Distributed primal-dual approximation algorithms for network design problems

Zeev Nutov
The Open University of Israel
nutov@openu.ac.il

Amir Sadeh*
The Open University of Israel
amirsadeh@yahoo.com

Abstract

We present a simple paradigm for efficient implementation of primal-dual approximation algorithms for several network design problems in the distributed environment. Our approximation ratios are the same as in the centralized primal-dual algorithms, and we establish that our message complexity is close to the best possible. Specifically, we show that the 2-approximation algorithm of [9] for covering an uncrossable set-family by a minimum cost set of edges can be implemented in distributed environment using $O(nD)$ communication rounds and $O(n^2 \log D)$ messages, where n is the number of nodes and D is the diameter of the communication network. In particular, within the same communication complexity we obtain:
(i) A 2-approximation for Steiner Forest, Point to Point Connection, and T-Join.
(ii) A 3-approximation for Survivable Network Design with requirements in $\{0, 1, 2\}$.
(iii) An $O(\log k)$-approximation for Steiner Network, for constant maximum requirement k.
Even for Steiner Forest, the previous best known algorithm [11] had expected ratio $O(\log n)$, while using $O(Sp \log^2 n)$ rounds and $O(|E|S \log n)$ messages, where $S \geq D$ is the "shortest path diameter" of the network, and $p = O(n)$ is the number of groups in the instance.

We also give a $2(k+1)$-approximation algorithm for k-Connected Subgraph with metric costs that uses $O(\log \log n)$ rounds and $O(|E|) = O(n^2)$ messages, which is optimal. This improves the result of [12] where was given an $O(k \log n)$-approximation algorithm that uses $O(\log \frac{n}{k})$ rounds and $O(nk \log \frac{n}{k})$ expected number of messages.

1 Introduction

1.1 Problems considered and results

Distributed approximation algorithms aim to establish a trade off between optimality of a distributed algorithm solution for the communication complexity – number of communication rounds and messages of the algorithm. Distributed algorithms inherently deal with the way nodes should

*Part of this work was done as a part of author's M.Sc. Thesis at The Open University of Israel.

exchange information in order to solve a common problem. The goal in network design problems is to design a "cheap" subnetwork (subgraph) that satisfies prescribed properties. Network connectivity is a fundamental property, naturally arising in distributed computing network design.

In the synchronous model all link delays are bounded and each processor keeps a local clock synchronized with every other clock in the network. A message sent in clock pulse p arrives before clock pulse $p+1$. So, in one communication round each node can send and receive one message from each of its neighbors and perform some *local computation*, which does not require any communication with other nodes. In the asynchronous model there is no synchronized local clock, messages arrive within some finite but unpredictable time, and messages arrival order may differ from its original transmission order. Unless stated otherwise, we assume the synchronous model when each message is restricted to $O(\log n)$ bits of information. It is straightforward to adapt our results to the general case when each message is restricted to $O(B)$ bits of information. Most of our results easily extend to the asynchronous model with invoking an $O(\log D)$ factor in the number of communication rounds, where D is the diameter of G.

In this paper we design efficient distributed primal-dual approximation algorithms for some network design connectivity problems. The instance to each of these problem contains a graph $G=(V,E)$ with edge costs $\{c(e) : e \in E\}$. Clearly, if all the data is known to some node s, then this node can apply any centralized algorithm for the problem. But usually, every node $v \in V$ knows only a "local" information about the edges incident to it and their costs. A distributed algorithm can be converted into a centralized one by sending all the information to one node s, and then distribute the solution computed by s to the nodes of the network. Assuming that the local information at each node and the solution size is $O(n)$, an attempt to convert a distributed algorithm into a centralized one requires $O(D|E|) = O(n^3)$ messages and the number of communication rounds is: $O(D|E|) = O(n^3)$ in the asynchronous model and $O(|E|) = O(n^2)$ in the synchronous model, where $n = |V|$ and D is the diameter of the communication network. A non-trivial distributed algorithm should use less rounds/messages than the one that transfers all the instance to a single node. The key idea behind our algorithms is that a total of $O(n)$ distinct messages are transferred during the algorithm (each message of size $O(\log n)$), but to all the nodes of the network.

In the next known statement, a leader election algorithm for trees is used as described in [21]; we provide a proof for completeness of exposition in Section 2.

Proposition 1.1 *Constructing an $O(D)$ diameter broadcasting tree T can be implemented using $O(n)$ rounds and $O(n \log n + |E|)$ messages. Distributing a pre-known information of N messages to all the nodes of the network over T can be done in the synchronous model using $O(N+D)$ rounds and $O(nN)$ messages. Electing a leader-minimum value over T takes $O(D)$ rounds and $O(n \log D)$ messages. In the asynchronous model, constructing T takes $O(nD)$ rounds and $O(n \log n + |E| + nD)$ messages, distributing N messages over T takes $O(ND)$ rounds and $O(nN)$ messages, and the leader election-minimum value over T takes $O(D \log D)$ rounds and $O(n \log D)$ messages.*

The generic problem we consider, which is the basis to most of our algorithms, is as follows.

Definition 1.1 *Let $\mathcal{F} \subseteq 2^V$ be a set-family of subsets of a ground-set V.*

- *$\mathcal{F}$ is* uncrossable *if $X \cap Y, X \cup Y \in \mathcal{F}$ or $X - Y, Y - X \in \mathcal{F}$ for any $X, Y \in \mathcal{F}$.*
- *An edge set I on V covers $\mathcal{F}$, or I is a $\mathcal{F}$-cover, if for every $X \in \mathcal{F}$ there is an edge in I with exactly one end-node in X.*
- *An inclusion minimal member of $\mathcal{F}$ is an $\mathcal{F}$-*core*, or simply a core if $\mathcal{F}$ is understood.*

Set-Family Edge-Cover
Instance: A graph $G = (V, E)$ with edge-costs $\{c(e) : e \in E\}$ and a set-family $\mathcal{F}$ on V.
Objective: Find a minimum cost $\mathcal{F}$-cover $I \subseteq E$.

