The Prize-Collecting Generalized Steiner Tree Problem via a new approach of Primal-Dual Schema

Kamal Jain[*] *Mohammad T. Hajiaghayi*[†]

Abstract

Primal-Dual schema is one of the most versatile techniques in the theory of approximation algorithms. A common starting point in all the primal-dual algorithms is to come up with a natural way to distribute the (approximate) cost of the primal solution to dual variables. For many problems (including the one in this paper, among unsolved problems the most notable are bidirected Steiner tree LP, factor 2 primal-dual for Steiner network problems, rent-or-buy problems) a natural way is either not clear or has bad examples. One reason for this difficulty is the use of auxiliary variable in the primal or dual programs. We propose to eliminate auxiliary variables using Farkas lemma towards obtaining a natural way to raise duals. We believe Farkas lemma is a major addition to primal-dual schema especially after noting the fact that primal-dual algorithms do not use the powerful strong duality theorem, which is equivalent to Farkas lemma. So by using Farkas lemma we actively used the strong duality theorem. Our scheme also answers a philosophical question both ways. Edmonds' primal-dual matching algorithm seems significantly more involved than the use of primal-dual algorithm in approximation algorithms (Edmonds both increased and decreased dual variables, whereas in approximation algorithm we only increase). Is this really the case or we can obtain better approximation algorithms following Edmonds' schema? We answer the question both ways. Yes, we can obtain better approximation algorithms by following the Edmonds' schema. Our algorithm when translated back on the original LP both increases and decreases the dual variables. In a separate publication [18], we answer this question the other way too. The Edmonds' LP can be converted into another LP where we only need to raise dual-variables.

1 Introduction

Consider a mailing company that wishes to ship packets overnight between several pairs of cities. To this end, this company can build connecting carriers between cities such that at the end by scheduling the carriers, the company is able to ship the packets overnight between pairs of connected cities. Assume the cost of connecting city i to city j is c_{ij} and the costs are symmetric. In addition, the company has the choice of leasing other companies for some pairs (i, j) of cities with cost π_{ij} so that without any worry the leased company do the shipment between cities i and j overnight. The goal is to build some carriers and lease some other companies such that the company do the shipments overnight with minimum total cost.

The above problem which has also several applications in design of telecommunications networks and wiring with different cable types which offer different amounts of capacity and carry different costs (see e.g. [2, 23]) is called the *prize-collecting generalized Steiner tree (PCGST)* problem. In this problem, given a graph $G = (V, E)$, a set of pairs $\mathcal{P} = \{(s_1, t_1), (s_1, t_1), \ldots, (s_k, t_k)\}$, a non-negative cost function $c : E \to \mathbf{Q}_+$, and finally a non-negative penalty function $\pi : \mathcal{P} \to \mathbf{Q}_+$, our goal is minimum-cost way of buying a set of edges and paying the penalty for those pairs which are not connected via bought edges. Without loss of generality we assume that $\mathcal{P} = V \times V$, since the penalty of any pair which is not required to be connected will be zero (In this paper, by $V \times V$ we mean all unordered pairs (i, j) where $i \neq j$.) When all sinks are identical, this problem is the classical prize collecting Steiner tree problem. When all penalties are infinity, this problem is the classical generalized Steiner tree problem. The best approximation algorithm for both of these problems is a $2 - \frac{1}{n-1}$ approximation algorithm (n is the number of vertices of the graph) due to Goemans and Williamson [11].

[*]Address: Microsoft Research. One Microsoft Way, Redmond, WA 98052, USA, E-mail: kamalj@microsoft.com.

[†]Address: Computer Science and Artificial Intelligence Laboratory, Massachusetts Institute of Technology, 200 Technology Square, Cambridge, MA 02139, U.S.A., E-mail: hajiagha@theory.lcs.mit.edu. This work was done while the author visited the Microsoft Research Theory Group.

1.1 Related work

Other problems with the same flavor of the PCGST problem which attracted so much interest recently are the *multicommodity connected facility location (MCFL)* problem and the *multicommodity rent-or-buy (MRoB)* problem (see e.g. [2, 13, 14, 19, 20]). In both of these problems, we are given an undirected graph $G = (V, E)$ with edge e possessing renting cost c_e and buying cost Mc_e for a parameter $M > 1$, and a set $V \times V = \{(s_1, t_1), (s_1, t_1), \ldots, (s_k, t_k)\}$ of pairs. In the former problem, our goal is to open a set F of facilities, assign sources and sinks to open facilities (on the vertices), and find a subgraph (V, H) of G such that for each h, if s_h is assigned to facility i_1 and t_h is assigned to facility i_2, then there is a path in (V, H) between i_1 and i_2. The cost of a solution is the sum of buying cost of edges in H and renting cost of edges in the shortest path (with respect to c) of each demand to its assigned facility. In the latter problem, our goal is to find a minimum-cost way of buying some edges, which allow unlimited use after payment of a large fixed cost, and renting some edges, with cost incurred on a per-unit of capacity basis, so that a unit of demand can be sent simultaneously from each source s_i to the corresponding sink t_i. Interestingly, Kumar et al. [20] show that any α-approximation algorithm for the MCFL problem gives a 2α-approximation algorithm for the MRoB problem. The best approximation algorithm for these problems is the randomized 8-approximation algorithm of Gupta et al. [13]. To the best of our knowledge, the only deterministic algorithm with extremely large constant approximation ratio [1] is a complicated primal-dual algorithm due to Kumar et al. [20] which is a combination of Goemans-Williamson [11] and Jain-Vazirani [16] primal-dual algorithms. Intuitively, the main difference between the MCFL problem and the PCGST problem, the problem considered in this paper, is that in the former problem the path between any source-sink pair (s_h, t_h) should go through rented edges, bought edges and then rented edges while in the latter problem the path either entirely should go through bought edges or entirely should go through leased (rented) edges.

1.2 Our results

We show a novel primal-dual framework which gives a 3-approximation algorithm for the prize-collecting generalized Steiner tree problem. In addition, we show that factor three is tight for our primal-dual framework. Finally, we show how we can improve the approximation factor to 2.54 using an LP-rounding technique. Indeed, both of the approaches introduced in this paper, i.e., the primal-dual approach and the LP-rounding approach, (though we mainly focus on the primal-dual approach) are general techniques for prize-collecting versions of combinatorial optimization problems. The prize-collecting generalized Steiner tree problem is just an example for which we have used both of our general approaches. It is worth mentioning that prize-collecting problems naturally appear in game theory and more specifically in the context of cost-sharing (in which cost share of an agent cannot be more than the utility derived). These problems also appear naturally if one wants to use the Lagarangian Relaxation technique of Jain-Vazirani [16] to obtain an approximation algorithm for corresponding k-version of the problem e.g., prize collecting Steiner tree problem is a Lagrangian relaxation of k-MST problem.

