A Group-Strategyproof Mechanism for Steiner Forests

J. Koenemann, S. Leonardi, G. Schaefer.

2004
A Group-Strategyproof Mechanism for Steiner Forests*

Jochen Könemann† Stefano Leonardi‡ Guido Schäfer‡

Abstract

In this paper we design an approximately budget-balanced and group-strategyproof cost-sharing mechanism for the Steiner forest game. An instance of this game consists of an undirected graph $G = (V, E)$, non-negative costs $c_e$ for all edges $e \in E$, and a set $R \subseteq V \times V$ of $k$ terminal pairs. Each terminal pair $(s, t) \in R$ is associated with an agent that wishes to establish a connection between nodes $s$ and $t$ in the underlying network. A feasible solution is a forest $F$ that contains an $s, t$-path for each connection request $(s, t) \in R$.

Previously, Jain and Vazirani [4] gave a 2-approximate budget-balanced and group-strategyproof cost-sharing mechanism for the Steiner tree game — a special case of the game considered here. Such a result for Steiner forest games has proved to be elusive so far, in stark contrast to the well known primal-dual $(2 - 1/k)$-approximate algorithms [1, 2] for the problem.

The cost-sharing method presented in this paper is 2-approximate budget-balanced. Our algorithm is an original extension of known primal-dual methods for Steiner forests [1].

An interesting byproduct of the work in this paper is that our Steiner forest algorithm is $(2 - 1/k)$-approximate despite the fact that the forest computed by our method is usually costlier than those computed by known primal-dual algorithms. In fact the dual solution computed by our algorithm is infeasible for the classical undirected cut relaxation for Steiner forests but we can still prove that its total value is at most the cost of a minimum-cost Steiner forest for the given instance.

Our algorithm gives rise to a new undirected cut dual for Steiner forests which we coin the lifted-cut dual. We show that this new dual is strictly stronger than the standard undirected cut dual for Steiner forests. Moreover, our Steiner forest algorithm produces a feasible solution for it.

1 Introduction

In this paper we consider the problem of designing cost-sharing methods that are approximately budget-balanced and cross-monotonic.

Consider a set $R$ of potential agents (or customers, players) that want to receive a common service, e.g., being connected to a network infrastructure. A cost-sharing method $\xi$ is an algorithm that, given any subset $Q \subseteq R$ of agents, computes a solution to service $Q$ and for each $j \in Q$ determines a non-negative cost-share $\xi_Q(j)$. The task is to devise a cost-sharing method $\xi$ that is $\alpha$-approximate budget-balanced and cross-monotonic:

---

*Partially supported by the EU within the 6th Framework Programme under contract 001907 (DELIS).
†Department of Combinatorics and Optimization, University of Waterloo, 200 University Avenue West, Waterloo, ON N2L 3G1, Canada. Email: jochen@uwaterloo.ca. This work was done while being on leave at the Dipartimento di Informatica e Sistemistica at Università di Roma “La Sapienza”, Italy.
‡Dipartimento di Informatica e Sistemistica, Università di Roma “La Sapienza”, Via Salaria 113, 00198 Roma, Italy. Email: {leon,schaefer}@dis.uniroma1.it.
\(\alpha\)-Approximate Budget-Balance: The sum of the cost-shares recovers at least the total cost \(c(Q)\) of the computed solution. Moreover, the sum of the cost-shares is at most \(\alpha \geq 1\) times the cost \(\text{opt}_Q\) of an optimum solution to service \(Q\). That is,
\[
c(Q) \leq \sum_{j \in Q} \xi_Q(j) \leq \alpha \cdot \text{opt}_Q.
\]

Cross-Monotonicity: The cost-share of each individual agent never decreases as the set of agents shrinks, i.e.,
\[
\forall Q' \subseteq Q, \forall j \in Q', \quad \xi_{Q'}(j) \geq \xi_Q(j).
\]

A cost-sharing method is budget-balanced if \(\alpha = 1\). Obtaining a budget-balanced cost-sharing method is computationally intractable if the underlying problem is NP-hard or if we additionally require cross-monotonicity (see for instance [3]).

Subsequently, we call a cost-sharing method that is \(\alpha\)-approximate budget-balanced and cross-monotonic an \(\alpha\)-approximate cross-monotonic cost-sharing method for short.

In this paper we consider the problem of devising an approximately cross-monotonic cost-sharing method for the Steiner forest problem. In this problem, we are given an undirected graph \(G = (V, E)\), a non-negative cost function \(c : E \rightarrow \mathbb{R}^+\) on the edges of \(G\), and a set of \(k > 0\) terminal pairs \(R = \{(s_1, t_1), \ldots, (s_k, t_k)\} \subseteq V \times V\). Each terminal pair \((s_j, t_j)\), \(1 \leq j \leq k\), is associated with an autonomous agent that wants to establish a connection between nodes \(s_j\) and \(t_j\). A feasible solution for terminal set \(R\) is a forest \(F \subseteq E\) such that nodes \(s_j\) and \(t_j\) are in the same tree of \(F\) for all \(1 \leq j \leq k\). The objective is to find a feasible solution of smallest total cost.

The main reason for being interested in the design of cross-monotonic cost-sharing methods lies in a result due to Moulin and Shenker [5]: Any (approximately) budget-balanced cross-monotonic cost-sharing method can be turned into an (approximately) budget-balanced group-strategyproof mechanism, i.e., a mechanism that encourages agents and coalitions of agents to reveal their true utility for receiving the service. (A more detailed description of the mechanism design problem that we consider is given below.)

**Related Work.** Though the designing of (approximately) cross-monotonic cost-sharing methods has recently received a growing attention in the computer science literature, such methods are known only for very few combinatorial problems: Moulin and Shenker [5] gave a cross-monotonic cost-sharing method for problems whose optimal cost function is a submodular function of the set \(Q\). However, this condition does not hold for many important network design problems such as Steiner trees and facility location. Jain and Vazirani [4] presented a cross-monotonic cost-sharing method for the minimum spanning tree game and therefore a 2-approximate cross-monotonic cost-sharing method for the Steiner tree game. Pál and Tardos [6] developed a 3-approximate cross-monotonic cost-sharing method for the facility location problem and a 15-approximate cross-monotonic cost-sharing method for the single-source rent-or-buy network design problem.

