Combined Column-and-Row-Generation for the Optimal Communication Spanning Tree Problem

Christian Tilk*,[a], Stefan Irnich[a]

[a] *Chair of Logistics Management, Gutenberg Scholl of Management and Economics, Johannes Gutenberg University Mainz, Jakob-Welder-Weg 9, D-55128 Mainz, Germany.*

Abstract

This paper considers the exact solution of the optimal communication spanning tree problem (OCSTP), which can be described as follows: Given an undirected graph with transportation costs on every edge and communication requirements for all pairs of vertices, the OCSTP seeks a spanning tree that minimizes the sum of the communication costs between all pairs of vertices, where the communication cost of a pair of vertices is defined as their communication requirement multiplied by the transportation cost of the unique tree path that connects the two vertices. Two types of compact formulations for OCSTP were presented in the literature. The first one is a four-index model based on a path formulation. The second is a three-index model in which a solution is an intersection of spanning trees, each rooted at a different vertex of the graph and modeled using a flow formulation for spanning tree problems. We present Dantzig-Wolfe reformulations for both compact models to be used in a combined column-and-row-generation algorithm. In the path-based reformulation, the pricing problems are simple shortest-path problems, one for each pair of vertices with a positive communication requirement. The pricing problems of the tree-based reformulation are fixed-cost network flow problems, one for each vertex of the graph. We apply different heuristic and exact methods for pricing and present optimal solutions for benchmark instances with up to 40 vertices.

Key words: optimal communication spanning tree, column generation, row generation, branch-and-price-and-cut

1. Introduction

The *optimal communication spanning tree problem* (OCSTP) was first described by Hu (1974): Given an undirected graph with transportation costs on every edge and communication requirements for all pairs of vertices, the OCSTP seeks a spanning tree that minimizes the sum of the communication costs between all pairs of vertices, where the communication cost of a pair of vertices is defined as their communication requirement multiplied by the transportation cost of the unique tree path that connects the two vertices. Johnson *et al.* (1978) have shown that the OCSTP is $\mathcal{NP}$-hard. Besides a broad range of applications in designing telecommunication and transportation networks, the OCSTP occurs as a subproblem in some network hub location problems (Contreras and Fernández, 2012).

Two types of compact formulations for OCSTP were presented in the literature. The first one is a four-index model based on a path formulation, which was introduced and used in Contreras *et al.* (2010) in a Lagrangian relaxation approach. The second is the three-index model by Fernández *et al.* (2013) in which a solution is an intersection of spanning trees, each rooted at a different vertex of the graph and modeled using a flow formulation for spanning tree problems. Results by Fernández *et al.* (2013) indicate the superiority of the linear relaxation of the latter compact formulation.

*Corresponding author.
Email address: tilk@uni-mainz.de (Christian Tilk)

The contribution of the paper at hand is the presentation of new and effective exact algorithms for OCSTP. Our starting point is the four-index (path-based) and three-index (tree-based) models, for which we derive Dantzig-Wolfe reformulations. Both reformulations have a huge number of variables and constraints. Therefore, we develop a combined column-and-row-generation algorithm for their resolution. In the path-based reformulation, the pricing problems are simple shortest-path problems, one for each pair of vertices with a positive communication requirement. The pricing problems of the tree-based reformulation are fixed-cost network flow problems, one for each vertex of the graph.

Our goal is the comparison of both reformulations to decide which of the column-and-row-generation algorithms are more effective from a computational point of view. We think that the comparison is also interesting from a theoretical point of view: In the path-based reformulation, the shortest-path models used for pricing have the integrality property, while in the tree-based reformulation no integral model is available for the pricing problems because they are $\mathcal{NP}$-hard. Hence, there is the tradeoff that the former subproblems can be solved much faster while the latter generally lead to a stronger linear relaxation (Lübbecke and Desrosiers, 2005).

We combine several algorithmic techniques in order to accelerate the solution process. We use partial column generation to reduce the number of pricing problems to solve in each iteration of the column-generation algorithm (see Gamache *et al.*, 1999). Moreover, for the tree-based reformulation, we show that the solution of any single pricing problem allows us to generate columns for all other pricing problems. At the same time, this procedure provides feasible solutions and upper bounds on the OCSTP in each iteration.

In order to finally compute optimal integer solutions, both column-and-row-generation algorithms are integrated into branch-and-bound. In addition, cycle-elimination inequalities can strengthen the path-based reformulation. The resulting branch-and-price-and-cut algorithms often quickly yield integer optimal solutions or otherwise tight lower and upper bounds. The remaining integrality gap is typically small and even for 50-vertex instances around 5 %. We present computational results for different variants of the branch-and-price-and-cut algorithms and show the usefulness of the proposed methods.

The paper is structured as follows: Section 2 briefly surveys exact solution algorithms for the OCSTP. The formal definition of the OCSTP, the compact four-index model of Contreras *et al.* (2010) as well as the compact three-index model of Fernández *et al.* (2013) are presented in Section 3. In Section 4, we describe the new extensive formulations to be solved with column-and-row-generation algorithms. Moreover, we formalize the pricing problems and discuss algorithms for their effective solution. Section 5 reports computational results. The paper closes with conclusions drawn in Section 6.

2. Literature

There exist several heuristic approaches to solve the OCSTP, e.g., with evolutionary algorithms, see for instance (Fischer and Merz, 2007) and (Steitz and Rothlauf, 2012). Approximation algorithms are presented by Wu *et al.* (2000), Sharma (2006) and Peleg and Reshef (1998).