As we consider undirected graphs, we may assume that $\mathcal{F}$ is *symmetric*, namely, that $S \in \mathcal{F}$ implies $V - S \in \mathcal{F}$. Given a partial solution I to Set-Family Edge-Cover, the *residual family* $\mathcal{F}_I$ consist from all members of $\mathcal{F}$ that are not covered by I. Goemans et al. [9] gave a 2-approximation algorithm for Set-Family Edge-Cover with uncrossable $\mathcal{F}$ provided that for any edge set I the $\mathcal{F}_I$-cores can be computed in polynomial time. In the distributed setting, we assume that:

Assumption 1:
There is a pre-known information built from $O(n)$ messages of $O(\log n)$ bits each, so that knowing this information and an edge-set $I \subseteq E$, any node $v \in V$ can locally compute the set of $\mathcal{F}_I$-cores.

Theorem 1.2 Set-Family Edge-Cover *admits a distributed 2-approximation algorithm that in the synchronous model uses $O(nD)$ rounds and $O(n^2 \log D)$ messages; in the asynchronous model the number of rounds is $O(nD \log D)$.*

An information as in Assumption 1 can be distributed to all nodes as described in Proposition 1.1. Comparing to the trivial algorithm that transfers the graph to a single node s, we use $O(nD)$ rounds versus $O(n^2)$ in the synchronous model, and $O(n^2 \log D)$ messages versus $O(n^3)$, while our ratio is the same; in the asynchronous model we use $O(nD \log D)$ rounds versus $O(n^3)$.

Several problems can be casted as Set-Family Edge-Cover with uncrossable $\mathcal{F}$. We give 3 examples. The instance to each problem contains a graph $G = (V, E)$ with edge costs $\{c(e) : e \in E\}$. Additional instance parts, objectives, and the corresponding family $\mathcal{F}$, are as follows, see [8].

Steiner Forest
Given a partition $R_1, \ldots, R_p$ of V find a minimum cost edge-set $I \subseteq E$ so that in the graph (V, I) every part belongs to the same connected component.
Here $\mathcal{F} = \{S \subseteq V : S \cap R_i \neq \emptyset, R_i - S \neq \emptyset \text{ for some part } R_i\}$.

T-Join
Given $T \subseteq V$ with $|T|$ even find a minimum cost edge-set $I \subseteq E$ so that $\deg_I(v)$ (the degree of v

w.r.t. I) is odd for every $v \in T$ and is even for every $v \in V - T$;
Here $\mathcal{F} = \{S \subseteq V : |S \cap T| \text{ is odd } \}$.

Point to Point Connection
Given disjoint subsets $P, Q \subseteq V$ with $|P| = |Q|$ find a minimum cost edge-set $I \subseteq E$ so that $|P \cap C| = |Q \cap C|$ for every connected component C of the graph (V, I).
Here $\mathcal{F} = \{S \subseteq V : |S \cap P| = |S \cap Q|\}$.

It was implicitly shown in [8] that Assumption 1 holds in each case. Thus we obtain:

Theorem 1.3 *Each one of the problems* Steiner Forest, T-Join, *and* Point to Point Connection, *admits a 2-approximation algorithm that in the synchronous model uses $O(nD)$ communication rounds and $O(n^2 \log D)$ messages; in the asynchronous model the number of rounds is $O(nD \log D)$.*

For Steiner Forest this improves the result of [11], where an $O(Sp \log^2 n)$ time algorithm with expected ratio $O(\log n)$ and message complexity $O(S|E| \log n)$ was given; here S is the so called "shortest path diameter" of the network, namely $S = \max_{u,v \in V} \ell(u, v)$ where $\ell(u, v)$ is the minimum length (number of the edges) in a cheapest uv-path. Note that $n \geq S \geq D \geq 1$.

We also establish lower bounds on the communication complexity for Steiner Forest, which does not depend on edge costs and hold even when G is a tree. Our lower bounds are valid under the following assumption on the way the problem instance is distributed among the nodes.

Assumption 2:
At the beginning, the local information available to every node v of G is: the ID of v, the edges of G incident to v, and the (ID of the) part R_i that contains v.

Assumption 1 is stronger than Assumption 2, as it allows more "global" information. Note however, that a distributed algorithm that relies on Assumption 1 has a preprocessing phase of redistributing the pre-known information (e.g., all the parts R_i) to all the nodes. Such a phase requires $\Theta(n^2)$ messages even for trees, see Proposition 1.1. Hence every algorithm that relies on Assumption 1 has implicitly a lower bound of $O(n^2)$ messages even for trees. This message complexity is dominated by other parts of our algorithm, and hence was assumed to be negligible. Assumption 2 is weaker than Assumption 1, but does not assume any preprocessing. We state our lower bound using parameters D and p.

