Optimal Steiner Trees Under Node and Edge Privacy Conflicts

Alessandro Hill · Roberto Baldacci · Stefan Voß

Abstract In this work, we suggest concepts and solution methodologies for a series of strategic network design problems that find application in highly data-sensitive industries, such as, for instance, the high-tech, governmental, or military sector. Our focus is on the installation of widely used cost-efficient tree-structured communication infrastructure. As base model we use the well-known Steiner tree problem, in which we are given terminal nodes, optional Steiner nodes, and potential network links between nodes. Its objective is to connect all terminals to a distributor node using a tree of minimum total edge costs. The novel, practically relevant side constraints are related to privacy concerns of customers, represented by terminals. In order to account for these, we study four privacy models that restrict the eligible infrastructure for the customer-distributor data exchange: (I) Selected pairs of terminals mutually exclude themselves as intermediate data-transmission nodes; (II) some pairs of terminals require disjoint paths to the distributor; (III) individual terminals forbid routing their data through allegedly untrustworthy links; and (IV) certain terminals do not allow the usage of doubtful links on their entire network branch. These topological data-privacy requirements significantly complicate

Alessandro Hill
Industrial and Manufacturing Engineering
California Polytechnic State University
San Luis Obispo, USA
E-mail: ahill29@calpoly.edu

Roberto Baldacci
Department of Electrical, Electronic, and Information Engineering "Guglielmo Marconi"
University of Bologna
Cesena, Italy
E-mail: r.baldacci@unibo.it

Stefan Voß
Institute of Information Systems (IWI)
University of Hamburg
Hamburg, Germany
E-mail: stefan.voss@uni-hamburg.de

the notoriously hard optimization problem. We clarify the model relationships by establishing dominance results, point out potential extensions and derive reduction tests. We present corresponding, strong non-compact integer programming (IP) formulations and embed these in efficient cutting plane methods. In addition, we develop constraint programming formulations that are used complementally to derive primal solutions. In a computational study, we analyze the performance of our methods on a diverse set of literature-based test instances.

Keywords Steiner tree problem · telecommunication · strategic network design · integer programming · constraint programming

1 Introduction

In this work, we address the case of centralized networks with minimal connectivity requirements. That is, we plan tree topologies in which exactly one path exists between two customer nodes in the network. These structures are commonly modeled using rooted Steiner trees in which a set of given customers, or terminals, is sought to be connected to a central distributor. Intermediate network nodes, or Steiner nodes, can be used in order to minimize the overall network installation cost and represent an essential mean at the modeling stage.

Very little work can be found in the literature in order to incorporate data privacy concerns into the strategic network design phase. In this section, we will describe and motivate our ideas, provide references to related work, and summarize the contribution of this paper.

1.1 Motivation

In strategic design of telecommunication infrastructure, the protection of data privacy is an important concern since the physical network structure is crucial in order to prevent from unwanted information leaks. A common requirement of information senders, or receivers, is that specific third-party network participants must not be able to tap the network path used to route privacy-sensitive information packages. We assume the case that data depersonalization, encryption technologies, and protocol changes are not sufficient in order to prevent from attacks such as for instance eavesdropping, traffic analysis, and camouflage marketing. Applications can be found, for example, when connecting high-tech industries, military facilities, or government agencies to a backbone network using fiber-optic cable structures.

We consider four topological concepts that allow the integration of a priori specification of privacy conflicts into the strategic network design model. In all cases we assume that the distributor is considered neutral and serves either as transmission source, transmission destination, or both. In a first *path-critical* model, we embed the customer requirement that no conflicting customer is allowed to be physically situated on the unique path connecting it to the distributor. Furthermore, we suggest a *branch-critical* model in which the two paths connecting two customers that are in a privacy conflict to the distributor node have to be edge disjoint. This paper extends these concepts from [9] by introducing two novel models that generalize the proposed models. We study the corresponding cases of terminal-edge conflicts which allow a more detailed specification of privacy concerns. For a conflict between a customer and an edge, we formulate an *edge-path-critical* model and an *edge-branch-critical* model in which the conflicting edge must not be part of the customer's data routing path or branch, respectively. These extensions are especially relevant because properties of a potential link such as proximity to competitors, physical condition, or accessibility may constitute an unacceptable risk for a customer's data flow. Hence, alternative network topologies that avoid these conflicting configurations are required in order to satisfy the customers.

To further motivate our concepts, we suggest their application in the context of social networks. To illustrate this, consider a social network between individuals or communities. Let us face the task of strengthening the (indirect) relationship between certain target network nodes. Then two nodes in the base network are connected by an edge if the relationship between the corresponding individuals can be potentially strengthened. Edge weights are given by a cost function that reflects the required effort. Obviously, a Steiner tree connecting the target nodes can be used to derive a possible strategy for interconnection. Conflicts can be useful to model incompatibility between intermediate individuals and target individuals. In other words, two target individuals are better connected if through paths that contain common "friends" (or, at least potential friends) only. Furthermore, we would like to point out that social network data may be a useful source for deriving conflict graphs. For instance, knowing a friendship between two network participants typically indicates that there might not be a major conflict and, therewith, reduces the number of potential conflict edges that need to be investigated.

1.2 Related Work

The Steiner tree problem (STP) without privacy concerns is known to be a challenging problem in combinatorial optimization itself and belongs to the class of NP-hard optimization problems in combinatorial optimization [14]. Many STP variants have been studied in the literature (see, e.g., [23,12,17,6]). In [5], the authors define conflicts between nodes such that conflicting nodes must not be both implemented in a solution network. So-called forbidden transitions are studied in [13]. They impose that selected pairs of incident edges must not both be used in a solution network. It is proven that even the minimum spanning tree problem is NP-hard under these side constraints. The most efficient exact approaches for the STP are based on branch-and-cut [8] and have already been studied in [3] and improved regarding their computational performance in [15]. Mathematical programming formulations for the

STP are given in [4] and [18]. More recently, path-extended formulations that yield stronger theoretical bounds are suggested [7]. Moreover, [19] suggest to solve exponentially many (w.r.t. the number of terminals) linear programs, based on laminar flows. Even though not being computationally competitive, the latter has the (theoretical) advantage of being polynomial for a fixed number of Steiner nodes. An overview on existing heuristic approaches is given in [22].

1.3 Contribution

In this work, we introduce new optimization problems that can be used to optimally plan under the described privacy-sensitive circumstances. We devise integer programming formulations and corresponding exact algorithms for the new models. Our methods are build upon strong cut-set formulations. Additionally, we present constraint programming formulations in order to evaluate the performance of state-of-the-art constraint programming solvers. Our detailed computational analysis shows the efficiency of our algorithmic techniques. To this end, we conduct experiments on diverse classes of test instances covering a broad range of privacy conflict densities for all models. Our main contributions are as follows:

1. Motivation and definition of two novel privacy-oriented Steiner tree problems; model relationships, preprocessing techniques and extensions.
2. Development of exact cutting plane algorithms based on strong integer programming formulations.
3. Elaboration of multi-commodity flow-based constraint programming formulations; paired with heuristic warm-start techniques.
4. Computational formulation, method and result analysis on a diverse set of instances derived from the literature.

All four optimization models are introduced in Section 2 and integer programming formulations are given in Section 3. The constraint programming approaches are presented in Section 4. Section 5 describes our extensive computational analysis. Model extensions are proposed in Section 6 before we conclude our work in Section 7.

2 Optimization Models

Before defining the privacy-oriented models, we recall the base Steiner tree problem (STP). In the STP, we are given a set of terminal (or customer) nodes T, a set of Steiner (or optional) nodes W, and a root (or distributor) node r. We denote the set of all nodes as V and assume that $V = T \mathbin{\dot{\cup}} W \mathbin{\dot{\cup}} \{r\}$ and $V' = T \mathbin{\dot{\cup}} W$. Additionally, an edge set $E \subseteq 2^V$ describes the possible links. Each edge $e \in E$ has an edge cost $c_e \geq 0$. A solution for the STP is a tree network B such that $T \subset V[B]$, $r \in V[B]$ and $E[B] \subseteq E$. The STP asks for a solution that minimizes the overall network cost $\sum_{e\in E[B]} c_e$. Note that we consider the rooted version of the STP, motivated by the application. In the following, we describe the concepts and the corresponding optimization models that generalize the STP. While two models have been introduced in [9], we introduce two new edge-conflict models which turn out to generalize the existing optimization problems. The following notation is used. For any $S \subseteq V'$, let $\delta^+(S)$ (respectively, $\delta^-(S)$) denote the set of arcs (i,j) with $i \in S$, $j \in V' \setminus S$ (respectively, with $i \in V' \setminus S$, $j \in S$). We use $\delta^+(i)$ (resp. $\delta^-(i)$) instead of $\delta^+(\{i\})$ (resp. $\delta^-(\{i\})$), $\forall i \in V'$. Moreover, we define $\delta(i)$ as the set of edges in E incident with node $i \in V$.

