Compact Representations of Graphs and Their Metrics

(Thesis Proposal)

D Ellis Hershkowitz

Compute Science Department
Carnegie Mellon University
dhershko@cs.cmu.edu

Thesis Committee

Bernhard Haeupler (CMU, Co-Chair)
R. Ravi (CMU, Co-Chair)
Anupam Gupta (CMU)
Michel Goemans (MIT)
Ola Svensson (EPFL)

Abstract

Graphs and metrics are two of the most ubiquitous, versatile and powerful tools in modern computing. Both are general enough to be widely applicable but structured enough to facilitate efficient algorithms. However, the modern proliferation of data has led to graphs and metrics of practical importance which are of unprecedented size. For this reason, now more than ever, understanding how to sparsify or otherwise effectively reduce the size of a graph or metric is of great importance. We propose new results in graph and metric sparsification using tools from combinatorial optimization, metric embeddings, approximation algorithms and online algorithms.

In the first part of this thesis proposal we provide new results on and directions for classical questions in graph and metric sparsification. First, we propose a new sort of tree embedding of metrics which embeds each point in a metric into boundedly-many copies. Using these embeddings we give the first non-trivial deterministic algorithm for online group Steiner tree and the demand robust version of many Steiner problems. Second, we give a new attack and preliminary results on one of the most basic sparsification questions in directed graphs: how to find a minimum cardinality subgraph which preserves all connectivity relationships between vertices.

In the second part of this proposal we investigate how the interactions between a graph and its induced metric affect compact representations. First, we study how to overcome the challenges of meaningfully incorporating graph structure in the form of hop-constraints into metric embeddings and network design problems. In particular, we give the first tree embeddings for the hop-constrained distances in a graph. Using these embeddings we give the first poly-log bicriteria algorithms for the hop-constrained version of many classic network design problems. Second, we investigate how graph structure can make metric sparsification easier. In particular, we study the Steiner point removal problem where we must vertex-sparsify a graph from a structured family. We give the first $O(1)$ distortion Steiner point removal solutions on series-parallel graphs as well as new approaches for Steiner point removal in planar graphs.

Contents

1 Introduction

Graphs and metrics are two of the most versatile tools in modern computing. Both are general enough to find wide-reaching applications but sufficiently structured enough to form the foundation of many algorithms. Indeed the rich and beautiful structure of graphs and metrics has driven algorithmic advances in areas as varied as computational biology, distributed computing, machine learning and chip design. In large part these applications are made possible by the the ability of graphs and metrics to model costs, congestion and distances in a variety of networks—including protein, transportation, manufacturing, computer, social and epidemiological networks. More generally, their ability to mathematically formalize a notion of "connectedness" and "closeness" lays the foundation for many applications such as how graphs and metrics form the basis of a great deal of work on planning and knowledge representation in AI.

While graphs and metrics form the foundation of much of modern computing, a proliferation of data has led to modern graphs and metrics that dwarf their predecessors. For instance, the social and knowledge graphs of companies like Google and Facebook have billions of vertices and hundreds of billions of edges [20]. Similarly, a flurry of work in molecular biology has led to the discovery of genomic and protein networks of interest whose interactions are described by singularly massive graphs [17]. Thus, modern algorithms for graphs and metrics are faced with the daunting task of computing over huge and complex inputs.

One of the most powerful and flexible tools for meeting the algorithmic challenges posed by massive graphs and metrics is sparsification and compression. Here, otherwise intractably large graphs and metrics are sparsified or otherwise made concise while preserving salient properties of the input. As exact compressions of a graph and metric are often impossible, these techniques often only approximately preserve certain properties of the graph. Applying these strategies can make otherwise intractably large problems significantly smaller, thereby opening the door to efficient algorithms.

> *In this thesis we give new results in graph sparsification and compression. We will use tools and perspectives from combinatorial optimization, metric embeddings, approximation algorithms and online algorithms.*

1.1 Part One Overview

In the first part of this thesis we revisit two classic questions regarding how to concisely represent metrics and graphs respectively.

1. **Tree embeddings of metrics:** Trees are among the simplest graphs. Indeed, by way of classical ideas—such as dynamic programming—their simple structure makes many otherwise intractable problems efficiently solvable. Naturally, then, a great deal of work has focused on how to approximate arbitrary metrics by trees. As an arbitrary metric is provably inapproximable by a single tree, conventionally, this work has focused on how to approximate a metric by a distribution over trees. However, the probabilistic nature of these embeddings makes these embeddings unsuitable for many well-studied settings—such as online settings with adaptive adversaries or demand-robust optimization.

   We investigate new tree embeddings of metrics which *deterministically* approximate an arbitrary metric by a tree. Our key insight is that the above barrier to approximating a metric by a single tree can be overcome if we allow our tree to contain (boundedely-many) copies of each vertex. We term these embeddings copy tree embeddings. We give constructions of copy tree embeddings and use these embeddings to build new approximation algorithms for several Steiner-type problems in online and demand-robust settings. Much of this is based on Haeupler et al. [36].

2. **Minimum connectivity-preserving subgraphs:** Similarly, a well-studied notion of graph sparsification is that of finding a small subgraph which preserves specified connectivity constraints. A classic such problem—the minimum strongly connected subgraph problem—tasks us with finding the strongly-connected (and spanning) subgraph with the fewest number of edges (minimum weight in the weighted case) of a strongly connected digraph. This problem is both of great practical interest—algorithms for this problem have been used to concisely represent the interactions of protein networks [11]—and great theoretical interest—it connects to many beautiful classic ideas in combinatorial optimization such as Lovász and Mader's splitting off theorems, Edmond's arborescence packing and Jain's uncrossing technique.

   While NP-hard, a variety of approximation algorithms have been proposed for the various problems in this space. The current best approximation for the weighted version of this problem is a simple 2 approximation while the unweighted problem admits a $3/2$ approximation [55] and the unweighted k-arc version admits a $1 + 1/k$ approximation [48].

   We propose a new attack on problems in this space that is simple yet general. We give some preliminary results that make this approach seem promising including a simple $3/2$ approximation for certain special cases of the unweighted MSCSS problem.

1.2 Part Two Overview

In the second part of this thesis we aim to examine the interplay between graphs and their induced metrics and how this relates to concise representations of graphs.

While every graph induces a metric, a graph can provide a metric with rich structure that a metric alone does not exhibit. Concretely, consider the following two senses in which a graph provides its induced metric with additional structure.

1. First, the set of all metrics induced by a minor-closed family of graphs is a strict subset of all metrics: for example, not every metric can be induced by a planar graph. In this way, considering a metric along with the graph that induces it can provide the induced metric with additional structure.
2. Second, while the metric we typically associate with a weighted graph is the shortest path metric—where the distance between two vertices is the minimum weight of a path connecting them—a weighted graph really induces two metrics. The first is the usual shortest path metric but the second is the hop-distance metric where the distance between two nodes is the minimum number of edges in a path connecting these nodes. These two metrics together define a notion of hop-constrained distances where the distance between two nodes is defined as the shortest path with at most some specified number of edges. A metric without the graph that induces it provides no such notion of hop-constrained distances.

On one hand, the structure a graph provides a metric with can be a boon to algorithms and embeddings. For example, many graph problems are significantly easier when the input metric is a tree, series-parallel, planar or, more generally, from a minor-closed family [5, 35, 56]. Similarly, metrics induced by well-behaved graphs admit embeddings that general metrics do not [16, 31, 46].

On the other hand, the additional structure afforded to a metric by the graph that induces it presents unique algorithmic challenges. Many problems become significantly more challenging when we require our input solution to conform not only to the input metric but also to the structure of the graph which induces the metric. For example, consider minimum spanning tree (MST). To solve MST on a weighted graph G one can just as easily solve MST on the metric completion of G (and then project back our solution to G in the natural way)

without any loss in the quality of our solution. Thus, in this sense MST does not meaningfully incorporate the structure of the input graph as one can assume without loss of generality that the problem is on a metric not a graph. In this way MST—and many other classical connectivity problems—are not really graph problems at all; rather they are really problems about metrics. However, if we desire a hop-constrained MST—where the hop-distance between any two nodes must not exceed some specified bound—no such reduction to MST on a metric is possible; a metric has no notion of hop-distance. Indeed, while MST is well-known to be solvable in polynomial-time, hop-constrained MST problem is known to admit no $o(\log n)$ approximation under the assumption P $\neq$ NP [4]. Thus, incorporating graph structure into a solution can make a problem significantly more algorithmically challenging.

Moreover, incorporating graph structure into a solution can be of great practical value. Consider, for example, the case of hop-constrained MST. MST has been extensively used as a subroutine to compute low-cost (computer) networks in which every node is able to communicate with every other node. A hop-constrained MST naturally lead to networks with lower latencies as messages between nodes must travel fewer links in the network [54, 57]. Additionally, if we imagine that a transmission over an edge fails with some probability—as has been observed to occur in practice—then by minimizing the number of edges messages must travel we can reduce the probability that a communication fails and achieve more reliable networks [54, 57].

Summarizing, in our search for compact representations, we cannot view graphs and metrics in isolation. Rather the interactions between a graph and its induced metric must be taken into account; both to exploit the additional structure that such interactions provide and to meet the important algorithmic challenges that arise from these interactions.

In the second half of this thesis we aim to better understand how the interactions between a graph's metric and structure affect compact representations.

1. **Concise representations of hop-constrained distances:** We first investigate compact representations of hop-constrained distances. Just as it is useful to embed a metric into simple structures such as trees, so too is it useful to embed hop-constrained distances into trees. However, the complexity inherent in hop-constrained distances makes tree-like summarizations significantly more challenging than the metric case. Indeed, not only are hop-constrained distances not metric (as they fail to satisfy the triangle inequality) but even the humble path graph demonstrates that they are, in some sense, inapproximable by metrics. This fact not only seems to rule out an embedding of hop-constrained distances into trees but as a distribution over metrics is itself a metric, it seems to rule out embedding hop-constrained distances into any distribution of metrics.