Our primal-dual algorithm has some similarities with Goemans-Williamson algorithm for the prize-collecting Steiner tree (PCST) problem, however our primal-dual LPs are considerably different from theirs. In fact, the primal-dual algorithm for PCST makes a crucial use of the assumption that all commodities have the same sink vertex and does not obviously extend to the PCGST problem. This is the same hardness mentioned by Kumar et al. [20], when they want to extend primal-dual algorithms for single-sink MCFL/MRoB to general MCFL/MRoB. Indeed, if the graph G is a tree the PCST problem can be solved in polynomial time using a simple dynamic programming. However, such a simple dynamic-programming approach does not work for the PCGST problem on trees, and we reduce the problem to an instance of max flow/min cut to solve it

[1] According to the authors, the approach of the paper only seems suitable to prove an approximation factor of at least several hundreds

polynomially (see Appendix E). It is worth mentioning, by having the solution of this problem on trees, one can obtain a logarithmic approximation ratio for the PCGST problem based on Bartal's machinery [3] (or its slight improvement by Fakcharoenphol et al. [8]) for probabilistically embedding general metrics into tree metrics.

The first barrier which arises when we consider the natural generalization of Goemans-Williamson primal-dual algorithm is dividing the initial penalty (which is in fact the initial potential) between vertices of each pair. We show that an incorrect dividing of the initial penalty can make the approximation factor very large. The second barrier arises in cost-sharing, the task of allocating the cost of an object to many users of the object in a "fair" manner, when two balls (connected components) merge in GW-algorithm. The last not the least barrier arises in the final removal of (unnecessary) edges in GW-algorithm. We overcome all these barriers by writing novel primal and dual LPs and then make a crucial use of Farkas' Lemma. Indeed some of the aforementioned barriers can be overcome if we can both increase and decrease the dual variables. To the best of our knowledge, the only primal-dual (exact and not even approximation) algorithms which both increase and decrease the dual variables are the max-flow algorithm and the seminal Edmonds' algorithm for minimum weighted perfect matching. In a separate paper, we show that we can obtain primal and dual LPs for these problems such that in the corresponding algorithms we only increase the dual variables (however, when translated back on the original LP both increase and decrease the dual variables) and still we can solve these problems in polynomial time. Thus, our framework makes a significant step towards answering the important open question that asks whether both increasing and decreasing of dual variables can be used to obtain better primal-dual approximation algorithms for combinatorial optimization problems.

2 Preliminary Definitions

All the graphs in this paper are undirected without loops or multiple edges, unless stated otherwise. The reader is referred to standard references for appropriate background [5, 27]. In addition, for exact definitions of various undefined NP-hard graph-theoretic problems in this paper, the reader is referred to Garey and Johnson's book on computers and intractability [9].

Our graph terminology is as follows. A graph G is represented by $G = (V, E)$, where V (or $V(G)$) is the set of vertices (or nodes) and E (or $E(G)$) is the set of edges. For a set $S \subset V$, we denote $|\{i, j\} \cap S| = 1$ by $S \odot (i, j)$. We denote a family of subsets of V by $\mathcal{S} = \{S_1, S_2, \ldots, S_k\}$. For a family $\mathcal{S}$, we write $\mathcal{S} \odot (i, j)$, if there exists an $S \in \mathcal{S}$ such that $S \odot (i, j)$.

3 Suitable LP relaxation using Farkas Lemma

The traditional LP relaxation for the problem can be written as:

$$\text{OPT} = \min \sum_{e \in E} c_e x_e + \sum_{i,j \in V} \pi_{ij} z_{ij}$$

subject to

$$\sum_{e \in \delta(S)} x_e + z_{ij} \geq 1 \qquad \forall S \subset V, (i, j) \in V \times V, S \odot (i, j)$$

$$x_e \geq 0 \qquad \forall e \in E$$

$$z_{ij} \geq 0 \qquad \forall (i, j) \in V \times V$$

The dual of this LP is:

$$\text{maximize} \quad \sum_{S \subset V, S \odot (i,j)} y_{Sij} \tag{1}$$

$$\text{subject to} \quad \sum_{S: e \in \delta(S), S \odot (i,j)} y_{Sij} \leq c_e \qquad \forall e \in E$$

$$\sum_{S: S \odot (i,j)} y_{Sij} \leq \pi_{ij} \qquad \forall (i,j) \in V \times V$$

$$y_{Sij} \geq 0 \qquad \forall S \subset V, S \odot (i,j)$$

The problem in this dual LP is that there are different dual variables for each pair of vertices. Consider a situation where two vertices i and j are replaced by a million of copies i.e., i is replaced by $i_1, i_2, \cdots$ and j is replaced by $j_1, j_2, \cdots$. Also the original penalty between i and j is replaced by its millionth between i_1 and j_1, and i_2 and j_2, and so on. By this reduction we did not logically change the original instance but the execution of a primal dual algorithm which treats each $y_{Si_1j_1}, y_{Si_2j_2}, \cdots$ independently can change significantly. Ability to raise dual variables independently to a specific goal is what really constitutes a primal-dual schema. In fact, it is not hard to create counterexamples that such an approach will not work. On the other hand if we raise $y_{Si_1j_1}, y_{Si_2j_2}, \cdots$ in some dependent manner then we will need to deal with cost-sharing i.e., distributing the rate of increase. If we raise uniformly then a pair of vertices can bias the dual raise in its favor by creating multiple copies as shown in the above example. In short, we do not see any direct solution to this problem, which is also a major hindrance in developing primal-dual schema based algorithm for many other problems (e.g., other version of the rent-or-buy problem, the Steiner tree problem based on bidirected LP, the generalized Steiner network problem). In this paper we propose a clean solution to this issue based on Farkas lemma. The problem we solve should be taken as an exemplification of our approach.

We create a new dual variable y_S which is the sum of all y_{Sij}'s, where i and j are separated by S. We will treat y_S as an independent dual variable in our LP. We will not fix in advance any arrangement of distributing y_S among the y_{Sij}'s. Instead we will let the dual solution adapts itself to accommodate y_{Sij}'s. A more rigorous way is to eliminate the issue of cost-sharing altogether by eliminating the dual variables y_{Sij}'s. This also inspires another paper [18] in which we show that Edmonds' matching primal-dual algorithm can be recast such that we only increase dual variables (originally Edmonds uses both increasing and decreasing of dual variables). With this substitution of y_S, the dual LP 1 becomes:

$$\text{maximize} \quad \sum_{S \subset V} y_S \tag{2}$$

$$\text{subject to} \quad y_S \leq \sum_{(i,j): S \odot (i,j)} y_{Sij} \qquad \forall S \subset V$$

$$\sum_{S: e \in \delta(S)} y_S \leq c_e \qquad \forall e \in E$$

$$\sum_{S: S \odot (i,j)} y_{Sij} \leq \pi_{ij} \qquad \forall i, j \in V$$

$$y_{Sij} \geq 0 \qquad \forall S \subset V, S \odot (i,j)$$

$$y_S \geq 0 \qquad \forall S \subset V$$

Lemma 1 *Dual LPs 1 and 2 are equivalent.*

Proof : Suppose we have a solution for dual LP 1; we can assign $y_S = \sum_{(i,j): S \odot (i,j)} y_{Sij}$ for each $S \subset V$,

which is a solution to dual LP 2. Suppose we have a solution for dual LP 2. Without loss of generality we can assume that the first set of constraint is tight; if not then we can decrease y_{Sij} without affecting the objective function (here we use that $y_S \geq 0$). Once we have $y_S = \sum_{(i,j):S \odot (i,j)} y_{Sij}$, we can replace y_S to get a solution for dual LP 1. □