Only a few exact approaches have been published: Ahuja and Murty (1987) propose a combinatorial branch-and-bound that is able to solve instances on sparse graphs with up to 40 vertices. Rothlauf (2009) investigates structural properties of optimal OCSTP solutions and heuristics exploiting this knowledge. Contreras *et al.* (2010) directly solve the four-index model for instances with up to 40 vertices and present a Lagrangian relaxation-based algorithm to compute tight lower and upper bounds for larger instances. Recently, Fernández *et al.* (2013) introduced a three-index, flow-based formulation for the OCSTP. They also present classes of valid inequalities and sketch some computational results obtained when the three-index formulation is directly solved with a MIP solver. Fischetti *et al.* (2002) present a column-and-row-generation algorithm for the minimum routing cost tree problem. This problem is a special case of the OCSTP, where the communication requirement between all pairs of vertices is one (or identical). They analyze different formulations and report optimal solutions for instances with up to 50 vertices.

3. Compact formulations

We start with a formal definition of the OCSTP: Let $G = (V, E)$ be a simple undirected graph with vertices $V = \{1, 2, \ldots, n\}$ and edges E. Let $n = |V|$ and $m = |E|$. A *transportation cost* c_{ij} is associated with each edge $\{i, j\} \in E$ and a *communication requirement* r_{ij} with each pair $(i, j) \in V \times V$. Any spanning tree $T = (V, E_T)$ with $E_T \subset E$ is a feasible solution to the OCSTP. Recall that it is necessary and sufficient that $|E_T| = n-1$ and $V = V(E_T)$ holds. The *communication cost of a tree* T is $\sum_{(i,j)\in V\times V} r_{ij} \sum_{(s,t)\in P_{ij}} c_{ij}$, where P_{ij} is the unique path connecting i and j in T. The objective of the OCSTP is to find a spanning tree with minimum communication cost.

For simplicity, we assume that all vertices $u \in V$ have at least some positive communication requirement. $\sum_{i\in V}(r_{iu} + r_{ui}) > 0$. Since G is undirected, there exist several possibilities to split the communication requirement between (i, j) and $(j, i) \in V \times V$. Let $o_{ij} = r_{ij} + r_{ji}$ be the total communication requirement between vertices i and j (in both directions). One obtains an equivalent OCSTP instance by redefining $r_{ij} := (o_{ij} - k_{ij})/2$ and $r_{ji} := (o_{ij} + k_{ij})/2$ for each k_{ij} with $0 \le k_{ij} \le o_{ij}$. For $k_{ij} = 0$, a symmetric communication requirement $r_{ij} = r_{ji}$ results. The fully asymmetric case results from setting $k = o_{ij}$ for all $i, j \in V$ with $i > j$ resulting in $r_{ij} = 0$ and $r_{ji} = o_{ij}$. We will exploit the above equivalence and use the fact that asymmetric instances are typically easier to solve, see Section 5.

Next we will present the two compact formulations from the literature. Both formulations utilize the complete directed graph $D = (V, A)$ underlying G with arc set $A = \{(i, j) \in V \times V : \{i, j\} \in E\}$. Note that A contains for each arc $(i, j) \in A$ its antiparallel counterpart $(j, i) \in A$. For each vertex $u \in V$, $\delta^+(u)$ denotes its *forward star* and $\delta^-(u)$ its *backward star* in D. Moreover, let $R := \{(s, t) \in V \times V : r_{st} > 0\}$ be the set of all pairs of vertices with a positive communication requirement.

3.1. Path-based formulation

The formulation by Contreras *et al.* (2010) exploits the fact that the communication requirement between a pair of vertices is satisfied by a path between them. The *communication cost* of such a path is defined as the sum of the transportation costs of the path multiplied by the communication requirement between its extremities. To fulfill the tree requirement, the overall number of arcs used in all paths must be limited to $n - 1$.

Two types of variables are used in the path-based formulation. Continuous variables f_{stij} are defined for each pair $(s, t) \in R$ and each arc $(i, j) \in A$ (four indices). A positive value of f_{stij} indicates that arc $(i, j) \in A$ is used for fulfilling the communication requirement between s and t, $(s, t) \in R$. Binary variables x_{ij} are defined for each edge $\{i, j\} \in E$ and $x_{ij} = 1$ indicates that the edge is in the spanning tree. The path-based formulation reads as follows:

$$\min \quad \sum_{(s,t)\in R} r_{st} \sum_{(i,j)\in A} c_{ij} f_{stij} \tag{1a}$$

$$\text{s.t.} \quad \sum_{(s,j)\in\delta^+(s)} f_{stsj} = 1 \qquad \forall (s,t) \in R \tag{1b}$$

$$\sum_{(i,j)\in\delta^+(i)} f_{stij} - \sum_{(h,i)\in\delta^-(i)} f_{sthi} = 0 \qquad \forall (s,t) \in R; i \in V \setminus \{s,t\} \tag{1c}$$

$$f_{stij} + f_{stji} \le x_{ij} \qquad \forall (s,t) \in R; \{i,j\} \in E \tag{1d}$$

$$\sum_{\{i,j\}\in E} x_{ij} = n - 1 \tag{1e}$$

$$x_{ij} \in \{0, 1\} \qquad \forall \{i,j\} \in E \tag{1f}$$

$$f_{stij} \ge 0 \qquad \forall (i,j) \in A; (s,t) \in R \tag{1g}$$

The objective (1a) minimizes the communication cost of the resulting tree. Constraints (1b) and (1c) are the flow constraints of the path between s and t for each positive communication requirement $r_{st} > 0$. The corresponding constraint that the flow into t is one, i.e., $\sum_{(i,t)\in\delta^-(t)} f_{stit} = 1$, is redundant and therefore

omitted. Constraints (1d) couple the f with the associated x variables. Finally, constraint (1e) ensures that the flow of each communication requirement can be fulfilled by the same spanning tree. The variables and their domains are given by (1f) and (1g).