   Despite these apparent roadblocks we demonstrate how to concisely represent hop-constrained distances by the metric of a weighted tree. The crucial insight we make is that the above impossibility results only hold if we our goal is to embed the hop-constrained distances between all nodes into a metric. Thus, we are able to approximate the hop-constrained distances by a tree by only embedding a (large) constant fraction of our nodes into a tree. These results are based on Haeupler et al. [37].

   We additionally investigate the hop-constrained analogues of classic concise representations of metrics such as analogues of shortest path trees and spanners.

2. **Hop-constrained network design:** Next, we investigate new approximation algorithms for various network design problems with hop-constraints. As mentioned above hop-constraints make network design problems significantly more challenging. For example, while Steiner forest admits a constant approximation, Steiner forest with hop-constraints is known to admit no $o(2^{\log^{1-\epsilon} n})$-approximation for any $\epsilon > 0$. Using our above embeddings of hop-constrained distances into trees and by relaxing our algorithms to be bicriteria, we overcome these impossibility results and give the first (poly-log,

poly-log) bicriteria approximations for the hop-constrained versions of many classic network design problems such as Steiner forest, group Steiner tree and group Steiner forest as well as the online and oblivious versions of these problems. We give many of these results in Haeupler et al. [37].

We also propose new attacks on improving the bounds for hop-constrained MST.

3. **Improved embeddings for structured graph families:** Lastly, we investigate how the structure of a graph can enable improved embeddings. Specifically, we focus on the Steiner point removal (SPR) wherein we are given a weighted graph and a collection of terminals and must compute a weighted minor just on the terminals which approximates the input metric. It is conjectured that if the input graph belongs to a (non-trivial) minor-closed family then such an embedding is possible with multiplicative distortion $O(1)$.

   We show that series-parallel graphs admit $O(1)$ distortion SPR solutions extending the frontier of what minor-closed families are known to admit $O(1)$ distortion SPR solutions. These result are detailed in Hershkowitz and Li [38]. Lastly, we propose new directions for SPR in planar graphs which extend our approach for series-parallel graphs.

2 Part One: Revisiting Classic Compact Representations

In this section we give new results for and propose future work in two classic questions regarding graph and metric sparsification: how to represent a metric by a tree and how to find sparse subgraphs which preserve connectivity.

2.1 New Tree Embeddings: Copy Tree Embeddings

Probabilistic embeddings of general metrics into distributions over trees are some of the best studied notions of graph sparsification and one of the most versatile tools in combinatorial and network optimization. The beauty and utility of these tree embeddings comes from the fact that their application is often simple, yet extremely powerful. Indeed, when modeling a network with length, costs, or capacities as a weighted graph, these embeddings often allow one to pretend that the graph is a tree. A common template for countless network design algorithms is to (1) embed the input weighted graph G into a randomly sampled tree T that approximately preserves the weight structure of G; (2) solve the input problem on T and; (3) project the solution on T back into G.

A long and celebrated line of work [2, 8, 23, 40] culminated in the embedding of Fakcharoenphol, Rao and Talwar [23]—henceforth the "FRT embedding"—which showed that any weighted graph on n nodes can be embedded into a distribution over weighted trees in a way that $O(\log n)$-approximately preserves distances in expectation. Together with the above template this reduces many graph problems to much easier problems on trees at the cost of an $O(\log n)$ approximation factor. This has lead to a myriad of approximation, online, and dynamic algorithms with poly-logarithmic approximations and competitive ratios for NP-hard problems such as for k-server [7], metrical task systems [9], group Steiner tree and group Steiner forest [3, 30, 51], buy-at-bulk network design [6] and (oblivious) routing [52]. For many of these problems tree embeddings are the only known way of obtaining such algorithms on general graphs.

Probabilistic tree embeddings have a drawback: Algorithms based on them naturally require randomization and their approximation guarantees only hold in expectation. For approximation algorithms—i.e., in the offline setting—there are derandomization tools, such as the FRT derandomizations given in [15, 23], to

overcome these issues. These derandomization results are so general that essentially any offline algorithm based on tree embeddings can be transformed into a deterministic algorithm with matching approximation guarantees (with only a moderate increase in running time). Unfortunately, these strategies are not applicable to online or dynamic settings where an adversary progressively reveals the input. Indeed, to our knowledge, all online and dynamic algorithms that use FRT are randomized (e.g. [3, 9, 21, 22, 25, 32, 34, 51]).[1].

This overwhelming evidence in the literature is driven by a well-known and fundamental barrier to the use of probabilistic tree embeddings in deterministic online and dynamic algorithms. More specifically and even worse, this is a barrier which prevents these algorithms from working against all but the weakest type of adversary. In particular, designing an online or dynamic algorithm which is robust to an oblivious adversary (which fixes all requests in advance, independently of the algorithm's randomness) is often much easier than designing an algorithm which is robust to an adaptive adversary (which chooses the next request based on the algorithm's current solution). As the actions of a deterministic algorithm can be fully predicted this distinction only holds for randomized algorithms—any deterministic algorithm has to always work against an adaptive adversary. For these reasons, many online and dynamic algorithms have exponentially worse competitive ratios in the deterministic or adaptive adversary setting than in the oblivious adversary setting. This is independent of computational complexity considerations.

The above barrier results from a repeatedly recognized and seemingly unavoidable phenomenon which prevents online algorithms built on FRT from working against adaptive adversaries. Specifically, there are graphs where every tree embedding must have many node pairs with polynomially-stretched distances [8]. There is nothing that prevents an adversary then from learning through the online algorithm's responses which tree was sampled and then tailoring the remainder of the online instance to pairs of nodes that have highly stretched distances. The exact same phenomenon occurs in the dynamic setting; see, for example, Guo et al. [32] and Gupta et al. [34] for dynamic algorithms with expected cost guarantees that only hold against oblivious adversaries because they are based on FRT. In summary, online and dynamic algorithms that use probabilistic tree embeddings seem inherently randomized and seem to necessarily only work against adversaries oblivious to this randomness.

Overall it seems fair to say that prior to this work tree embeddings seemed fundamentally incapable of enabling adaptive-adversary-robust and deterministic algorithms in several well-studied settings.

2.1.1 Copy Tree Embeddings and Their Applications

We propose a new type of tree embedding which addresses these issues by *deterministically* embedding a graph into a single tree containing $O(\log n)$ copies of each vertex while preserving the connectivity structure of every subgraph and $O(\log^2 n)$-approximating the cost of every subgraph.

Using this embedding we obtain several new algorithmic results: We reduce an open question of Alon et al. [3]—the existence of a deterministic poly-log-competitive algorithm for online group Steiner tree on a general graph—to its tree case. We give a poly-log-competitive deterministic algorithm for a closely related problem—online partial group Steiner tree—which, roughly, is a bicriteria version of online group Steiner tree. Lastly, we give the first poly-log approximations for demand-robust Steiner forest, group Steiner tree and group Steiner forest. In demand-robust problems an algorithm is given a collection of possible problems for which it is responsible, buys a partial solution with discounted costs in a first stage and then an (adaptive)

[1] We exclude so called "oblivious" solutions in this statement which can be interpreted as solving "simple" online problems that allow for "oblivious" solutions where choices for each part of the input can be made independently—most problems provably do not admit oblivious solutions.

adversary reveals the problem which the algorithm must complete its partial solution in the second stage with inflated costs.

To precisely define our embeddings we define a copy mapping ϕ which maps a vertex v to its copies.

Definition 1 (Copy Mapping)**.** *Given vertex sets V and V' we say $\phi : V \to 2^{V'}$ is a copy mapping if every node has at least one copy (i.e. $|\phi(v)| \geq 1$ for all $v \in V$), copies are disjoint (i.e. $\phi(v) \cap \phi(u) = \emptyset$ for $u \neq v$) and every node in V' is a copy of some node (i.e. for every $v' \in V'$ there is some $v \in V$ where $v' \in \phi(v)$). For $v' \in V'$, we use the shorthand $\phi^{-1}(v')$ to stand for the unique $v \in V$ such that $v' \in \phi(v)$.*

A copy tree embedding for a weighted graph G now simply consists of a tree T on copies of vertices of G with one distinguished root and two mappings $\pi_{G \to T}$ and $\pi_{T \to G}$ which map subsets of edges from G to T and from T to G in a way that preserves connectivity and approximately preserves costs. We say that *two vertex subsets U, W are connected* in a graph if there is a $u \in U$ and $w \in W$ such that u and w are connected. We also say that a mapping $\pi : 2^E \to 2^{E'}$ is *monotone* if for every $A \subseteq B$ we have that $\pi(A) \subseteq \pi(B)$. A rooted tree $T = (V, E, w)$ is *well-separated* if for all edges e if e' is a child edge of e in T then $w(e') \leq \frac{1}{2} w(e)$.

Definition 2 (α-Approximate Copy Tree Embedding with Copy Number χ)**.** *Let $G = (V, E, w)$ be a weighted graph with some distinguished root $r \in V$. An α-approximate copy tree embedding with copy number χ consists of a weighted rooted tree $T = (V', E', w')$, a copy mapping $\phi : V \to 2^{V'}$ and edge mapping functions $\pi_{G \to T} : 2^E \to 2^{E'}$ and $\pi_{T \to G} : 2^{E'} \to 2^E$ where $\pi_{T \to G} : 2^{E'} \to 2^E$ is monotone and:*

1. ***Connectivity Preservation:*** *For all $F \subseteq E$ and $u, v \in V$ if u, v are connected by F, then $\phi(u), \phi(v) \subseteq V'$ are connected by $\pi_{G \to T}(F)$. Symmetrically, for all $F' \subseteq E'$ and $u', v' \in V'$ if u' and v' are connected by F' then $\phi^{-1}(u')$ and $\phi^{-1}(v')$ are connected by $\pi_{T \to G}(F')$.*
2. ***α-Cost Preservation****: For any $F \subseteq E$ we have $w(F) \leq \alpha \cdot w'(\pi_{G \to T}(F))$ and for any $F' \subseteq E'$ we have $w'(F') \leq w(\pi_{T \to G}(F'))$.*
3. ***Copy Number:*** *$|\phi(v)| \leq \chi$ for all $v \in V$ and $\phi(r) = \{r'\}$ where r' is the root of T.*

A copy tree embedding is efficient if T, ϕ, and $\pi_{T \to G}$ are deterministically poly-time computable and well-separated if T is well-separated.