Theorem 1.4 *For any p, D with $2p + D = n$, there exist and instance of* Steiner Forest *with the input graph G being a tree, so that under Assumption 2, the following holds. Any distributed algorithm that computes an inclusion minimal solution to the problem and uses messages of size $\leq B$ needs $\Omega(D + p/B)$ rounds in the synchronous model, $\Omega(Dp/B)$ rounds in the asynchronous model, and $\Omega(Dp/B)$ messages in both models.*

Proof: The proof reduces the following mailing problem, that has the required lower bounds, to the inclusion minimal Steiner Forest problem:

Mailing Problem: Given a sender node s and a receiver node t that are connected by a single path of length D, send a vector $Y = (b_1, \cdots, b_p)$ of p bits from s to t.

Given an instance of the Mailing Problem, construct an instance of Steiner Forest (without costs) as follows. The graph G is obtained by adding $2p$ "virtual" nodes: p nodes $s_1, .., s_p$ connected to s and p nodes $t_1, .., t_p$ connected to t. The nodes s and t will simulate all the activities of the virtual nodes in relation to running a distributed algorithm. The connectivity requirements are $r(s_i, t_i) = 1$ if $b_i = 1$ and $r(u, v) = 0$ otherwise; t will mark every t_i as belonging to part (with ID) i, and s will mark s_i as belonging to part (with ID) i only if $b_i = 1$ (if $b_i = 0$ then s_i may be marked by label $> p$). The obtained network will run the distributed Steiner Forest algorithm. The edge (t, t_i) will be chosen to the solution if, and only if, $b_i = 1$. Hence t will be able to resolve Y. □

Substituting $D = p = n/3$ and $B = O(\log n)$ in Theorem 1.4, we obtain the lower bounds: $\Omega(n^2/\log n)$ on the number of messages, and $\Omega(n^2/\log n)$ and $\Omega(n)$ on the number of rounds in the asynchronous and the synchronous model, respectively. This essentially implies that the communication complexity of our algorithm that uses $O(n^2 \log D)$ messages, and $O(nD \log D)$ rounds in the asynchronous model, are close to the optimal. It remains an open question whether our number of rounds $O(nD)$ in the synchronous case can be improved.

There are two additional problems we consider that do not admit an immediate reduction to Set-Family Edge-Cover, but use the algorithm from Theorem 1.2 as a subroutine.

Steiner Network
Given connectivity requirements $\{r(u, v) : u, v \in V\}$ find a minimum cost edge-set $I \subseteq E$ so that in the graph (V, I) there are $r(u, v)$ edge-disjoint uv-paths for every $u, v \in V$.

The Survivable Network Design (SND) problem is the same as Steiner Network, except that the paths are required to be internally node disjoint, and not only edge disjoint. We consider SND with $r(u, v) \in \{0, 1, 2\}$, and call this problem $\{0, 1, 2\}$-SND for short.

Theorem 1.5 Steiner Network *and* $\{0, 1, 2\}$-SND *admit an* $H(k)$*-approximation algorithm that uses* $O(knD)$ *rounds in the synchronous model and* $O(kn^2 \log n)$ *messages, where* $k = \max_{u,v \in V} r(u, v)$ *is the maximum requirement, and* $H(k)$ *is the* k*th harmonic number; in the asynchronous model the number of rounds is* $O(knD \log D)$*.*

Finally, we consider the k-Connected subgraph problem with metric costs (when the input graph is complete and the edge costs satisfy the triangle inequality) and prove:

Theorem 1.6 k-Connected subgraph *with metric costs admits an* $O(k)$*-approximation algorithm that uses* $O(\log \log n)$ *communication rounds and* $O(n^2)$ *messages.*

This improves the result of [12] where was given an $O(k \log n)$-approximation algorithm that uses $O(\log \frac{n}{k})$ rounds and with *expected* message complexity $O(nk \log \frac{n}{k})$.

Theorems 1.2, 1.5, and 1.6 are proved in Sections 2, 3, and 4, respectively.

The following table summarizes our and previous results for problems considered.

Problem	**Approximation**	**Rounds**	**Messages**
Set-Family Edge-cover	2	$O(nD)$	$O(n^2 \log D)$
Steiner Forest [11]	expected $O(\log n)$	$O(Sp \log^2 n)$	$O(S\|E\| \log n)$
Steiner Forest	2	$O(nD)$	$O(n^2 \log D)$
T-Join	2	$O(nD)$	$O(n^2 \log D)$
Point to Point Connection	2	$O(nD)$	$O(n^2 \log D)$
$\{0,1,2\}$-SND	3	$O(nD)$	$O(n^2 \log D)$
Steiner Network	$2H(k)$	$O(knD)$	$O(kn^2 \log D)$
Metric k-Connected Subgraph [12]	$O(k \log n)$	$O(\log \frac{n}{k})$	expected $O(nk \log \frac{n}{k})$
Metric k-connected Subgraph	$O(k)$	$O(\log \log n)$	$O(n^2)$

Table 1: Our and previous results for problems considered. Results without references are proved in this paper. For Steiner Forest p is the number of groups and S is the shortest path diameter, $n \geq S \geq D \geq 1$ For Steiner Network and k-connected Subgraph $k = \max_{u,v \in V} r(u,v)$ is the maximum requirement.

1.2 Related work

All the problems considered in this paper except T-join are NP-hard. These problems were studied extensively in the centralized setting, c.f., [8, 15, 20, 7, 19]. Both k-Connected Subgraph and Steiner Forest generalize the famous Minimum Spanning Tree problem, which was extensively studied, and for which optimal time and message complexity distributed algorithms were devised, c.f., [1, 17]. In addition, the approximate distributed version was studied by Elkin [6] et al., with some important result on the trade off between the number of rounds/messages and approximation ratios.