3.2. Tree-based formulation

The idea of the three-index formulation by Fernández *et al.* (2013) is that each vertex $u \in V$ distributes the commodity u to all other vertices $V \setminus \{u\}$. The demand of vertex i for commodity u is r_{ui}. The flow of each commodity u induces a tree rooted at vertex u with communication cost $\sum_{i \in V \setminus \{u\}} r_{ui} \sum_{(k,l) \in P_{ui}} c_{kl}$, where P_{ui} is the tree path connecting u and i. The model seeks for a spanning tree that minimizes the sum of the communication costs of all commodities.

The three-index formulation uses three types of variables: Binary variables y_{uij} are defined for vertices $u \in V$ and arcs $(i,j) \in A$. The value $y_{uij} = 1$ indicates that arc (i,j) is used for distributing commodity u. Continuous variables f_{uij} give the amount of flow of commodity u on arc (i,j). Binary variables x_{ij} are defined for each edge $\{i,j\} \in E$ and indicate whether the edge is in the spanning tree. The compact three-index formulation is:

$$\min \quad \sum_{u \in V} \sum_{(i,j) \in A} c_{ij} f_{uij} \tag{2a}$$

$$\text{s.t.} \quad \sum_{(u,j) \in \delta^+(u)} f_{uuj} = \sum_{i \in V \setminus \{u\}} r_{ui} \qquad \forall u \in V \tag{2b}$$

$$\sum_{(i,j) \in \delta^+(i)} f_{uij} - \sum_{(h,i) \in \delta^-(i)} f_{uhi} = -r_{ui} \qquad \forall u \in V; i \in V \setminus \{u\} \tag{2c}$$

$$f_{uij} \leq M y_{uij} \qquad \forall u \in V; (i,j) \in A \tag{2d}$$

$$\sum_{(i,j) \in A} y_{uij} \leq n - 1 \qquad \forall u \in V \tag{2e}$$

$$y_{uij} + y_{uji} \leq x_{ij} \qquad \forall u \in V; \{i,j\} \in E \tag{2f}$$

$$\sum_{\{i,j\} \in E} x_{ij} = n - 1 \tag{2g}$$

$$x_{ij} \in \{0,1\} \qquad \forall \{i,j\} \in E \tag{2h}$$

$$y_{uij} \in \{0,1\}, f_{uij} \geq 0 \qquad \forall u \in V; (i,j) \in A \tag{2i}$$

The objective (2a) minimizes the sum of the communication cost of all commodities. Constraints (2b) and (2c) are the flow conservation constraints of every commodity at every vertex. Constraints (2d) link the y and the corresponding f variables using an appropriately defined big number $M > 0$. Constraints (2f) couple pairs of the y variables with the associated x variable. The cardinality constraints (2e) ensure that the flow of commodity u forms a tree. Finally, constraint (2g) together with (2f) ensures that the flow of each commodity can be expressed by the same spanning tree. The variables and their domains are given by (2h) and (2i).

4. Column-and-row generation and branch-and-price-and-cut

In this section, we first derive Dantzig-Wolfe reformulations of both models (1) and (2). Then, we present the column-and-row generation problems and algorithms for their solution. Finally, we discuss branching and cutting in the overall branch-and-price-and-cut algorithms.

4.1. Dantzig-Wolfe reformulation of the path-based formulation

The path-based formulation (1) can be decomposed by communication requirements $(s,t) \in R$ so that all f_{stij} variables for a fixed (s,t) form one block. For each $(s,t) \in R$, constraints (1b), (1c), and (1g) describe

an s-t-path in D. Using this decomposition into blocks, the Dantzig-Wolfe reformulation replaces the f_{stij} variables by variables representing s-t-paths, keeps the x_{ij} variables and the constraints (1e)–(1f) in the master problem, and reformulates (1d) with the new path variables. Let $w(s,t)$ be the number of s-t-paths in D so that they can be indexed by $p \in P(s,t) := \{1,2,\ldots,w(s,t)\}$. Hence, there is one continuous variable λ^p_{st} for $p \in P(s,t)$, i.e, for every possible s-t-path. The set of these paths is given by $\{(f^p_{stij}) : p \in P(s,t)\}$. Moreover, let c^p_{st} be the communication cost of the pth path. Then, the *integer master problem* of the path-based formulation (IMP-Path) is:

$$\min \quad \sum_{(s,t)\in R} \sum_{p\in P(s,t)} c^p_{st}\lambda^p_{st} \tag{3a}$$

$$\text{s.t.} \quad \sum_{p\in P(s,t)} (f^p_{stij} + f^p_{stji})\lambda^p_{st} \le x_{ij} \qquad \forall (s,t) \in R; \{i,j\} \in E \tag{3b}$$

$$\sum_{\{i,j\}\in E} x_{ij} = n-1 \tag{3c}$$

$$x_{ij} \in \{0,1\} \qquad \forall \{i,j\} \in E \tag{3d}$$

$$\sum_{p\in P(s,t)} \lambda^p_{st} = 1 \qquad \forall (s,t) \in R \tag{3e}$$

$$\lambda^p_{st} \ge 0 \qquad \forall (s,t) \in R; p \in P(s,t) \tag{3f}$$

The objective (3a) minimizes the communication costs of paths for all communication requirements $(s,t) \in R$. Constraints (3b)–(3d) are (identical) reformulations of (1d)–(1f). Finally, (3e) are the convexity constraints forcing the selection of exactly one s-t-path for each $(s,t) \in R$, while (3f) describe the domain of the path variables.

The reformulated model (3) is identical to the model presented by Fischetti *et al.* (2002) for the minimum routing-cost-tree problem except that the coefficients c^p_{st} here model communication costs for generally non-unary communication requirements.