In most of the methods that were proposed so far to obtain (approximately) cross-monotonic cost-sharing methods, the cost-shares are closely related to a feasible dual solution generated by the algorithm and therefore approximate budget-balance is an immediate consequence of the approximation guarantee achieved by the algorithm.

On the other hand, combinatorial problems that are well-behaved with respect to their approximability may prove hard when looking for approximately cross-monotonic cost-sharing methods. In a recent paper [3], Immorlica, Mahdian, and Mirrokni provide lower bounds on the budget-balance factor \(\alpha\) of
cross-monotonic cost-sharing methods for several problems. Among other results they prove (maybe most surprisingly) lower bounds of $\Omega(n)$ and $\Omega(n^{1/3})$ for the budget-balance factor of the set cover and the vertex cover problems, respectively. Observe that these lower bounds are achieved by using cross-monotonicity only.

**Our Contribution.** While the 2-approximation achieved by primal-dual algorithms for the Steiner tree problem is matched by a 2-approximate cross-monotonic cost-sharing method [4], a similar result for the Steiner forest problem has proved to be elusive so far. This contrasts the optimization version of the problem where primal-dual $(2 - 1/k)$-approximation algorithms for both problems exist [1, 2].

In this paper we present a cross-monotonic cost-sharing method that is 2-approximate budget-balanced. Our algorithm is an original extension of the classical methods for Steiner forests [1, 2]. The Steiner forests produced by our algorithm are generally more expensive than those computed by the algorithms in [1, 2], since a terminal pair $(s_j, t_j)$ will continue to contribute to construct the forest even after $s_j$ and $t_j$ are connected.

An interesting byproduct of the work in this paper is that our Steiner forest algorithm is $(2 - 1/k)$-approximate despite the fact that the forest computed by our method is usually costlier than those computed by known primal-dual algorithms in [1, 2]. In fact the dual solution computed by our algorithm is infeasible for the classical undirected cut relaxation for Steiner forests but we can still prove that its total value is at most the cost of a minimum-cost Steiner forest for the given instance.

We end our paper by presenting a new LP relaxation for Steiner forests, the *lifted-cut dual*. We prove that this dual is strictly stronger than the well-studied undirected cut dual for Steiner forests. Moreover, the dual solution computed by our algorithm is feasible for the lifted-cut dual but may not be feasible for the usual undirected cut formulation. This raises the dazzling question of the existence of better primal-dual algorithms for Steiner trees and forests.

**Mechanism Design Problem.** The mechanism design problem that we consider can be described as follows. Consider a service provider whose set of potential agents (or customers) is $R$. Each agent $j$ in $R$ has a *utility* $u_j$ which corresponds to the maximum prize $j$ is willing to pay for the service. Moreover, each agent $j$ makes a *bid* $b_j$ for receiving the service. A *cost-sharing mechanism* is an algorithm that, based on the bids $\{b_j\}_{j \in R}$, (i) determines a set $Q \subseteq R$ of agents that receive the service, (ii) computes a solution to service the agents in $Q$, and (iii) for each $j \in Q$ fixes a *prize* $x_j$ that $j$ has to pay for receiving the service.

We define the *benefit* of an agent $j$ to be $u_j - x_j$ if $j \in Q$, and zero otherwise. We assume that each agent is selfish and therefore may lie about the prize she is willing to pay so as to maximize her benefit. The task is to design a cost-sharing mechanism that encourages agents to bid their true utility: No agent or group of agents should be able to benefit from lying about their utilities. A cost-sharing mechanism is *strategyproof* if the dominant strategy of each agent is to bid her utility; it is said to be *group-strategyproof* if the same holds even if agents collude.

Moulin and Shenker [5] showed that, given a cross-monotonic cost-sharing method $\xi$ for the underlying problem, the following cost-sharing mechanism is group-strategyproof: Initially, let $Q = R$. If for each agent $j \in Q$ the cost share $\xi_Q(j)$ is less than or equal to her bid $b_j$, stop. Otherwise, remove from $Q$ all agents whose cost shares are larger than their bids, and repeat. Jain and Vazirani [4] later extended this framework to approximately budget-balanced cross-monotonic cost-sharing methods.
Organization of the Paper. Our algorithm is based on a primal-dual algorithm for Steiner forests. We introduce the key ideas underlying this algorithm in the following section. Subsequently, we state our cross-monotonic algorithm for the Steiner forest problem in Section 3 and we analyze it in Sections 4 and 5. We comment on the approximation guarantee of our algorithm in Section 6. Finally, we present the new lifted-cut dual for Steiner trees in Section 7.

2 A primal-dual Steiner forest algorithm

We review the algorithm of Agrawal, Klein, and Ravi [1] and subsequently refer to it as AKR. AKR is a primal-dual algorithm. This means that the algorithm constructs both a feasible and integral primal and a feasible dual solution for a linear programming formulation of the Steiner forest problem and its dual, respectively. A standard integer programming formulation for the Steiner forest problem has a binary variable $x_e$ for all edges $e \in E$. Variable $x_e$ has value 1 if edge $e$ is part of the resulting forest. We let $\mathcal{U}$ contain exactly those subsets $U$ of $V$ that separate at least one terminal pair in $R$. In other words, $U \in \mathcal{U}$ iff there is $(s,t) \in R$ with $|\{s,t\} \cap U| = 1$.