The primal-dual approach for distributed algorithm is not new. It was already implemented for Vertex Cover by Dubhashi, Grandoni, and Panconesi [4]. This algorithm runs in polylogarithmic time. Elkin [6] showed that for $D \geq 3$ the MST, and thus also Steiner Forest have an $\Omega(n/\log^t n)$-approximation threshold for distributed algorithm that use polylogarithmic number of rounds, where $t \geq 0$ is some universal constant. Hence unlike Vertex Cover, MST and Steiner Forest do not admit reasonable approximation ratios using polylogarithmic number of rounds.

As mentioned, the centralized versions of Set-Family Edge-Cover admits a 2-approximation algorithm [9]. Steiner Forest, T-Join, and Point to Point Connection are particular cases, hence also admit a 2-approximation algorithm [8, 22]. However, none of these primal-dual algorithms was known to admit a non-trivial distributed implementation. The only distributed algorithm for Steiner Forest

was given by [11]; they have devised a probabilistic algorithm with $O(\log n)$ expected approximation, based on a so called "least element lists" data structure, that uses $O(Sk \log^2 n)$ communication rounds and $O(|E|S \log n)$ messages.

Both Steiner Network and $\{0,1,2\}$-SND admit a 2-approximation algorithm, see [10] and [7], respectively. These algorithm are based on the iterative rounding method, that repeatedly solves linear programs. The known primal-dual approximation algorithms have ratio $2H(k)$, see [9] and [20]; note that $k = 2$ and $H(k) = 3$ for $\{0,1,2\}$-SND. We note that for general requirements SND is unlikely to admit even a polylogarithmic approximation even for very restricted instances [13, 16, 2]. However, in the case of metric costs, SND admits an $O(\log k)$-approximation algorithm [3].

The best known ratio for k-Connected Subgraph is $O(\log k \log \frac{n}{n-k})$ [19]. For metric costs, the best known ratio is $2+\frac{k}{n}$ [14]. The situation in the distributed settings is far from optimal. The only distributed algorithm for k-Connected Subgraph was for the case of metric costs [12]; this algorithm has ratio $O(k \log n)$, rounds complexity $O(\log \frac{n}{k})$, and expected message complexity $O(nk \log \frac{n}{k})$.

For centralized approximation algorithms for connectivity problems see a survey in [15].

2 Proof of Theorem 1.2

We start by proving Proposition 1.1. Initially, we compute some BFS tree T, which is used as a broadcasting tree for communication between the nodes. Note that the diameter of T is at most $2D$. This is done by electing a "leader" over a general spanning tree. The leader then commences a BFS tree T rooted to it, afterward every node knows its neighbors in T. To perform communication along T, every node broadcasts its pre-known encoded information over T; upon receiving a new information - message M by a node v, v will save locally M and will act as relay in order to continue broadcasting M over T.

MST construction due to Awerbuch [1] uses $O(n)$ rounds and $O(|E| + n \log n)$ messages. The leader election algorithm for trees (similar to rings) needs additional $O(diameter(MST)) = O(n)$ rounds and $O(n \log n)$ messages. The BFS algorithm uses $O(D)$ rounds and $O(|E|)$ messages. The tree broadcasting uses $O(D + N)$ rounds and $O(nN)$ messages. Electing a leader-minimum value over T takes $O(D)$ rounds with $O(n \log D)$ messages. We use a slightly modified version of the leader election for trees described by Santoro [21] et al. Each active node will try conqueror its 2^j neighborhood in mega phase j, after $2 \log D$ mega phases the "leader" of T knows the minimum value and will broadcast its value over T.

The asynchronous case analysis is similar. The leader election takes $O(n \log n)$ rounds, the BFS takes $O(nD)$ rounds, and the broadcasting mini-phase takes $O(Nn)$ rounds with no change in the message complexity. Electing a leader-minimum value over T takes $O(D \log D)$ rounds with $O(n \log D)$ messages; this follows immediately from the leader election for trees, c.f., [21]. This

finishes the proof of Proposition 1.1.

We briefly describe the 2-approximation algorithm of [9] for Set-Family Edge-Cover with uncrossable $\mathcal{F}$. Let $\delta(S)$ be the set of edges in E with exactly one endnode in S. Consider the following LP-relaxation **(P)** for the problem and its dual program **(D)**:

$$\begin{array}{llll} \textbf{(P)} & \min & \sum_{e\in E} c(e)x(e) & \\ & \text{s.t.} & \sum_{e\in\delta(S)} x_e \geq 1 & \forall\ S\in\mathcal{F} \\ & & x_e \geq 0 & \forall\ e\in E \end{array} \qquad \begin{array}{llll} \textbf{(D)} & \max & \sum_{S\in\mathcal{F}} y_S & \\ & \text{s.t.} & \sum_{\delta(S)\ni e} y_S \leq c_e & \forall\ e\in E \\ & & y_S \geq 0 & \forall\ S\in\mathcal{F} \end{array}$$

Given a solution y to **(D)**, an edge $e \in E$ is *tight* if the inequality in **(D)** that corresponds to e holds with equality. Equivalently, setting $\bar{c}(e) \leftarrow c(e) - \sum\{S \in \mathcal{F} : e \in \delta(S)\}y_S$ to be the *residual costs*, we have that an edge e is tight if, and only if, $\bar{c}(e) = 0$. The algorithm produces an $\mathcal{F}$-cover $I \subseteq E$ (so the characteristic vector of I is a feasible solution to **(P)**) and a solution y to **(D)** so that all the edges in I are tight. Here is the description of the algorithm. The algorithm has two phases.