2.1 Path-Privacy Conflicts

Two customers may want to obviate that neither of their information is routed through the other customer's node in a solution network B. Let $C_P \subseteq 2^T$ be the set of these *path-privacy conflicts* between two terminals. Then we define the STP+CP as the corresponding STP with the additional side constraints that for each $\{i,j\} \in C_P$, both, a path exists between i and r in $B \setminus j$, and a path exists between j and r in $B \setminus i$. In Figure 1 (left), an optimal solution for an STP instance with 8 customers, 4 Steiner nodes and $E = 2^V$ is depicted, along with an optimal solution for a corresponding STP+CP instance with $C_P = \{\{1,2\},\{3,4\},\{6,8\}\}$ (right).

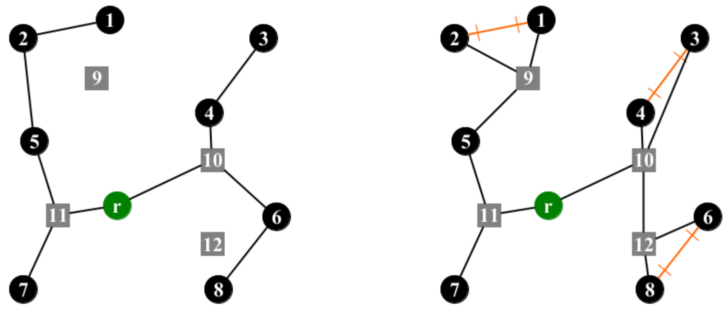

Fig. 1 Optimal networks for the STP and the STP+CP with eight terminals, four Steiner nodes and three privacy conflicts.

2.2 Path-Edge-Privacy Conflicts

A customer may not only fear the loss of sensitive information when using a non-trustworthy intermediate node for data transmission. A significant risk could already stem from the physical course of the link or the technology used. Assume that we are given such *path-edge-privacy conflicts* in form of a set $C_{PE} \subseteq (T \times E)$. Then we define the STP with path-edge-privacy conflicts (STP+CPE) as the corresponding STP with the additional requirement that for each $(i, e) \in C_{PE}$, the path P connecting i and r in a solution does not contain the edge e; i.e., $e \notin E[P]$. Figure 2 (left) illustrates an optimal solution for an STP+CPE instance with $C_{PE} = \{(2, \{2, 5\}), (1, \{2, 9\}), (7, \{11, r\}), (6, \{10, r\})\}$. Note that a terminal may be in conflict with an edge between non-terminals.

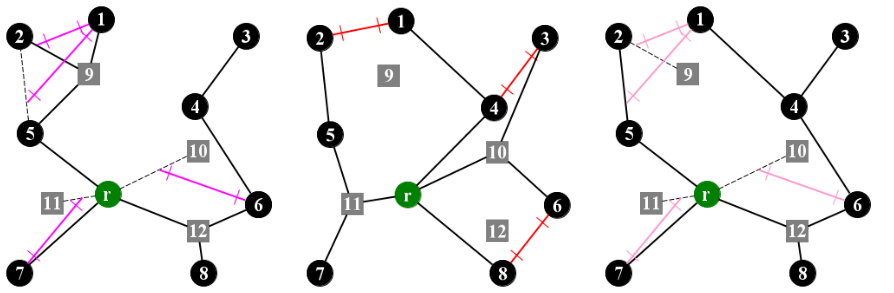

Fig. 2 Optimal networks for instances of the STP+CPE, the STP+CB and the STP+CBE with eight terminals and four Steiner nodes.

Furthermore, in contrast to path-privacy conflicts, edge-privacy conflicts may cause infeasibility for an STP+CPE instance. To see this, consider the case that $(i, \{i, j\}) \in C_{PE}\ \forall j \in \delta(i)$ for a terminal node $i \in T$. Since i must not be connected to r using any incident edge, no solution can be found for the STP. However, the feasibility problem can be solved efficiently by sequentially checking potential r-connectivity for each terminal i in the graph $(V, E \setminus C_{PE}(i))$.

2.3 Branch-Privacy Conflicts

An even higher level of security can be achieved if we require two customers $i, j \in T$ to be on separate branches of the solution network B. Let $C_B \subseteq 2^T$ be a set of *branch-privacy conflicts*. Then the STP with branch-privacy constraints (STP+CB) is defined as the corresponding STP with the side constraint that i and j have to be in distinct connected components of $B \setminus r$ for $\{i, j\} \in C_B$. Figure 2 (center) illustrates an optimal solution for an STP+CB instance with $C_B = \{\{1, 2\}, \{3, 4\}, \{6, 8\}\}$.

2.4 Branch-Edge-Privacy Conflicts

A customer $i \in T$ may fear the infiltration of the sub-network induced by its branch via an individual edge $\{j, k\} \in E$ in the network B. In other words, the connecting infrastructure used for this link is considered vulnerable with respect to the privacy of i, independent of the types of nodes j and k. Let $C_{BE} \subseteq (T \times E)$ be the set of such *branch-edge-privacy conflicts*. Then we define the STP with branch-edge-privacy conflicts (STP+CBE) as the corresponding STP with the additional requirement that for each $(i, e) \in C_{BE}$, node i and edge e have to be in distinct connected components of $B \setminus r$ if e is selected in the solution. Figure 2 (right) illustrates an optimal solution for an STP+CBE instance with $C_{BE} = \{(1, \{2, 5\}), (1, \{2, 9\}), (7, \{11, r\}), (6, \{10, r\})\}$.

2.5 Reductions

In order to decrease the problem instance size, we can apply the following preprocessing techniques, also called reduction tests.

– For two conflicting terminals i and j, neither in the STP+CP nor in the STP+CB the edge $e = \{i, j\}$ may be used in a solution. Hence, e can be removed from E.
– Note that in the case that $i \in e$ for a terminal-edge conflict (i, e) (in C_{PE} or C_{BE}), edge e can be deleted from E. Therefore, we assume that $i \notin e \ \forall (i, e)$ throughout the paper.
– An edge $e \in E$ which is in conflict with all terminals, i.e., $(i, e) \in C_{PE} \ \forall i \in T$, can be removed.

2.6 Model Relationships

Note that STP+CP, STP+CPE, STP+CB and STP+CBE reduce to the STP in the case that their corresponding conflict sets are empty. Hence, all optimization problems are NP-hard since the STP is [14]. Furthermore, the STP+CPE can be used to model the STP+CP which is expressed by the following lemma.

Lemma 1 *The STP+CPE generalizes the STP+CP.*

Proof Let P be an STP+CP instance and C_P the path-privacy conflict set. Then the set of solutions for the corresponding STP+CPE instance with $C_{PE} = \bigcup_{i \in T} \{(i, e) : j \in C_P(i), \ e \in \delta(j)\}$ equals the solution set for P; in particular, the optimal objective function values are identical. □

Similarly, a corresponding relation exists for the STP+CB.

Lemma 2 *The STP+CBE generalizes the STP+CB.* □

Let opt(P) denote the cost of an optimal solution for a problem instance. Then we can relate the optimal objective function values for models STP+CP and STP+CB as follows.

Lemma 3 *Let P be an STP+CP instance and C_P the path-privacy conflict set; P' an STP+CB instance and $C_B = C_P$. Then opt(P) $\leq$ opt(P').*

Proof STP+CP is a relaxation of STP+CB when $C_B = C_P$ [9]. □

3 Integer Programming Approaches

In this section, we provide strong integer formulations for all four models described in Section 2. While formulations for the STP+CP and the STP+CB were developed in [9], we introduce first formulations for the STP+CPE and the STP+CBE. All formulations are non-compact and we present corresponding cutting-plane techniques.