4.2. Dantzig-Wolfe reformulation of the tree-based formulation

The tree-based formulation (2) can be decomposed by commodities $u \in V$ so that the variables y_{uij} and f_{uij} for a fixed u form a block. For each $u \in V$, the constraints (2b)–(2e) and (2i) describe a directed spanning tree rooted at vertex u together with flows satisfying the communication requirements for commodity u. Using this definition of blocks, the Dantzig-Wolfe reformulation replaces the variables y_{uij} and f_{uij} by variables representing the spanning trees with associated flows and reformulates constraints (2f)–(2h) in these new variables. Let $\ell(u)$ be the number of directed spanning trees rooted at u. Moreover, we define the index set $Q(u) := \{1,2,\ldots,\ell(u)\}$ so that the set of all spanning trees rooted at u with flows can be described by $\{(f^q_{uij}, y^q_{uij}) : q \in Q(u)\}$. Let, the communication cost of the qth spanning tree be c^q_u. The *integer master problem* of the tree-based formulation (IMP-Tree) is then:

$$\min \quad \sum_{u\in V} \sum_{q\in Q(u)} c^q_u\lambda^q_u \tag{4a}$$

$$\text{s.t.} \quad \sum_{q\in Q(u)} (y^q_{uij} + y^q_{uji})\lambda^q_u \le x_{ij} \qquad \forall u \in V; \{i,j\} \in E \tag{4b}$$

$$\sum_{\{i,j\}\in E} x_{ij} = n-1 \tag{4c}$$

$$x_{ij} \in \{0,1\} \qquad \{i,j\} \in E \tag{4d}$$

$$\sum_{q\in Q(u)} \lambda^q_u = 1 \qquad \forall u \in V \tag{4e}$$

$$\lambda^q_u \ge 0 \qquad \forall u \in V; q \in Q(u) \tag{4f}$$

The objective (4a) calls for the minimization of the overall communication costs of all commodities. Constraints (4b)–(4d) are (identical) reformulations of the constraints (2f)–(2h). The convexity constraints (4e) require the selection of exactly one spanning tree for each commodity. The domain of the spanning tree variables is given by (4f).

4.3. Combined column-and-row generation

For solving the mixed-integer programs (3) and (4) we employ a branch-and-price-and-cut algorithm (Lübbecke and Desrosiers, 2005; Desaulniers *et al.*, 2005), meaning that first integrality is relaxed, the respective linear relaxations are then solved with combined column-and-row generation algorithms, and integrality is finally established by integrating the procedure into a branch-and-bound scheme with an optional cutting-plane procedure.

In the following, the linear relaxations of formulations (3) and (4) are denoted as the *master programs* of the path-based and tree-based formulation, respectively. They do not contain the integer constraints (3d) and (4d). Moreover, the *restricted master problems* for the path-based formulation (RMP-Path) and the tree-based formulation (RMP-Tree) have a reduced set of variables and constraints. The RMP-Path contains a subset of the path variables λ^p_{st} as well as a subset of the $|E||R|$ coupling constraints (3b). RMP-Path is initialized with paths that are solutions to the s-t-shortest-path problems for all $(s,t) \in R$. Similarly, the RMP-Tree contains a reduced set of the tree variables λ^q_u as well as a subset of the $|E||V|$ coupling constraints (4b). We initialize RMP-Tree with trees that are solutions to the minimum spanning tree problem and to s-to-all shortest-path problems for all $s \in V$. For each tree, we add a column per commodity $u \in V$. The cost of such a column can be easily computed by solving a modified u-to-all shortest-path problem.

We start with the description of the combined column-and-row generation algorithms while branching and cutting is discussed later in Section 4.4.

4.3.1. Column Generation for the path-based reformulation

Let $\pi_{stij} \leq 0$ be the dual prices of the constraints (3b) of RMP-Path, and let $\mu_{st} \in \mathbb{R}$ be the dual prices of the convexity constraints (3e). There is a pricing problem for each pair $(s,t) \in R$ asking for a negative reduced-cost s-t-path:

$$
\begin{aligned}
\min \quad & \sum_{(i,j)\in A} (r_{st}c_{ij} - \pi_{stij}) f_{stij} - \mu_{st} \\
\text{s.t.} \quad & \sum_{(s,j)\in\delta^+(s)} f_{stsj} = 1 \\
& \sum_{(i,j)\in\delta^+(i)} f_{stij} - \sum_{(j,i)\in\delta^-(i)} f_{stji} = 0 && \forall i \in V \setminus \{s,t\} \\
& f_{stij} \geq 0 && \forall (i,j) \in A
\end{aligned}
$$

This is the network-flow formulation of the s-t-shortest-path problem on $D = (V, A)$ with arc costs $\tilde{c}_{ij} := (r_{st}c_{ij} - \pi_{stij})$. Due to the non-negativity of the coefficients $\tilde{c}_{ij}$, the pricing problem can be easily solved with Dijkstra's algorithm (Dijkstra, 1959). We can generate more than one negative reduced-cost column per iteration by using the following modification. Every time vertex t is labeled, we check if the corresponding path has negative reduced cost and if so, we add the corresponding variable to RMP-Path.