Phase 1 starts with partial solution $I = \emptyset$, and applies a sequence of *iterations*. At each iteration, exactly one edge is selected to be added to I, until $\mathcal{F}_I$ has no cores, that is I is an $\mathcal{F}$-cover. Selecting an edge to add is done as follows. The algorithm maintains (implicitly) a feasible solution y for **(D)**. Initially, $y_S = 0$ for all $S \in \mathcal{F}$. Now, if I is still not an $\mathcal{F}$-cover (so there is at least one $\mathcal{F}_I$ core), we increase uniformly (possibly by zero) the dual variables y_S corresponding to $\mathcal{F}_I$-cores, until some edge $e \in E - I$ that covers some $\mathcal{F}_I$-core becomes tight; then e is added to I at this iteration.

Phase 2 applies on I "reverse delete", which means the following. Let $I = \{e_1, \ldots, e_j\}$, where e_i was added at iteration i. For $i = j$ downto 1, we delete e_i from I if $I - e_i$ is still a $\mathcal{F}$-cover. At the end of the algorithm, I is output.

In [9] it is proved that this algorithm has approximation ratio 2. Here we only need to show that under Assumption 1 the algorithm can be implemented in distributed environment using $O(nD)$ rounds and $O(n^2 \log D)$ messages.

Claim 2.1 *I is a forest at the end of Phase 1.*

Proof: Suppose to the contrary that I has a cycle C. Let $e = uv$ be the edge of C added to C last, and let I' be the accumulated partial solution before e was added. Then e covers some $\mathcal{F}_{I'}$-core S, say $u \in S$ and $v \in V - S$. However, I' contains the uv-path $C - e$, hence I' already covers S. This gives a contradiction. □

Each time an edge is added to I at Phase 1, this edge is distributed to all the nodes in the network. As the number of edges is $O(n)$, this can be implemented using $O(nD)$ rounds and $O(n^2)$

messages, by Proposition 1.1.

We describe how to implement the edge selection step in Phase 1. Every $v \in V$ calculates the residual costs of the edges in $\delta_{E-I}(v)$, the quantity $\varepsilon(v) = \min\{\bar{c}(e)/t(e) : e \in \delta_E(v)\}$, and the edge e_v for which the minimum is attained, where $t(e)$ is the number of (at most 2) $\mathcal{F}_I$-cores covered by e. Then, a leader election-minimum distinct value algorithm is applied to select among the edges $\{e_v : v \in V\}$ an edge e with $\bar{c}(e)/t(e)$ minimum; the node id's are used to break ties. After $e = e_u$ is chosen, u sends e_u over the BFS tree. Upon receiving the edge e_u, each node will add e_u to I and will update the dual variables, which are needed to calculate the residual costs. This edge selection mini-phase takes $O(D)$ rounds with $O(n \log D)$ message for one edge, by Proposition 1.1. Hence for all the $O(n)$ edges derived from Claim 2.1, $O(nD)$ rounds and $O(n^2 \log D)$ messages suffice.

Phase 2 can be implemented using local computations only. The edges and their augmentation order in I is known to every node in the network. Hence by Assumption 1 each node in the network can check locally whether a set $I - e$ (for any $e \in I$) is an $\mathcal{F}$-cover by insuring that every edge e removable from I does not create any $\mathcal{F}_{I-e}$-core. This "reverse delete" implementation takes 0 communication rounds and 0 messages.

The proof of Theorem 1.2 is complete.

3 Proof of Theorem 1.5

3.1 Algorithm for Steiner Network

We describe a variation of the (centralized) algorithm of [9] for Steiner Network. For every i the relation $\{(u, v) \in V \times V : r(u, v) \geq i\}$ is an equivalence, and let $\mathcal{R}_i$ be the partition of V into its the equivalence classes. We will assume that the requirements are given by the partitions $\mathcal{R}_1, \ldots \mathcal{R}_k$, and at the end of this section will discuss the case of pairwise requirement $r(u, v)$. Let $\mathcal{R}_0 = \{V\}$.

The algorithm has k iterations. Let $\lambda_H(u, v)$ be the maximum number of edge disjoint uv-paths in H. At the beginning of iteration i the partial solution H satisfies $\lambda_H(u, v) \geq i - 1$ for all $u, v \in \mathcal{R}_{i-1}$. During iteration i the algorithm computes an edge set I so that $\lambda_{H+I}(u, v) \geq i$ for all $u, v \in \mathcal{R}_i$. Hence after k iterations the solution H is feasible.

The augmenting edge set I is computed using the 2-approximation algorithm for Set-Family Edge-Cover described in Section 2. The corresponding set-family $\mathcal{F}$ is defined by

$$\mathcal{F} = \{S \subseteq V : S \cap R \neq \emptyset, R - S \neq \emptyset \text{ for some } R \in \mathcal{R}_i, \deg_H(S) = i - 1\} ,$$

where $\deg_H(S)$ is the number of edges in H from S to $V - S$. It is proved in [9] that this $\mathcal{F}$ is uncrossable, and that $c(I) \leq 2\mathsf{opt}/(k - i)$. This implies the approximation ratio $2H(k)$.

For the distributed implementation, all we need to show is that Assumption 1 holds during each iteration, namely, knowing an edge-set I, any node $v \in V$ can locally compute the set of $\mathcal{F}_I$-cores.

Here however we assume that the pre-known information is built from $O(kn)$ messages of $O(\log n)$ bits each. Thus we assume that at the beginning of iteration i every node knows the partial solution H computed so far. Then computing the $\mathcal{F}_I$ cores is done using standard flow methods. For every pair u, v that belongs to the same part $R \in \mathcal{R}_i$, compute in $H+I$ the maximum uv flow. If this flow has value $i-1$, compute the minimum cut C_{uv} containing u and the minimum cut C_{vu} containing v. The minimal inclusion sets among the sets computed are the $\mathcal{F}_I$-cores.