Note that y_{Sij}'s do not appear in the objective function, so they can be classified as auxiliary variables. So the idea of the primal dual algorithm is to raise dual variables y_S's (according to some rule) and let the LP itself keep accommodating the auxiliary variables (which may increase or decrease). In other words we raise dual variables y_S's and keep testing whether feasible y_{Sij}'s exist, so in essence we do not decrease the actual dual variable but may decrease the auxiliary dual variables. Intuitively, we freeze some y_S if increasing it by infinitesimally small epsilon results in the non-existence of feasible y_{Sij}'s. Formally, we need to find the condition when feasible y_{Sij}'s do not exist. This can be done using Farkas lemma.

Suppose $\alpha : 2^V \rightarrow \mathcal{R}^+$ is a function from the powerset of V to the non-negative real numbers. For convenience we use the notation α_S to denote the value of α at S. Consider a feasible solution of the dual LP 2. Using the first set of constraints we get:

$$\sum_{S \subset V} \alpha_S y_S \leq \sum_{S \subset V} \alpha_S \sum_{(i,j):S \odot (i,j)} y_{Sij}$$

Changing the order of summation on the right-hand side we get:

$$\sum_{S \subset V} \alpha_S y_S \leq \sum_{(i,j) \in V \times V} \sum_{S:S \odot (i,j)} \alpha_S y_{Sij}$$

The right hand side can be increased further to yield:

$$\sum_{S \subset V} \alpha_S y_S \leq \sum_{i,j \in V} \sum_{S:S \odot (i,j)} (\max_{S:S \odot (i,j)} \alpha_S) y_{Sij}$$

This simplifies to:

$$\sum_{S \subset V} \alpha_S y_S \leq \sum_{i,j \in V} (\max_{S:S \odot (i,j)} \alpha_S) \sum_{S:S \odot (i,j)} y_{Sij}$$

Along with the third set of constraints in dual LP 2 this gives us:

$$\sum_{S \subset V} \alpha_S y_S \leq \sum_{i,j \in V} (\max_{S:S \odot (i,j)} \alpha_S) \pi_{ij}$$

This inequality does not have y_{Sij}. Farkas lemma tells us that if we fix the y_S's so that the feasible y_{Sij}'s do not exist then we must have a function α (dual variables corresponding to the first set of inequalities in dual LP 2) so that the above inequality is violated (see Schrijver books on linear and integer programming [24], lemma 7.1.f.) Here, dual variables corresponding to the third set of inequalities are $(\max_{S:S \odot (i,j)} \alpha_S)$. Thus in essence if we enforce the above inequality for all functions α, we get the condition on those y_S's for which feasible y_{Sij}'s exist. Hence the dual LP 2 becomes:

$$\begin{aligned}
\text{maximize} \quad & \sum_{S \subset V} y_S & & (3)\\
\text{subject to} \quad & \sum_{S: e \in \delta(S)} y_S \leq c_e & \forall e \in E & \\
& \sum_{S \subset V} \alpha_S y_S \leq \sum_{i,j \in V} (\max_{S: S \odot (i,j)} \alpha_S) \pi_{ij} & \forall \alpha & \\
& y_S \geq 0 & \forall S \subset V &
\end{aligned}$$

The second set of constraints is actually infinity in number, but only a finite number of them are actually necessary. This is because the set of feasible y_S's is a polytope. Let us now try to simplify dual LP 3.

Lemma 2 *It is sufficient to consider α's having only one positive value in its range.*

Proof : Take an α. Suppose it is taking two positive values in its range. Suppose M_1 is the maximum value in its range and M_2 is the second maximum value in its range. Consider α_1 which takes value $M_1 - M_2$ whenever α takes value M_1, otherwise α_1 takes value zero. Consider $\alpha_2 = \alpha - \alpha_1$. Note that the constraint corresponding to α can be derived by adding the constraints corresponding to α_1 and α_2. Furthermore, the cardinality of the range of α_2 is decreased by 1. Rest of the proof can be completed by induction. □

Corollary 3 *It is sufficient to consider α's taking value from $\{0, 1\}$ i.e., α can be thought of denoting a family of subsets of V.*

We denote a family of subsets of V by $\mathcal{S}$. The dual LP 3 becomes:

$$\begin{aligned}
\text{maximize} \quad & \sum_{S \subset V} y_S & & (4)\\
\text{subject to} \quad & \sum_{S: e \in \delta(S)} y_S \leq c_e & \forall e \in E & \\
& \sum_{S \in \mathcal{S}} y_S \leq \sum_{(i,j) \in V \times V, \mathcal{S} \odot (i,j)} \pi_{ij} & \forall \text{ family } \mathcal{S} & \\
& y_S \geq 0 & \forall S \subset V &
\end{aligned}$$

Define a function $f : 2^{2^V} \rightarrow \mathcal{R}^+$. For every family, $\mathcal{S}$, $f(\mathcal{S})$ is defined as the right-hand side of the inequality corresponding to $\mathcal{S}$ in dual LP 4, i.e.,

$$f(\mathcal{S}) = \sum_{(i,j) \in V \times V, \mathcal{S} \odot (i,j)} \pi_{ij}.$$

Lemma 4 *f is a submodular function, i.e., $f(\mathcal{S}_1) + f(\mathcal{S}_2) \geq f(\mathcal{S}_1 \cap \mathcal{S}_2) + f(\mathcal{S}_1 \cup \mathcal{S}_2)$*

Proof : We note that for $(i, j) \in V \times V$, if $\mathcal{S}_1 \odot (i, j)$ or $\mathcal{S}_2 \odot (i, j)$, then $\mathcal{S}_1 \cup \mathcal{S}_2 \odot (i, j)$, and visa versa. However, for $(i, j) \in V \times V$, if $\mathcal{S}_1 \cap \mathcal{S}_2 \odot (i, j)$, then $\mathcal{S}_1 \odot (i, j)$ and $\mathcal{S}_2 \odot (i, j)$. The proof immediately follows from these two facts. □

Note that this lemma is analogous to the submodularity of the st-cut. Indeed this lemma is also stronger than the notion of proper functions, weak supermodularity, and weak submodularity considered in the context of Steiner edge connectivity and Steiner element connectivity problems. There we usually have two

inequalities and the function needs to satisfy one of them. The simplicity and the strength of this lemma is not a surprise. If we go back and observe the second set of inequalities in Dual LP 4, it looks like the Hall's condition for bipartite graphs. This is more formally captured as the following lemma:

Lemma 5 *Construct a bipartite graph. One side of the bipartite graph is 2^V and the other side of the bipartite graph is $V \times V$. We put a directed edge of infinite capacity from S to ij if $S \odot ij$. In addition we have two more nodes s and t. We put directed edge of infinite capacity from s to all S's. For every ij we put a directed from ij to t of capacity π_{ij}. The Dual LP 4 is maximizing the flow in this graph from s to t subject to the first set of constraint in Dual LP 4.*

Proof : Proof follows from the classical max-flow min-cut theorem.