Based on the formulations, we designed branch-and-cut methods for the four models. The corresponding algorithms have been built within the IBM ILOG CPLEX 12.90 framework by using the CPLEX callback functions. Through these functions, the programmer can almost completely customize the general approach embedded into CPLEX. For example, one can choose the next node to explore in the enumeration tree, choose the branching variable, or define a problem dependent branching scheme, separate and add his own cutting planes, apply his own heuristic methods, etc. For additional details about the use of the different callback functions, the reader is referred to the documentation of the CPLEX callable library [11].

3.1 Formulations

For an edge set D, we denote the set of all arcs $\{(i,j),(j,i) : \{i,j\} \in D\}$ as $A[D]$.

3.1.1 STP

Our formulations use a binary *edge variable* y_e for each edge $e \in E$ in order to encode the Steiner tree that corresponds to an integer-feasible solution. We write $y_{i,j}$ to denote y_e for $e = \{i,j\}$ whenever convenient. Let A be the set of arcs obtained from all possible orientations of edges in E, i.e., $A = \{(i,j) : \{i,j\} \in E\}$. For each arc $a \in A$, we define a binary *arc variable* x_a. An edge (arc) is installed in a solution if and only if $y_e = 1$ ($x_a = 1$). We build the formulations based on the following non-compact cut-set-based formulation for the STP that is suggested in [24].

$$(F) \quad \min \sum_{e \in E} c_e y_e \tag{1a}$$

$$\text{s.t.} \quad x_{i,j} + x_{j,i} = y_{i,j} \qquad (\{i,j\} \in E) \tag{1b}$$

$$\sum_{a \in \delta^-(i)} x_a = 1 \qquad (i \in T) \tag{1c}$$

$$\sum_{a \in \delta^-(S)} x_a \geq 1 \qquad (S \cap T \neq \emptyset, S \subseteq V') \tag{1d}$$

$$x_a \in \{0,1\} \qquad (a \in A) \tag{1e}$$

$$y_e \in \{0,1\} \qquad (e \in E) \tag{1f}$$

In formulation (F), inequalities (1d) are *connectivity inequalities*, and also ensure that the tree is rooted at the root node r. The following flow-balance inequalities are known to be valid for the formulations above [18, 9].

$$\sum_{a\in\delta^-(i)} x_a \;\leq\; \sum_{a\in\delta^+(i)} x_a \qquad (i \in W) \quad (2)$$

Inequalities (2) ensure that the number of arcs leaving Steiner node $i \in W$ is at least the number of arcs entering i. The flow can be further balanced by

$$x_{i,j} \;\leq\; \sum_{(k,i)\in\delta^-(i):k\neq j} x_{k,i} \qquad ((i,j) \in A, i \in W). \quad (3)$$

We will denote the extensions of the formulations introduced below, say formulation (F), obtained after adding inequalities (2) and (3) by (F^+).

3.1.2 STP+CP

In order to extend formulation (F) to the STP+CP, we define the set of all conflicting nodes for $i \in T$ as $C_P(i) = \{j \in T : \{i,j\} \in C_P\}$. The following strong formulation for the STP+CP is suggested in [9].

$$\begin{aligned}
(F_{CP}) \quad \min \quad & \sum_{e\in E} c_e y_e \\
\text{s.t.} \quad & (1b) - (1f) \\
& \sum_{a\in\delta^-(S\setminus C_P(i))} x_a \;\geq\; 1 \qquad (S \subseteq V' : i \in S, i \in T) \quad (4a)
\end{aligned}$$

Path-privacy connectivity inequalities (4a) ensure that connectivity with respect to r is established using nodes that are not in conflict with terminal $i \in T$. A more detailed polyhedral discussion on formulation (F_{CP}) can be found in [9].

3.1.3 STP+CPE

A formulation for the more general STP+CPE can be derived using a connectivity argument. Let $C_{PE}(i)$ denote the set of edges that are in path-conflict with node $i \in T$ in an STP+CPE instance; i.e., $C_{PE}(i) = \{e \in E : (i,e) \in C_{PE}\}$. Then the STP+CPE can be formulated as follows.

$$\begin{aligned}
(F_{CPE}) \quad \min \quad & \sum_{e\in E} c_e y_e \\
\text{s.t.} \quad & (1b) - (1f) \\
& \sum_{a\in\delta^-(S)\setminus A[C_{PE}(i)]} x_a \;\geq\; 1 \qquad (S \subseteq V' : i \in S, i \in T) \quad (5a)
\end{aligned}$$

Path-edge-privacy connectivity inequalities (5a) ensure that connectivity with respect to r is established without the usage of edges that are in conflict with terminal $i \in T$. Similarly, as shown for inequalities (4a) in [9], a weaker formulation for the STP+CPE can be obtained by using the following disaggregated version of inequalities (5a) that considers one conflict at a time.

$$\sum_{a \in \delta^-(S) \setminus \{(j,k),(k,j)\}} x_a \geq 1 \qquad (S \subseteq V' : i \in S, (i, \{j,k\}) \in C_{PE})$$

Note that formulation (F_{CPE}) is equivalent to formulation (F_{CP}) if $C_E = \{(i, \{i,j\}) : j \in C_P(i), i \in T\}$. Hence, it reduces to the strongest known STP+CP formulation ([9]) for STP+CP instances.

3.1.4 STP+CB

A strong formulation for the STP+CB can be obtained by considering multiple mutually conflicting terminals. Let $\mathcal{Q}_B$ be the set of all cliques in the graph (T, C_B). Then the STP+CB can be formulated as follows [9].

$$\begin{aligned}
(F_{CB}) \ \min \quad & \sum_{e \in E} c_e y_e \\
\text{s.t.} \quad & (1b) - (1f) \\
& \sum_{a \in \delta^-(Q)} x_a \geq |Q| && (Q \in \mathcal{Q}_B, S \subseteq V' : Q \subseteq S) \quad (7a) \\
& \sum_{a \in \delta^-(i)} x_a \leq 1 && (i \in W) \quad (7b)
\end{aligned}$$

Inequalities (7a) imply the installation of at least $|Q|$ tree branches to connect the terminals in a clique $Q \in \mathcal{Q}_B$. Note that every elementary branch-conflict in C_B is contained in $\mathcal{Q}_B$ as a clique of cardinality two; i.e., $C_B \subseteq \mathcal{Q}_B$. Steiner in-degree inequalities (7b) are necessary in order to ensure a proper tree structure [9].

3.1.5 STP+CBE

In order to formulate the STP+CBE, the most general privacy-conflict model, we use a slightly more elaborate connectivity argument. Let $C_{BE}(i) = \{e : (i, e) \in C_{BE}\}$ for $i \in T$. We define the total branch-edge conflict graph G_{CBE} to have a node for each node in V and each edge in E; i.e., $V[G_{CBE}] = \{v_i, v_e : i \in V, e \in E\}$. Its edge set contains two types of edges: Between two nodes that represent nodes in V and the corresponding edge is a conflict edge; between a node that represents a terminal and a node that represents an edge which are in a privacy conflict; i.e., $E[G_{CBE}] = \{v_i, v_e : \exists(i,e) \in C_{BE}\} \cup \{e : \exists i \in Te \in C_{BE}(i)\}$. Then the STP+CBE can be formulated as follows.

$$(F_{CBE}) \ \min \ \sum_{e \in E} c_e y_e$$

$$\text{s.t.} \ \ (1b) - (1f), (7b)$$

$$\sum_{a \in \delta^-(S)} x_a \ \geq \ x_{j,k} + x_{k,j} + 1 \qquad (S \subseteq V' : \{i,j,k\} \subseteq S, (i,\{j,k\}) \in C_{BE}) \tag{8a}$$

$$\sum_{a \in \delta^-(S) \setminus \{(r,k)\}} x_a \ \geq \ 1 \qquad (S \subseteq V' : i \in S, (i,\{r,k\}) \in C_{BE}) \tag{8b}$$

Inequalities (8a) ensure that there exist two disjoint paths that connect terminal $i \in T$ and its conflicting edge $\{j,k\} \in C_{BE}(i)$ to r if the edge is implemented. They can be separated efficiently on the edge set $E[x^*] \cap E[C_{BE}]$, where $E[x^*]$ represents the edges of the support graph. In the case that a conflicting edge e is incident with r, we cannot require this double-connectivity, but single-connectivity for i in the network that does not contain e (Inequalities (8b)). Furthermore, they correspond to Inequalities (5a) on a restricted set of branch-edge-privacy conflicts. Inequalities (8b) can be strengthened by incorporating all conflicting edges for i as follows.

$$\sum_{a \in \delta^-(S) \setminus A[C_{BE}(i)]} x_a \ \geq \ 1 \qquad (S \subseteq V' : i \in S, (i,\{r,k\}) \in C_{BE}) \tag{9a}$$

3.2 Cut Separation Algorithms

Cutting-plane algorithms for formulation (F) suffer from an extensive number of model cuts (1d). To overcome this computational challenge, [15] suggest several techniques that aim at generating multiple diverse and at the same time effective connectivity cuts. Our approach incorporates the ideas of these authors. Note that these techniques do not improve the dual bounds obtained by formulation F and solely speed up convergence of the cut generation process.