If we are given pairwise requirements $r(u,v)$, we can convert them into requirements in the partition form $\mathcal{R}_1, \ldots \mathcal{R}_k$ as follows. Note that for any equivalence class $R_j \in \mathcal{R}_i$, $r(u,v) \geq i$ holds for every $u, v \in R_j$. Hence the equivalence classes of $\mathcal{R}_i$ could be defined by the minimum node ID in each equivalence class.

$$u \in R_j,\ R_j \in \mathcal{R}_i \Longleftrightarrow j = \min\{id(u), \min_{r(u,v)\geq i,\ v\in V} id(v)\}$$

Each $u \in V$ will broadcast its subpartition identifier in $\mathcal{R}_i$ denoted by j, i.e., for all $1 \leq i \leq k$. By Proposition 1.1 this adds $nk + D$ rounds in the synchronous model and nkD rounds in the asynchronous model with a total of kn^2 messages. These number of rounds and messages are dominated by the ones of the algorithm.

3.2 Algorithm for $\{0,1,2\}$-SND

We start by presenting a centralized algorithm for $\{0,1,2\}$-SND, which is a modification of the algorithm of Ravi and Williamson [20]. The first iteration of the algorithm is the same as in the case of Steiner Network. Let H be the partial solution computed. In [20] it is proved that $c(H) \leq \mathsf{opt}$.

After resetting the costs of the edges in H to 0, we get the following "residual" problem:

Instance: A graph $H = (V, E_H)$, an edge set E on V with costs $\{c(e) : e \in E\}$, and a collection $\mathcal{D}$ of pairs of V so that every pair belongs to the same component of H.

Objective: Find a minimum cost edge set $I \subseteq E$ so that $H+I$ contains 2 internally disjoint uv-paths for every pair $\{u,v\} \in \mathcal{D}$.

We describe a construction from [18] that reduces the latter problem to Set-Family Edge-Cover with uncrossable $\mathcal{F}$. Start by modifying the instance $H, E, c, \mathcal{D}$ (see Figure 1). A node a is a *cut-node* of H if $H - a$ has more (connected) components than H. The components of $H - a$ that are not components of H are the *sides* of a. Let Q be the set of cut-nodes of H. For every $a \in Q$ with sides $A_1, \ldots, A_k$ do the following (see Figure 1): add new nodes $a_1, \ldots, a_k$, add the edges $aa_1, \ldots, aa_k$ of the weight 0 each to E_H, and for every edge $ua \in E_H \cup E$ with $u \in A_i$ replace its end-node a by a_i; the set $\mathcal{D}$ of 2-connectivity demand pairs remains the same. Clearly, the construction is polynomial. Note that only edges in E that are incident to a node in Q and have both end-nodes in the same component of H are affected. Note that for subsets of E the

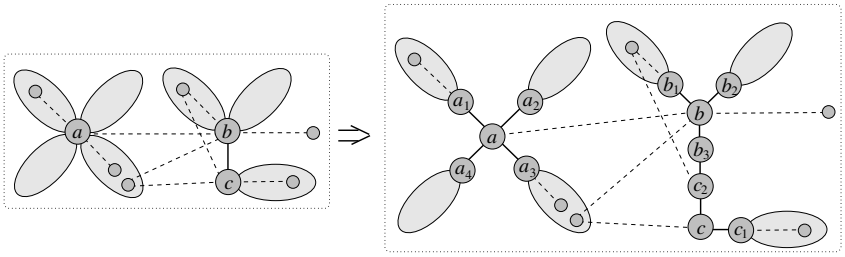

Figure 1: Modification of the residual instance. Some edges in E are shown by dashed lines. The set of original cut-nodes of H is $Q = \{a, b, c\}$ and the added nodes are $a_1, a_2, a_3, a_4, b_1, b_2, b_3, c_1, c_2$.

transformation is cost preserving, since all original edges in E keep their weights, while the added edges have weight 0. It is also easy to see that $I \subseteq E$ is a feasible solution to the original instance if, and only if, I is a feasible solution to the modified instance; the weight of I is the same in both instances. Henceforth $H = (V, E_H), E, c, \mathcal{D}$ is the modified instance, and Q is the set of original cut-nodes. We now define our family $\mathcal{F}$ on this modified instance.

Definition 3.1 *A* set-pair *is a partition $\{X, X'\}$ of $V - a$ for some $a \in Q$ so that no edge in E_H connects X and X'. A set-pair $\{X, X'\}$ is* violated *if there is a demand pair $\{x, x'\} \in \mathcal{D}$ so that $x \in X$ and $x' \in X'$. A set $X \subseteq V$ is violated if it is a part of some violated set-pair. Let $\mathcal{F}^+$ be the family of all violated sets, let $\mathcal{F}^- = \{V - X : X \in \mathcal{F}^+\}$, and let $\mathcal{F} = \mathcal{F}^+ \cup \mathcal{F}^-$.*

Note that $X \in \mathcal{F}^+$ if, and only if, $V - X \in \mathcal{F}^-$, thus $\mathcal{F}$ is *symmetric*. It is a routine to show that $I \subseteq E$ is a feasible solution for the modified instance if, and only if, I covers $\mathcal{F}$.

Lemma 3.1 ([18]) *The family $\mathcal{F}$ in Definition 3.1 is uncrossable.*

Hence we can apply the 2-approximation algorithm of [9] to compute an edge set $I \subseteq E$ so that $H + I$ is a feasible solution to $\{0, 1, 2\}$-SND.