We say that an edge $e \in E$ is *tight*, if for e, the first constraint of dual LP 4 holds with equality. We call a family $\mathcal{S} \in \mathcal{F}$ is *tight*, if for $\mathcal{S}$, the second constraint of dual LP 4 holds with equality.

Corollary 6 *Suppose y is a feasible solution to dual LP 4. Suppose the constraints corresponding to family's $\mathcal{S}_1$ and $\mathcal{S}_2$ are tight, then the constraint corresponding to family's $\mathcal{S}_1 \cup \mathcal{S}_2$ and $\mathcal{S}_1 \cap \mathcal{S}_2$ are also tight.*

Proof : By Lemma 4, since $f(\mathcal{S})$ is submodular, we have

$$\sum_{(i,j)\in V\times V:\, \mathcal{S}_1 \odot (i,j)} \pi_{ij} + \sum_{(i,j)\in V\times V:\, \mathcal{S}_2 \odot (i,j)} \pi_{ij} \geq \sum_{(i,j)\in V\times V:\, \mathcal{S}_1\cap\mathcal{S}_2 \odot (i,j)} \pi_{ij} + \sum_{(i,j)\in V\times V:\, \mathcal{S}_1\cup\mathcal{S}_2 \odot (i,j)} \pi_{ij}.$$

Since y is feasible and the left hand side is modular,

$$\sum_{S\in\, \mathcal{S}_1\cap\mathcal{S}_2} y_S + \sum_{S\in\, \mathcal{S}_1\cup\mathcal{S}_2} y_S = \sum_{S\in\, \mathcal{S}_1} y_S + \sum_{S\in\, \mathcal{S}_2} y_S$$

These two facts show that both $\mathcal{S}_1 \cap \mathcal{S}_2$ and $\mathcal{S}_1 \cup \mathcal{S}_2$ are tight. □

4 The Primal-Dual Algorithm

Now, we are ready to present a primal-dual algorithm for the PCGST problem.

Algorithm A
Input: An undirected graph $G = (V, E)$, edge costs $c_e \geq 0$ for $e \in E$, and penalty $\pi_{ij} \geq 0$ for $(i, j) \in V \times V$.
Output: A forest F', and a set of pairs $\mathcal{Q}$ not connected via F'.
begin
1 **let** $F \leftarrow \emptyset$
2 *implicitly set* $y_S \leftarrow 0$ *for all* $S \subset V$
3 **let** $\mathcal{G} \leftarrow \{\{v\} : v \in V\}$
4 **for** each $(i, j) \in V \times V$
5 **set** (i, j) *unmarked*
6 **for** each connected component $C \in \mathcal{G}$
7 **set** C *active*
8 **let** $y_C \leftarrow 0$
9 **while** there exists an *active* connected component $C \in \mathcal{G}$
10 **find** minimum ϵ_1 s.t. if we increase y_C of each *active* $C \in \mathcal{G}$ by ϵ_1 we get a new tight edge $e = \{i, j\} \in E$ with $i \in C_p \in \mathcal{G}$, $j \in C_q \in \mathcal{G}$, $C_p \neq C_q$ (see section 5 or Appendix C)
11 **find** minimum ϵ_2 s.t. if we increase y_C of each *active* $C \in \mathcal{G}$ by ϵ_1 we get a new tight family $\mathcal{S}$ (see Section 5 or Appendix C)

let $\epsilon = \min(\epsilon_1, \epsilon_2)$
let $y_C = y_C + \epsilon$ for all *active* $C \in \mathcal{G}$
if $\epsilon = \epsilon_2$
    set all pairs $(i, j) \in V \times V$ which have run out of potential *marked* (see Section 5 or Appendix C)
    set all $S \in \mathcal{S}$ *inactive*
else
    let $F \leftarrow F \cup \{e\}$
    let $\mathcal{G} \leftarrow \mathcal{G} \cup \{C_p \cup C_q\} - \{C_p\} - \{C_q\}$
    let $y_{C_p \cup C_q} \leftarrow 0$
    set $C_p \cup C_q$ *active*, except when there is no $(i, j) \in V \times V$ such that $C_p \cup C_q \odot (i, j)$
let F' is derived from F by removing as many edges as possible so that every *unmarked* pair is connected in F'
let $\mathcal{Q}$ be all pairs not connected via F'
end

Since we only grow y variables for a polynomial number of sets, checking whether an edge is tight is easy. In Section 5 and Appendix C, we show how we can find out the next tightness event for families. Note that this is especially important, since the number of families is doubly exponential.

5 Algorithm to find next event

Let y^*_S's be the current solution to Dual LP 4 and $\mathcal{S}$ be the current family of active sets. We want to find out what is the maximum ϵ we can add in all the active dual. There are two restrictions on maximum ϵ, first set of constraints and the second set of constraints in Dual LP 4. There are only linear number of first set of constraints and one can find maximum ϵ, say ϵ_1, which this set of constraints allow to add in all the active duals. So we only need to find maximum ϵ, say ϵ_2, which the second set of constraints allow to add in all the active sets. Following lemma 5 we can route y^*_S's dual in the bipartite graph described in lemma 5. We want to route additional maximum flow, which should be equal through all the active sets. We call such a flow *equal flow*, which in spirit is quite similar to the balanced flow used in [7]. The following lemma characterize the maximum equal flow (ϵ_2) through the active sets.

Lemma 7 *Construct the bipartite graph as done in lemma 5. Delete all the vertices except those which corresponds to either positive duals (according to y^*_S) or active duals. Note that we kept only polynomial number of vertices and have deleted all others. Change the capacity on edges from s to S to y^*_S, if S is not an active set; and to $y^*_S + \epsilon_2$ if S is an active set. No other change is required.*

ϵ_2 is maximum equal-flow which can be routed if and only if there are at least two minimum st-cut, one separating all active sets (in fact all sets) from s and another keeping at least one active set on the s side.

Proof : Note that y^*_S flow through inactive sets and $y^*_S + \epsilon_2$ flow through active sets can be routed, hence if these are the corresponding capacities then s separated from all other vertices is a minimum st-cut (its capacity is the same as a flow).