For a subset $U$ of the nodes we also let $\delta(U)$ denote the set of those edges that have exactly one endpoint in $U$. We then obtain the following integer linear programming formulation for the Steiner forest problem:

$$\begin{align*}
\min & \sum_{e \in E} c_e \cdot x_e \\
\text{s.t.} & \sum_{e \in \delta(U)} x_e \geq 1 \quad \forall U \in \mathcal{U} \\
& x \text{ integer}
\end{align*}$$

The linear program dual of the LP-relaxation $(LP)$ of $(IP)$ has a variable $y_U$ for all node sets $U \in \mathcal{U}$. There is a constraint for each edge $e \in E$ that limits the total dual assigned to sets $U \in \mathcal{U}$ that contain exactly one endpoint of $e$ to be at most $c_e$.

$$\begin{align*}
\max & \sum_{U \in \mathcal{U}} y_U \\
\text{s.t.} & \sum_{U \in \mathcal{U}, e \in \delta(U)} y_U \leq c_e \quad \forall e \in E \\
& y \geq 0
\end{align*}$$

Algorithm AKR constructs a primal solution for $(LP)$ and a dual solution for $(D)$. The algorithm has two goals:

**Compute a feasible solution for the given Steiner forest instance.** The algorithm reduces the degree of infeasibility as it progresses.

**Create a dual feasible packing of sets of largest possible total value.** The algorithm raises dual variables of certain subsets of nodes at all times. The final dual solution is going to be maximal in the sense that no single set can be raised without violating a constraint of type (1).

We think of an execution of algorithm AKR as a process over time and let $x^t$ and $y^t$ be the primal incidence vector and feasible dual solution at time $t$. We also use $F^t$ to denote the forest corresponding to
x. Initially, we let $x^0_e = 0$ for all $e \in E$ and $y^0_U = 0$ for $U \in \mathcal{U}$. In the following we say that an edge $e \in E$ is tight if the corresponding constraint (1) holds with equality.

Assume that the forest $F'$ at time $t$ is infeasible. We use $\tilde{F}^t$ to denote the subgraph of $G$ that is induced by the tight edges for dual $y^t$. A connected component $U$ of $\tilde{F}^t$ is active iff $U$ separates at least one terminal pair, i.e., iff $U \in \mathcal{U}$. Let $\mathcal{A}^t$ be the set of all active connected components of $\tilde{F}^t$ at time $t$. AKR raises the dual variables for all sets in $\mathcal{A}^t$ uniformly at all times $t \geq 0$.

Suppose now that two active connected components $U_1$ and $U_2$ collide at time $t$ in the execution of AKR. In other words, there are terminals $u_1 \in U_1$ and $u_2 \in U_2$ such that a path between $u_1$ and $u_2$ becomes tight as a consequence of increasing $y_{U_1}$ and $y_{U_2}$. If this happens, we add the path to $F'$ and continue. $U_1$ and $U_2$ are part of the same connected component of $\tilde{F}^t$ for $t' > t$.

Let $T$ be a tree of the final forest $F$ constructed by AKR. We define the age $\text{age}(T)$ of $T$ to be the point of time at which $T$ was formed, i.e., the first time $t$ such that $T \subseteq F'$. The following is the main theorem of [1]:

**Theorem 1.** Suppose that algorithm AKR outputs a forest $F$ consisting of trees $T_1, \ldots, T_l$ and a feasible dual solution $\{y_U\}_{U \in \mathcal{U}}$. We then have

$$c(F) \leq 2 \cdot \sum_{U \in \mathcal{U}} y_U - 2 \cdot \sum_{i=1}^l \text{age}(T_i) \leq \left(2 - \frac{1}{k}\right) \cdot \text{opt}_R,$$

where $\text{opt}_R$ is the minimum-cost of a Steiner forest for the given input instance with terminal set $R$.

### 3 A cross-monotonic algorithm for Steiner forests

In this section we use the ideas presented in the last section to develop a cross-monotonic algorithm for the Steiner forest problem. We refer to this algorithm by CSF throughout the remainder of this paper.

Define the time of death $d(s,t)$ for each terminal pair $(s,t) \in R$ as

$$d(s,t) = \frac{1}{2} \cdot c(s,t),$$

where $c(s,t)$ denotes the cost of the minimum-cost $s,t$-path in $G$. We assume for ease of presentation that each vertex $v \in V$ has at most one terminal on it. This assumption is without loss of generality since we can replace each vertex in $V$ by a sufficient number of copies and link these copies by 0-cost edges.

We extend the death time notion to individual nodes and define $d(r) = d(s,t)$ for terminals $r,s,t \in R$ iff $r \in \{s,t\}$.

Recall from the last section that AKR raises only node-sets in $\mathcal{U}$ and as a consequence, $y^t$ is a feasible dual solution for (D) at all times $t \geq 0$. Algorithm CSF on the other hand will also raise subsets of $V$ that do not separate terminal pairs.

Using the notation introduced in Section 2 we obtain CSF by modifying the definition of $\mathcal{A}^t$. We say that a connected component $U$ of $\tilde{F}^t$ is active at time $t$ if it contains at least one terminal $r \in U$ with death time at least $t$, i.e., $U$ is active iff there exists $r \in U$ with $d(r) \geq t$. CSF grows all active connected components in $\mathcal{A}^t$ uniformly at all times $t \geq 0$.

What is the intuition behind this? Consider a terminal pair $(s,t) \in R$ and imagine running the primal-dual Steiner forest algorithm AKR on the instance consisting of this terminal pair only. In this case, AKR grows two moats corresponding to $s$ and $t$, respectively, at all times $t \leq d(s,t)$. At time $d(s,t)$ the moats of $s$ and $t$ meet and a path connecting the terminals is added. In CSF a terminal pair $(s,t)$ is active for the
time it would take \( s \) and \( t \) to connect in the absence of any other terminals. The death time of \( s \) and \( t \) is \textit{independent} of other terminal pairs that are present. This independence is the crucial property leading to cross-monotonicity.

Consider an arbitrary terminal pair \((s, t) \in R\). Observe that our choice of the death time \( d(s, t) \) in (2) implies that \( s \) and \( t \) end up in the same connected component of the final forest \( F \) and thus \( \text{CSF} \) constructs a feasible solution for the given Steiner forest instance.

We now detail the cost-share computation. For a terminal \( r \in R \) and for \( t \leq d(r) \) we let \( U'(r) \) be the connected component in \( \bar{F}' \) that contains \( r \). Also let \( d'(r) \) be the number of terminals in \( U'(r) \) whose death time is at least \( t \). We then define the cost-share of terminal node \( r \in R \) as

\[
\xi_r(r) = 2 \cdot \int_{t=0}^{d(r)} \frac{1}{d'(r)} \, dt
\]

and we let \( \xi_s(s, t) = \xi_r(s) + \xi_r(t) \) for all \((s, t) \in R\).