Now we describe the distributed implementation of the algorithm. As in Steiner Network, let $\mathcal{R}_1$ be the be the partition of V into the equivalence classes of the relation $\{(u, v) \in V \times V : r(u, v) \geq i\}$. A collection $\mathcal{R}_2$ of subsets of V is defined as the blocks (2-node connectivity components) of the graph (V, F) where $F = \{uv : r(u, v) = 2\}$. It is well known that $\sum_{R \in \mathcal{R}_2} |R| \leq 2n - 1 = O(n)$. Thus we can apply Proposition 1.1 to distribute the collection $\mathcal{R}_2$ of sets to all the nodes of the network within the claimed communication complexity.

As in the centralized algorithm the distributed implementation has two iteration. Both iterations will run the Set-Family Edge-Cover algorithm described in Section 2. Iteration 1 runs the Steiner Forest algorithm as in theorem 1.3, so the first iteration is identical to the first iteration in the Steiner Network algorithm. Iteration 2 will run the Set-Family Edge-Cover algorithm on the the family $\mathcal{F}$ as in Definition 3.1. At the beginning of iteration 2, every node will modify its locally constructed graph H from iteration 1 according to the reduction described above (Figure 1). To

run the Set-Family Edge-Cover algorithm with the family $\mathcal{F}$ as in Definition 3.1, every cut-node of H will simulate locally the activities of its copies created by the reduction. The graph H constructed by the reduction has $O(n)$ nodes/edges, and $\sum_{R\in\mathcal{R}_2}|R| = O(n)$. This is the pre-known information used by every node. It is not hard to verify that knowing $H, \mathcal{R}_2$ and the partial solution I, any node $v \in V$ can locally compute the set of $\mathcal{F}_I$-cores. Hence Assumption 1 holds, and the communication complexity of iteration 2 is as in Theorem 1.2.

The proof of Theorem 1.5 is complete.

4 Proof of Theorem 1.6

We start by describing a centralized $2(k+1)$-approximation algorithm for k-Connected Subgraph with metric costs. This algorithm is known, and we provide proofs only for completeness of exposition. Given a spanning tree T in a metric graph G, a well known heuristic constructs a Hamiltonian cycle H of cost $c(H) \leq 2c(T)$ in linear time. The centralized algorithm is as follows.

1. Compute a minimum spanning tree T in G.
2. Construct a Hamiltonian cycle H_T of cost $c(H_T) \leq 2c(T)$, and label the nodes by $1,\ldots,n$ according to their order in H_T.
3. Obtain a graph H by connecting each node i to the nodes $i+1, i+2, \ldots, i+\min\{k, n-i\}$.

Lemma 4.1 *The algorithm computes a k-connected graph H of costs $c(H) \leq k(k+1)/2 \cdot c(H_T)$.*

Proof: We prove that H is k-connected. It is sufficient to prove that if H is a graph on $V = \{1,\ldots,n\}$ with $n \geq k+1$ so that every $i \leq n-1$ has at least $\min\{k, n-i\}$ neighbors in $\{i+1,\ldots,n\}$, then H is k-connected. If $n = k+1$ then H is a complete graph. Assume that $n \geq k+2$ and suppose to the contrary that G is not k-connected. Then there is $C \subseteq V$ with $|C| \leq k-1$ so that $G-C$ is disconnected. Let X, Y be distinct connected component of $G-C$ with $n \notin X$, and let $i_X = \max_{i\in X} i$ and $i_Y = \max_{i\in Y} i$. We must have $\{i_X+1,\ldots,n\} \subseteq C$; hence $i_X > i_Y$ and $n \notin Y$. Thus the same argument applied on Y gives $i_Y > i_X$. This is a contradiction.

We prove that $c(H) \leq k(k+1)/2 \cdot c(H_T)$. Since the costs are metric, $c(i,j) \leq \sum_{\ell=i}^{j-1} c(\ell, \ell+1)$ for every $i < j \leq n$. By the construction, if $ij \in H$ then $i - j \leq k$. Thus

$$\sum_{ij\in H} c(i,j) \leq \sum_{ij\in H} \sum_{\ell=i}^{j-1} c(\ell,\ell+1) \leq [k + (k-1) + \cdots + 1] \cdot c(H_T) = k(k+1)/2 \cdot c(H_T) \ .$$

□

Proposition 4.2 *Any k-edge-connected graph H with edge costs $c(e)$ has a spanning tree T with $c(T) \leq 2c(H)/k$.*

Proof: Edmonds [5] proved, that if a directed graph has k edge disjoint paths from r to any other node, then it contains k edge-disjoint arborescences rooted at r. Thus the bidirection D of H obtained by replacing every (undirected) edge $e = uv$ by two opposite directed edges uv, vu of the same cost as e, contains k edge-disjoint arborescences rooted at r. Let T be the underlying tree of the least cost arborescence among them. Then $c(T) \leq c(D)/k = 2c(H)/k$. □

Corollary 4.3 *The algorithm computes a k-connected graph H of costs $c(H) \leq 2(k+1) \cdot \mathsf{opt}$.*

Proof: We have $c(H) \leq k(k+1)/2 \cdot c(H_T) \leq k(k+1)c(T) \leq k(k+1) \cdot 2\mathsf{opt}/k = 2(k+1)\mathsf{opt}$, by Lemma 4.1 and Proposition 4.2. □

Now we describe a distributed implementation of the algorithm. Assume every node $v \in V$ has a unique ID $id(v) \in \{1, 2, \cdots, n\}$. The implementation is as follows.

