Kruskal-based approximation algorithm for the multi-level Steiner tree problem

Reyan Ahmed, Faryad Darabi Sahneh, Keaton Hamm, Stephen Kobourov, Richard Spence

University of Arizona

ESA 2020

September 7-9, 2020

Introduction

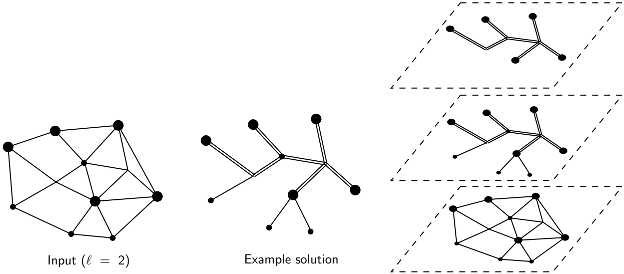
- Let $G(V, E)$ be an undirected graph, and let $T \subseteq V$ be a subset of terminals. A *Steiner tree* of G over T is a subtree which spans T

Introduction

- Let $G(V, E)$ be an undirected graph, and let $T \subseteq V$ be a subset of terminals. A *Steiner tree* of G over T is a subtree which spans T
- Computing a minimum-weight Steiner tree is NP-hard
- Simple $\left(2 - \frac{2}{|T|}\right)$-approximation; approximable with ratio $\rho \approx 1.39$ [3]
- NP-hard to approximate with ratio $\frac{96}{95} \approx 1.01$ [5]

Introduction

- We study a generalization of the Steiner tree problem in graphs where terminals require one of ℓ priorities, levels, or quality of service (QoS) requirements
- Motivated by multi-level graph visualization, we call this the *multi-level Steiner tree (MLST) problem*
- Also known in literature as *multi-level network design*, *Quality of Service Multicast tree*, *Priority Steiner tree*, *Multi-Tier Tree*, etc.



Input (ℓ = 2)

Example solution

Problem Definition

Definition 1 (Multi-level Steiner tree (MLST))

Let $G = (V, E)$ be a connected graph, and $T \subseteq V$ be a subset of terminals. Each terminal $t \in T$ has a priority $P(t) \in \{1, 2, \ldots, \ell\}$. A multi-level Steiner tree (MLST) is a tree G' with edge rates $y(e) \in \{1, 2, \ldots, \ell\}$ such that for any two terminals u, $v \in T$, the u–v path in G' uses edges of rate greater than or equal to $\min\{P(u), P(v)\}$.

We use ℓ for the highest priority or level, and 1 for the lowest.

Problem Definition

Reyan Ahmed, Faryad Darabi Sahneh, Keaton Hamm, Stephen Kobourov, Richard Spence

- Let $c_i(e)$ denote the cost of including edge e with rate i.

Problem Definition

- Let $c_i(e)$ denote the cost of including edge e with rate i.
- The cost of a solution is defined as the sum of all edge costs at their respective rates $y(\cdot)$, i.e.,

$$\text{Cost} = \sum_{e \in E(G')} c_{y(e)}(e).$$

Problem Definition

Reyan Ahmed, Faryad Darabi Sahneh, Keaton Hamm, Stephen Kobourov, Richard Spence

- Let $c_i(e)$ denote the cost of including edge e with rate i.
- The cost of a solution is defined as the sum of all edge costs at their respective rates $y(\cdot)$, i.e.,

$$\text{Cost} = \sum_{e \in E(G')} c_{y(e)}(e).$$

- Edge costs *proportional* if $c_i(e) = i \cdot c_1(e)$ for all $e \in E$

Problem Definition

- Let $c_i(e)$ denote the cost of including edge e with rate i.
- The cost of a solution is defined as the sum of all edge costs at their respective rates $y(\cdot)$, i.e.,

$$\text{Cost} = \sum_{e \in E(G')} c_{y(e)}(e).$$

- Edge costs *proportional* if $c_i(e) = i \cdot c_1(e)$ for all $e \in E$

MLST problem: Given a graph $G = (V, E)$, terminals T, and priorities, compute a minimum cost MLST.