We let the final forest produced by \( \text{CSF}(R) \) be denoted by \( F \) and we use \( \{y_U \}_{U \subseteq V} \) for the dual computed by our method.

### 4 Analysis: Proving cross-monotonicity

In order to prove the cross-monotonicity of \( \text{CSF} \) we consider an arbitrary terminal pair \((s, t) \in R\) and let \( R_0 = R \setminus \{(s, t)\} \). In this section we study the effect of the removal of \((s, t)\) on the cost-shares of all other terminal pairs \((s', t') \in R_0\).

Let us first introduce some simplifying notation. Assume that \( \text{CSF}(R) \) terminates at time \( t^* \) with forest \( F \). Similarly, \( \text{CSF}(R_0) \) finishes at time \( t_0^* \) with a forest \( F_0 \). Moreover, for all times \( t \) we let \( \mathcal{C}' \) and \( \mathcal{C}'_0 \) be the sets of connected components of \( \bar{F}' \) and of \( \bar{F}'_0 \), respectively. The next lemma shows that \( \mathcal{C}'_0 \) is a refinement of \( \mathcal{C}' \).

**Lemma 1.** For all times \( t \leq t^* \) and for all \( U_0 \in \mathcal{C}'_0 \) there must be a set \( U \in \mathcal{C}' \) such that \( U_0 \subseteq U \).

**Proof.** The proof is by induction on the time \( t \). It is clear that the claim is true for \( t = 0 \) since \( \mathcal{C}'_0 = \mathcal{C}'_0^0 = V \).

Consider a point in time \( 0 \leq t < t^* \) and assume the claim is true at time \( t \). \( \text{CSF}(R_0) \) grows active sets in \( \mathcal{C}'_0 \) and these are the only sets that can potentially violate the claim at any time \( t + \varepsilon \) for \( \varepsilon > 0 \). Let \( U_0 \in \mathcal{C}'_0 \) be an active set at time \( t \) in \( \text{CSF}(R_0) \), i.e., there exists a terminal \( r \in U_0 \) with \( d(r) \geq t \). From the induction hypothesis we know that there is a connected component \( U \) of \( F' \) that contains \( U_0 \). Then \( U \) must be active in \( \text{CSF}(R) \) at time \( t \) and hence \( \text{CSF}(R) \) grows \( U \) at time \( t \). The claim follows.

This claim immediately implies cross-monotonicity. Let \( \xi(r) \) and \( \xi_0(r) \) be the cost-share of terminal \( r \in R_0 \) in \( \text{CSF}(R) \) and in \( \text{CSF}(R_0) \), respectively.

**Corollary 1.** Algorithm \( \text{CSF} \) is cross-monotonic, i.e., for each \( r \in R_0 \) we have

\[
\xi_0(r) \geq \xi(r).
\]

**Proof.** Let \( U'(r) \) and \( U_0'(r) \) be the moats containing terminal \( r \) at time \( t \) in \( \text{CSF}(R) \) and \( \text{CSF}(R_0) \), respectively. Similarly, let \( d'(r) \) and \( d_0'(r) \) be the number of terminals with death time at least \( t \) in \( U'(r) \) and \( U_0'(r) \). Lemma 1 implies that \( U_0'(r) \subseteq U'(r) \) and hence \( d_0'(r) \leq d'(r) \) for all \( t \leq t^* \) and for all \( r \in R_0 \). Hence we obtain

\[
\xi(r) = 2 \cdot \int_{t=0}^{d(r)} \frac{1}{d'(r)} \, dt \leq 2 \cdot \int_{t=0}^{d(r)} \frac{1}{d_0'(r)} \, dt = \xi_0(r)
\]

for all \( r \in R_0 \) and the corollary follows. \( \square \)
5 Analysis: Competitiveness and cost-recovery

Recall that we let \( \{ y_U \}_{U \subseteq V} \) denote the dual solution computed by CSF\((R)\) and let \( F \) be the corresponding forest.

**Lemma 2.** Suppose that algorithm CSF outputs a forest \( F \) and a (possibly infeasible) dual solution \( \{ y_U \}_{U \subseteq V} \). We then have
\[
    c(F) \leq 2 \cdot \sum_{U \subseteq V} y_U = \sum_{(s,t) \in R} \xi_R(s,t).
\]

*Proof.* The proof of Theorem 1 implies that \( c(F) \leq 2 \cdot \sum_{U \subseteq V} y_U \). Using Definition (3) it can then be seen that the cost-share sum on the right-hand side of (4) increases by \( 2 \varepsilon \) whenever the total dual value increases by \( \varepsilon \) for some \( \varepsilon > 0 \). Hence we must have \( \sum_{(s,t) \in R} \xi_R(s,t) = 2 \cdot \sum_{U \subseteq V} y_U \). \( \square \)

This does not mean that the cost \( c(F) \) of the forest \( F \) produced by our cost-sharing method is at most twice that of an optimum Steiner forest. In fact, \( \{ y_U \}_{U \subseteq V} \) is not a feasible solution for (D) since our algorithm raises duals for active sets that are not in \( \mathcal{U} \). Surprisingly, we can however show that the total dual \( \sum_{U \subseteq V} y_U \) is bounded by the cost \( \text{opt}_R \) of an optimum Steiner forest for the given instance on terminal set \( R \).

**Lemma 3.** Let \( y \) be the (infeasible) dual computed by CSF\((R)\) and let \( \text{opt}_R \) be the minimum-cost of any feasible Steiner forest for the given instance. Then, we have
\[
    \sum_{U \subseteq V} y_U \leq \text{opt}_R.
\]

Lemma 2 and 3 imply the following corollary on the approximate budget-balance of CSF.

**Corollary 2.** Let \( F \) be the Steiner forest computed by CSF\((R)\). We then have
\[
    c(F) \leq \sum_{(s,t) \in R} \xi_R(s,t) \leq 2 \cdot \text{opt}_R.
\]

We now prove Lemma 3.

