

[RFK20] Daniel Rehfeldt, Henriette Franz, and Thorsten Koch. *Optimal Connected Subgraphs: Formulations and Algorithms.* Tech. rep. 20–23. Zuse Institute Berlin, 2020.

[RK19] Daniel Rehfeldt and Thorsten Koch. “Combining NP-hard reduction techniques and strong heuristics in an exact algorithm for the maximum-weight connected subgraph problem”. In: *SIAM Journal on Optimization* 29.1 (2019), pp. 369–398.

[RKM19] Daniel Rehfeldt, Thorsten Koch, and Stephen J Maher. “Reduction techniques for the prize collecting Steiner tree problem and the maximum-weight connected subgraph problem”. In: *Networks* 73.2 (2019), pp. 206–233.

[RSS13] Federica Ricca, Andrea Scozzari, and Bruno Simeone. “Political districting: from classical models to recent approaches”. In: *Annals of Operations Research* 204.1 (2013), pp. 271–299.

[SC98] Dong X Shaw and Geon Cho. “The critical-item, upper bounds, and a branch-and-bound algorithm for the tree knapsack problem”. In: *Networks: An International Journal* 31.4 (1998), pp. 205–216.

[SRC11] María Angélica Salazar-Aguilar, Roger Z Ríos-Mercado, and Mauricio Cabrera-Ríos. "New models for commercial territory design". In: *Networks and Spatial Economics* 11.3 (2011), pp. 487–507.

[Sch20] Stephan Schwartz. *An Overview of Graph Covering and Partitioning.* Tech. rep. 20–24. Zuse Institute Berlin, 2020.

[Seg+07] JA Segura-Ramiro, Roger Z Ríos-Mercado, Ada M Álvarez-Socarrás, and Karim de Alba Romenus. "A location-allocation heuristic for a territory design problem in a beverage distribution firm". In: *Proceedings of the 12th Annual International Conference on Industrial Engineering Theory, Applications, and Practice (IJIE).* 2007, pp. 428–434.

[Shi05] Takeshi Shirabe. "A model of contiguity for spatial unit allocation". In: *Geographical Analysis* 37.1 (2005), pp. 2–16.

[VZ00] Yehuda Vardi and Cun-Hui Zhang. "The multivariate L1-median and associated data depth". In: *Proceedings of the National Academy of Sciences* 97.4 (2000), pp. 1423–1426.

[WBB17] Yiming Wang, Austin Buchanan, and Sergiy Butenko. "On imposing connectivity constraints in integer programs". In: *Mathematical Programming* 166.1-2 (2017), pp. 241–271.

[Won84] Richard Wong. "Dual Ascent Approach for Steiner Tree Problems on a Directed Graph". In: *Mathematical Programming* 28 (Oct. 1984), pp. 271–287.

[Yam+09] Takanori Yamamoto, Hideo Bannai, Masao Nagasaki, and Satoru Miyano. "Better decomposition heuristics for the maximum-weight connected graph problem using betweenness centrality". In: *International Conference on Discovery Science.* Springer. 2009, pp. 465–472.

[ZS05] Andris A Zoltners and Prabhakant Sinha. "Sales territory design: Thirty years of modeling and implementation". In: *Marketing Science* 24.3 (2005), pp. 313–331.

[ÁMLM13a] Eduardo Álvarez-Miranda, Ivana Ljubić, and Petra Mutzel. "The maximum weight connected subgraph problem". In: *Facets of Combinatorial Optimization.* Springer, 2013, pp. 245–270.

[ÁMLM13b] Eduardo Álvarez-Miranda, Ivana Ljubić, and Petra Mutzel. "The rooted maximum node-weight connected subgraph problem". In: *International Conference on AI and OR Techniques in Constriant Programming for Combinatorial Optimization Problems.* Springer. 2013, pp. 300–315.

[ÁMS17] Eduardo Álvarez-Miranda and Markus Sinnl. "A Relax-and-Cut framework for large-scale maximum weight connected subgraph problems". In: *Computers & Operations Research* 87 (2017), pp. 63–82.