Known Results

- The MLST problem admits constant-approximations if the edge costs are proportional, e.g., [4, 8, 1]
  - Charikar et al. [4] give 4ρ- (Alg. C_1) and $e\rho$-approximations with proportional costs, which were improved subsequently in [8, 1]
- Approximable with ratio $O(\log|T|)$ with unrestricted edge costs [4]
- Admits no $c \log\log n$-approximation unless $\mathrm{NP} \subseteq \mathrm{DTIME}(n^{O(\log\log\log n)})$ [6]

Our Contributions

Reyan Ahmed, Faryad Darabi Sahneh, Keaton Hamm, Stephen Kobourov, Richard Spence

1. We consider generalizations of Kruskal's and Prim's algorithms (for MSTs) to the multi-level or multi-priority problem
2. We describe a $(2 \ln |T|)$-approximation algorithm for the MLST problem based on Kruskal's algorithm, which matches the best-known approximation ratio by Charikar et al. [4]. Interestingly, a Prim-based generalization could perform arbitrarily poorly
3. We provide an experimental comparison between these two logarithmic approximations and compare the quality of the MLSTs returned with respect to $|V|$, ℓ using a variety of graph generation methods

$O(\log|T|)$-approximation by Charikar et al.

- Charikar et al. [4] give a simple $2(\ln|T|+1)$-approximation (which we call Algorithm $\mathrm{C}_{2\mathrm{a}}$), which greedily connects terminals one by one in order of priority

$O(\log|T|)$-approximation by Charikar et al.

- Charikar et al. [4] give a simple $2(\ln|T|+1)$-approximation (which we call Algorithm $\mathrm{C}_{2\mathrm{a}}$), which greedily connects terminals one by one in order of priority
  - Sort the terminals in non-increasing priority
  - For each terminal v, connect it to the existing tree using the cheapest possible path using edges of rate $P(v)$

Kruskal-based approximation algorithm

Instead of connecting terminals by order of priority, we instead connect the "cheapest" pair (u, v) of terminals at each iteration

- While all terminals are not yet connected, find the pair of terminals u, v which minimizes the cost of connecting them using edges of rate $\min\{P(u), P(v)\}$
- Add the corresponding u-v path; remove the lower-priority terminal from the set of terminals

This is also a $2 \ln |T|$-approximation, independent of ℓ

ILP

$$\text{Minimize} \quad \sum_{i=1}^{\ell} \sum_{(u,v)\in E} c_i'(u,v) x_{uv}^i \text{ subject to} \tag{1}$$

$$\sum_{(v,w)\in E} f_{vw}^i - \sum_{(u,v)\in E} f_{uv}^i = \begin{cases} |T_i| - 1 & \text{if } v = r \\ -1 & \text{if } v \in T_i \setminus \{r\} \\ 0 & \text{else} \end{cases} \tag{2}$$

$$x_{uv}^i \leq x_{uv}^{i-1} \tag{3}$$

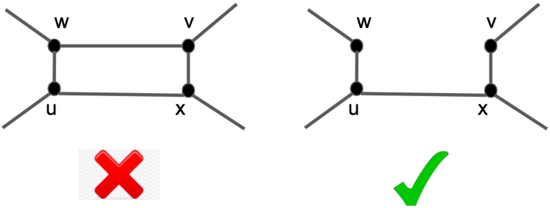
$$0 \leq f_{uv}^i \leq (|T_i| - 1) \cdot x_{uv}^i \tag{4}$$

$$x_{uv}^i \in \{0, 1\} \tag{5}$$

ILP

Reyan Ahmed, Faryad Darabi Sahneh, Keaton Hamm, Stephen Kobourov, Richard Spence

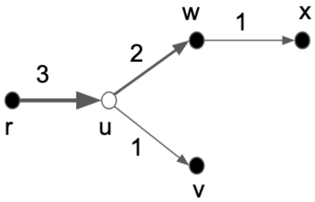
- The objective is to minimize $\sum_{i=1}^{\ell} \sum_{(u,v)\in E} c_i'(u,v)x_{uv}^i$
- Here, $c_i'(u,v)$ is the incremental cost of edge (u,v) with rate i
- Hence, $c_i'(u,v) = c_i(e) - c_{i-1}(e)$ where $e = (u,v)$ and $c_0(e) = 0$
- The variable $x_{uv}^i = 1$ if (u,v) appears in the solution with rate greater than or equal to i, and 0 otherwise.