- We use the following slightly different, but equivalent, version of inequalities (1d) which enforces connectivity for each terminal node separately. This may result in the separation of multiple cuts.

$$\sum_{a \in \delta^-(S)} x_a \geq 1 \qquad (i \in S, S \subseteq V', i \in T) \tag{10}$$

- Forward and backward cuts: When separating inequalities (10), we compute two minimal cuts: The $r - i$ cut and the $i - r$ cut which are of equal weight but may differ in terms of cut sets.

– Steiner node connectivity: Similar to cut inequalities (10), we derive the following connectivity cuts for each Steiner node which ensure connectivity dependent on whether the node is used in the tree or not.

$$\sum_{a \in \delta^-(S)} x_a \geq \sum_{a \in \delta^-(\{i\})} x_a \qquad (i \in S, S \subseteq V', i \in W) \tag{11}$$

– Cut ranking: We limit ourselves to adding the 15 most violated cuts for each connectivity-type inequality. We measure the violation of an inequality $g(y,x) \leq b$ by solution (y^*, x^*) as $g(y^*, x^*) - b$.
– Connected component connectivity. Whenever the support graph w.r.t. y is disconnected, then we add dynamically connectivity inequalities (1d) in which S contains the component nodes (especially during the first rounds).
– Flow-balance inequalities (2) are added to the initial formulation, but inequalities (3) are separated dynamically.

The separation problem for connectivity inequalities (10) and (11) corresponds to finding an $r - i$ cut of minimal weight in the support graph. Conflict-connectivity inequalities (4a), (5a), (7a), (8a), (8b), (8b), and (9a) can be separated similarly on an auxiliary network. For a more detailed description we refer to [10] and [9]. For inequalities (7a), we explicitly compute the set of all maximal cliques $\mathcal{Q}$ a-priori using the Bron-Kerbosch algorithm [2]; we limit ourselves to 10000 cliques.

4 Constraint Programming Approaches

In this section we propose alternative solution approaches based on constraint programming. To this end, we develop constraint programming formulations for the STP+CP, the STP+CPE, the STP+CB and the STP+CBE. In contrast to the IP methods in Section 3, the compact formulations below can be directly fed into any constraint programming solver, without needing to develop cutting plane techniques.

4.1 CP Formulations

As opposed to [20], we derive our formulation from the multi-commodity formulation for the STP introduced in [24] (formulation (F) in [18]). This compact IP formulation is known to produce dual bounds for the STP, but is computationally inefficient since having a large number of variables ($\mathcal{O}(|T||A|)$). Nevertheless, extensions of its CP counterpart can be used to obtain primal solutions for our models without the need for a cutting plane algorithm, as our computational results will show.

For each node $i \in V$, we introduce a binary *node variable* w_i which takes value 1 if node i is in the network, and 0 otherwise. We define an integer *arc commodity-flow variable* f_a^i for each arc $a \in A$ and terminal $i \in T$, which describes the units of flow of commodity i on arc a. To encode the usage of an arc, we also define a binary *arc* variable z_a for each arc $a \in A$. Then the STP can be formulated as the following constraint program.

$$
\begin{aligned}
(G)\ \min\quad & \underset{a\in A}{\texttt{sum}}(c_a \;*\; z_a) && && \text{(12a)}\\
\text{s.t.}\quad & \underset{a\in\delta^+(i)}{\texttt{sum}}(f_a^i) \;=\; \underset{a\in\delta^-(i)}{\texttt{sum}}(f_a^i) - 1 && (i \in T) && \text{(12b)}\\
& \underset{a\in\delta^+(i)}{\texttt{sum}}(f_a^k) \;=\; \underset{a\in\delta^-(i)}{\texttt{sum}}(f_a^k) && (k \neq i \in T, i \in T) && \text{(12c)}\\
& \underset{a\in\delta^+(r)}{\texttt{sum}}(f_a^k) \;=\; \underset{a\in\delta^-(r)}{\texttt{sum}}(f_a^k) + 1 && (k \in T) && \text{(12d)}\\
& z_{i,j} + z_{j,i} \;\leq\; 1 && (\{i,j\} \in E) && \text{(12e)}\\
& z_{i,j} \;\leq\; \underset{k\in T}{\texttt{sum}}(f_{i,j}^k) && ((i,j) \in A) && \text{(12f)}\\
& z_{i,j} \;\geq\; f_{i,j}^k && (k \in T, (i,j) \in A) && \text{(12g)}\\
& \texttt{alternative}\big(w_i, z_{j,i} | \{j,i\} \in \delta(i)\big) && (i \in V') && \text{(12h)}\\
& w_i = 1 && (i \in V \setminus W) && \text{(12i)}\\
& \underset{i\in V}{\texttt{sum}}(w_i) \;=\; \underset{a\in A}{\texttt{sum}}(z_a) + 1 && && \text{(12j)}\\
& w_i \ \texttt{integer variable in}\ \{0,1\} && (i \in V) && \text{(12k)}\\
& z_a \ \texttt{integer variable in}\ \{0,1\} && (a \in A) && \text{(12l)}\\
& f_a^i \ \texttt{integer variable in}\ \{0,1\} && (i \in T, a \in A) && \text{(12m)}
\end{aligned}
$$

In formulation (G), the objective (12a) is to minimize the network edge costs. The flow conservation for the commodity is ensured via constraints (12b)-(12d). We forbid reverse arcs in inequality (12e). In IP formulations, we can rely on the fact that every solution of a linear program (LP) at a branching minimizes the objective; in particular, this is true for integer solutions. In CP though, we have to explicitly assure that every feasible assignment of values to variables corresponds to a feasible solution to the problem. Therefore, constraints (12f) are used to ensure that flow is traversing an arc that is installed. Conversely, an arc has to be used if it carries flow (12g). We formulate the in-degree requirement for used network nodes using the alternative constraint (12h). The latter ensures that exactly one incident in-arc is chosen if the corresponding node's node variable w_i is set to 1. Variable w_i is forced to one if i is a terminal node. (12j) forces the number of edges to be equal to the number of nodes minus one in each solution and was shown to be computationally advantageous [20]. In our model, the alternative constraint directly implies the latter, making it redundant. The correctness of formulation (G) follows from the fact that it equals to the corresponding LP formulation in [24] (using an equivalent clique inequality for constraint (12h)).

STP formulation (G) allows us to derive formulations for the privacy conflict models by limiting corresponding commodity flows. The STP+CP can be formulated as follows.

$$(G_{CP}) \quad \min \quad \underset{e\in E}{\texttt{sum}}(c_e \, * \, u_e) \tag{13a}$$

$$\text{s.t.} \quad (12b) - (12m)$$

$$\underset{\{i,k\}\in\delta(i)}{\texttt{sum}}(f^j_{i,k} + f^j_{k,i}) + \underset{\{j,k\}\in\delta(j)}{\texttt{sum}}(f^i_{j,k} + f^i_{k,j}) \;=\; 0 \quad (\{i,j\} \in C_P) \tag{13b}$$

For a path-privacy conflict $\{i,j\} \in C_P$, constraints (13b) force in-flows and out-flows of conflicting commodity i to zero for terminal j. The STP+CPE can be formulated as follows.

$$(G_{CPE}) \quad \min \quad \underset{e\in E}{\texttt{sum}}(c_e \, * \, u_e) \tag{14a}$$

$$\text{s.t.} \quad (12b) - (12m)$$

$$f^i_{k,j} + f^i_{j,k} \;=\; 0 \qquad (\{j,k\} \in C_{PE}(i), i \in T) \tag{14b}$$

For an edge-privacy conflict $(i,\{j,k\}) \in C_E$, constraints (14b) force forward-flows and backward-flows of commodity i to zero on edge $\{j,k\}$.