The following theorem summarizes the guarantees of our copy tree embedding construction.

Theorem 1. *There is a poly-time deterministic algorithm which given any weighted graph $G = (V, E, w)$ and root $r \in V$ computes an efficient and well-separated $O(\log^2 n)$-approximate copy tree embedding with copy number $O(\log n)$.*

We next apply our copy tree embeddings to obtain new results for several online and demand-robust connectivity problems whose history we briefly summarize now. Group Steiner tree and group Steiner forest are two well-studied generalizations of set cover and Steiner tree. In the group Steiner tree problem, we are given a weighted graph $G = (V, E, w)$ and groups $g_1, \ldots, g_k \subseteq V$ and must return a subgraph of G of minimum weight which contains at least one vertex from each group. The group Steiner forest problem generalizes group Steiner tree. Here, we are given $A_i, B_i \subseteq V$ pairs and for each i we must connect some vertex from A_i to some vertex in B_i. Alon et al. [3] and Naor et al. [51] each gave a poly-log approximation for online group Steiner tree and forest respectively but both of these approximation guarantees are randomized and only hold against oblivious adversaries because they rely on FRT. Indeed, Alon et al. [3] posed the existence of a deterministic poly-log approximation for online group Steiner tree as an open question which has since been restated several times [12, 13]. Similarly, while demand-robust minimum spanning tree and special cases of demand-robust Steiner tree have received considerable attention [19, 41, 42], there are no known poly-log approximations for demand-robust Steiner tree, group Steiner tree or group Steiner forest.

The following theorems summarize our result for deterministic online group Steiner tree and forest as well as for the aformentioned partial group Steiner tree problem where "$1-\epsilon$-connection competitive" means that our algorithm satisfies all but an ϵ fraction of its connectivity requirements.

Theorem 2. *If there exists an α-competitive poly-time deterministic algorithm for group Steiner tree (resp. group Steiner forest) on well-separated trees then there exists an $O(\log n \cdot \alpha)$-competitive poly-time deterministic algorithm for group Steiner tree (resp. group Steiner forest) on general graphs.*

Theorem 3. *There is a deterministic poly-time algorithm for online partial group Steiner tree which given any $\epsilon > 0$ is $O\left(\frac{\log^3 n}{\epsilon}\right)$-cost-competitive and $(1-\epsilon)$-connection competitive.*

We next generalize copy tree embeddings to demand-robust copy tree embeddings. Roughly, these are copy tree embeddings which simultaneously work well for every possible demand-robust scenario. We then adapt our analysis from our previous constructions to show that these copy tree embeddings exist. Lastly, we apply demand-robust copy tree embeddings to give poly-log approximations for the demand-robust versions of several Steiner problems—Steiner forest, group Steiner tree and group Steiner forest—for which, prior to this work, nearly nothing was known. In particular, the only non-trivial algorithms known for demand-robust Steiner problems prior to this work are an algorithm for demand-robust Steiner tree [19] and an algorithm for demand-robust Steiner forest *on trees* with exponential scenarios [24] (which is, in general, incomparable to the usual demand-robust setting). To show these results, we apply our demand-robust copy tree embeddings to reduce these problems to their tree case. Thus, we also give our results on trees which are themselves non-trivial.

Theorem 4. *There is a randomized poly-time $O(\log^4 n)$-approximation algorithm for the demand-robust group Steiner tree problem on weighted graphs.*

Theorem 5. *There is a randomized poly-time $O(\log^6 n)$-approximation for the demand-robust group Steiner forest problem on weighted graphs with polynomially-bounded aspect ratio.*

Demand-robust group Steiner forest generalizes demand-robust Steiner forest and prior to this work no poly-log approximations were known for demand-robust Steiner forest; thus the above result gives the first poly-log approximation for demand-robust Steiner forest. We solve the tree case of the above problems by observing a connection between demand-robust and online algorithms. In particular, we exploit the fact that for certain online rounding schemes a demand-robust problem can be seen as an online problem with two time steps provided certain natural properties are met. Notably, these properties will be met for these problems *on trees*. Thus, we emphasize that going through the copy tree embedding is crucial for our application—a more direct approach of using online rounding schemes on the general problem does not seem to yield useful results.

2.1.2 Future Directions

There are at least two directions which we feel would be interesting to explore in regards copy tree embeddings.

1. Bienkowski et al. [12] recently gave a deterministic algorithm for online non-metric facility location—which is equivalent to online group Steiner tree on trees of depth 2—with a poly-log-competitive ratio and stated that they expect their techniques will extend to online group Steiner tree on trees. A very exciting direction for future work would thus be to extend these techniques to general depth trees which, when combined with our reduction to the tree case, would prove the existence of a deterministic poly-log-competitive algorithm for online group Steiner tree, settling the open question of Alon et al. [3].

2. It would be interesting to prove lower bounds on copy tree embedding parameters, such as, more rigorously characterizing the tradeoffs between the number of copies and the cost approximation factor. One should also consider the possibility of improved constructions. For example: Is it possible to get a logarithmic approximation with few copies, maybe even a constant number of copies? It is easy to see that with an exponential number of copies—one for each possible subgraph—a perfect cost approximation factor of one is possible. Can one show that a sub-logarithmic distortion is impossible with a polynomial number of copies? We currently do not even have a proof that excludes a constant cost approximation factor with a constant copy number.

2.2 Connectivity-Preserving Subgraphs: MSCSS

One of the most well studied notions of graph sparsification is that of a small weight subgraph of a weighted graph which preserves certain connectivity relationships in the input graph. In this section we revisit one of the simplest such problems on directed graphs: the minimum strongly connected spanning subgraph (MSCSS) problem. In the MSCSS problem we are given a weighted digraph $D = (V, A, w)$ and our goal is to find a subdigraph $H \subseteq D$ of minimum edge cardinality.

On the lower bounds side the MSCSS problem is know to be APX-hard and the natural LP, given below, is known to have an integrality gap of at least $4/3$ [11].

$$\min \sum_{a \in A} x_a \text{ s.t.} \qquad \text{(MSCSS LP)}$$

$$\sum_{a \in \delta^+(S)} x_a \geq 1 \qquad \forall S \text{ s.t. } \emptyset \subset S \subset V$$

On the upper bounds side a series of works culminated in an intricate combinatorial algorithm of Vetta [55] which gives a $3/2$ approximation (as well as an upper bound of $3/2$ on the integrality gap of the natural LP) by way of a great deal of case analysis. A tantalizing open question is whether the current $3/2$ approximation can be improved to a $4/3$ approximation or if there exists an instance with integrality gap at least $3/2$.

There are two well-studied generalizations of MSCSS. In the weighted MSCSS problem we must find not a minimum edge-cardinality subgraph but a minimum edge-weight subgraph. For this problem the natural LP has an integrality gap of at least $3/2$ [48] and a simple 2 approximation is known. Second, in the k-MSCSS problem the goal is to find a k-arc connected subdigraph of minimum arc cardinality. A series of works resulted in a $\min\left(\frac{5}{3}, 1 + \frac{1}{k}\right)$ approximation for this problem. Notice that for $k = 1$ the mentioned algorithm has a worse approximation guarantee than the $3/2$ approximation of Vetta [55]. It is also known that no approximation better than $1 + O(\frac{1}{k})$ is possible [29].

2.2.1 A New Attack on MSCSS

We propose a new general approach for MSCSS-type problems.

Consider the (unweighted) MSCSS problem as described above. It is well-known and easy to verify that given a fixed root $r \in V$ the union of an r-in-arborescence and r-out-arborescence gives a 2 approximation. A natural local-search-type algorithm, then, is one which starts with an r-in-arborescence I and r-out-arborescence O, and argues that either $|I \cup O|$ is close to the cost of the optimal solution or that I and O can be modified to new r-in-arborescences and r-out-arborescences I' and O' respectively where $|I' \cup O'| < |I \cup O|$.

On the one hand, if I and O share many edges then we ought to be happy with our current solution. On the other hand, if I and O do not overlap we would like to argue that there is a reason why they cannot be made to overlap or else a way to make them overlap more.

An elementary calculation which formalizes this intuition conveniently shows that if we would like to argue that $I \cup O$ is a $3/2$ approximation then it suffices to argue that OPT $\geq |I \Delta O|$ where Δ gives the symmetric difference. In particular, we have the following.

- If $|I \cap O| \geq \frac{1}{2}n$ then $|I \cup O| = \frac{3}{2}n \leq \frac{3}{2}$ OPT since OPT $\geq n$.
- If $|I \cap O| < \frac{1}{2}n$ then applying the fact that OPT $\geq |I \Delta O|$ and our assumption that $|I \cap O| < \frac{1}{2}n$ shows that

$$\begin{aligned} |I \cup O| &= |I \cap O| + |I \Delta O| \\ &\leq .5n + \text{OPT} \\ &\leq \frac{3}{2}\,\text{OPT} \end{aligned}$$

Thus, our general approach is to either argue that we can find a lower bound which witnesses OPT $\geq |I \Delta O|$ or that $|I \cap O|$ can be increased. This approach extends naturally to the k-MSCSS problem where we instead maintain k in- and out-arborescences and try to increase the overlap of our in- and out-arborescences or else argue that there is a lower bound witnessing the cost of our solution.