ILP

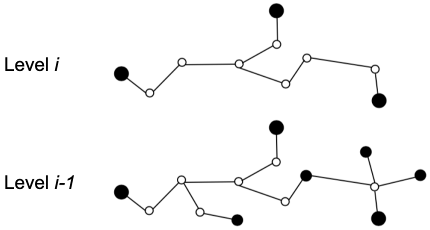
$$\sum_{(v,w)\in E} f^i_{vw} - \sum_{(u,v)\in E} f^i_{uv} = \begin{cases} |T_i| - 1 & \text{if } v = r \\ -1 & \text{if } v \in T_i \setminus \{r\} \\ 0 & \text{else} \end{cases} \quad \forall v \in V; 1 \le i \le \ell \tag{2}$$

- For every edge $e = (u, v)$ we define two flow variables f^i_{uv} and f^i_{vu}.
- The flow constraint ensures that the edges of rate greater than or equal to i form a Steiner tree over T_i.



ILP

Reyan Ahmed, Faryad Darabi Sahneh, Keaton Hamm, Stephen Kobourov, Richard Spence



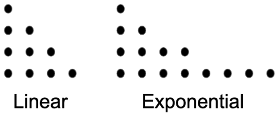
$$x^i_{uv} \leq x^{i-1}_{uv} \qquad \forall (u,v) \in E; 2 \leq i \leq \ell \qquad (3)$$

$$0 \leq f^i_{uv} \leq (|T_i| - 1) \cdot x^i_{uv} \qquad \forall (u,v) \in E; 1 \leq i \leq \ell \qquad (4)$$

$$x^i_{uv} \in \{0,1\} \qquad \forall (u,v) \in E; 1 \leq i \leq \ell \qquad (5)$$

Experimental Results

- We consider two types of edge costs:
  - Proportional
  - Non-proportional
- We consider four types of graph generation:
  - Erdős–Rényi (ER) [7]
  - Watts–Strogatz (WS) [10]
  - Barabási–Albert (BA) [2]
  - SteinLib instances [9]
- The number of priorities $\ell \in \{2, \ldots, 7\}$
- We adopt two methods for selecting terminal sets:
  - Linear
  - Exponential



Experimental Results

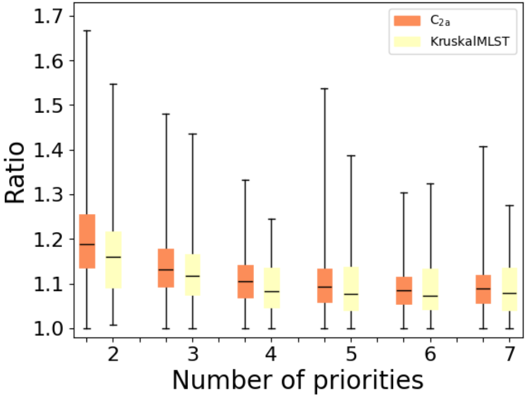


Figure: Performance of $\mathrm{C_{2a}}$ [4] and $\mathrm{KruskalMLST}$ on Erdős–Rényi graphs w.r.t. ℓ with non-proportional edge weights. There are 190 instances for a specific ℓ.

Experimental Results

Reyan Ahmed, Faryad Darabi Sahneh, Keaton Hamm, Stephen Kobourov, Richard Spence

Graph Generator	ER		WS		BA		SteinLib	
Algorithm	C_1	K	C_1	K	C_1	K	C_1	K
Equal to OPT	73	**133**	391	**679**	94	**202**	4	**8**
Mean	1.048	**1.044**	1.016	**1.012**	1.028	**1.021**	1.2355	**1.1918**
Median	1.044	**1.037**	1.006	**1.0**	1.019	**1.016**	1.2072	**1.1707**
Min	**1.0**	**1.0**	**1.0**	**1.0**	**1.0**	**1.0**	**1.0**	**1.0**
Max	1.263	**1.202**	1.31	**1.18**	1.212	**1.126**	1.7488	**1.6404**
Best Approx.	40.53%	**54.29%**	24.92%	**50.78%**	30.62%	**69.38%**	31.50	**59.12%**

Table: Statistics of Algorithms C_1 [4] and $\mathrm{KruskalMLST}$ (abbreviated K) with proportional edge cost. Best Approx. reports the percentage of instances (out of 1140) that each algorithm achieved strictly better experimental approximation ratio. Best performance in each category is bolded. The statistics correspond to the experimental approximation ratio.