The STP+CB can be formulated as follows.

$$(G_{CB}) \quad \min \quad \underset{e\in E}{\texttt{sum}}(c_e \, * \, u_e) \tag{15a}$$

$$\text{s.t.} \quad (12b) - (12m)$$

$$f^i_{k,l} + f^i_{l,k} + f^j_{k,l} + f^j_{l,k} \;\leq\; 1 \qquad (\{k,l\} \in E, \{i,j\} \in C_B) \tag{15b}$$

Constraint (15b) allows at most one unit of conflicting commodities i and j on forward and backward arcs of each potential edge in order to force terminals i and j on distinct tree branches.

The STP+CBE can be formulated as follows.

$$(G_{CBE}) \quad \min \quad \underset{e\in E}{\texttt{sum}}(c_e \, * \, u_e) \tag{16a}$$

$$\text{s.t.} \quad (12b) - (12m), (14b)$$

$$f^i_{j',k'} + f^i_{k',j'} + f^{i'}_{j',k'} + f^{i'}_{k',j'} + f^{i'}_{j,k} + f^{i'}_{k,j} \;\leq\; 2$$

$$(\{j',k'\} \in E, i' \in T \setminus i, \{j,k\} \in C_{BE}(i), i \in T) \tag{16b}$$

Note that we include constraint (14b) for conflicts in C_{BE}. In this way we only need to cover the case that a conflicting edge is not on the terminal's path to r. To understand constraint (16b), we first observe that if edge $e = \{j,k\}$ and terminal i are on the same branch, there must be a terminal i' that is connected to r via e. Furthermore, there has to be an edge $e' = \{j',k'\}$ which is used by commodities i and i'. In order to forbid this, constraint (16b) only allows flow combinations with at most two of the variables (f^i_e, $f^{i'}_{e'}$, and $f^{i'}_e$) set to 1.
In formulation (F_{CBE}), we deal with a large number of constraints (16b) which cannot be handled by a CP solver. To overcome this, we observe that it is

sufficient to apply these constraints to edges $\{j', k'\} \in \delta(r)$. In this case, we obtain:

$$f_a^i + f_a^{i'} + f_{j,k}^{i'} + f_{k,j}^{i'} \ \leq \ 2 \ (a \in \delta^+(d), i' \in T \setminus i, \{j,k\} \in C_{BE}(i), i \in T) \tag{17a}$$

The CP formulations above can be used to derive corresponding IP formulations. It is well-known that multi-commodity flow formulations, such as formulation (G), are computationally inefficient for the STP. In preliminary experiments, we could confirm this for formulations (G_{CP}) and (G_{CB}).

4.2 Heuristic Starting Point Construction

Our CP-based approaches benefit from starting points representing a feasible solution. This effect has been observed for applications of CP to hard combinatorial problems [9]. This is especially important when the CP solver is struggling to find a feasible (or decent) initial solution itself; possibly due to the large number of variables or constraints.

To overcome this limitation, we define a simple generic constructive heuristic (**HEU**) that can be used to derive an initial solution for instances of all the presented models. We are especially interested in finding solutions for dense underlying networks G having a large number of variables in the multi-commodity flow formulations from Section 4.1. Note that known shortest-path-based heuristics for the STP (see, e.g., [22]) cannot be applied to our extended models due to the conflicts.

For an instance of STP+X (X$\in$ {CP,CB,CPE,CBE}), we repeatedly run the constructive heuristic that is described in Algorithm 1 in a multi-start fashion. The procedure $\mathit{selectTerminal}(T', \sigma)$ returns a terminal in $T' \subseteq T$ using the strategy σ. A shortest path w.r.t. the edge costs between node $t_1 \in V$ and node $t_2 \in V \setminus t_1$ in the network H is returned by $\mathit{shortestPath}(t_1, t_2, H)$, if it exists. An empty path is returned, otherwise. $CX(t)$ denotes the set of nodes/edges that are in conflict with terminal t. Note that Algorithm 1 fails to return a solution if for a terminal t, no path can be found that connects it to the root r. For σ we use the following priorities.

a) Descending terminal node index.
b) Ascending terminal node index.
c) No direct connection to r first, then descending terminal node index.
d) Descending number of conflicting nodes/edges, then descending terminal node index.

In a second run, all the strategies are repeated while forcing a direct edge-connection to r to be chosen if it exists. In total, this leads to eight runs of Algorithm 1.

Algorithm 1: Constructive STP+X Heuristic (**HEU**)

```
  Input:  $P$ (STP+X instance)
          $S$ (STP+X solution)
          $\sigma$ (Construction strategy)
  Output: [$N$ (Solution network for $P$)]
1 $N \leftarrow (\emptyset, \emptyset)$;
2 while $\big(T \not\subseteq V[N]\big)$ do
3     $t \leftarrow selectTerminal(T \setminus V[N], \sigma)$;
4     $P \leftarrow shortestPath(r, t, (V \setminus T[N], E) \setminus CX(t))$;
5     if $\big(P = 0\big)$ then
6         return;
7     else
8         $N \leftarrow N \cup P$;
9 return $N$;
```

5 Computational Analysis

In this section, we deduct an empirical evaluation of the developed optimization methods. After explaining the used test instance set, we present and compare the results of our exact approaches. Moreover, we study the conflict/cost trade-offs and analyze the topological conflict impacts.

5.1 Experimental Setup

In this section, we report and analyze the results obtained by our algorithms. They are implemented in C++ and CPLEX 12.90 and run on an Intel Core i7-7600 2.80 GHz machine with 16 GB RAM.

We evaluate the performance of our approaches using a diverse set of instances[1]. We derive 1560 instances in total with up to 100 nodes from 30 base STP instances from [1][2] (Steinb$k$, $k \in \{3, 4, 5, 16, 17, 18\}$) and the SteinLib[3] [16] (from test sets PUC, ES20FST, P4Z, and P4E); 390 instances for each model where conflicts for STP+CP and STP+CB are identical [9]. We randomly draw pairs of conflict nodes from T using a uniform distribution with two different seeds. For $\gamma \in \{0, 0.25, 0.5, 0.75, 1.0\}$, we incrementally generate $\lceil \gamma |T| \rceil$ conflicts for sparse instances ($|E[G]| < 0.5\binom{n}{2}$) and $\lceil \gamma |T|(|T| - 1)/2 \rceil$ conflicts for dense instances. That is, for a base instance and a seed, the con-

[1] The test instances can be obtained from the authors upon request.
[2] `http://people.brunel.ac.uk/~mastjjb/jeb/orlib/steininfo.html`
[3] `http://steinlib.zib.de`

flicts obtained from γ_2 contain all conflicts for γ_1 if $\gamma_1 \leq \gamma_2$. The node of highest degree in G, with lowest index (in case of ties), is assigned to be the root node r. Similarly, edge conflicts are randomly generated by sampling a terminal first, followed by an edge. For sparse instances, we generate $\lceil \gamma |E| \rceil$ conflicts and for dense instances, we generate $\lceil \gamma |T||E|/5 \rceil$ conflicts. To avoid infeasible instances, we initially experimented with 20 random seeds and only selected the first seeds for which formulation G_{CE} found a solution for all values of γ.

5.2 Results

We first introduce the notation used for the presentation of our computational results. We run our methods on the instances after applying preprocessing techniques from Section 2.5. The number of instances of a **Type** (Sparse or Dense) and a conflict density $\boldsymbol{\gamma}$ ($\boldsymbol{\gamma} \in \{0, 0.25, 0.5, 0.75, 1\}$) is given in column **#**. Note that the reduced number of instances for $\boldsymbol{\gamma}$=0 (no conflicts) are due to no random seeds being applied. Column **sols** lists the relative number (in %) of instances for which a feasible solution can be found within 600 seconds (no time limit for solving the LP). We use ***** to describe the relative number (in %) of instances that are solved to optimality. Let lb' and ub' denote the best lower and upper bounds that we found for an instance. Then we provide the average relative optimality gap that a lower bounding method yields by $\boldsymbol{\Delta^{LB}}$. For an instance and a lower bound lb, this gap is computed as $100 * (ub' - lb)/ub'$. Similarly, the best average relative gap for an upper bounding method can be found in column $\boldsymbol{\Delta^{UB}}$. The relative gap with respect to both best bounds $(100 * (ub' - ub')/ub')$ is denoted by $\boldsymbol{\Delta}$. The average run time in seconds is contained in column **t**.