Suppose there is another minimum st-cut which keeps an active set on s side. Suppose there are k active sets. If we increase ϵ_2 by some amount, say δ, then the capacity of this cut will be strictly smaller than the capacity of the cut in which s is separated from all other vertices. Hence we will not be able to route additional $\epsilon_2 + \delta$ equal-flow.

On the other hand suppose all the minimum st-cuts separate all the active sets from s. In other words if we insist to keep at least one active set on the s side then the capacity of the minimum cut will be strictly more than the capacity of the actual minimum cut. Let D be the difference between these capacities. If we increase ϵ_2 by D/k then the capacity of all the minimum cut increases by D. Hence we can route an additional $\epsilon_2 + (D/k)$ equal-flow. □

This lemma gives us an iterative algorithm to find maximum equal-flow. We start with an upper bound on ϵ_2. A good upperbound is $(\sum_{ij} \pi_{ij} - \sum_S y_S^*)/k$. Set $\epsilon_2 = (\sum_{ij} \pi_{ij} - \sum_S y_S^*)/k$. We compute the minimal minimum st-cut (i.e., s side is set theoretically minimal). Suppose this minimum cut, call it T, separates all the active sets from s then we have the condition of the lemma. So assume that T separates only k' number of active sets from s, where $k' < k$. Let the capacity of T is D less than the capacity of the minimum cut which separates s from all the sets. We decrease ϵ_2 by $D/(k - k')$, which makes the capacity of T equals to the capacity of the cut which separates s from all other vertices. We compute the minimal minimum cut again. If this new cut separates s from all the other vertices then again we have the condition of the lemma. If not then the claim is that the new cut separates strictly more than k' active vertices from s. Indeed, otherwise the decrease in the capacity of cuts, which separates less than or equal to k' active vertices, is at most $k'D/(k - k')$, which is also the decrease in the capacity of T. So among these cuts T is still the minimum cut. This proves the claim.

Theorem 8 *ϵ_2 can be found by at most linear number of maxflow computations.*

The active sets which need to be freeze can also be obtained from this bipartite graph. Use ϵ_2 as found above and find the maximal minimum st-cut. All the active sets on the s side need to be frozen. The reason is if we add any further ϵ to any of the active dual on s side then S separated from all other vertices does not remain a minimum st-cut.

6 Analysis

First we consider the time complexity of the algorithm. The main loop of the algorithm terminates when all connected components of F are inactive. Since in each iteration the sum of the number of components and the number of active components decreases, the loop terminates after at most $2n - 1$ iterations. Since the most time-consuming part of the loop is finding a tight family which can be done in polynomial time according to Subsection C, the total running time of the algorithm is polynomial.

Now, we consider the approximation factor of the algorithm.

Theorem 9 *The Algorithm A outputs a forest F' and a set of pairs $\mathcal{Q}$ which are not connected via F' such that*

$$\sum_{e \in F'} c_e + \sum_{(i,j) \in \mathcal{Q}} \pi_{ij} \leq (3 - \frac{2}{n}) \sum_{S \subset V} y_S \leq (3 - \frac{2}{n})\text{OPT}$$

The second inequality is easy since y_S's form a feasible solution for the dual. The proof of the first inequality follows immediately from the following two lemmas.

Lemma 10 *The sum of penalties of marked pairs in F is at most $\sum_{S \subset V} y_S$.*

Proof : We mark a pair only if it belongs to a tight family. Using corollary 6 union of tight families is a tight family. The constraint corresponding to this tight family proves the lemma. □

Lemma 11 $\sum_{e \in F'} c_e \leq (2 - \frac{2}{n}) \sum_{S \subset V} y_S$.

Proof : The idea of the proof is similar to that of Goemans-Williamson [11]and is also not an emphasizing point of this paper. The proof is moved to the appendix D □

Finally it is worth mentioning that the ratio three between the cost of the resulting solution from Algorithm A and the sum of dual variables, i.e., $\sum_{S \subset V} y_S$ is tight. Consider the following example. Graph G contains

a path $a, s_1, \ldots, s_n, b$ where all edges of this path have length $2+\epsilon$. Also there is a node t whose distance from all vertices in the path is ∞. The penalty for pair (a,b) is ∞, the penalty for pair (s_i,t), $1 \le i \le k$ is 1, and the rest of the penalties are zero. One can easily observe that for this graph and these penalties, the algorithm stops when $\sum_{S \subset V} y_S = n + o(1)$. However the total cost of the optimal solution, which buys all edges of the path and pays the penalties 1 for pairs (s_i,t), $1 \le i \le k$, is $3n + o(1)$. Note that in this case, Algorithm A outputs the optimal solution also.

7 LP Rounding

In this section, we use the LP-rounding technique to give an approximation algorithm with ratio $\frac{1}{1-e^{-1/2}} \approx 2.54$ for this problem. First, we write the LP as follows

$$\begin{aligned}
\text{OPT} = \min & \sum_{e \in E(G)} c_e x_e + \sum_{(s,t) \in V \times V} \pi_{st} z_{st} \\
\text{such that} & \\
\sum_{e \in \delta(S)} x_e + z_{st} & \ge 1 \qquad \forall (s,t) \in V \times V \ \& \ \forall S \subset V \text{ such that} S \odot (s,t) \\
x_e & \ge 0 \qquad \forall e \in E(G) \\
z_{st} & \ge 0 \qquad \forall (s,t) \in V \times V
\end{aligned}$$

Consider a parameter $0 \le \alpha < 1$. For all pairs (s,t) with $z_{st} \ge \alpha$, we round z_{st} to one, i.e., we pay the penalties for these pairs. We call these pairs $\mathcal{Q}$. One can easily observe that if we set $x'_e = \frac{1}{1-\alpha} x_e$. Then $\sum_{e \in \delta(S)} x'_e \ge 1$ for all $(s,t) \in \mathcal{Q}$ and for all $S \subset V$ such that $S \odot (s,t)$. It means x' is a fractional solution of generalized Steiner tree for pairs in $\mathcal{Q}$. Using the 2-approximation algorithm of Goemans-Williamson we can obtain an integer solution with cost at most $2 \sum_{e \in E(G)} c_e x'_e$. It means we can obtain a solution of total cost at most $2 \sum_{e \in E(G)} \frac{1}{1-\alpha} c_e x_e + \sum_{(s,t) \in \mathcal{Q}} \pi_{st}$. Let $0 \le \beta < 1$ be a parameter to be fixed later. We choose α uniformly at random within interval $[0, \beta]$. Then we have

$$E[\text{SOL}] = E[2 \sum_{e \in E(G)} \frac{1}{1-\alpha} c_e x_e + \sum_{(s,t) \in \mathcal{Q}} \pi_{st}] = E[2 \sum_{e \in E(G)} \frac{1}{1-\alpha} c_e x_e] + E[\sum_{(s,t) \in V \times V \& z_{st} \ge \alpha} \pi_{st}]$$

$$= E[\frac{2}{1-\alpha}] \sum_{e \in E(G)} c_e x_e + \sum_{(s,t) \in V \times V} \pi_{st} \Pr[z_{st} \ge \alpha] \le (\int_0^\beta \frac{1}{\beta} \frac{2}{1-\alpha} d\alpha) \sum_{e \in E(G)} c_e x_e$$

$$+ \sum_{(s,t) \in V \times V} \pi_{st} (\int_0^{z_{st}} \frac{1}{\beta} d\alpha) = \frac{2 \ln(1/(1-\beta))}{\beta} \sum_{e \in E(G)} c_e x_e + \frac{1}{\beta} \sum_{(s,t) \in V \times V} \pi_{st} z_{st}$$