5.1 A proof of Lemma 3

Recall the definition of the death time \( d(s,t) \) of a terminal pair \( (s,t) \in R \). In the following, let \( R = \{(s_1,t_1), \ldots, (s_k,t_k)\} \) such that \( d(s_1,t_1) \leq \ldots \leq d(s_k,t_k) \).

We define a total order on the set of terminal nodes as follows. Let \( u \in \{s_i,t_i\}\) and \( v \in \{s_j,t_j\}\) for \( i, j \in \{1, \ldots, k\} \) such that \( u \neq v \). We then define \( u < v \) if \( i < j \) or if \( i = j \) and \( u = s_i \).

Let \( U' \) be an active connected component in CSF\((R)\) at some time \( t \geq 0 \). A terminal node \( v \in U' \) is responsible for the growth of \( U' \) iff there does not exist a terminal \( u \in U' \) different from \( v \) with \( v < u \). This way, each active moat in CSF has a unique responsible terminal node.
Figure 1: A tree $T$ spanning terminals $R(T) = \{s_i,t_i\}_{1 \leq i \leq 3}$. The set of responsible terminal nodes at time $t$ is $R'(T) = \{s_1,s_2,s_3,t_1,t_3\}$ (where we assume that $d(s_1) > d(t_2)$). The corresponding moats in $U'(T) = \{U'(v)\}_{v \in R'(T)}$ are pairwise disjoint and each moat loads at least one edge of $T$.

For a terminal node $v \in R$ and a time $t \geq 0$, let $r'(v) = 1$ if $v$ is responsible at time $t$ and 0 otherwise. We then define the total responsibility time of a terminal $v \in R$ as

$$r(v) = \int_{t=0}^{d(v)} r'(v) dt.$$  (5)

As before we let $U'(v)$ be the connected component of $\bar{F}^t$ containing terminal $v \in R$. We can show that a terminal $v \in R$ is responsible for a unique moat at all times $0 \leq t < r(v)$.

Claim 1. Let $v \in R$ be a terminal and let $r(v)$ be its total responsibility time. Then, for any point of time $0 \leq t < r(v)$, $v$ is responsible for $U'(v)$ in $\text{CSF}(R)$.

Proof. Assume for the sake of contradiction that there is a point of time $t \in [0,r(v))$ such that $v$ is not responsible for $U'(v)$. Since $U'(v)$ is active, we know that there must be a terminal $u \in U'(v)$ that is responsible. We therefore must have $v < u$ and also $d(v) \leq d(u)$. Since $u$ and $v$ are contained in the same active moat in $\text{CSF}$ after time $t$, this means that $v$ cannot be responsible after time $t$ and hence $r(v) \leq t$; a contradiction.

Definition (5) also implies that

$$\sum_{U \subseteq V} y_U = \sum_{u \in R} r(u)$$

and hence it suffices to bound the sum on the right-hand side in order to prove Lemma 3.

Let $F^*$ be a minimum-cost Steiner forest for the given instance with terminal set $R$. Consider a tree $T$ in $F^*$ and suppose that $T$ connects the terminals $R(T) = \{v_1, \ldots, v_p\}$; see Figure 1. The idea is to bound the total growth of the moats of terminals in $R(T)$ by the cost $c(T)$ of the tree $T$ spanning $R(T)$.

We let $R'(T)$ be the set of terminal nodes in $R(T)$ that are responsible at time $t$, i.e.,

$$R'(T) = \{v \in R(T) : r'(v) = 1\}.$$ 

The following claim shows that at any time $t$ the moats in

$$U'(T) = \{U'(v)\}_{v \in R'(T)}$$
Lemma 4. the same connected component of \( U^i \) among the nodes in \( T \) are pairwise disjoint.

\[ \text{Lemma 4.} \quad \text{the same connected component of } U^i \quad \text{among the nodes in } T \quad \text{are pairwise disjoint.} \]

Proof. Assume for the sake of contradiction that \( U^i(\cdot) \) and \( U^i(u) \) are not disjoint. Since both \( U^i(\cdot) \) and \( U^i(u) \) are connected components of \( T^i \) it must therefore be the case that \( U^i(\cdot) = U^i(u) \). Claim 1 implies that both \( v \) and \( u \) are responsible for this moat and hence, we must have \( v = u \). This contradicts our choice of \( u \) and \( v \).

Let \( w \in R(T) \) be the terminal node with highest responsibility time. It is tempting to believe that \( w \) is the node with largest death time among nodes in \( R(T) \) and that at time \( t = r(w) \) all nodes in \( R(T) \) are in the same connected component \( U^i(w) \) of \( T^i \). However, this need not necessarily be true; see Figure 2.

Claim 2. Consider a point of time \( t \) and two terminal nodes \( v, u \in R^i(T), v \neq u \). The two moats \( U^i(v) \) and \( U^i(u) \) must be disjoint.

Proof. Assume for the sake of contradiction that \( U^i(v) \) and \( U^i(u) \) are not disjoint. Since both \( U^i(v) \) and \( U^i(u) \) are connected components of \( T^i \) it must therefore be the case that \( U^i(v) = U^i(u) \). Claim 1 implies that both \( v \) and \( u \) are responsible for this moat and hence, we must have \( v = u \). This contradicts our choice of \( u \) and \( v \).

Let \( w \in R(T) \) be the terminal node with highest responsibility time. It is tempting to believe that \( w \) is the node with largest death time among nodes in \( R(T) \) and that at time \( t = r(w) \) all nodes in \( R(T) \) are in the same connected component \( U^i(w) \) of \( T^i \). However, this need not necessarily be true; see Figure 2.

Lemma 4. If \( \delta(U^i(w)) \cap T \neq \emptyset \) for all \( 0 \leq t < r(w) \) then we must have \( \sum_{v \in R(T)} r(v) \leq c(T) \).