2.2.2 Preliminary Results

Using the above strategy we are able to give a $3/2$ approximation for certain special instances of MSCSS which is much simpler than that of Vetta [55].

In particular, since every vertex has out-degree 1 in I and in-degree 1 in O, it follows that the graph induced by $I \cap O$ is a collection vertex-disjoint paths. If these paths obey certain structure then we can give a fairly simple $3/2$ approximation. Roughly, the structure we require is that the graph induced by $I \cup O$ after contracting $I \cap O$ is an "anti-parallel tree"; that is, it is an acyclic digraph where if $a = (u, v)$ is an arc in this graph then so is the reversal of a, namely $\bar{a} = (v, u)$.

2.2.3 Proposed Work

For MSCSS we have 4 directions we would like to explore further.

1. First, our hope is that we will be able to generalize the above $3/2$ for the aforementioned special case to the fully general case, thereby greatly simplifying the result of Vetta [55].
2. Assuming we are successful in the previous goal an exciting, but perhaps optimistic goal, would be to try to extend these results to a $4/3$ approximation for MSCSS. In particular, a calculation analogous to the above also shows that to give a $4/3$ approximation it suffices to argue that either we can increase $|I \cap O|$ or OPT $\geq |I \Delta O| + \frac{1}{2}|I \cap O|$; thus, if we could strengthen our lower bounds in this way then we could give an integrality-gap tight approximation for MSCSS.
3. Next, we would like to explore how well the above strategy generalizes to the k-MSCSS problem. In particular, as earlier mentioned the current best approximation algorithm for k-MSCSS achieves an approximation of $\min\left(\frac{5}{3}, 1 + \frac{1}{k}\right)$; notably for $k = 1$ this is worse than the $3/2$ approximation of

Vetta [55]. However, whereas it is not necessarily clear how to generalize the approach of Vetta [55] for MSCSS to the k-MSCSS problem as described earlier our approach seems to naturally generalize. Thus, it is our hope that for the k-MSCSS problem our approach could give something like a $1 + \frac{1}{2k}$ approximation.

4. Lastly, we would like to give some thought to an alternative approach to another simple $3/2$ approximation for MSCSS. It is well-known that given a solution x to MSCSS LP it is possible to sample an in- or out-arborescence which includes arc a with probability at most x_a. Applying the fact that the support of any basic feasible to MSCSS LP has size at most $3n$ (see, e.g. [28]) and some simple convexity arguments shows that taking the union of an in and out arborescence sampled in this way achieves a $\frac{5}{3}$ approximation. However, it is not too hard to see by standard combinatorial-optimization-type arguments that if there does not exist a vertex v such that $x(\delta^+(v)) = x(\delta^-(v))$ then, in fact, this algorithm achieves a $3/2$ approximation.[2] Thus, one approach would be to argue that either (1) such a vertex existed in which case we could do something like apply Mader's directed splitting off theorem to remove this vertex and induct or (2) no such vertex exists in which case we have a $3/2$ approximation and are done. Along these lines we have a new proof of the LP sparsity of $3n$ of the above LP which seems to greatly simplify that of Gabow [28]. Also along this lines we were able to give a tight 1.5 approximation for weighted MSCSS on half-integral instances; see Hershkowitz et al. [39].

3 Part Two: Effects of Graphs and Metric Interactions On Sparsification

In this second part of this thesis we would like to explore how graph structure can both hurt and help compact representations of notions of distance in a graph. We give an overview of results we have obtained in this direction as well as directions for future work.

3.1 Concise Representations of Hop-Constrained Distances

As alluded to in the introduction a graph can be understood as inducing two metrics: the hop metric and the shortest path metric. Given weighted graph $G = (V, E, w)$, the shortest path distance between vertices u and v is defined as $d(u, v) := \min\{w(P) \mid \text{path } P \text{ in } G \text{ between } u, v\}$ where $w(P) := \sum_{e \in P} w(e)$. Just as the shortest path distances are the natural notion of distance to consider in network design problems, the hop-constrained distances—defined below—are the natural notion of distance to consider when doing hop-constrained network design. Here, $\text{hop}(P)$ gives the number of edges in P.

Definition 3 (Hop-Constrained Distances)**.** *For a (complete) weighted graph $G = (V, E, w)$ and a hop constraint $h \geq 1$ we define the h-hop distance between any two nodes $u, v \in V$ as*

$$d_G^{(h)}(u, v) := \min\{w(P) \mid \textit{path } P \textit{ in } G \textit{ between } u, v \textit{ with } \text{hop}(P) \leq h\}.$$

3.1.1 Why Hop-Constrained Distances Seem Poorly Behaved

At a glance, the hop-constrained distances seem quite poorly behaved. It is easy to verify that $d_G^{(h)}$ is symmetric, i.e., $d^{(h)}(u, v) = d^{(h)}(v, u)$, and satisfies the identity of indiscernibles, i.e., $d^{(h)}(u, v) = 0 \Leftrightarrow u = v$ and so one might be tempted to include that they induce a metric. However, it is simple to see that hop-constrained distances are not necessarily metrics since they do not obey the triangle inequality. Indeed, the

[2] Here $x(\delta^+(v)) := \sum_{a=(v,\cdot)} x_a$ and $x(\delta^-(v))$ is defined symmetrically.

existence of a short h-hop path from u to v and a short h-hop path from v to w does not imply the existence of a short h-hop path between u and w. More formally it is possible that $d^{(h)}(u,w) \gg d^{(h)}(u,v) + d^{(h)}(v,w)$.

Of course with a factor 2 relaxation in the hop constraint the relaxed triangle inequality $d^{(2h)}(u,w) \leq d^{(h)}(u,v) + d^{(h)}(v,w)$ holds for any graph G and any $u,v,w \in V(G)$. This suggests—albeit incorrectly—that one might be able to approximate hop-constrained distance by allowing constant slack in the hop constraint and length approximation as in the following definition.

Definition 4. *A distance function $\tilde{d}$ approximates the h-hop constrained distances $d_G^{(h)}$ for a weighted graph $G = (V, E, w)$ where $h \geq 1$ with **distance stretch** $\alpha \geq 1$ and **hop stretch** $\beta \geq 1$ if for all $u, v \in V$ we have*

$$d_G^{(\beta h)}(u,v) \leq \tilde{d}(u,v) \leq \alpha \cdot d_G^{(h)}(u,v).$$

Unfortunately, as we observe even with such a relaxation hop-constrained distances can be arbitrarily far from a metric.

Lemma 1. *Given any graph $G = (V, E, w)$ with aspect ratio L and a distance stretch α and hop stretch β satisfying $\alpha(\beta h + 1) \geq L$, we have that $\alpha \cdot d_G$ approximates $d_G^{(h)}$ with distance stretch α and hop stretch β where L is the aspect ratio of G.*

The main goal of the remainder of this section is to explore to what extent $d^{(h)}$ admits some of the well-studied and useful compact representations that the normal shortest path metric admits despite being arbitrarily far from being metric. We will be particularly interested in how well $d^{(h)}$ can be approximated by a distribution over trees; that is, to what extent $d^{(h)}$ admits probabilistic tree embeddings.

3.1.2 Metric Embeddings of Hop-Constrained Distances

As the expected distances induced by a distribution over trees induces a metric, Lemma 1 rules out the possibility of probabilistic tree embeddings in the classic sense for hop-constrained distances. However, it does not rule out the possibility that a distribution over trees might approximate hop-constrained distances on *most* vertices; this observation motivates the following definition of a partial metric and its stretch.

Definition 5 (Partial Metric)**.** *Any metric d defined on a set V_d is called a **partial metric** on V if $V_d \subseteq V$.*

Of course, a partial metric on the empty set trivially approximates $d_G^{(h)}$ and we are ultimately interested in estimating $d^{(h)}$ on all pairs of nodes. For this reason, we give the following notions of exclusion probability and how a distribution over partial metrics can induce a distance function between all nodes.

Definition 6 (Distances of Partial Metric Distributions)**.** *Let $\mathcal{D}$ be a distribution of partial metrics of V for weighted graph $G = (V, e, w)$. We say $\mathcal{D}$ has **exclusion probability** ϵ if for all $v \in V$ we have $\Pr_{d\sim\mathcal{D}}[v \in V_d] \geq 1 - \epsilon$. If $\epsilon \leq \frac{1}{3}$ then we say that $\mathcal{D}$ induces the distance function $d_{\mathcal{D}}$ on V, defined as*

$$d_{\mathcal{D}}(u,v) := \mathop{\mathbb{E}}_{d\sim\mathcal{D}}[d(u,v) \cdot \mathbb{I}[u,v \in V_d]].$$

Definition 7 (Stretch of Partial Metric Distribution)**.** *A distribution $\mathcal{D}$ of partial metrics on V with exclusion probability at most $\frac{1}{3}$ approximates $d^{(h)}$ on weighted graph $G = (V, E, w)$ for hop constraint $h \geq 1$ with **worst-case distance stretch** $\alpha_{WC} \geq 1$ and **hop stretch** $\beta \geq 1$ if each d in the support of $\mathcal{D}$ approximates $d_G^{(h)}$ on V_d with distance stretch α_{WC} and hop stretch β, i.e. for each d in the support of $\mathcal{D}$ and all $u, v \in V_d$ we have*

$$d_G^{(\beta h)}(u,v) \leq d(u,v) \leq \alpha \cdot d_G^{(h)}(u,v).$$

*Furthermore, $\mathcal{D}$ has **expected distance stretch** $\alpha_{\mathbb{E}}$ if for all $u, v \in V$ we have*

$$d_{\mathcal{D}}(u,v) \leq \alpha_{\mathbb{E}} \cdot d_G^{(h)}(u,v).$$

As we prove it is indeed possible to embed most of V into a distribution over trees with poly-logarithmic distance and hop stretch if each of these trees is only on most of the vertices of V.