4.3.2. Column Generation for the tree-based reformulation

Let $\pi_{uij} \leq 0$ be the dual prices of the constraints (4b) of the RMP-Tree, and let $\mu_u \in \mathbb{R}$ be the dual prices of the convexity constraints (4e). There is one pricing problem for each commodity $u \in V$ asking for

a negative-reduced cost spanning tree with flows:

$$\min \quad \sum_{(i,j)\in A} c_{ij} f_{uij} - \sum_{(i,j)\in A} \pi_{uij} y_{uij} - \mu_u \tag{5a}$$

$$\text{s.t.} \quad \sum_{(u,j)\in\delta^+(u)} f_{uuj} = \sum_{h\in V\setminus\{u\}} r_{uh} \tag{5b}$$

$$\sum_{(h,j)\in\delta^+(h)} f_{uhj} - \sum_{(i,h)\in\delta^-(h)} f_{uih} = -r_{uh} \qquad \forall h \in V \setminus \{u\} \tag{5c}$$

$$f_{uij} \le M y_{uij} \qquad \forall (i,j) \in A \tag{5d}$$

$$\sum_{(i,j)\in A} y_{uij} \le n - 1 \tag{5e}$$

$$y_{uij} \in \{0,1\}, f_{uij} \ge 0 \qquad \forall (i,j) \in A \tag{5f}$$

This problem is a *fixed-cost network flow problem* (FCNFP). Its objective (5a) is to minimize the sum of fixed costs and flow costs. The decision that an arc (i,j) is used, indicated by $y_{uij} = 1$, results in a fixed cost $-\pi_{uij} \ge 0$, while flow along that arc, given by the value of f_{uij}, imposes flow costs $c_{ij} f_{uij}$. Constraints (5b) and (5c) guarantee that the supply of commodity u flows from vertex u to every other vertex. Consistency between flow and arc variables is ensured by (5d) using big-M constant. Constraints (5e) limit the number of arcs in the spanning tree and (5f) describe the domains of the variables.

The pricing problem (5) is guaranteed to generate valid spanning trees only if all vertices $i \in V \setminus \{u\}$ have a positive demand $r_{ui} > 0$. Otherwise, less than $|V| - 1$ arcs may be needed to reach all positive-demand vertices possibly creating undirected cycles. The following inequalities can be added to exclude such non-tree solutions:

$$\sum_{(h,i)\in\delta^-(i)} y_{uhi} \le 1 \qquad\qquad i \in V \setminus \{u\}$$

The FCNFP is $\mathcal{NP}$-hard as it generalizes the Steiner tree problem (Garey and Johnson, 1979). Many heuristic and exact methods have been developed for the FCNFP.

Heuristic solution of the FCNFP. We decided to apply a *dynamic slope scaling procedure* (DSSP, Kim and Pardalos, 1999) to heuristically solve the pricing problem. DSSP completely removes the binary variables from the model by approximating costs of an optimal FCNFP solution with new variable costs $\widehat{c}_{uij}$ that simultaneously reflect the original fixed costs $-\pi_{uij}$ and variable costs c_{ij}. The coefficients $\widehat{c}_{uij}$ are approximated by solving successive u-to-many *minimum-cost flow* (MCF) problems and iteratively updating the coefficients depending on the computed values. The u-to-many MCF problems can be solved with Dijkstra's algorithm (Dijkstra, 1959). Each solution provides a feasible solution of the FCNFP and therefore each iteration may generate a negative reduced-cost tree with flows.

Exact solution of the FCNFP. We use the branch-and-cut algorithm of Ortega and Wolsey (2003) for the exact solution of FCNFP. Ortega and Wolsey (2003) introduce *dicut inequalities* and their variants together with several heuristics for their separation. Standard branching on the y_{uij} variables is applied. Every time the branch-and-cut finds a feasible integer solution, we add the corresponding variable to RMP-Tree if it has negative reduced cost.

Moreover, it can occur that a tree solution contains leaf vertices with demand zero. (A leaf is a vertex with degree one.) We modify such a solution by pruning zero-demand leaf vertices (note that (5e) is an inequality). This avoids the generation of equivalent columns with equal cost. Pruning is also used in some heuristics for the Steiner tree problem (Voß, 1992).

Overall pricing strategy. In each iteration of the RMP-Tree, one pricing problem per commodity results. As they are $\mathcal{NP}$-hard, solving all of them requires a large amount of computation time. However, it suffices to terminate pricing when one negative reduced-cost solution is found. This method is known as partial column generation and was previously applied, e.g., to large-scale aircrew rostering problems (Gamache *et al.*, 1999). Our implementation of partial column generation computes priorities for the commodities $u \in V$ to determine the order in which the pricing problems in each iteration are examined. The priorities are updated depending on the number of columns generated in the previous iterations.

Moreover, a tree computed for a specific commodity u may at the same time be a solution to the pricing problem of every other commodity. We exploit this fact using the following heuristic: First, non-spanning trees are extended to spanning trees using the original edge costs c. This can be done in several ways (we check all): Use a minimum spanning-tree algorithm starting with the given tree. Alternatively, for each $j \in V \setminus \{u\}$, shrink the tree into a single super-vertex and compute a shortest-path tree rooted at vertex j. Second, for each commodity $i \in V \setminus \{u\}$ prune zero-demand leaf vertices as explained before. As a by-product of this procedure, we obtain an upper bound for the OCSTP by simply summing up the communication costs for all commodities.

4.3.3. Row generation

RMP-Path and RMP-Tree are typically highly degenerate linear programs. The reason is that the path-based formulation (3) consists of $\mathcal{O}(|E||R|)$ constraints while an integer basic solution comprises no more than $|R| + (|V| - 1)$ positive variables. The tree-based formulation has $\mathcal{O}(|E||V|)$ constraints, but an integer basic solution consists of no more than $|V| + (|V| - 1)$ positive variables. To overcome degeneracy problems, we generate the coupling constraints (3b) and (4b) of RMP-Path and RMP-Tree dynamically. Fischetti *et al.* (2002) use a similar row-generation method to solve the minimum routing-cost-tree problem with a path-based formulation.