Proof. Consider any point of time \( t \geq 0 \) where there are at least two terminals in \( R(T) \) that are responsible, i.e., \( |R^i(T)| > 1 \). By Claim 2 we have that the moats in \( U^i(T) \) are pairwise disjoint. On the other hand, the nodes in \( R^i(T) \) are connected by \( T \) and hence, each of the moats in \( U^i(T) \) loads a distinct part of the edges of \( T \); see Figure 1.

Consider now a time \( t \) where \( |R^i(T)| = 1 \). It must be the case that \( w \) is the only remaining responsible terminal among the nodes in \( R(T) \), i.e., \( R^i(T) = \{w\} \). By assumption, \( U^i(w) \) loads at least one edge of \( T \). This concludes the proof of the lemma.

In the following, let \( \bar{w} \) be the mate of \( w \), i.e., \( (w, \bar{w}) \in R(T) \) and \( d(w) = d(w, \bar{w}) \). From now on we will assume that there is a time \( t_0 \in [0, r(w)) \) such that \( \delta(U^{i_0}(w)) \cap T = \emptyset \) and hence \( T \subseteq E(U^{i_0}(w)) \), where \( E(U^{i_0}(w)) \) denotes the subset of those edges in \( E \) that have both endpoints in \( U^{i_0}(w) \).

We also must have \( |R^i(T)| = 1 \) for all \( t \in [t_0, r(w)) \) since all nodes of \( R(T) \) are in the same connected component of \( T^i \). Furthermore, since \( w \) is responsible until time \( r(w) \) we must have \( R^i(T) = \{w\} \) for all \( t \in [t_0, r(w)) \) and thus \( u \prec w \) and \( u \prec \bar{w} \) for all \( u \in R(T) \setminus \{w, \bar{w}\} \).

Let \( P_w \) be the unique \( w, \bar{w} \)-path in \( T \). We define \( I^i(T) \) as the set of responsible terminal pairs in \( R^i(T) \setminus \{w, \bar{w}\} \) that inflict dual load on path \( P_w \) in \( CSF(R) \) at time \( t \), i.e.,

\[ I^i(T) = \{v \in R^i(T) \setminus \{w, \bar{w}\} : \delta(U^i(v)) \cap P_w = \emptyset \}. \]

Claim 3. Consider a point in time \( t \) and a terminal \( v \in I^i(T) \). Then \( U^i(v) \) does not contain either \( w \) or \( \bar{w} \).

Proof. By definition of \( I^i(T) \), we know that \( v \not\in \{w, \bar{w}\} \). We also know that \( v \prec w \) and \( v \prec \bar{w} \). The claim follows as \( v \) is responsible for the growth of \( U^i(v) \) and hence \( \{w, \bar{w}\} \cap U^i(v) = \emptyset \).
For a time \( t \) and a node \( v \in I'(T) \), let \( p'_w(v) \) be the number of intersections of \( P_w \) and \( U'(v) \) at time \( t \):

\[
p'_w(v) = |\delta(U'(v)) \cap P_w|.
\]  

We use \( s_1 \) to denote the cost of that part of \( P_w \) that does not feel any dual load from any of the terminals in \( R(T) \). Let \( l_w \) and \( l_\bar{w} \) be the total load on \( P_w \) coming from terminals \( w \) and \( \bar{w} \), respectively. We can then express the cost of \( P_w \) as

\[
c(P_w) = l_w + l_\bar{w} + s_1 + \int_0^{t_0} \sum_{v \in F(T)} p'_w(v) dt.
\]  

We obtain the following lemma.

**Lemma 5.** If there is a \( t_0 \in [0, r(w)) \) with \( \delta(U^0(w)) \cap T = \emptyset \) then we must have \( \sum_{v \in R(T)} r(v) \leq c(T) \).

**Proof.** Similar to the proof of Lemma 4, consider a time \( t < r(w) \) where \( R'(T) \) contains more than one terminal. The corresponding moats in \( U'(T) \) are pairwise disjoint by Claim 2 and the nodes in \( R'(T) \) are connected by \( T \). Hence, each of the moats in \( U'(T) \) loads a distinct part of \( T \).

Moreover, using the definition of \( p'_w(v) \) in (6), for all \( t \in [0, t_0) \) and \( v \in I'(T) \) moat \( U'(v) \) loads at least \( p'_w(v) \) edges of \( T \).

Recall that \( s_1 \) is the cost of the segments of \( P_w \) that do not feel any load from terminals in \( R(T) \). Furthermore, \( w \) loads edges of \( T \) until time \( t_0 \) and hence we must have

\[
s_1 + \int_0^{t_0} \sum_{v \in F(T)} (p'_w(v) - 1) dt + t_0 + \sum_{v \in R(T) \setminus \{w\}} r(v) \leq c(T).
\]  

The death time of node \( w \) is at most half of the cost of \( P_w \). Using (7) we therefore obtain

\[
r(w) \leq \frac{l_w + l_\bar{w} + s_1}{2} + \frac{1}{2} \int_0^{t_0} \sum_{v \in F(T)} p'_w(v) dt
\]

\[
\leq t_0 + s_1 + \int_0^{t_0} \sum_{v \in F(T)} (p'_w(v) - 1) dt,
\]

where the second inequality uses the fact that \( \max\{l_w, l_\bar{w}\} \leq t_0 \) and that by Claim 3, \( p'_w(v) \geq 2 \) for all \( v \in I'(T) \). Combining (8) and (9) yields the lemma. \( \square \)

We can now sum over all trees \( T \) in the forest \( F^* \). Lemmas 4 and 5 imply that

\[
\sum_{T \in F^*} \sum_{v \in R(T)} r(v) \leq \sum_{T \in F^*} c(T) = \text{opt}_R.
\]

This finishes the proof of Lemma 3.

### 6 Algorithmic consequences

In the previous section we have shown that the dual solution \( \{y_U\}_{U \subseteq V} \) computed by our algorithm CSF, although being possibly infeasible for (D), yields a lower bound on the optimum cost \( \text{opt}_R \):

\[
\text{opt}_R \geq \sum_{U \subseteq V} y_U.
\]
Following the proof of Theorem 1 of Agrawal, Klein, and Ravi [1], we can use this fact and prove that our algorithm achieves the same approximation guarantee as the known primal-dual algorithms [1, 2]. This is surprising, since the forest constructed by our algorithm is usually costlier than those computed by the algorithms in [1, 2].