5.2.1 Linear Programming

In Table 1, we summarize the results obtained by solving the linear programs (LP) derived from our IP formulations for the STP+CP and the STP+CB. Corresponding results for the STP+CPE and the STP+CBE can be found in Table 2. No start solution is provided to the solver. As observed in [9], it can be seen that solving the LP is more time consuming for branch-constrained models than it is for path-constrained problems. On average, the time spent on solving the LP for the STP+CB (STP+CBE) is about 4.7 (2.6) times higher than for the STP+CP (STP+CPE). At the same time, the relative number of instances for which the LP solution is integer-feasible (and, therewith, optimal) is lower when considering branch-privacy models. For the STP+CP (STP+CPE), it is 81.8% (57.2%), whereas only 52.1% (16.9%) integer-feasible LP solutions could be found for the STP+CB (STP+CBE). Accordingly, the observed average optimality gaps are 0.9% (9.4%) and 9.5% (23.9%), respectively. Note that the highest LP solve times are 2550 seconds and 1770 seconds. We observe that 94.1% of the dense STP+CP instances can be solved to optimality by solving the LP, but only 11.2% of the sparse STP+CBE instances has integral LP solutions. All dense instances without privacy constraints are having integral LP solutions but yield optimality gaps up to 96.3%.

Table 1 LP results for STP+CP and STP+CB instances.

Instances			**STP+CP**			**STP+CB**		
Type	γ	#	*	Δ^{LB}	t	*	Δ^{LB}	t
Sparse	0	13	69.2	1.7	8.8	69.2	1.7	15.6
	0.25	39	64.1	1.7	12.1	41.0	1.0	3.2
	0.5	39	61.5	1.7	12.2	38.5	0.7	5.5
	0.75	39	69.2	1.7	13.9	38.5	1.1	4.8
	1	39	66.7	1.7	14.4	35.9	1.1	6.7
	All	169	65.7	1.7	12.8	40.8	1.0	5.9
Dense	0	17	100.0	0.0	17.4	100	0.0	16.3
	0.25	51	90.2	0.0	35.1	41.2	28.4	149.4
	0.5	51	90.2	0.0	35.3	39.2	28.0	329.5
	0.75	51	100.0	0.0	32.3	49.0	21.7	410
	1	51	94.1	0.0	31.5	100.0	0.0	5.9
	All	221	94.1	0.0	32.3	60.6	18.0	207.7
All	All	390	81.8	0.9	22.6	52.1	9.5	106.8

Table 2 LP results for STP+CPE and STP+CBE instances.

Instances			**STP+CPE**			**STP+CBE**		
Type	γ	#	*	Δ^{LB}	t	*	Δ^{LB}	t
Sparse	0	13	69.2	1.7	12.3	69.2	1.7	6.2
	0.25	39	64.1	1.6	15.2	17.9	2.7	7.7
	0.5	39	56.4	1.5	10.3	5.1	4.4	8.5
	0.75	39	56.4	1.5	6.3	2.6	7.1	10.2
	1	39	51.3	1.2	3.9	0.0	8.7	9.4
	All	169	58	1.5	9.2	11.2	5.4	8.7
Dense	0	17	100.0	0.0	18.9	100.0	0.0	12.4
	0.25	51	60.8	9.6	44.1	27.5	29.8	73.8
	0.5	51	58.8	22	53.9	9.8	47.9	144.5
	0.75	51	41.2	20.7	76.0	11.8	51.2	219.6
	1	51	51.0	22.8	86.8	9.8	54.7	310.5
	All	221	56.6	17.3	61.6	21.3	42.4	173.7
All	All	390	57.2	9.4	35.4	16.9	23.9	91.2

5.2.2 Integer Programming

Table 3 and Table 4 show the results obtained after running our branch-and-cut algorithm with a time limit of 600 seconds for instances of the STP+CP and the STP+CB, and the STP+CPE and the STP+CBE, respectively. For

98.2%/82.3%/88.7% of the STP+CP/STP+CB/STP+CPE instances we are able to find integer-feasible solutions. However, we find solutions for only 67.2% of the STP+CBE instances. Similarly, the lowest number of optimally solved instances is for STP+CBE instances (57.7%). Note that the average relative optimality gap derived from the solutions found is relatively low for all models (0.0%-1.1%) with a maximum of 22.4%. For STP+CP instances, the lower bounds yield relatively high average optimality gaps (21.1%) compared to the other models (0.5%-9.6%).

Table 3 IP results for STP+CP and STP+CB instances.

Instances			**STP+CP**				**STP+CB**			
Type	γ	#	**sols**	*	$\mathbf{\Delta^{LB}}$	$\mathbf{\Delta^{UB}}$	**sols**	*	$\mathbf{\Delta^{LB}}$	$\mathbf{\Delta^{UB}}$
Sparse	0	13	100.0	84.6	0.6	0.6	100.0	84.6	0.6	0.6
	0.25	39	100.0	84.6	0.8	0.8	100.0	100.0	0.0	0.0
	0.5	39	94.9	84.6	1.0	0.6	100.0	97.4	0.0	0.0
	0.75	39	94.9	84.6	1.2	0.8	100.0	100.0	0.0	0.0
	1	39	92.3	84.6	1.0	0.4	100.0	100.0	0.0	0.0
	All	169	95.9	84.6	1.0	0.6	100.0	98.2	0.0	0.0
Dense	0	17	100.0	100.0	0.0	0.0	100.0	100.0	0.0	0.0
	0.25	51	100.0	100.0	0.0	0.0	54.9	54.9	28.1	0.0
	0.5	51	100.0	100.0	0.0	0.0	49.0	47.1	27.8	0.0
	0.75	51	100.0	100.0	0.0	0.0	60.8	56.9	21.7	0.1
	1	51	100.0	100.0	0.0	0.0	100.0	100.0	0.0	0.0
	All	221	100.0	100.0	0.0	0.0	68.8	67.4	17.9	0.0
All	All	390	98.2	93.3	0.5	0.3	82.3	80.8	9.0	0.0

Table 4 IP results for STP+CPE and STP+CBE instances.

Instances			**STP+CPE**				**STP+CBE**			
Type	γ	#	**sols**	*	$\mathbf{\Delta^{LB}}$	$\mathbf{\Delta^{UB}}$	**sols**	*	$\mathbf{\Delta^{LB}}$	$\mathbf{\Delta^{UB}}$
Sparse	0	13	100.0	84.6	0.7	0.7	100.0	84.6	0.8	0.8
	0.25	39	100.0	84.6	0.7	0.7	97.4	84.6	0.8	0.6
	0.5	39	100.0	87.2	0.3	0.3	100.0	87.2	0.2	0.3
	0.75	39	100.0	94.9	0.1	0.1	94.9	76.9	2.5	1.6
	1	39	100.0	100.0	0.0	0.0	94.9	74.4	2.9	2.0
	All	169	100.0	91.1	0.3	0.3	97.0	81.1	1.5	1.1
Dense	0	17	100.0	100.0	0.0	0.0	100.0	100.0	0.0	0.0
	0.25	51	90.2	88.2	9.3	0.1	68.6	62.7	27.7	0.7
	0.5	51	74.5	74.5	21.7	0.0	37.3	31.4	45.9	1.9
	0.75	51	76.5	70.6	20.1	0.2	31.4	27.5	48.8	2.6
	1	51	72.5	70.6	22.3	0.1	21.6	17.6	54.2	1.4
	All	221	80.1	77.8	16.9	0.1	44.3	39.8	40.7	1.2
All	All	390	88.7	83.6	8.6	0.2	67.2	57.7	21.1	1.1

5.2.3 Constructive Heuristic

Results for our simple construction heuristic that serves to find a CP starting point can be found in Table 5. The run time never exceeds one second. We recall that the motivation for these starting points is to provide a solution when the CP solver cannot find a feasible solution or only finds a very cost-intense solution. In our experiments, this seems to be mostly the case when considering dense instances which lead to larger a number of variables. **HEU** is able to construct solutions for all dense instances, but finds solutions for at most 33.5% of sparse instances for each model. With respect to the best found lower bounds, the achieved average optimality gaps range from 35.0% to 56.7% for the different models.

Table 5 Heuristic construction (**HEU**) results for STP+CP, STP+CB, STP+CPE and STP+CBE instances.