Experimental Results

Graph Generator	ER		WS		BA	
Algorithm	C_{2a}	K	C_{2a}	K	C_{2a}	K
Equal to OPT	16	**26**	16	**30**	10	**26**
Mean	1.123	**1.109**	1.099	**1.081**	1.121	**1.097**
Median	1.109	**1.099**	1.087	**1.067**	1.096	**1.08**
Min	**1.0**	**1.0**	**1.0**	**1.0**	**1.0**	**1.0**
Max	1.667	**1.54**	1.863	**1.601**	1.941	**1.667**
Best Approx.	37.20%	**61.22%**	34.83%	**63.85%**	30.62%	**68.24%**

Table: Statistics of Algorithms C_{2a} [4] and KruskalMLST (abbreviated K) with non-proportional edge cost. Best Approx. reports the percentage of instances (out of 1140) that each algorithm achieved strictly better experimental approximation ratio. Best performance in each category is bolded.

Experimental Results

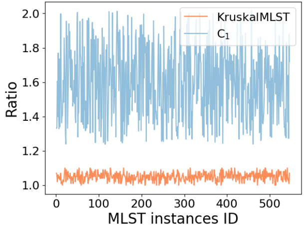


Figure: A class of graphs for which the Algorithm $\mathrm{KruskalMLST}$ significantly outperforms Algorithm C_1 [4]. The *x*–axis is the instance number and carries no meaning of time; the *y*–axis is the approximation ratio.

Conclusion and Future work

Reyan Ahmed, Faryad Darabi Sahneh, Keaton Hamm, Stephen Kobourov, Richard Spence

- We showed that the Kruskal-based algorithm is a logarithmic approximation, matching the best approximation guarantee of Charikar et al. [4]
- Formulated an ILP for the general MLST problem
- We demonstrated that $\mathrm{KruskalMLST}$ compares favorably to other algorithms
- A natural question is whether the analysis of any of these algorithms $\mathrm{GreedyMLST}$, $\mathrm{KruskalMLST}$, or $\mathrm{C_{2a}}$ can be tightened

Abu Reyan Ahmed, Patrizio Angelini, Faryad Darabi Sahneh, Alon Efrat, David Glickenstein, Martin Gronemann, Niklas Heinsohn, Stephen Kobourov, Richard Spence, Joseph Watkins, and Alexander Wolff.
Multi-level Steiner trees.
In *17th International Symposium on Experimental Algorithms, (SEA)*, pages 15:1–15:14, 2018.

Albert-László Barabási and Réka Albert.
Emergence of scaling in random networks.
science, 286(5439):509–512, 1999.

Jaroslaw Byrka, Fabrizio Grandoni, Thomas Rothvoß, and Laura Sanità.
Steiner tree approximation via iterative randomized rounding.
J. ACM, 60(1):6:1–6:33, 2013.

M. Charikar, J. Naor, and B. Schieber.

Resource optimization in QoS multicast routing of real-time multimedia.
IEEE/ACM Transactions on Networking, 12(2):340–348, April 2004.

Miroslav Chlebík and Janka Chlebíková.
The Steiner tree problem on graphs: Inapproximability results.
Theoret. Comput. Sci., 406(3):207–214, 2008.

Julia Chuzhoy, Anupam Gupta, Joseph (Seffi) Naor, and Amitabh Sinha.
On the approximability of some network design problems.
ACM Trans. Algorithms, 4(2):23:1–23:17, 2008.

Paul Erdös and Alfréd Rényi.
On random graphs, i.
Publicationes Mathematicae (Debrecen), 6:290–297, 1959.

Marek Karpinski, Ion I. Mandoiu, Alexander Olshevsky, and Alexander Zelikovsky.

Improved approximation algorithms for the quality of service multicast tree problem.
Algorithmica, 42(2):109–120, 2005.

T. Koch, A. Martin, and S. Voß.
SteinLib: An updated library on Steiner tree problems in graphs.
Technical Report ZIB-Report 00-37, Konrad-Zuse-Zentrum für Informationstechnik Berlin, Takustr. 7, Berlin, 2000.

Duncan J Watts and Steven H Strogatz.
Collective dynamics of ‘small-world’networks.
nature, 393(6684):440, 1998.