Let $T$ be a tree in the final forest $F$ constructed by CSF. We define the age $\text{age}(T)$ of $T$ to be the point of time at which the final moat that contains $T$ stops growing, i.e., $\text{age}(T) = \max \{r(v) : v \in R(T)\}$.

**Theorem 2.** Suppose that algorithm CSF outputs a forest $F$ consisting of trees $T_1, \ldots, T_l$ and a (possibly infeasible) dual solution $\{y_U\}_{U \subseteq V}$. We then have

$$c(F) \leq 2 \cdot \sum_{U \subseteq V} y_U - 2 \cdot \sum_{i=1}^{l} \text{age}(T_i) \leq \left(2 - \frac{1}{k}\right) \cdot \text{opt}_R,$$

where $\text{opt}_R$ is the minimum-cost of a Steiner forest for the given input instance with terminal set $R$.

Since CSF also raises dual variables for node-sets that do not separate any terminal pair, one could hope that CSF always constructs a better lower bound than those obtained from the feasible dual solution of the algorithms in [1, 2].

In fact, depending on the underlying instance, CSF may yield a significantly stronger lower bound than the one obtained from the feasible dual solution of AKR. As an example, consider the instance given in Figure 3. The optimal cost $\text{opt}_R$ to connect all terminal pairs in $R$ is $2^k$. The total dual raised during the execution of AKR equals $2^k \cdot \frac{1}{2} = k$, while the total dual of the solution constructed by CSF is $(2^k - 1) \cdot \frac{1}{2} + \frac{1}{2}c(s_1, t_1) = 2k - 1 = \text{opt}_R$. That is, for this particular instance, the lower bound of CSF proves optimality of the computed solution. Observe that this example also shows that the bound stated in Lemma 3 is tight.

On the other hand, the above statement is unfortunately not true in general. One can construct Steiner forest instances for which the lower bounds of CSF and AKR are equally close to the optimum, or on which the lower bound of AKR is better than to the one of CSF.

**A special case: Rooted Steiner tree games.** The rooted Steiner tree game is a special case of the Steiner forest game. In the Steiner tree game, we are given a subset $R' \subseteq V$ of terminal nodes that want to be connected to a designated root node $r$; that is, agents correspond to nodes and the root node in particular is not part of the agents-set. A feasible solution is a tree that spans $R' \cup \{r\}$.

Jain and Vazirani [4] gave a 2-approximate cross-monotonic cost-sharing method for the Steiner tree game. Their method is based on a budget balanced cross-monotonic cost-sharing method for the minimum spanning tree game with a pre-specified root.

We can use algorithm CSF to obtain a 2-approximate budget-balanced cost-sharing mechanism $\xi^{ST}$ for the Steiner tree game: Define the set of terminal pairs as $R = \{(r, v)\}_{v \in R'}$ and let algorithm CSF run on this instance. Recall that the root node $r$ is not part of the agent-set in the Steiner tree game. We therefore define the cost-share of a terminal node $v \in R'$ as $\xi^{ST}_R(v) = \xi_R(r, v)$. By Corollary 1 and Corollary 2, $\xi^{ST}$ is a 2-approximate budget-balanced cross-monotonic cost-sharing method for the Steiner tree game.
7 A stronger LP relaxation for Steiner forests

Consider an instance of the Steiner forest problem given by an undirected graph $G = (V, E)$, non-negative edge costs $c_e$ for all edges $e \in E$, and a set of $k \geq 1$ terminal pairs $R \subseteq V \times V$. For a subset $U \subseteq V$, let $S(U)$ denote the set of terminal pairs in $R$ that are separated by $U$, i.e.,

$$S(U) = \{(s, t) \in R : |\{s, t\} \cap U| = 1\}.$$

Also let $s(U)$ be the cardinality of $S(U)$. A subset $U \subseteq V$ is then in $\mathcal{U}$ if $s(U) \geq 1$.

Recall the definition of the death time $d(s, t)$ of a terminal-pair $(s, t) \in R$ from (2). We also remind the reader of the definition of the precedence relation $\prec$ that ranks the terminals in $R$ in order of non-decreasing death-time. We extend this relation to terminals pairs and say that $(s_1, t_1) \prec (s_2, t_2)$ for two terminal pairs $(s_1, t_1), (s_2, t_2) \in R$ if $s_1 \prec s_2$.

For a set $U \subseteq V$, we let $R(U)$ denote the set of terminal nodes that are contained in $U$. Consider a terminal $w \in \{s, t\}$ and let $\tilde{w}$ be $w$’s mate in the Steiner forest instance (i.e., $(w, \tilde{w}) \in R$). We let $\mathcal{U}_w$ be the set of Steiner cuts that contain $w$ but not $\tilde{w}$ and for which $w$ is the separated terminal of highest rank:

$$\mathcal{U}_w = \{U \in \mathcal{U} : w \in S(U), v \prec w \text{ for all } v \in S(U)\}.$$

We also let $\mathcal{U}_{w, \tilde{w}}$ be the set of all non-Steiner cuts containing $w$ and $\tilde{w}$, where $(w, \tilde{w})$ is the terminal pair of highest rank:

$$\mathcal{U}_{w, \tilde{w}} = \{U \subseteq V : s(U) = 0, \ (w, \tilde{w}) \subseteq R(U), (s, t) \prec (w, \tilde{w}) \text{ for all } (s, t) \in R(U)\}.$$

We will also say that a terminal $w \in R$ is responsible for set $U \subseteq V$ if $U \in \mathcal{U}_w \cup \mathcal{U}_{w, \tilde{w}}$.