Theorem 6. *For any (complete) weighted graph G, any hop-constraint $h \geq 1$, and any $0 < \epsilon < \frac{1}{3}$ there is a distribution $\mathcal{D}$ over well-separated tree metrics each of which is a partial metric on $V(G)$ such that $\mathcal{D}$ has exclusion probability at most ϵ and approximates $d_G^{(h)}$ with expected distance stretch $\alpha_{\mathbb{E}} = O(\log n \cdot \log \frac{\log n}{\epsilon})$, worst-case distance stretch $\alpha_{\text{WC}} = O(\frac{\log^2 n}{\epsilon})$ and hop stretch $\beta = O(\frac{\log^2 n}{\epsilon})$.*

3.1.3 Preliminary Results

In addition to studying how well hop-constrained distances can be approximated by distributions over trees, we also have some additional preliminary results on how well hop-constrained distances can be compactly represented by other classic metric structures.

1. First, we have observed that the greedy spanner when applied to hop-constrained distances provides an essentially optimal spanner for hop-constrained distances.
2. Second, we have a construction analogous to that of a shortest path tree for hop-constrained distances. In particular, given root r we have shown that there exists a tree subgraph whose metric distances approximate all hop-constrained distances to r with an $O(\log n)$ loss in the hop stretch and an $O(1)$ loss in the distance stretch. Matching this we have a lower bound construction which shows that any such subgraph with a constant distance stretch incurs an $\Omega(\log n)$ in the hop stretch.

Lastly, we also have a lower bound construction which shows that if one wants a constant exclusion probability in a partial tree embedding of hop-constrained distances as above then one must incur an $\Omega(\log n)$ in both the hop and distance stretches of one's embedding.

3.1.4 Future Directions

There are at least three directions that we would like to explore regarding concise representations of hop-constrained distances.

1. First, the main open direction in this space is to tighten our upper and lower bounds and to more tightly characterize what trade-offs are possible and necessary between distance stretch, hop stretch and exclusion probability.
2. Second, as has been studied for the shortest path metric [1], it would be interesting to understand how well we can embed hop-constrained distances into trees which are *subtrees* of the input graph. There are some trivial senses in which this is impossible and so a non-trivial result in this space would rely crucially on finding the right notion of such an embedding.
3. Lastly, it would be interesting to explore what other sort of concise approximations hop-constrained distances admit such as, for example, an analogue of light approximate shortest path trees (LASTs) [44] for hop-constrained distances.

3.2 Hop-Constrained Network Design

We next explore new algorithms for the hop-constrained versions of many classic network design problems.

As detailed in Table 1, we give the first (poly-log, poly-log) bicriteria algorithms for hop-constrained network design problems that relax both the cost and hop constraint of the solution. As noted in the introduction, bicriterianess is necessary for any poly-log approximation for Steiner forest and its generalizations. Furthermore, while the results in Table 1 are stated in utmost generality, many special cases of our results were to our knowledge not previously known. For example, our algorithm for hop-constrained oblivious Steiner forest immediately gives new algorithms for hop-constrained Steiner forest, hop-constrained online Steiner tree and hop-constrained online Steiner forest, as well as min-cost h-spanner. Similarly, our algorithm for oblivious network design immediately gives new algorithms for the hop-constrained version of the well-studied buy-at-bulk network design problem [6].

All of our algorithms for these problems—with the exception of the min-cost hop-constrained arborescence problem—use the above mentioned tree embedding template with our h-hop partial tree embeddings. In particular, we apply the embedding result from the previous section, solve the resulting non-hop constrained problem and then project the resulting solution back into the original graph. As we discuss in the next section—and unlike in the non-hop-constrained case—arguing that this strategy works with our embeddings requires a non-trivial amount of work.

Hop-Constrained Problem	**Cost Apx.**	**Hop Apx.**
Offline Problems		
Arborescence	$O(\log n)$	$O(\log n)$
Relaxed k-Steiner Tree	$O(\log^2 n)$	$O(\log^3 n)$
k-Steiner Tree	$O(\log^3 n)$	$O(\log^3 n)$
Group Steiner Tree	$O(\log^5 n)$	$O(\log^3 n)$
Group Steiner Forest	$O(\log^7 n)$	$O(\log^3 n)$
Online Problems		
Group Steiner Tree	$O(\log^6 n)$	$O(\log^3 n)$
Group Steiner Forest	$O(\log^8 n)$	$O(\log^3 n)$
Oblivious Problems		
Steiner Forest	$O(\log^3 n)$	$O(\log^3 n)$
Network Design	$O(\log^4 n)$	$O(\log^3 n)$

Table 1: Our bicriteria approximation results. All results are for poly-time algorithms that succeed with high probability (at least $1 - \frac{1}{\text{poly}(n)}$). For some of the problems we assume certain parameters are poly(n) to simplify presentation. All results are new except for the k-Steiner tree result which is implied by [43].

3.2.1 Realizing The Tree Embedding Template for Hop-Constrained Network Design

While turning classic probabilistic tree embeddings (such as that of FRT) into something which can be used for network design problems is more or less trivial, doing so for our above embedding is much more technically fraught. The following series of definitions formalizes the notion of an embedding we would like when using the usual tree embedding template; these are somewhat analogous to those of the previous sections with some minor differences regarding how to map from the the tree to the graph. Throughout this section we will use $w_G : E \to \mathbb{R}_{\geq 0}$ to refer to the weight function of weighted graph $G = (V, E, w_G)$.

Definition 8 (Partial Tree Embedding)**.** *A partial tree embedding on weighted graph $G = (V(G), E(G), w_G)$ consists of a rooted and weighted tree $T = (V(T), E(T), w_T)$ with $V(T) \subseteq V(G)$ and a path $T_e^G \subseteq G$ for every $e \in E(T)$ between e's endpoints satisfying $w_G(T_e^G) \leq w_T(e)$.*

Definition 9 (h-Hop Partial Tree Embedding)**.** *A partial tree embedding $(T, \{T_e^G\}_{e \in E(T)})$ is an h-hop partial tree embedding with **distance stretch** $\alpha \geq 1$ and **hop stretch** $\beta \geq 1$ for graph $G = (V(G), E(G), w_G)$ if*

1. $d_G^{(\beta h)} \leq d_T(u, v) \leq \alpha \cdot d^{(h)}(u, v)$ *for all* $u, v \in V(T) \subseteq V(G)$;
2. $\text{hop}(T_{uv}^G) \leq \beta h$ *for all* $u, v \in V(T) \subseteq V(G)$.

Definition 10 (Distances of Partial Tree Embedding Distributions)**.** *Let $\mathcal{D}$ be a distribution of partial tree embeddings on weighted graph $G = (V, E, w)$. We say $\mathcal{D}$ has **exclusion probability** ϵ if for all $v \in V$ we have $\Pr_{(T,\cdot)\sim\mathcal{D}}[v \in V(T)] \geq 1 - \epsilon$. If $\epsilon \leq \frac{1}{3}$ then we say that $\mathcal{D}$ induces the distance function $d_{\mathcal{D}}$ on V, defined as*

$$d_{\mathcal{D}}(u, v) := \mathop{\mathbb{E}}_{(T,\cdot)\sim\mathcal{D}}[d_T(u, v) \cdot \mathbb{I}[u, v \in V(T)]].$$

Definition 11 (Stretch of Partial Tree Embedding Distribution)**.** *A distribution $\mathcal{D}$ of h-hop partial tree embeddings on V with exclusion probability at most $\frac{1}{3}$ approximates $d^{(h)}$ on weighted graph $G = (V, E, w)$ for hop constraint $h \geq 1$ with **worst-case distance stretch** $\alpha_{WC} \geq 1$ and **hop stretch** $\beta \geq 1$ if each $(T, \cdot)$ in the support of $\mathcal{D}$ approximates $d_G^{(h)}$ on $V(T)$ with distance stretch α_{WC} and hop stretch β, i.e. for each $(T, \cdot)$ in the support of $\mathcal{D}$ and all $u, v \in V(T)$ we have*

$$d_G^{(\beta h)}(u, v) \leq d_T(u, v) \leq \alpha \cdot d_G^{(h)}(u, v).$$

*Furthermore, $\mathcal{D}$ has **expected distance stretch** $\alpha_{\mathbb{E}}$ if for all $u, v \in V$ we have*

$$d_{\mathcal{D}}(u, v) \leq \alpha_{\mathbb{E}} \cdot d_G^{(h)}(u, v).$$

The following summarizes our construction of partial tree embeddings for hop-constrained distances where we pick up an extra $\log n$ on the hop stretch as a result of the $\log n$ depth of the trees into which we embed.

Theorem 7. *Given weighted graph $G = (V, E, w)$, $0 < \epsilon < \frac{1}{3}$ and root $r \in V$, there is a poly-time algorithm which samples from a distribution over h-hop partial tree embeddings whose trees are well-separated and rooted at r with exclusion probability ϵ, expected distance stretch $\alpha_{\mathbb{E}} = O(\log n \cdot \log \frac{\log n}{\epsilon})$, worst-case distance stretch $\alpha_{WC} = O(\frac{\log^2 n}{\epsilon})$ and hop stretch $\beta = O(\frac{\log^3 n}{\epsilon})$.*

When using FRT for network design one must argue that it is possible to project a subgraph of G to the tree embedding in a way that preserves the connectivity between vertices and approximately preserves the weight of the subgraph in order to argue that there exists a cheap solution for the induced problem on the tree embedding. However, the fact that our embeddings are partial and deal with hop-constrained distances makes it unclear how to project from a subgraph of the input graph to a subgraph of the tree into which we embed in a way that preserves both connectivity and cost. Nonetheless we show that, up to some small constants, such a projection is possible and so the above embedding result can be turned into a general tool for hop-constrained network design. The following definition formalizes what projection we use.