Now, by setting $\beta = 1 - e^{-1/2}$,

$$E[\text{SOL}] \le \frac{1}{1-e^{-1/2}} (\sum_{e \in E(G)} c_e x_e + \sum_{(s,t) \in V \times V} \pi_{st} z_{st}) = \frac{1}{1-e^{-1/2}} \text{OPT}$$

and thus we obtained the claim approximation ratio $\frac{1}{1-e^{-1/2}}$.

We note that here we applied the general idea of Goemans according to [25] to choose the best α for each input instead of setting it once (otherwise, we get 3-approximation). Indeed, it is not hard to show that our distribution is the best distribution for this instance of LP rounding.

8 Discussion and open problems

In this paper we see that a straight forward LP sometimes can raise cost sharing issues. These issues could be tackle more easily with a different LP which altogether could be impossible to imagine. In Appendix A we give the primal LP for dual LP 4, which is really the LP we used. As one can see Primal LP 5 is very hard to imagine directly - it has doubly exponential number of variable. We also see a systematic procedure to eliminate cost sharing issues using Farkas lemma. If we go back and see there is not a lot we used from the Goemans-Williamsons paper on constraint forest [11]. As it seems this technique of primal-dual is a general method to solve other prize collecting problems[2] Indeed this is also the case with the rounding technique we presented in this paper. Prize collecting problems appear naturally in cooperative game theory. Penalty function π can be thought of the maximum price a user is ready to pay for a shared resource. So we would like to distribute a cost of shared resource(s) among its users. As shown in numerous papers [15, 17, 22, 6] primal-dual schema has an important role to play in designing cost sharing methods. This makes our primal-dual technique even more useful.

An immediate open problem is that whether we can obtain a primal-dual algorithm for the PCGST problem with a better approximation factor (ideally two). Towards this goal we present in Appendix B another dual LP with even more combinatorial structure than dual LP 4. Another important problem is to get better approximation ratios for multicommodity connected facility location and multicommodity rent-or-buy problems. A starting point could be to simplify the proof of Gupta et. al. algorithm [13] via cost sharing. Cost sharing is closely related with dual LPs and it will be very interesting to come up with a dual LP which can give better understanding of their algorithm. The framework proposed in this paper could play an instrumental role not only in getting better understanding of their algorithm but even getting a better algorithm or a better analysis of their algorithm.

Solving the PCGST problem can be considered as the first step (see e.g. [4, 10, 16, 21]) towards obtaining constant-factor approximation algorithms for related generalized Steiner tree problems such as *k-forest*, in which we want to connect at least k given pairs of vertices with minimum cost, *budgeted generalized Steiner tree*, in which our objective is to find a set S of pairs whose generalized Steiner tree costs at most B and maximize the profit of connected pairs (here each pair has some profit instead of penalty). Even, solving (or obtaining constant factor approximation algorithms for) these problems on trees seems challenging. Considering the PCGST and related problems when we have costs on the nodes instead of edges (see e.g. [1, 12, 21]) is another interesting open area.

Finally, it is worth mentioning that if want to use the Lagrangian Relaxation technique of Jain and Vazirani [16] to obtain a constant factor approximation algorithm for the k-forest problem, since PCGST is the Lagrangian relaxation of k-forest, we need to satisfy the following condition. If the optimum solution for PCGST is $C_{\text{OPT}} + P_{\text{OPT}}$ (C is the buying cost and P is the renting cost), we should obtain an approximation algorithm with solution of total cost at most $\alpha C_{\text{OPT}} + P_{\text{OPT}}$, for some constant α (i.e. the co-efficient of P should be one). Goemans-Williamson primal dual algorithm has this property and thus can be used for k-Steiner tree. However, none of constant factor approximation algorithms mentioned in this paper for PCGST has this property (the logarithmic approximation algorithm based on the Bartal's machinery has this property). Obtaining constant factor approximation algorithms with such a property is another interesting open problem (the dual LP of Appendix B which has more combinatorial structure might be useful to this end also.)

[2] Algorithm to find tight sets in section 5 should also be applicable for a range of problems. We purpose even a more general algorithm to find tight sets in Appendix C

9 Acknowledgement

The authors would like to thank Baruch Awerbuch for his help in the early phase of our work. The authors would also like to thanks Laci Lovasz and Michel Goemans for fruitful discussions.

References

[1] A. Agrawal, P. Klein, and R. Ravi, *When trees collide: an approximation algorithm for the generalized Steiner problem on networks*, SIAM J. Comput., 24 (1995), pp. 440–456.

[2] B. Awerbuch and Y. Azar, *Buy-at-bulk network design*, in Proceedings of the 38th Annual Symposium on Foundations of Computer Science (FOCS '97), IEEE Computer Society, 1997, p. 542.

[3] Y. Bartal, *On approximating arbitrary metrices by tree metrics*, in Proceedings of the thirtieth annual ACM symposium on Theory of computing, ACM Press, 1998, pp. 161–168.

[4] A. Blum, R. Ravi, and S. Vempala, *A constant-factor approximation algorithm for the k-MST problem*, J. Comput. System Sci., 58 (1999), pp. 101–108. Twenty-eighth Annual ACM Symposium on the Theory of Computing (Philadelphia, PA, 1996).

[5] J. A. Bondy and U. S. R. Murty, *Graph Theory with Applications*, American Elsevier Publishing Co., Inc., New York, 1976.

[6] N. R. Devanur, M. Mihail, and V. V. Vazirani, *Strategyproof cost-sharing mechanisms for set cover and facility location games*, ACM Conference on Electronic Commerce, (2003), pp. 108–114.

[7] N. R. Devanur, C. H. Papadimitriou, A. Saberi, and V. V. Vazirani, *Market equilibrium via a primal-dual-type algorithm*, in Manuscript, 2003.

[8] J. Fakcharoenphol, S. Rao, and K. Talwar, *A tight bound on approximating arbitrary metrics by tree metrics*, in Proceedings of the thirty-fifth ACM symposium on Theory of computing, ACM Press, 2003, pp. 448–455.