We now present the lifted-cut dual for the Steiner forest problem:

$$\text{opt}_{LC-D} := \max_{U \subseteq V} \sum_{U \subseteq V} y_U$$

s.t.

$$\sum_{U \subseteq V, e \in \delta(U)} y_U \leq c_e \quad \forall e \in E \quad (10)$$

$$\sum_{U \in \mathcal{U}_w} y_U + \sum_{U \in \mathcal{U}_{w, \tilde{w}}} y_U \leq d(w) \quad \forall w \in R \quad (11)$$

$$y \geq 0$$

Notice that a feasible solution to (LC-D) may assign positive values to non-Steiner cuts $U \in 2^V \setminus \mathcal{U}$. The constraints of type (11) are necessary as the objective function value of (LC-D) would be unbounded in their absence. It is an easy exercise to verify that CSF produces a dual solution that is feasible for (LC-D).

Fix a set $U \subseteq V$ and consider the occurrence of $y_U$ in constraints (11). If $U \in \mathcal{U}_w$, $y_U$ only occurs in the constraint for $w$. Otherwise, if $U \in \mathcal{U}_{w, \tilde{w}}$, $y_U$ occurs in the constraint for $w$ and in the constraint for $\tilde{w}$.

The following LP is the dual of (LC-D):

$$\text{opt}_{LC-P} := \min \sum_{e \in E} c_e \cdot x_e + \sum_{w \in R} d(w) x_w \quad (LC-P)$$

s.t.

$$\sum_{e \in \delta(U)} x_e \geq 1 \quad \forall U \in \mathcal{U}_w, \forall w \in R \quad (12)$$

$$\sum_{e \in \delta(U)} x_e + x_w + x_{\tilde{w}} \geq 1 \quad \forall U \in \mathcal{U}_{w, \tilde{w}}, \forall (w, \tilde{w}) \in R \quad (13)$$

$$x \geq 0$$
Lemma 6. Let $F$ be a feasible solution for the underlying Steiner forest instance. We can then construct a solution $x$ that is feasible for (LC-P) and satisfies:
\[
\sum_{e \in E} c_e \cdot x_e + \sum_{w \in R} d(w)x_w \leq c(F).
\]

In particular, this implies that $\text{opt}_{\text{LC-D}} = \text{opt}_{\text{LC-P}} \leq \text{opt}_{R}$.

Proof. We construct a solution $x$ that is feasible for (LC-P) and show that for each tree $T \in F$
\[
\sum_{e \in E(T)} c_e \cdot x_e + \sum_{w \in R(T)} d(w)x_w \leq c(T).
\]

The lemma then follows by summing over all trees in $F$.

Consider a tree $T \in F$. Let $(w, \bar{w})$ be the responsible terminal pair for the non-Steiner cut $V(T)$, the set of all vertices spanned by $T$. Moreover, let $P$ denote the unique $w, \bar{w}$-path in $T$. We set $x_e = \frac{1}{2}$ for each edge $e \in E(P)$ and $x_e = 1$ for each edge $e \in E(T \setminus P)$. Moreover, we assign $x_w = x_{\bar{w}} = \frac{1}{2}$ and $x_r = 0$ for all terminals $r \in T \setminus \{w, \bar{w}\}$.

The objective value for $x$ is
\[
c(T) - \frac{1}{2}c(P) + \frac{1}{2}d(w) + \frac{1}{2}d(\bar{w}) = c(T) - \frac{1}{2}c(P) + d(w, \bar{w}) \leq c(T),
\]
where the last inequality holds since $d(w, \bar{w}) \leq \frac{1}{2}c(P)$, by the definition of death time.

It remains to be shown that $x$ is feasible for (LC-P). We show for each terminal $r \in R(T)$ that the cuts in $\mathcal{U}_r$ for each terminal pair $(r, \bar{r}) \in R(T)$ satisfy the constraints in (LC-P).

Consider a cut $U \in \mathcal{U}_r$ for some $r \in R(T)$. If $r \in \{w, \bar{w}\}$, constraint (12) holds, since $U$ intersects $P$ and $x_r = \frac{1}{2}$. Now let $r \notin \{w, \bar{w}\}$. As $(w, \bar{w})$ is responsible for $V(T)$ we must have $(r, \bar{r}) \prec (w, \bar{w})$ and hence $(w, \bar{w}) \cap U = 0$. It can now be seen that $U$ either intersects at least one edge $e$ of $T$ that is not on $P$ (and hence $x_e = 1$) or it intersects at least two edges $e_1$ and $e_2$ on $P$ (and therefore $x_{e_1} = x_{e_2} = 1/2$). Thus, constraint (12) holds in this case as well.

Next consider a cut $U \in \mathcal{U}_r, \gamma$ for a terminal pair $(r, \bar{r}) \in R(T)$. If $(r, \bar{r}) \neq (w, \bar{w})$ then $(w, \bar{w}) \setminus U = 0$ as before and $U$ crosses at least one edge of $T$ that is not on $P$ or at least two edges of $P$. Hence constraint (13) holds. Otherwise, $U$ crosses no edge of $T$ but $x_w + x_{\bar{w}} = 1$ and thus (13) is satisfied.

We end this section by examining a well-known cycle instance for the Steiner tree problem that shows that the IP/LP gap of (D) is approximately 2. Consider an even-length cycle $C = (v_0, \ldots, v_{n-1})$ on $n$ nodes with unit edge costs. Pick an arbitrary one of the $n$ nodes as the root and let all other nodes be the terminals of our instance. The optimal Steiner tree for this instance has cost $\text{opt}_{R} = n - 1$.

The total dual constructed by AKR is $\sum_{U \subseteq \mathcal{V}} y_U = n/2$. Observe that this is an optimal solution for (D) as there is a half-integral solution for the LP relaxation having the same cost: set $x_e = \frac{1}{2}$ for each edge $e$ of the cycle. That is, any feasible dual solution $y$ must satisfy $\sum_{U \subseteq \mathcal{V}} y_U \leq n/2$.

On the other hand, the total dual constructed by CSF is
\[
\sum_{U \subseteq \mathcal{V}} y_U = \frac{1}{2} \cdot (n - 1) + \frac{1}{2} \cdot n \cdot 2 = \frac{3n}{4} - \frac{1}{2}
\]
and this solution is feasible for (LC-D). The latter term is strictly larger than $n/2$ if $n > 2$. The IP/LP gap of (LC-D) is therefore at least $4/3$ and we are not aware of a worse example for Steiner trees.
References