Definition 12 ($T(H, h)$)**.** *Let $(T, \cdot)$ be a partial tree embedding. Then $T(H, h) := \bigcup T_{uv}$ where the $\bigcup$ is taken over u, v such that $u, v \in V(T)$ and $\text{hop}_H(u, v) \leq h$.*

The following theorem shows that, up to some small constants, our projection is indeed cost and hop stretch preserving.

Theorem 8. *Fix $h \geq 1$, let H be a subgraph of weighted graph $G = (V, E, w_G)$ and let $(T, \cdot)$ be an $8h$-hop partial tree embedding of G with worst-case distance stretch α. Then $w_T(T(H, h)) \leq 4\alpha \cdot w_G(H)$.*

3.2.2 Proposed Work and Preliminary Results

There are two directions we would like to explore regarding new algorithms for hop-constrained network design problems.

1. First, it would be exciting to achieve a bicriteria approximation algorithm for a hop-constrained network design problem which achieves a constant approximation in one parameter. Here, the most promising direction seems to be that of hop-constrained MST and, in particular, achieving a constant in the cost approximation for MST and a poly-log in the hop approximation. Along these lines we have reduced such a result to finding a path of low depth for each vertex which intersects each of the components of the "forest representation" of the MST at most a constant number of times. On the other hand, we have a polynomial integrality gap instance for the natural LP which holds even for bicriteria algorithms; along these lines we have given some thought to the use of LP hierarchies for this problem in the hopes of finding stronger lower bounds and would like to further explore this direction.

2. Second, it would be exciting to understand what sort of bicriteria approximation algorithms are possible for the hop-constrained versions of cut problems. Some previous work [49] gave a simple h-approximation for h-hop constrained min cut which matches an integrality gap lower bound but it is possible that poly $\log n$ bicriteria approximation algorithms may be possible. While tree embeddings have proven useful for cut problems—see e.g. work on requirement cuts [50]—the partialness of our embeddings seems to rule out their usage in cut problems and so, it seems, a new approach would be required.

3.3 Improved Embeddings for Structured Graph Families: Steiner Point Removal

We next explore how the structure of a graph can make compact representations of salient features of the graph easier; specifically, we study the Steiner point removal problem.

Compact representations of graphs are particularly interesting when we assume that G, while prohibitively large and so in need of compact representations, is a member of a minor-closed graph family such as tree, cactus, series-parallel or planar graphs.[3] As many algorithmic problems are significantly easier on such families—see e.g. [5, 35, 56]—it is desirable that G' is not only a simple approximation of G's metric but that it also belongs to the same family as G.

Steiner point removal (SPR) formalizes the problem of producing a simple G' in the same graph family as G that preserves G's metric. In SPR we are given a weighted graph $G = (V, E, w)$ and a terminal set $V' \subseteq V$. We must return a weighted graph $G' = (V', E', w')$ where:

1. G' is a minor of G;
2. $d_G(u, v) \leq d_{G'}(u, v) \leq \alpha \cdot d_G(u, v)$ for every $u, v \in V'$;

and our aim is to minimize the multiplicative distortion α where we refer to a G' with distortion α as an α-SPR solution. In the above d_G and $d_{G'}$ give the distances in G and G' respectively.

[3] A graph G' is a minor of a graph G if G' can be attained (up to isomorphism) from G by edge contractions as well as vertex and edge deletions. A graph is F-minor-free if it does not have F as a minor. A family of graphs $\mathcal{G}$ is said to be minor-closed if for any $G \in \mathcal{G}$ if G' is a minor of G then $G' \in \mathcal{G}$. A seminal work of Robertson and Seymour [53] demonstrated that every minor-closed family of graphs is fully characterized by a finite collection of "forbidden" minors. In particular, if $\mathcal{G}$ is a minor-closed family then there exists a finite collection of graphs $\mathcal{M}$ where $G \in \mathcal{G}$ iff G does not have any graph in $\mathcal{M}$ as a minor. Here and throughout this work we will use "minor-closed" to refer to all non-trivial minor-closed families of graphs; in particular, we exclude the family of all graphs which is minor-closed but trivially so.

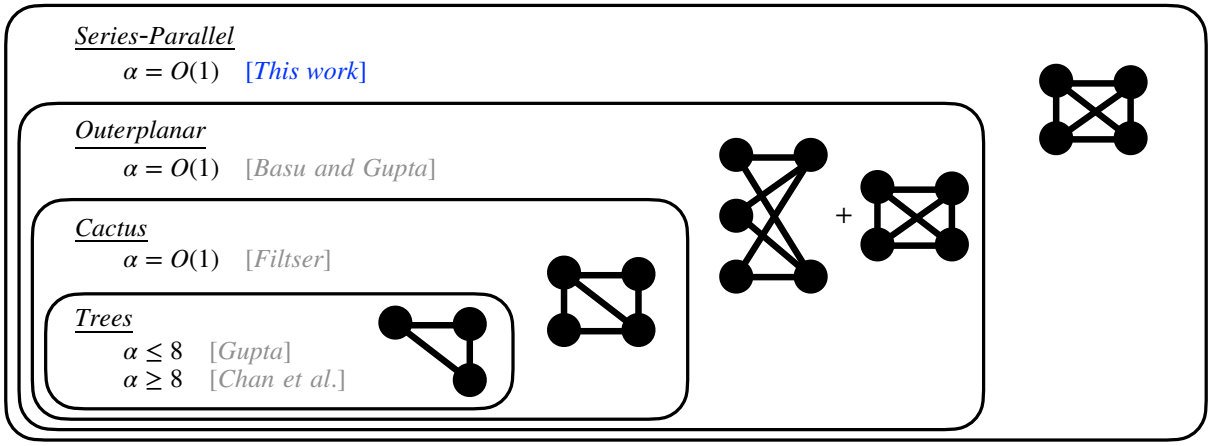


Figure 1: A summary of the SPR distortion for K_h-minor-free graphs achieved in prior work and our own. Graph classes illustrated according to containment. We also give the forbidden minors for each graph family.

If we only required that G' satisfy the second condition then we could always achieve $\alpha = 1$ by letting G' be the complete graph on V' where $w'(\{u, v\}) = d_G(u, v)$ for every $u, v \in V'$. However, such a G' forfeits any nice structure that G may have exhibited. Thus, the first condition ensures that if G belongs to a minor-closed family then so does G'. The second condition ensures that G''s metric is a good proxy for G's metric. G' is simpler than G since it is a graph only on V' while G' is a proxy for G's metric by approximately preserving distances on V'.

As Gupta [33] observed, even for the simple case of trees we must have $\alpha > 1$. For example, consider the star graph with unit weight edges where V' consists of the leaves of the star. Any tree $G' = (V', E', w')$ has at least two vertices u and v whose connecting path consists of at least two edges. On the other hand, the length of any edge in G' is at least 2 and so $d_{G'}(u, v) \geq 4$. Since $d_G(u, v) = 2$ it follows that $\alpha \geq 2$. While this simple example rules out the possibility of 1-SPR solutions on trees, it leaves open the possibility of small distortion solutions for minor-closed families.

In this vein several works have posed the existence of $O(1)$-SPR solutions for minor-closed families as an open question: see, for example, [10, 14, 18, 26, 47] among other works. A line of work (summarized in Figure 1) has been steadily making progress on this question for the past two decades. Gupta [33] showed that trees (i.e. K_3-minor-free graphs) admit 8-SPR solutions.[4] Filtser et al. [27] recently gave a simpler proof of this result. Chan et al. [14] proved this was tight by showing that $\alpha \geq 8$ for trees which remains the best known lower bound for K_h-minor-free graphs. In an exciting recent work, Filtser [26] reduced $O(1)$-SPR in K_h-minor-free graphs to computing "$O(1)$ scattering partitions" and showed how to compute these partitions for several graph classes, including cactus graphs (i.e. all F-minor-free graphs where F is K_4 missing one edge). Lastly, a work of Basu and Gupta [10] generalizes these results by showing that outerplanar graphs (i.e. graphs which are both K_4 and $K_{2,3}$-minor-free) have $\alpha = O(1)$ solutions.

3.3.1 $O(1)$ Steiner Point Removal in Series-Parallel Graphs

We advance the state-of-the-art for Steiner point removal in minor-closed graph families. We show that series-parallel graphs (i.e. graphs which are K_4-minor-free) have $O(1)$-SPR solutions. Series-parallel graphs

[4] Strictly speaking trees are not the class of all K_3-minor-free graphs nor are they really minor-closed since they are not closed under vertex and edge deletion. Really, the class of K_3-minor-free graphs are all *forests*. The stated results regarding trees also hold for forests but the literature seems to generally gloss over this detail and so we do so here and throughout the paper.

are a strict superset of all of the aforementioned graph classes for which $O(1)$-SPR solutions were previously known; again, see Figure 1. The following theorem summarizes our main result in this area.

Theorem 9. *Every series-parallel graph $G = (V, E, w)$ with terminal set $V' \subseteq V$ has a weighted minor $G' = (V', E', w')$ such that for any $u, v \in V'$ we have*

$$d_G(u, v) \leq d_{G'}(u, v) \leq O(1) \cdot d_G(u, v).$$

Moreover, G' is poly-time computable by a deterministic algorithm.

Our result will be based on a new graph partition introduced by Filtser [26], the scattering partition. Roughly speaking, a scattering partition of a graph is a low-diameter partition which respects the shortest path structure of the graph.[5]

Definition 13 (Scattering Partition)**.** *Given weighted graph $G = (V, E, w)$, a partition $\mathcal{P} = \{V_i\}_i$ of V is a (τ, Δ) scattering partition if:*

1. ***Connected:*** *Each $V_i \in \mathcal{P}$ is connected;*
2. ***Low Weak Diameter:*** *For each $V_i \in \mathcal{P}$ and $u, v \in V_i$ we have $d_G(u, v) \leq \Delta$;*
3. ***Scattering:*** *Every shortest path P in G of length at most Δ satisfies $|\{V_i : V_i \cap P \neq \emptyset\}| \leq \tau$.*

Filtser [26] extended these partitions to the notion of a scatterable graph.