[9] M. R. Garey and D. S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-completeness*, W. H. Freeman and Co., San Francisco, Calif., 1979.

[10] N. Garg, *A 3-approximation for the minimum tree spanning k vertices*, in 37th Annual Symposium on Foundations of Computer Science (Burlington, VT, 1996), IEEE Comput. Soc. Press, Los Alamitos, CA, 1996, pp. 302–309.

[11] M. X. Goemans and D. P. Williamson, *A general approximation technique for constrained forest problems*, SIAM J. Comput., 24 (1995), pp. 296–317.

[12] S. Guha, A. Moss, J. S. Naor, and B. Schieber, *Efficient recovery from power outage (extended abstract)*, in Proceedings of the thirty-first annual ACM symposium on Theory of computing, ACM Press, 1999, pp. 574–582.

[13] A. Gupta, A. Kumar, M. Pal, and T. Roughgarden, *Approximation via cost-sharing: a simple approximation algorithm for the multicommodity rent-or-buy problem*, in Proceedings of the 44rd Symposium on Foundations of Computer Science, IEEE Computer Society, 2003, p. 606.

[14] A. Gupta, A. Kumar, and T. Roughgarden, *Simpler and better approximation algorithms for network design*, in Proceedings of the thirty-fifth ACM symposium on Theory of computing, ACM Press, 2003, pp. 365–372.

[15] K. Jain and V. V. Vazirani, *Application of approximation algorithms to cooperative games*, 33rd Annual ACM Symposium of the theory of computing, (2001), pp. 364–372.

[16] ——, *Approximation algorithms for metric facility location and k-median problems using the primal-dual schema and Lagrangian relaxation*, J. ACM, 48 (2001), pp. 274–296.

[17] ——, *Equitable cost allocations via primal-dual-type algorithms*, 34th Annual ACM Symposium of the theory of computing, (2002), pp. 313–321.

[18] K. Jain, V. V. Vazirani, and M. Hajiaghayi, *Algorithms for perfect matching using a new primal-dual schema*. In preparation, 2004.

[19] D. R. Karger and M. Minkoff, *Building steiner trees with incomplete global knowledge*, in Proceedings of the 41st Annual Symposium on Foundations of Computer Science, IEEE Computer Society, 2000, p. 613.

[20] A. Kumar, A. Gupta, and T. Roughgarden, *A constant-factor approximation algorithm for the multicommodity*, in Proceedings of the 43rd Symposium on Foundations of Computer Science, IEEE Computer Society, 2002, p. 333.

[21] A. Moss and Y. Rabani, *Approximation algorithms for constrained node weighted steiner tree problems*, in Proceedings of the thirty-third annual ACM symposium on Theory of computing, ACM Press, 2001, pp. 373–382.

[22] M. Pal and E. Tardos, *Strategy proof mechansims via primal-dual algorithm*, 44th Annual IEEE Symposium on the Foundations of Computer Science, (2003).
[23] F. S. Salman, J. Cheriyan, R. Ravi, and S. Subramanian, *Approximating the single-sink link-installation problem in network design*, SIAM J. on Optimization, 11 (2000), pp. 595–610.
[24] A. Schrijver, *Theory of linear and integer programming*, John Wiley & Sons, New York, 1986.
[25] D. B. Shmoys, *Using linear programming in the design and analysis of approximation algorithms: two illustrative problems*, in Approximation algorithms for combinatorial optimization (Aalborg, 1998), vol. 1444 of Lecture Notes in Comput. Sci., Springer, Berlin, 1998, pp. 15–32.
[26] E. Tardos, *A strongly polynomial algorithm to solve combinatorial linear programs*, Oper. Res., 34 (1986), pp. 250–256.
[27] D. B. West, *Introduction to Graph Theory*, Prentice Hall Inc., Upper Saddle River, NJ, 1996.

A The primal LP

Our primal LP whose dual is dual LP 4 is as follows (let $\mathcal{F}$ be the powerset of the powerset of V):

$$\text{OPT} = \min \sum_{e \in E} c_e x_e + \sum_{\mathcal{S} \in \mathcal{F}} z_{\mathcal{S}} \Big(\sum_{(i,j) \in V \times V : \mathcal{S} \odot (i,j)} \pi_{ij} \Big)$$

such that

$$\sum_{e \in \delta(S)} x_e + \sum_{\mathcal{S} \in \mathcal{F}:\, S \in \mathcal{S}} z_{\mathcal{S}} \geq 1 \qquad \forall S \subset V \qquad (1)$$

$$x_e \geq 0 \qquad \forall e \in E \qquad (2)$$

$$z_{\mathcal{S}} \geq 0 \qquad \forall \mathcal{S} \in \mathcal{F} \qquad (3)$$

We note that the optimum solution of the PCGST problem is a feasible solution for the primal LP (by setting x_e one for each bought edge e and all z's zero except $z_{\mathcal{S}} = 1$ when $\mathcal{S}$ is the family corresponding to partition of connected components of bought edges (an isolated vertex is a connected component by itself).

B A more combinatorial dual LP

We say a subset $S \subset V$ is a *union of parts* in partition $\mathcal{C} = \{C_1, C_2, \cdots, C_k\}$ if and only if $S = C_{i_1} \cup C_{i_2} \cup \cdots \cup C_{i_h}$ ($h \leq k$) for $C_{i_1}, C_{i_2}, \ldots, C_{i_h} \in \mathcal{C}$.

Lemma 12 *Consider a partition $\mathcal{C} = \{C_1, C_2, \cdots, C_k\}$ of V. Let $\mathcal{F}_{\mathcal{C}} = \{S | S$ is a union of parts in $\mathcal{C}\}$. In dual LP 3, it is sufficient to consider only $\mathcal{F}_{\mathcal{C}}$ for all partitions $\mathcal{C}$ of V.*

Proof : Consider a family of sets $\mathcal{S}$. Define an equivalence binary relation on V. Two elements i and j are equivalent if they are either both present or both absent for any set $S \in \mathcal{S}$. Consider the corresponding partition for this equivalence relation. Denote it by $\mathcal{C}$. Consider the constraint corresponding to $\mathcal{F}_{\mathcal{C}}$. The right hand side of this constraint is the same as the right hand side of the constraint for $\mathcal{S}$, the left hand side has only additional terms, which are assumed to be non-negative. Hence the constraint corresponding to $\mathcal{F}_{\mathcal{C}}$ implies the constraint for $\mathcal{S}$. The proof follows. □

The dual LP 4 now becomes:

$$\text{maximize} \quad \sum_{S \subset V} y_S \qquad (5)$$

$$\text{subject to} \quad \sum_{S : e \in \delta(S)} y_S \leq c_e \qquad \forall e \in E$$

$$\sum_{S \text{ is a union of parts in C}} y_S \leq \sum_{(i,j) \in V \times V, \mathcal{C} \odot (i,j)} \pi_{ij} \quad \forall \text{ partition } \mathcal{C}$$

$$y_S \geq 0 \qquad \forall S \subset V$$

Theorem 13 *Dual LPs 1 and 5 are equivalent.*

C A more general algorithm for finding out the next event

Suppose y_S^* is the current dual solution. Suppose $\mathcal{S}$ is the family of active sets. We can find the timing of the next event by solving the following LP (note that in the following LP y_S^* are constant).