Step 1 is implemented by applying the algorithm of [17] that uses $O(\log \log n)$ rounds and $O(n^2)$ messages. (This does not contradicts the lower bound by Elkin [6] since Elkin's lower bound is valid only for $D \geq 3$, while in a complete graph $D = 1$).

Step 2 is implemented as follows. The network will transmit the MST to the network leader which is the node with the maximum $id(\cdot)$. This could be done using $O(1)$ rounds and $O(n^2)$ messages. In order to choose the leader every node v sends its $id(v)$ over all its links; this takes only one round and $O(n^2)$ messages. Afterwords every node can compute locally who is the leader. In order to send the MST edges to the leader every node $v \in V$ will send the $deg_T(v)$ to the leader. The leader then will assign to every v exactly $\deg_T(v)/2$ distinct nodes that will help him to send its edges (as relays) to the leader with additional $O(1)$ rounds and $O(n)$ messages for all the nodes in the network altogether. The complexities follows.

After the MST arrives to the leader, the leader will compute the Hamiltonian cycle and the new node labels locally. Afterward the leader will send to each node its new label. This computation takes 1 round with $O(n)$ messages.

Step 3 is implemented as follows. Each node will send to every node in the network its new label; this computation takes 1 round with $O(n)$ messages. Then each node with label i will update locally its local edge tables so that it will be connected to the nodes labeled with $i+1, i+2, \ldots, i+\min\{k, n-i\}$ and to the nodes $i-1, i-2, \ldots, i-\max\{k, 1\}$ (because the network is undirected).

The proof of Theorem 1.6 is complete.

References

[1] B. Awerbuch. Optimal distributed algorithms for minimum weight spanning tree, counting, leader election, and related problems. In *STOC*, pages 230–240, 1987.

[2] T. Chakraborty, J. Chuzhoy, and S. Khanna. Network design for vertex connectivity. In *STOC*, pages 167–176, 2008.

[3] J. Cheriyan and A. Vetta. Approximation algorithms for network design with metric costs. In *STOC*, pages 167–175, 2005.

[4] D. Dubhashi, F. Grandoni, and A. Panconesi. Distributed Approximation Algorithms via LP - Duality and Randomization. In T. F. Gonzalez, editor, Chapter 13 in *Handbook of Approximation Algorithms and Metaheuristics*. Chapman & Hall/CRC, 2007.

[5] J. Edmonds. Matroid intersection. *Annals of discrete Mathematics*, pages 185–204, 1979.

[6] M. Elkin. Unconditional lower bounds on the time-approximation tradeoffs for the distributed minimum spanning tree problem. In *STOC*, pages 331–340, 2004.

[7] L. Fleischer, K. Jain, and D. P. Williamson. Iterative rounding 2-approximation algorithms for minimum-cost vertex connectivity problems. *J. Comput. Syst. Sci.*, 72(5):838–867, 2006.

[8] M. Goemans and D. P. Williamson. Primal-dual. In D. S. Hochbaum, editor, Chapter 4 in *Approximation Algorithms for NP-hard problems*, pages 144–190. PWS, 1995.

[9] M. X. Goemans, A. V. Goldberg, S. Plotkin, D. B.Shmoys, E. Tardos, and D. P. Williamson. Improved approximation algorithms for network design problems. In *SODA*, pages 223–232, 1994.

[10] K. Jain. A factor 2 approximation algorithm for the generalized Steiner network problem. *Combinatorica*, 21(1):39–60, 2001.

[11] M. Khan, F. Kuhn, D. Malkhi, G. Pandurangan, and K. Talwar. Efficient distributed approximation algorithms via probabilistic tree embeddings. In *PODC*, pages 263–272, 2008.

[12] M. Khan, G. Pandurangan, and V. S. Anil Kumar. A simple randomized scheme for constructing low-weight k-connected spanning subgraphs with applications to distributed algorithms. *Theor. Comput. Sci.*, 385(1-3):101–114, 2007.

[13] G. Kortsarz, R. Krauthgamer, and J. R. Lee. Hardness of approximation for vertex-connectivity network design problems. *SIAM Journal on Computing*, 33(3):704–720, 2004.

[14] G. Kortsarz and Z. Nutov. Approximating node connectivity problems via set covers. *Algorithmica*, 37:75–92, 2003.

[15] G. Kortsarz and Z. Nutov. Approximating minimum-cost connectivity problems. In T. F. Gonzalez, editor, Chapter 58 in *Approximation Algorithms and Metaheuristics*. Chapman & Hall/CRC, 2007.

[16] Y. Lando and Z. Nutov. Inapproximability of survivable networks. In *APPROX-RANDOM*, pages 146–152, 2008.

[17] Z. Lotker, E. Pavlov, B. Patt-Shamir, and D. Peleg. MST construction in $O(\log \log n)$ communication rounds. In *SPAA*, pages 94–100, 2003.

[18] Z. Nutov. Approximating steiner networks with node weights. In *LATIN*, pages 411–422, 2008.

[19] Z. Nutov. An almost $O(\log k)$-approximation for k-connected subgraphs. In *SODA*, 2009. To appear.

[20] R. Ravi and D. P. Williamson. An approximation algorithm for minimum-cost vertex-connectivity problems. *Algorithmica*, 18:21–43, 1997.

[21] N. Santoro. *Design and Analysis of Distributed Algorithms*. Wiley-Interscience, 2006.

[22] D. P. Williamson. *On the design of approximation algorithms for a class of graph problems*. PhD thesis, Massachusetts Institute of Technology, Cambridge, MA, USA, 1993.