Definition 14 (Scatterable Graph)**.** *A weighted graph $G = (V, E, w)$ is τ-scatterable if it has a (τ, Δ)-scattering partition for every $\Delta \geq 0$.*

We will say that G is deterministic poly-time τ-scatterable if for every $\Delta \geq 0$ a (τ, Δ)-scattering partition is computable in deterministic poly-time.

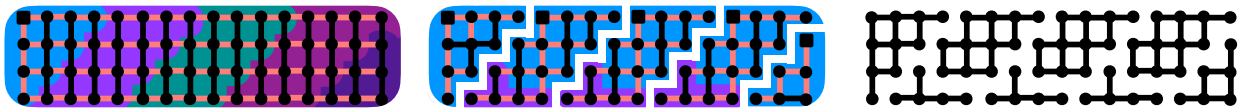
Lastly, the main result of Filtser [26] is that solving SPR reduces to showing that every induced subgraph is scatterable. In the following $G[A]$ is the subgraph of G induced by the vertex set A.

Theorem 10 (Filtser [26])**.** *A weighted graph $G = (V, E, w)$ with terminal set $V' \subseteq V$ has an $O(\tau^3)$-SPR solution if $G[A]$ is τ-scatterable for every $A \subseteq V$. Furthermore, if $G[A]$ is deterministic poly-time scatterable for every $A \subseteq V$ then the $O(\tau^3)$-SPR solution is computable in deterministic poly-time.*

Thus, given a series-parallel graph G and some $\Delta \geq 1$, our goal is to compute an $(O(1), \Delta)$-scattering partition for G. Such a partition has two non-trivial properties to satisfy: (1) each constituent part must have weak diameter at most Δ and (2) each shortest path of length at most Δ must be in at most $O(1)$ parts (a property we will call "scattering").

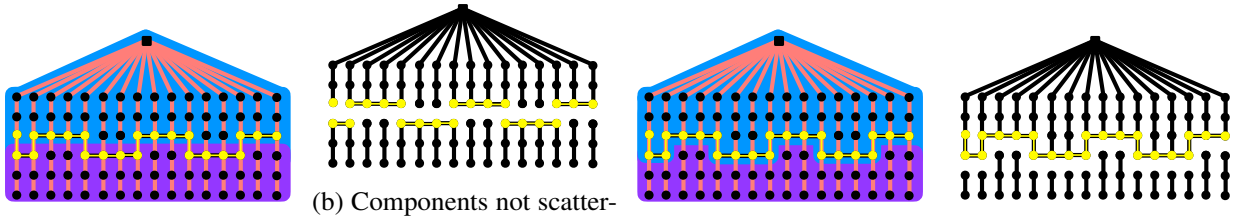
A well-known technique of Klein et al. [45]—henceforth "KPR"—has proven useful in finding so-called low diameter decompositions for K_h-minor-free graphs and so one might reasonably expect these techniques to prove useful for finding scattering partitions. Specifically, KPR shows that computing low diameter decompositions in a K_h-minor-free graph can be accomplished by $O(h)$ levels of recursive "Δ-chops". Fix a root r and a BFS tree T_{BFS} rooted at r. Then, a Δ-chop consists of the deletion of every edge with one vertex at depth $i \cdot \Delta$ and another vertex at depth $i \cdot \Delta + 1$ for every $i \in \mathbb{Z}_{\geq 1}$; that is, it consists of cutting edges between each pair of adjacent Δ-width annuli. KPR proved that if one performs a Δ-chop and then recurses on each of the resulting connected component then after $O(h)$ levels of recursive depth in a K_h-minor free graph the resulting components all have diameter at most $O(\Delta)$. We illustrate KPR on the grid graph in Figure 3.

[5] We drop one of the parameters of the definition of Filtser [26] as it will not be necessary for our purposes.



(a) First Δ-chop. (b) Second Δ-chop. (c) Resulting connected components.

Figure 2: Two levels of Δ-chops on the grid graph for $\Delta = 3$. We give the edges of the BFS trees we use in pink; roots of these trees are given as squares. Background colors give the annuli of nodes.



(a) A Δ-chop (b) Components not scattering. (c) A perturbed Δ-chop. (d) Components scattering.

Figure 3: An example where a Δ-chop does not produce a scattering partition but how perturbing said chop does. Here, we imagine that the root is at the top of the graph and each edge incident to the root has length $\Delta - 3$. We highlight the path P that either ends up in many or one connected component depending on whether we perturb our Δ-chop in yellow.

Thus, we could simply apply Δ-chops $O(h)$ times to satisfy our diameter constraints (up to constants) and hope that the resulting partition is also scattering. Unfortunately, it is quite easy to see that (even after just one Δ-chop!) a path of length at most Δ can end up in arbitrarily many parts of the resulting partition. For example, the highlighted shortest path in Figures 3a and 3b repeatedly moves back and forth between two annuli and ends up in arbitrarily many parts after a single Δ-chop. Nonetheless, this example is suggestive of the basic approach of our work. In particular, if we merely perturbed our first Δ-chop to cut "around" said path as in Figures 3c and 3d then we could ensure that this path ends up in a small number of partitions.

More generally, the approach we take in this work is to start with the KPR chops but then slightly perturb these chops so that they do not cut *any* shortest path of length at most Δ more than $O(1)$ times. That is, all but $O(1)$ edges of any such path will have both vertices in the same (perturbed) annuli. We then repeat this recursively on each of the resulting connected components to a constant recursion depth. Since each subpath of a shortest path of length at most Δ is itself a shortest path with length at most Δ, we know that each such shortest path is broken into a constantly-many-more shortest paths at each level of recursion. Moreover, since we recurse a constant number of times, each path ends up in a constant number of components.

Implementing this strategy requires meeting two challenges. First, it is not clear that the components resulting from KPR still have low diameter if we allow ourselves to perturb our chops. Second, it is not clear how to perturb a chop so that it works *simultaneously* for every shortest path. Solving the first challenge will be somewhat straightforward while solving the second will be significantly more involved. In particular, what makes the second challenge difficult is that we cannot, in general, perturb a chop on the basis of one violated shortest path as in the previous example; doing so might cause other paths to be cut too many times which will then require additional, possibly conflicting, perturbations and so on. Rather, the approach we take is to perturb our chops in a way that takes every shortest path into account all at once.

Scattering Chops. The easier issue to solve will be how to ensure that our components have low diameter even if we perturb our chops. Here, by closely tracking various constants through a known analysis of KPR we show that the components resulting from KPR with (boundedly) perturbed cuts are still low diameter.

We summarize this fact and the above discussion with the idea of a scattering chop. A (τ, Δ)-scattering chop consists of cutting all edges at *about* every Δ levels in the BFS tree in such a way that no shortest path of length at most Δ is cut more than τ times. Our analysis shows that if all K_h-minor-free graphs admit $(O(1), \Delta)$-scattering chops for every Δ then they are also $O(1)$-scatterable and therefore also admit $O(1)$-SPR solutions; this holds even for $h > 4$.

Hammock Decompositions and How to Use Them. The more challenging issue we must overcome is how to perturb our chops so that every shortest path of length at most Δ is only cut $O(1)$ times. Moreover, we must do so in a way that does not perturb our boundaries by too much so as to meet the requirements of a scattering chop. We solve this issue with our new metric decomposition for series-parallel graphs, the hammock decomposition.

Consider a shortest path P of length at most Δ. Such a path can be partitioned into a (possibly empty) prefix consisting of only edges in T_{BFS}, a middle portion whose first and last edges are cross edges of T_{BFS} and a (possibly empty) suffix which also only has edges in T_{BFS}. Thus, if we want to compute a scattering chop, it suffices to guarantee than any shortest path of length at most Δ which is either fully contained in T_{BFS} or which is between two cross edges of T_{BFS} is only cut $O(1)$ times by our chop; call the former a BFS path and the latter a cross edge path.

Next, notice that all BFS paths are only cut $O(1)$ times by our initial KPR chops. Specifically, each BFS path can be partitioned into a subpath which goes "up" in the BFS tree and a subpath which goes "down" in the BFS tree. As our initial KPR chops are Δ apart and each such subpath is of length at most Δ, each such subpath is cut at most $O(1)$ times. Thus provided our perturbations do not interfere *too much* with the initial structure of our KPR chops we should expect that our BFS paths will only be cut $O(1)$ times.

Thus, our goal will be to perturb our KPR chops to not cut any cross edge path more than $O(1)$ times while mostly preserving the initial structure of our KPR chops. Our hammock decompositions will allow us to do exactly this. They will have two key components.

The first part is a "forest of hammocks." Suppose for a moment that our input graph had a forest subgraph F that contained all cross edge paths of our graph which were also shortest paths. Then, it is not too hard to see how to use F to perturb our chops to be scattering for all cross edge paths. Specifically, for each tree T in our forest F we fix an arbitrary root and then process edges in a BFS order. Edges which we process will be marked or unmarked where initially all edges are unmarked. To process an edge $e = \{u, v\}$ we do the following. If e is marked or u and v both belong to the same annuli then we do nothing. Otherwise, e is unmarked and u is in some annuli A but v is in some other annuli A' (before any perturbation). We then propagate A an additional $\Theta(\Delta)$ deeper into T; that is if we imagine that v is the child of u in F then we move all descendants of u in F within $\Theta(\Delta)$ of u into A. We then mark all edges in T whose endpoints are descendants of u and within $\Theta(\Delta)$ of u. A simple amortized analysis shows that after performing these perturbations every cross edge path is cut $O(1)$ times.