In both reformulations, we initialize the RMP with no coupling constraints. For each edge $\{i, j\} \in E$, violated coupling constraints (3b) and (4b) can be identified by inspection. We add violated coupling constraints only if there are no more negative reduced-cost columns in the current RMP. Hence, in our implementation of combined row-and-column generation, a valid lower bound for the OCSTP results whenever no negative reduced-cost columns are found. In Section 5, we will quantify the positive impact of a dynamic row generation on the basis of a computational study.

4.4. Branching and cutting

Branching on the edge variables $x_{ij} \in \{0, 1\}$ ensures integrality in both reformulations (3) and (4). We use the following variable-selection strategy. Among all edges $\{i, j\}$ with fractional value $\bar{x}_{ij}$ we choose one closest to 0.7, i.e., $|\bar{x}_{ij} - 0.7|$ minimum. Pretests have revealed that such a rule is often able to well balance the branch-and-bound tree. Ties are broken by preferring edges $\{i, j\}$ with higher cost c_{ij}.

Solving the path-based formulation (3) requires massive branching. We therefore decided to use strong branching: Every time the branching procedure is called, we choose up to five candidate edges with fractional values $\bar{x}_{ij}$ closest to 0.7. The two son nodes of each candidate edge $\{i, j\}$ are solved and their lower bounds $LB^1_{\{i,j\}}$ and $LB^2_{\{i,j\}}$ are stored. Among the candidate edges, we choose the one that maximizes $\min\{LB^1_{\{i,j\}}, LB^2_{\{i,j\}}\}$.

In order to strengthen the master program bounds for the path-based formulation (3), we use the following *cycle-elimination inequalities* (Padberg and Wolsey, 1983):

$$\sum_{\{i,j\} \in E(S)} x_{ij} \leq |S| - 1 \qquad \forall \varnothing \neq S \subsetneq V$$

Herein, $E(S)$ is the set of all edges with both endpoints in S. Violated cycle-elimination inequalities can be separated solving maximum-flow problems (see Padberg and Wolsey, 1983). These inequalities are valid but redundant for the tree-based formulation (4).

5. Computational results

This section presents the computational results for both branch-and-price-and-cut algorithms for the OCSTP. All computations are performed on a standard PC with an Intel(R) Core(TM) i7-2600 at 3.4 GHz processor with 16 GB of main memory. Algorithms are coded in C++ and compiled in release mode with MS-Visual Studio 2010. The callable library of CPLEX 12.5 is used to iteratively re-optimize both RMPs in the combined column-and-row-generation algorithm as well as to solve the FCNFP pricing problems of the tree-based formulation by branch-and-cut. We test our algorithms on the set of benchmark instances from Contreras *et al.* (2010). There are 20 instances divided into five groups each containing four instances with 10, 20, 30, 40, and 50 vertices, respectively. Distances and demand are uniformly distributed in [0,100] and approximately 50 % of the vertex pairs have a positive communication requirement.

For all runs, we set a hard time limit of 7,200 seconds. Pre-tests have shown that the following strategies are beneficial:

(1) Define the communication requirements in the most asymmetric way in both formulations.

For the path-based formulation:

(2) Use the primal simplex algorithm for reoptimizing RMP-Path.

(3) Do not apply partial column generation. Shortest-path computations using the Dijkstra algorithm are already very fast.

For the tree-based formulation:

(4) Use the barrier optimizer of CPLEX to reoptimize RMP-Tree. The dual prices are stabilized and oscillate less compared to using the primal or dual simplex algorithm.

(5) Use partial column generation and terminate pricing as soon as a negative reduced-cost column is found. Solving the pricing problem exactly is by far the most time-consuming component.

(6) Reuse negative reduced-cost trees of one commodity for all other commodities and use them to compute upper bounds.

Tables 1 and 2 show the impact of dynamic row generation for both formulations (3) and (4). The table entries have the following meaning:

$\#rows$	the average number of rows present at the last RMP solved
T_{RMP}	the average time for solving one iteration of RMP-Path/RMP-Tree in seconds
gap	the average gap in percent between the best lower and upper bound found
T	the average time in seconds needed to solve the instance to optimality
$\#solved$	the number of instances solved to optimality

	all rows					dynamic row generation				
Group	$\#rows$	T_{RMP}	gap	T	$\#solved$	$\#rows$	T_{RMP}	gap	T	$\#solved$
`Raidl10`	1 094	0.01	0.00	0.03	4/4	185	0.01	0.00	0.03	4/4
`Raidl20`	16 429	0.07	0.00	13.11	4/4	1 516	0.01	0.00	4.22	4/4
`Raidl30`	106 281	2.88	0.00	1 841.34	4/4	6 520	0.11	0.00	148.94	4/4
`Raidl40`	308 507	19.64	5.29	6 568.79	1/4	12 915	0.88	4.03	5 428.75	1/4
`Raidl50`	773 616	42.86	8.12	7 200.00	0/4	22 700	2.22	6.54	7 200.00	0/4
All	241 185	13.09	2.68	3 124.65	13/20	8 767	0.64	2.11	2 556.39	13/20

Table 1: Impact of applying dynamic row generation in the path-based reformulation (3)

Table 1 quantifies the benefit of using dynamic row generation in the path-based reformulation. On average only 3.6 % of all coupling constraints are present in the RMP resulting in a reduction of the computation time from 13 to 0.64 seconds. The average gap is reduced by 0.5 %.

	all rows					dynamic row generation				
Group	$\#rows$	T_{RMP}	gap	T	$\#solved$	$\#rows$	T_{RMP}	gap	T	$\#solved$
`Raidl10`	312	0.01	0.00	8.79	4/4	120	0.01	0.00	24.93	4/4
`Raidl20`	2 962	0.12	0.00	927.59	4/4	635	0.04	0.00	637.94	4/4
`Raidl30`	11 446	2.55	8.32	7 200.00	0/4	1 380	0.28	2.45	7 200.00	0/4
`Raidl40`	26 555	6.59	13.25	7 200.00	0/4	2 160	0.78	6.43	7 200.00	0/4
`Raidl50`	53 945	22.93	38.42	7 200.00	0/4	3 025	2.19	11.58	7 200.00	0/4
All	19 044	6.44	12.00	4 507.28	8/20	1464	0.66	4.09	4 452.58	8/20