$$\begin{aligned}
\text{maximize} \quad & \epsilon & & (6)\\
\text{subject to} \quad & y_S = y_S^* + \epsilon & \forall S \in \mathcal{S} \\
& y_S = y_S^* & \forall S \notin \mathcal{S} \\
& y_S \leq \sum_{(i,j):S\odot(i,j)} y_{Sij} & \forall S \subset V \\
& \sum_{S:e\in\delta(S)} y_S \leq c_e & \forall e \in E \\
& \sum_{S:ij\odot S} y_{Sij} \leq \pi_{ij} & \forall (i,j) \in V \times V \\
& y_{Sij} \geq 0 & \forall S \subset V, ij \odot S \\
& y_S \geq 0 & \forall S \subset V
\end{aligned}$$

Note that this LP has exponentially many y_S's, but only polynomial number of those are positive (the ones we raised during the algorithm). Thus we only need to consider these y_S's and corresponding y_{Sij}'s. Hence, this is a polynomial-sized combinatorial LP, which can be solved in strongly polynomial time [26]. Solving this LP gives us the timing of the next event. Let the new y_S^*'s be the output y_S's of this program, i.e., if we solve the LP 6 we get zero as the optimum value. One can check each edge to see if there is a new edge which went tight. If there is a new edge which went tight treat it as the next event and do appropriate steps. If there is no new edge which went tight then it must be the case that some set went tight because of the penalty equation (constraint 5). Since y_S and penalties are not directly related to each other, it seems to be a difficult problem to solve. We again use a combinatorial LP to find the active sets to be made inactive. We start with all the active sets. For each active set, S, we perform the following test:

Assume only S is active. Solve the LP 6. If the optimum solution is zero then make S inactive.

Lemma 14 *There is at least one set which becomes inactive in the above test.*

Proof : Suppose this is false. So for every active set, S, there is a positive ϵ_S which can be added in y_S^* and dual LP 2 can be kept feasible. Let us say that ϵ_{min} be the smallest of all these ϵ_S's. Let the number of active sets is k. One can create k feasible solution, each for every active set S by adding ϵ_{min} in y_S^*. Average of feasible solutions is feasible. Take the average of these k feasible solutions. This can be thought as created by adding ϵ_{min}/k in all the active sets i.e., if we resolve the LP 6 we get a positive answer. A contradiction. □

Hence the test will make inactive at least one set. We can again use a similar technique to find out which pairs (i, j) have run out of potential (and thus mark them in the algorithm). We take one pair and try to maximize the surplus dual only for that pair. Suppose we done the surplus with ϵ. We maximize ϵ so that feasible y_S's remain feasible, when we replace π_{ij} by $\pi_{ij} - \epsilon$. If ϵ turns out to be positive, i and j still have potential. If ϵ turns out to be zero then i and j have run out of potential.

In the Section 5 we gave a combinatorial algorithm based on lemma 5 to find the next event. The method used in this section is still important because it could be applicable in other problems.

D Proof of lemma 11

Proof : The idea of the proof is similar to that of Goemans and Williamson [11]. Since all edges $e \in F'$ are tight, i.e., $\sum_{S\subset V:e\in\delta(S)} y_S = c_e$, we need to show that

$$\sum_{e\in F'} \sum_{S\subset V:e\in\delta(S)} y_S \leq (2 - \frac{2}{n}) \sum_{S\subset V} y_S,$$

or by rewriting terms,

$$\sum_{S \subset V} y_S |F' \cap \delta(S)| \leq (2 - \frac{2}{n}) \sum_{S \subset V} y_S.$$

We can prove this invariant by induction on the number of iterations of the main loop. Consider an iteration and the set of components of $\mathcal{G}$ in this iteration. Form a graph H in which active and inactive components are vertices and the edges are $e \in F' \cap \delta(A)$ for active A. We remove all isolated inactive vertices. Now let N_a be the set of active vertices and N_i be the set of inactive vertices in H. In this iteration, we increase the left-hand side of the inequality by $\epsilon(\sum_{v \in N_a} d(v))$, while we increase the right-hand side by $\epsilon(2 - \frac{2}{n})|N_a|$ ($d(v)$ is the degree $v \in V(H)$.) Thus we only need to show that $\sum_{v \in N_a} d(v) \leq (2 - \frac{2}{n})|N_a|$. To do this, we show that all leaves of H must be active vertices. Let v be an inactive leaf of H, adjacent to edge e, and let C_v be the inactive component corresponding to v which is deactivated at some time before. Any pair $(i, j) \in V \times V$ such that $C_v \odot (i, j)$ is marked. Furthermore, since v is a leaf, no vertex in C_v can lie on the path between vertices of an unmarked pair. Therefore, no inactive node can be a leaf. It means,

$$\sum_{v \in N_a} d(v) \leq \sum_{v \in N_a \cup N_i} d(v) - \sum_{v \in N_i} d(v) \leq 2(|N_a \cup N_i| - 1) - 2|N_i| = (2 - \frac{2}{n})|N_a|.$$

Here the inequality holds since all inactive vertices has degree at least two, and since the number of active components is always at most n. □

E PCGST on trees

Theorem 15 *The PCGST problem on trees can be solved in polynomial time.*

Proof : We reduce the problem to an instance of max flow/min cut as follows. We form a bipartite graph $G' = (X \cup Y, E')$ where there is a vertex $x_e \in X$ with weight of c_e for each edge $e \in E(G)$ (G is our original graph which is a tree), there is a vertex y_{ij} with weight π_{ij} for each $(i, j) \in V \times V$, and there is an edge from x_e to y_{ij} if and only if edge e is on the (unique) path between i and j in G. Now, it is easy to see that solving the PCGST problem on G is equivalent to find a minimum weighted vertex cover[3] in G' (for each selected vertex $x_e \in X$ we buy edge e, and for each selected vertex $y_{ij} \in Y$ we pay the penalty.) The minimum weighted vertex cover on bipartite graphs can be easily solved by the max flow/min cut theorem, see e.g. [27]. □

It is worth mentioning, by having the solution of this problem on trees, one can obtain a logarithmic approximation ratio for the PCGST problem based on Bartal's machinery [3] (or its slight improvement by Fakcharoenphol et al. [8]) for probabilistically embedding general metrics into tree metrics. Finally as mentioned in Section 8, solving the k-forest problem on trees is an important open problem.

[3] A minimum weighted vertex cover is a set of vertices with minimum weight such that for each edge, at least one of its endpoints is in the set.