Instances			**STP+CP**		**STP+CB**		**STP+CPE**		**STP+CBE**	
Type	γ	#	**sols**	$\mathbf{\Delta^{UB}}$	**sols**	$\mathbf{\Delta^{UB}}$	**sols**	$\mathbf{\Delta^{UB}}$	**sols**	$\mathbf{\Delta^{UB}}$
Sparse	0	13	15.4	31.3	15.4	31.1	15.4	31.3	0.0	
	0.25	39	15.4	31.8	15.4	28.1	17.9	35.9	0.0	
	0.5	39	15.4	32.7	15.4	28.0	17.9	33.9	0.0	
	0.75	39	15.4	33.7	15.4	25.0	20.5	35.6	0.0	
	1	39	15.4	33.1	15.4	24.9	17.9	33.5	0.0	
	All	169	15.4	32.7	15.4	26.9	18.3	34.5	0.0	
Dense	0	17	100.0	70.0	100.0	70.0	100.0	70.0	100.0	70.0
	0.25	51	100.0	67.9	100.0	61.0	100.0	68.7	100.0	66.8
	0.5	51	100.0	66.8	100.0	53.6	100.0	67.7	100.0	64.9
	0.75	51	100.0	65.9	100.0	42.6	100.0	66.0	100.0	63.9
	1	51	100.0	65.0	100.0	6.8	100.0	64.2	100.0	62.6
	All	221	100.0	66.7	100.0	43.2	100.0	66.9	100.0	65.0
All	All	390	63.3	49.7	63.3	35.0	64.6	50.7	56.7	65.0

5.2.4 Constraint Programming

The results for our CP approaches (using **HEU** starting points) are summarized in Table 6 and Table 7. Except for sparse STP+CB instances, solutions could be found for every instance. CP finds more solutions than IP. It is able to improve the start solutions provided by **HEU**. The obtained relative optimality gaps for sparse instances are higher than the ones from IP but still below 11% (STP+CP: 5.9%, STP+CB: 5.5%, STP+CPE: 5.4%, STP+CBE: 10.6%). However, they are between 30.6% and 45.5% for dense instances. Even without providing starting points, CP can find feasible solutions for 100.0%, 92.8%, 97.4%, and 55.9% of the STP+CP, STP+CB, STP+CPE, and STP+CBE instances. Note that for some dense STP+CBE instances the model could not be loaded due to insufficient memory.

Table 6 Constraint programming (using starting point) results for STP+CP and STP+CB instances.

Instances			**STP+CP**				**STP+CB**			
Type	γ	#	**sols**	*	$\boldsymbol{\Delta}^{\mathbf{LB}}$	$\boldsymbol{\Delta}^{\mathbf{UB}}$	**sols**	*	$\boldsymbol{\Delta}^{\mathbf{LB}}$	$\boldsymbol{\Delta}^{\mathbf{UB}}$
Sparse	0	13	100	7.7	30.5	4.4	38.5	0	40.8	3
	0.25	39	100	0	33.2	6	38.5	0	42.2	4.2
	0.5	39	100	7.7	31	5.9	38.5	0	43.6	7.2
	0.75	39	100	10.3	30.2	6.1	38.5	0	46.5	6.3
	1	39	100	12.8	29.7	6.1	38.5	0	47.6	5.1
	All	169	100	7.7	31	5.9	38.5	0	44.7	5.5
Dense	0	17	100	5.9	48.9	47.2	100	5.9	48.9	47.2
	0.25	51	100	5.9	51.7	45.5	100	3.9	73.7	41.4
	0.5	51	100	5.9	53	45.3	100	2	78.7	38
	0.75	51	100	5.9	53.3	46	100	0	82.3	30.7
	1	51	100	5.9	52.9	42.4	100	0	85.7	6.7
	All	221	100	5.9	52.4	45	100	1.8	77.6	30.6
All	All	390	100	6.7	41.7	25.4	73.3	1	61.1	18

Table 7 Constraint programming (using starting point) results for STP+CPE and STP+CBE instances.

Instances			**STP+CPE**				**STP+CBE**			
Type	γ	#	**sols**	*	$\boldsymbol{\Delta}^{\mathbf{LB}}$	$\boldsymbol{\Delta}^{\mathbf{UB}}$	**sols**	*	$\boldsymbol{\Delta}^{\mathbf{LB}}$	$\boldsymbol{\Delta}^{\mathbf{UB}}$
Sparse	0	13	100	7.7	30.5	4.4	100	15.4	28.4	4.9
	0.25	39	100	10.3	29.9	5.9	100	2.6	35.5	8
	0.5	39	100	2.6	33.2	5.3	97.4	7.7	36.7	8.4
	0.75	39	100	10.3	30.4	5	94.9	5.1	41.2	12.9
	1	39	100	7.7	31.5	5.8	89.7	10.3	42.4	15.7
	All	169	100	7.7	31.2	5.4	95.9	7.1	38	10.6
Dense	0	17	100	5.9	48.9	47.2	100	5.9	48.9	47.2
	0.25	51	100	3.9	56	44.8	68.6	2	58.7	40.8
	0.5	51	100	3.9	63.7	45	70.6	0	72.5	43.6
	0.75	51	100	2	65.7	47.9	66.7	0	72.1	46
	1	51	96.1	2	65.2	44	60.8	0	72.1	51.8
	All	221	99.1	3.2	61.6	45.5	69.2	0.9	65.5	45.5
All	All	390	99.5	5.1	46.4	25.5	80.8	3.6	51.8	28.1

5.2.5 Overall Best

Finally, we report our best overall results in Table 8. These contain best lower (upper) bounds found by LP, IP and CP. For the STP+CP, the STP+CB and the STP+CPE, we can prove optimality for over 80% of the instances. For the STP+CBE, 40.3% of the instances remain unsolved.

Figure 3 illustrates optimal Steiner trees and the corresponding conflict graphs for a dense STP+CB instance (SteinLib base instance p402) with $|T| = 5$, $|W| = 94$, and different values of γ. It can be seen that the original STP instance ($\gamma = 0$) is optimally solved by a path network. The optimal cost increases by 37% ($\gamma = 0.25$), 103% ($\gamma = 0.5$), 124% ($\gamma = 0.75$), and 936%

Table 8 Best results for STP+CP, STP+CB, STP+CPE, and STP+CBE instances.

Instances			**STP+CP**		**STP+CB**		**STP+CPE**		**STP+CBE**	
Type	γ	#	**opt**	$\boldsymbol{\Delta}$	**opt**	$\boldsymbol{\Delta}$	**opt**	$\boldsymbol{\Delta}$	**opt**	$\boldsymbol{\Delta}$
Sparse	0	13	84.6	0.6	84.6	0.6	84.6	0.7	84.6	0.8
	0.25	39	84.6	0.8	100	0	84.6	0.7	84.6	0.8
	0.5	39	84.6	1	97.4	0	87.2	0.3	87.2	0.2
	0.75	39	84.6	1.2	100	0	94.9	0.1	76.9	2.5
	1	39	84.6	1	100	0	100	0	74.4	2.9
	All	169	84.6	1	98.2	0	91.1	0.3	81.1	1.5
Dense	0	17	100	0	100	0	100	0	100	0
	0.25	51	100	0	54.9	28.1	88.2	9.3	62.7	27.9
	0.5	51	100	0	47.1	27.7	74.5	21.7	31.4	49.3
	0.75	51	100	0	56.9	21.5	70.6	20.1	27.5	51.3
	1	51	100	0	100	0	70.6	22.3	17.6	56
	All	221	100	0	67.4	17.8	77.8	16.9	39.8	42.6
All	All	390	93.3	0.5	80.8	8.9	83.6	8.6	57.7	22.1

($\gamma = 1$). Also, the number of network branches and number of Steiner nodes increase from 2 to 5, and 5 to 13, respectively. Note that in the case of $\gamma = 1$, the branch conflicts $C_B = 2^T$ force every terminal to be a leaf on a separate branch.