Table 2: Impact of applying dynamic row generation in the tree-based reformulation (4)

In the tree-based reformulation, the impact of row generation is smaller as can be seen from Table 2. On average there are only 7.7 % of all coupling constraints present when applying dynamic row generation and the average computation time for solving RMP-Tree decreases by almost factor 10 (from 6.44 to 0.66 seconds). Therefore, the overall solution time also decreases, but since most instances cannot be solved to proven optimality, the reduction in average computation time is small. The average gap decreases by around 8 %. The impact of row generation increases significantly for larger instances in both reformulations.

Table 3 compares the solutions of the two reformulations (3) and (4) when dynamic row generation is applied. The table entries have the following meaning:

gap_{root}	the average gap at the root node in percent
T_{root}	the time to solve the root node in seconds
gap_{closed}	the percentage of the root gap closed at the end of the optimization
$\#N$	the number of branch-and-bound nodes solved
$\#PP$	the number of pricing problems solved

	path-based formulation (3)					tree-based formulation (4)				
Group	gap_{root}	T_{root}	gap_{closed}	$\#N$	$\#PP$	gap_{root}	T_{root}	gap_{closed}	$\#N$	$\#PP$
`Raidl10`	0.27	0.02	100.00	3	795	0.07	4.90	100.00	2	774
`Raidl20`	2.38	0.20	100.00	46	21 939	0.68	241.28	100.00	10	1 941
`Raidl30`	2.41	2.75	100.00	108	210 391	2.45*	7 200.00	0.00	0	3 233
`Raidl40`	6.77	11.45	55.45	921	1 547 252	6.43*	7 200.00	0.00	0	2 861
`Raidl50`	8.66	56.53	27.58	234	1 589 344	11.58*	7 200.00	0.00	0	2 890
All	4	14.19	76.61	262	673 944	4.24	4 369.23	40.00	2	2 340

Table 3: Comparison of both reformulations solved with dynamic row generation

The tree-based formulation was not able to solve the root node of a single instance when the number of vertices exceeds 20. The values marked with * in the table show the gap between the best Lagrangian lower bound and the best upper bound found at the root node. As clear from theory, the root lower bound of the tree-based reformulation dominates the root lower bound of the path-based formulation, but the solution times of the tree-based formulation are on average more than 300 times larger. Therefore, the path-based formulation is able to close 76 % of the root gap by branching and cutting while the tree-based formulation is not even able to solve the root node for instances with 30 or more vertices. The last two columns indicate that many more shortest-path pricing problems can be solved within the time limit compared to FCNFP pricing problems. In turn, more branch-and-bound nodes can be solved in the path-based reformulation leading to smaller optimality gaps at the end.

Table 4 compares all four formulations, compact and extensive as well as path-based and tree-based, respectively. The table entries have the following meaning: *gap* denotes the gap between the computed lower bound and the best known solution and T the solution time in seconds. The best known solution values are taken from the work of Contreras *et al.* (2010).

	compact path-based		compact tree-based		extensive path-based		extensive tree-based	
Instance	*gap*	*T*	*gap*	*T*	*gap*	*T*	*gap*	*T*
Raidl10a	0.00	0.09	0.00	1.22	0.00	0.07	0.00	83.47
Raidl10b	0.00	0.06	0.00	0.25	0.00	0.02	0.00	1.91
Raidl10c	0.00	0.09	0.00	0.11	0.00	0.02	0.00	7.83
Raidl10d	0.00	0.13	0.00	0.27	0.00	0.05	0.00	6.52
Raidl20a	0.00	4.73	0.01	3 586.83	0.00	4.19	0.00	1 671.82
Raidl20b	0.00	1.89	0.01	885.16	0.00	2.62	0.00	413.58
Raidl20c	0.00	4.12	0.01	2 569.18	0.00	3.40	0.00	386.28
Raidl20d	0.00	1.50	0.00	30.03	0.00	0.27	0.00	80.11
Raidl30a	0.00	161.92	3.98	7 200.00	0.00	101.90	0.54	7 200.00
Raidl30b	0.00	157.36	3.76	7 200.00	0.00	162.14	1.78	7 200.00
Raidl30c	0.00	383.13	3.57	7 200.00	0.00	111.47	1.69	7 200.00
Raidl30d	0.00	221.32	3.69	7 200.00	0.00	106.29	1.19	7 200.00
Raidl40a	0.00	485.51	2.77	7 200.00	0.00	115.08	1.12	7 200.00
Raidl40b	2.13	7 200.00	7.27	7 200.00	0.16	7 200.00	4.33	7 200.00
Raidl40c	2.87	7 200.00	7.32	7 200.00	0.47	7 200.00	4.46	7 200.00
Raidl40d	8.31	7 200.00	11.51	7 200.00	4.65	7 200.00	9.71	7 200.00
Raidl50a	-	-	8.39	7 200.00	1.50	7 200.00	7.23	7 200.00
Raidl50b	-	-	11.36	7 200.00	4.61	7 200.00	10.66	7 200.00
Raidl50c	-	-	13.40	7 200.00	5.62	7 200.00	11.79	7 200.00
Raidl50d	-	-	11.32	7 200.00	4.68	7 200.00	10.23	7 200.00
Solved		13/20		8/20		13/20		8/20

Table 4: Comparison of compact and extensive formulations

The compact path-based formulation is not able to solve instances with 50 vertices due to memory limitations. The results show that the extensive formulations produce consistently better results than the compact formulations. Moreover, the combined column-and-row generation algorithm of the path-based formulation (3) is superior to the one for the tree-based formulation (4). Both compact formulations (1) and (2) solved by CPLEX are clearly inferior.

6. Conclusions

This paper has presented a path-based and a tree-based extensive formulation of the optimum communication spanning tree problem. Due to their large number of variables and constraints, both Dantzig-Wolfe reformulations are solved with combined column-and-row generation. Branch-and-bound finally ensures integrality of solutions.

Our findings concerning a reasonable design of combined column-and-row-generation algorithms are the following: Dynamic row generation significantly accelerates the solution process. Partial pricing is advantageous for the tree-based but not for the path-based reformulation. For the former pricing problem, partial pricing includes the use of FCNFP heuristics as well as prioritizing the pricing problems (there is one for each vertex/commodity) and stopping pricing whenever a negative reduced cost tree with flows is found. Other important components of the algorithm for the tree-based formulation are the computation of upper bounds from trees computed in the pricing problems and the exchange of trees between pricing problems of different commodities.

In summary, the combined column-and-row-generation algorithm for the path-based reformulation is superior on the benchmark instances although its linear relaxation bound is dominated by the one of the tree-based formulation. The decisive point is that the speed, in which simple shortest-path pricing problems can be solved in the path-based approach, overcompensates the weaker linear relaxation. The tree-based combined column-and-row-generation approach suffers too much from the time-consuming FCNFP pricing problems, which are finally solved by branch-and-cut. Overall, the path-based formulation is able to solve instances with up to 40 vertices to optimality, while increasing the number of vertices leads to optimality gaps that can easily reach 5 % and more.

References

Ahuja, R. and Murty, V. (1987). Exact and heuristic algorithms for the optimum communication spanning tree problem. *Transportation Science*, **21**(3), 163–170.

Contreras, I. and Fernández, E. (2012). General network design: A unified view of combined location and network design problems. *European Journal of Operational Research*, **219**(3), 680–697.

Contreras, I., Fernández, E., and Marín, A. (2010). Lagrangean bounds for the optimum communication spanning tree problem. *Top*, **18**(1), 140–157.

Desaulniers, G., Desrosiers, J., and Solomon, M., editors (2005). *Column Generation*. Springer, New York, NY.

Dijkstra, E. (1959). A note on two problems in connexion with graphs. *Numerische Mathematik*, **1**, 269–271.

Fernández, E., Luna-Mota, C., Hildenbrandt, A., Reinelt, G., and Wiesberg, S. (2013). A flow formulation for the optimum communication spanning tree. *Electronic Notes in Discrete Mathematics*, **41**(0), 85–92.

Fischer, T. and Merz, P. (2007). A memetic algorithm for the optimum communication spanning tree problem. In *Hybrid Metaheuristics*, pages 170–184. Springer.

Fischetti, M., Lancia, G., and Serafini, P. (2002). Exact algorithms for minimum routing cost trees. *Networks*, **39**(3), 161–173.

Gamache, M., Soumis, F., and Marquis, G. (1999). A column generation approach for large-scale aircrew rostering problems. *Operations Research*, **47**(2), 247–263.

Garey, M. and Johnson, D. (1979). *Computers and Intractability: A guide to the Theory of NP-completness*. Freeman, New York.

Hu, T. C. (1974). Optimum communication spanning trees. *SIAM Journal on Computing*, **3**(3), 188–195.

Johnson, D. S., Lenstra, J. K., and Rinnooy Kan, A. (1978). The complexity of the network design problem. *Networks*, **8**(4), 279–285.

Kim, D. and Pardalos, P. M. (1999). A solution approach to the fixed charge network flow problem using a dynamic slope scaling procedure. *Operations Research Letters*, **24**(4), 195–203.

Lübbecke, M. E. and Desrosiers, J. (2005). Selected topics in column generation. *Operations Research*, **53**(6), 1007–1023.

Ortega, F. and Wolsey, L. A. (2003). A branch-and-cut algorithm for the single-commodity, uncapacitated, fixed-charge network flow problem. *Networks*, **41**(3), 143–158.

Padberg, M. W. and Wolsey, L. A. (1983). Trees and cuts. In C. Berge, D. Bresson, P. Camion, J. Maurras, and F. Sterboul, editors, *Combinatorial Mathematics Proceedings of the International Colloquium on Graph Theory and Combinatorics*, volume 75 of *North-Holland Mathematics Studies*, pages 511–517. North-Holland.

Peleg, D. and Reshef, E. (1998). Deterministic polylog approximation for minimum communication spanning trees. In K. Larsen, S. Skyum, and G. Winskel, editors, *Automata, Languages and Programming*, volume 1443 of *Lecture Notes in Computer Science*, pages 670–681. Springer Berlin Heidelberg.

Rothlauf, F. (2009). On optimal solutions for the optimal communication spanning tree problem. *Operations Research*, **57**(2), 413–425.

Sharma, P. (2006). Algorithms for the optimum communication spanning tree problem. *Annals of Operations Research*, **143**(1), 203–209.

Steitz, W. and Rothlauf, F. (2012). Using penalties instead of rewards: Solving OCST problems with guided local search. *Swarm and Evolutionary Computation*, **3**, 46–53.

Voß, S. (1992). Steiner's problem in graphs: heuristic methods. *Discrete Applied Mathematics*, **40**(1), 45 – 72.

Wu, B. Y., Chao, K.-M., and Tang, C. Y. (2000). Approximation algorithms for some optimum communication spanning tree problems. *Discrete Applied Mathematics*, **102**(3), 245 – 266.