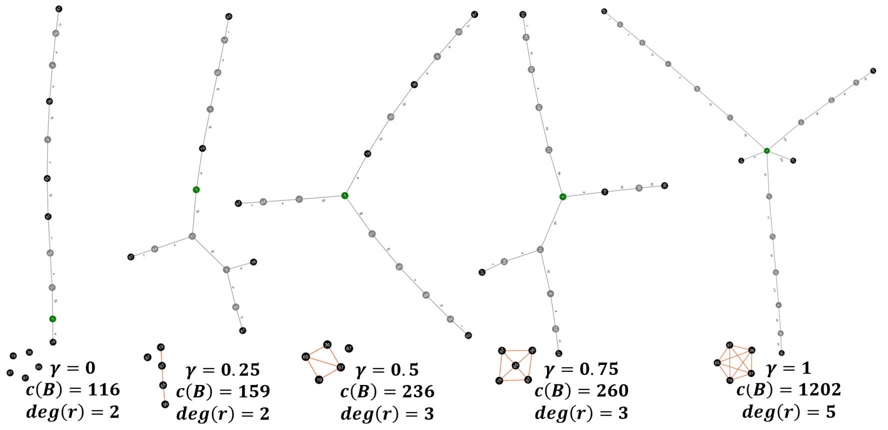

Fig. 3 Optimal networks for the STP+CB instance derived from SteinLib instance p402 for increasing values of γ.

6 Model Extensions

In the following, we describe several practically relevant model extensions and relationships to other known network design models.

Asymmetric Distrust. In the above models, we define path-privacy (branch-privacy) conflicts to be mutual for two terminals i and j with $\{i, j\} \in C_P$ ($\{i, j\} \in C_B$). Certainly, asymmetric versions may be defined in which we allow distrust coming from one customer only. In this case one could define a directed privacy-conflict relation $\subset (T \times T)$. We do not further explore this direction in this work, since the presented models and techniques for the symmetric case can be adapted accordingly.

Steiner Node Conflicts. Furthermore, we do not consider privacy conflicts involving Steiner nodes. A possible model extension could be the generalization which allows $C_P \subset (V' \times V')$, $C_E \in (V' \times E)$, and $C_B \subset (V' \times V')$, respectively.

Hop-Based Restrictions. Hop constraints are known to be of practical importance in telecommunication applications [23]. For two network nodes $i, j \in V$ and a hop limit h, they ensure that the (in tree models unique) path connecting i and j does not contain more than h edges. A hop-privacy-oriented model could be defined requiring that the number of hops between i and j is at least h. Assuming that the distance between two nodes on different branches is ∞, the resulting model falls between the STP+CP and the STP+CB. More detailed, setting $h = 1$ models the STP+CP, and $h = \infty$ corresponds to the $STP + CB$.

Graph Coloring. The NP-hard [14] graph coloring problem asks for an assignment of labels, or colors, to nodes of a graph such that adjacent nodes are labeled differently, and the number of used labels is minimized. Similarly, terminals need to be assigned to tree branches while avoiding conflicts in the STP+CB. The more restrictive total graph coloring problem [21] additionally asks for edge labels, while respecting edge-node and edge-edge incidence conflicts. The conflicts between customers and edges in the studied edge-conflict models are closely related. This plays an important role when formulating strong polyhedral descriptions for the STP+CBE in Section 3.

7 Conclusion

In this paper, we studied extensions of the Steiner tree problem in graphs for strategic network design. We considered model variants of practical relevance that take into account privacy concerns between pairs of terminals, and introduced more detailed conflicts between terminals and edges. We showed the relationships between four models and presented cut-set based integer programming (IP) formulations. Additionally, we formulated all models using constraint programming (CP) based on commodity-flows. In order to warm-start the latter approach, we devised a construction heuristic.

In a computational study on more than 1500 instances, we identified that the dense occurrence of terminal-edge conflicts yields the most difficult instances for our approaches. Also, it seemed significantly easier to optimally solve instances with very low or very high conflict density. In our experiments, IP outperformed CP. Strong formulations paired with efficient separation algorithms seemed to lead to well-performing branch-and-cut methods. However, the compact CP formulations were able to produce more feasible solutions. We suggest that CP approaches could benefit from both, further improved starting points and more aggressive reduction techniques. Moreover, the hybridization of IP and CP could be worth investigating for these hard combinatorial problems.

References

1. Beasley, J.E.: An SST-based algorithm for the Steiner problem in graphs. Networks **19**(1), 1–16 (1989)
2. Bron, C., Kerbosch, J.: Algorithm 457: Finding all cliques of an undirected graph. Commun. ACM **16**(9), 575–577 (1973)
3. Chopra, S., Gorres, E.R., Rao, M.: Solving the Steiner tree problem on a graph using branch and cut. ORSA Journal on Computing **4**(3), 320–335 (1992)
4. Chopra, S., Rao, M.R.: The Steiner tree problem i: Formulations, compositions and extension of facets. Mathematical Programming **64**(1-3), 209–229 (1994)
5. Cornet, A., Laforest, C.: Total domination, connected vertex cover and Steiner tree with conflicts. Discrete Mathematics & Theoretical Computer Science **19**(3) (2017). URL `http://dmtcs.episciences.org/4154`
6. Di Puglia Pugliese, L., Gaudioso, M., Guerriero, F., Miglionico, G.: An algorithm to find the link constrained Steiner tree in undirected graphs. In: G.M. Greuel, T. Koch, P. Paule, A. Sommese (eds.) Mathematical Software – ICMS 2016, pp. 492–497. Springer International Publishing, Cham (2016)
7. Filipecki, B., Van Vyve, M.: Stronger path-based extended formulation for the Steiner tree problem. Networks **75**, 3–17 (2020)
8. Gamrath, G., Koch, T., Maher, S.J., Rehfeldt, D., Shinano, Y.: SCIP-Jack — a solver for STP and variants with parallelization extensions. Mathematical Programming Computation **9**(2), 231–296 (2017)
9. Hill, A., Baldacci, R., Voß, S.: Branch-and-cut algorithms for Steiner tree problems with privacy conflicts. In: Computing and Combinatorics, Lecture Notes in Computer Science Vol. 11653, pp. 266–278. Springer International Publishing (2019)
10. Hill, A., Schwarze, S.: Exact algorithms for bi-objective ring tree problems with reliability measures. Computers & Operations Research **94**, 38–51 (2018)
11. IBM CPLEX: IBM ILOG CPLEX 12.90 callable library (2018)
12. Johnson, D.S., Minkoff, M., Phillips, S.: The prize collecting Steiner tree problem: theory and practice. In: Proceedings of the eleventh annual ACM-SIAM symposium on discrete algorithms, pp. 760–769. Society for Industrial and Applied Mathematics (2000)
13. Kanté, M.M., Laforest, C., Momège, B.: Trees in graphs with conflict edges or forbidden transitions. In: T.H.H. Chan, L.C. Lau, L. Trevisan (eds.) Theory and Applications of Models of Computation, pp. 343–354. Springer Berlin Heidelberg (2013)
14. Karp, R.M.: Reducibility among combinatorial problems. In: Complexity of computer computations, pp. 85–103. Springer (1972)
15. Koch, T., Martin, A.: Solving Steiner tree problems in graphs to optimality. Networks **32**(3), 207–232 (1998)
16. Koch, T., Martin, A., Voß, S.: Steinlib: An updated library on Steiner tree problems in graphs. In: X.Z. Cheng, D.Z. Du (eds.) Steiner Trees in Industry, pp. 285–325. Springer US, Boston, MA (2001)
17. Leggieri, V., Haouari, M., Triki, C.: The Steiner tree problem with delays: A compact formulation and reduction procedures. Discrete Applied Mathematics **164**, 178–190 (2014)
18. Polzin, T., Daneshmand, S.V.: A comparison of Steiner tree relaxations. Discrete Applied Mathematics **112**(1), 241–261 (2001)

19. Siebert, M., Ahmed, S., Nemhauser, G.: A linear programming based approach to the Steiner tree problem with a fixed number of terminals. arXiv preprint arXiv:1812.02237 (2018)
20. Una, D.D., Gange, G., Schachte, P., Stuckey, P.J.: Steiner tree problems with side constraints using constraint programming. In: Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence (AAAI-16), pp. 3383–3389. Association for the Advancement of Artificial Intelligence (2016)
21. Vizing, V.G.: Some unsolved problems in graph theory. Russian Mathematical Surveys **23**(6), 125–141 (1968)
22. Voß, S.: Steiner's problem in graphs: heuristic methods. Discrete Applied Mathematics **40**(1), 45–72 (1992)
23. Voß, S.: The Steiner tree problem with hop constraints. Annals of Operations Research **86**, 321–345 (1999)
24. Wong, R.T.: A dual ascent approach for Steiner tree problems on a directed graph. Mathematical Programming **28**(3), 271–287 (1984)