Unfortunately, it is relatively easy to see that such an F may not exist in a series-parallel graph. The forest of hammocks component of our decompositions is a subgraph which will be "close enough" to such an F, thereby allowing us to perturb our chops similarly to the above strategy. A hammock graph consists of two subtrees of a BFS tree and the cross edges between them. A forest of hammocks is a graph partitioned into hammocks where every cycle is fully contained in one of the constituent hammocks. While the above perturbation will guarantee that our cross edge paths are not cut too often, it is not clear that such a perturbation does not change the structure of our initial chops in a way that causes our BFS paths to be cut too many times.

The second part of our hammock decompositions is what we use to guarantee that our BFS paths are not cut too many times by preserving the structure of our initial KPR chops. Specifically, the forest structure of

our hammocks will reflect the structure of T_{BFS}. In particular, we can naturally associate each hammock H_i with a single vertex, namely the lca of any u and v where u is in one tree of H_i and v is in the other. Then, our forest of hammocks will satisfy the property that if hammock H_i is a "parent" of hammock H_j in our forest of hammocks then the lca corresponding to H_i is an ancestor of the lca corresponding to H_j in T_{BFS}; even stronger, the lca of H_j will be contained in H_i. Roughly, the fact that our forest of hammocks mimics the structure of T_{BFS} in this way will allow us to argue that the above perturbation does not alter the initial structure of our KPR chops too much, thereby ensuring that BFS paths are not cut too many times.

The computation of our hammock decompositions constitutes the bulk of our technical work but is somewhat involved. The basic idea is as follows. We will partition all cross edges into equivalence classes where each cross edge in an equivalence class share an lca in T_{BFS} (though there may be multiple, distinct equivalence classes with the same lca). Each such equivalence class will eventually correspond to one hammock in our forest of hammocks. To compute our forest of hammocks we first connect up all cross edges in the same equivalence class. Next we connect our equivalence classes to one another by cross edge paths which run between them. We then extend our hammocks along paths towards their lcas to ensure the above-mentioned lca properties. Finally, we add so far unassigned subtrees of T_{BFS} to our hammocks. We will argue that when this process fails it shows the existence of a K_4-minor and, in particular, a clawed cycle.

3.3.2 Preliminary Results and Proposed Work

The next open question in this line of work to tackle and the question to which we would like to give more thought is whether planar graphs admit $O(1)$ Steiner point removal solutions.

To this end we have shown a theorem analogous to Theorem 10 but where we do not require a scattering partition but only an *approximate* scattering partition. An approximate scattering partitions is defined as a scattering partition but where instead of the scattering condition we require that for every pair of vertices u and v and every shortest path P between u and v there is *some* path of length $O(1) \cdot w(P)$ between u and v which satisfies $|\{V_i : V_i \cap P \neq \emptyset\}| \leq \tau$. Our hope is that we will be able to show that every planar graph admits such an approximate scattering partition with $\tau = O(1)$ which by this theorem would be sufficient to show the existence of $O(1)$ Steiner point removal solutions for planar graphs.

As a first step towards this goal we plan to give some thought to a special case of planar graphs which we call spider grid graphs. A spider grid graph is a planar graph which consists of edge-disjoint paths $P_1, P_2, \ldots$ which pairwise have some root vertex in common r. Any edge $e = (u, v)$ not in one of these paths is such that if $u \in P_i$ then $v \in P_{i+1}$. Considering the series-parallel analogue of these graphs is what led us to our $O(1)$ SPR solutions for series-parallel graphs and so we hope to accomplish something similar with planar graphs.

Another interesting open question to explore would be that of $O(1)$ Steiner point removal solutions in bounded treewidth graphs. In particular, prior work of Filtser [26] showed that trees (i.e. treewidth 1 graphs) are $O(1)$ scatterable. A graph has treewidth 2 iff it is series-parallel and so our work shows that treewidth 2 graphs have $O(1)$ SPR solutions. Thus, another natural next question to consider would be whether bounded treewidth graphs admit $O(1)$ SPR solutions.

References

[1] Ittai Abraham and Ofer Neiman. Using petal-decompositions to build a low stretch spanning tree. pages 395–406, 2012.

[2] Noga Alon, Richard M Karp, David Peleg, and Douglas West. A graph-theoretic game and its application to the k-server problem. *SIAM Journal on Computing*, 24(1):78–100, 1995.

[3] Noga Alon, Baruch Awerbuch, Yossi Azar, Niv Buchbinder, and Joseph Naor. A general approach to online network optimization problems. 2(4):640–660, 2006.

[4] Ernst Althaus, Stefan Funke, Sariel Har-Peled, Jochen Könemann, Edgar A Ramos, and Martin Skutella. Approximating k-hop minimum-spanning trees. *Operations Research Letters*, 33(2):115–120, 2005.

[5] Sanjeev Arora, Michelangelo Grigni, David R Karger, Philip N Klein, and Andrzej Woloszyn. A polynomial-time approximation scheme for weighted planar graph tsp. volume 98, pages 33–41, 1998.

[6] Baruch Awerbuch and Yossi Azar. Buy-at-bulk network design. pages 542–547. IEEE, 1997.

[7] Nikhil Bansal, Niv Buchbinder, Aleksander Madry, and Joseph Naor. A polylogarithmic-competitive algorithm for the k-server problem. pages 267–276. IEEE, 2011.

[8] Yair Bartal. Probabilistic approximation of metric spaces and its algorithmic applications. pages 184–193. IEEE, 1996.

[9] Yair Bartal, Avrim Blum, Carl Burch, and Andrew Tomkins. A polylog (n)-competitive algorithm for metrical task systems. pages 711–719, 1997.

[10] Amitabh Basu and Anupam Gupta. Steiner point removal in graph metrics. *Unpublished Manuscript, available from http://www. math. ucdavis. edu/˜ abasu/papers/SPR. pdf*, 1:25, 2008.

[11] Piotr Berman, Bhaskar DasGupta, and Marek Karpinski. Approximating transitivity in directed networks. *arXiv preprint arXiv:0809.0188*, 2008.

[12] Marcin Bienkowski, Björn Feldkord, and Paweł Schmidt. A nearly optimal deterministic online algorithm for non-metric facility location. *arXiv preprint arXiv:2007.07025*, 2020.

[13] Niv Buchbinder and Joseph Naor. *The design of competitive online algorithms via a primal-dual approach*. Now Publishers Inc, 2009.

[14] T-H Hubert Chan, Donglin Xia, Goran Konjevod, and Andrea Richa. A tight lower bound for the steiner point removal problem on trees. pages 70–81. Springer, 2006.

[15] Moses Charikar, Chandra Chekuri, Ashish Goel, Sudipto Guha, and Serge Plotkin. Approximating a finite metric by a small number of tree metrics. pages 379–388. IEEE, 1998.

[16] Chandra Chekuri, Anupam Gupta, Ilan Newman, Yuri Rabinovich, and Alistair Sinclair. Embedding k-outerplanar graphs into l1. *SIAM Journal on Discrete Mathematics*, 20(1):119–136, 2006.

[17] James Cheng, Yiping Ke, Shumo Chu, and M Tamer Özsu. Efficient core decomposition in massive networks. In *2011 IEEE 27th International Conference on Data Engineering*, pages 51–62. IEEE, 2011.

[18] Yun Kuen Cheung, Gramoz Goranci, and Monika Henzinger. Graph minors for preserving terminal distances approximately-lower and upper bounds. 2016.

[19] Kedar Dhamdhere, Vineet Goyal, R Ravi, and Mohit Singh. How to pay, come what may: Approximation algorithms for demand-robust covering problems. pages 367–376. IEEE, 2005.

[20] Laxman Dhulipala. Provably efficient and scalable shared-memory graph processing.

[21] Matthias Englert and Harald Räcke. Reordering buffers with logarithmic diameter dependency for trees. pages 1224–1234. SIAM, 2017.

[22] Matthias Englert, Harald Räcke, and Matthias Westermann. Reordering buffers for general metric spaces. 2007.

[23] Jittat Fakcharoenphol, Satish Rao, and Kunal Talwar. A tight bound on approximating arbitrary metrics by tree metrics. *Journal of Computer and System Sciences*, 69(3):485–497, 2004.

[24] Uriel Feige, Kamal Jain, Mohammad Mahdian, and Vahab Mirrokni. Robust combinatorial optimization with exponential scenarios. pages 439–453. Springer, 2007.

[25] Amos Fiat and Manor Mendel. Better algorithms for unfair metrical task systems and applications. *SIAM Journal on Computing*, 32(6):1403–1422, 2003.

[26] Arnold Filtser. Scattering and sparse partitions, and their applications. 2020.

[27] Arnold Filtser, Robert Krauthgamer, and Ohad Trabelsi. Relaxed voronoi: A simple framework for terminal-clustering problems. 2018.

[28] Harold N Gabow. On the l-infinity-norm of extreme points for crossing supermodular directed network lps. In *International Conference on Integer Programming and Combinatorial Optimization*, pages 392–406. Springer, 2005.

[29] Harold N Gabow, Michel X Goemans, Éva Tardos, and David P Williamson. Approximating the smallest k-edge connected spanning subgraph by lp-rounding. *Networks: An International Journal*, 53(4):345–357, 2009.

[30] Naveen Garg, Goran Konjevod, and R Ravi. A polylogarithmic approximation algorithm for the group steiner tree problem. *Journal of Algorithms*, 37(1):66–84, 2000.

[31] Gramoz Goranci, Monika Henzinger, and Pan Peng. Improved guarantees for vertex sparsification in planar graphs. *SIAM Journal on Discrete Mathematics*, 34(1):130–162, 2020.

[32] Xiangyu Guo, Janardhan Kulkarni, Shi Li, and Jiayi Xian. On the facility location problem in online and dynamic models. Schloss Dagstuhl-Leibniz-Zentrum für Informatik, 2020.

[33] Anupam Gupta. Steiner points in tree metrics don't (really) help. pages 682–690, 2001.

[34] Varun Gupta, Ravishankar Krishnaswamy, and Sai Sandeep. Permutation strikes back: The power of recourse in online metric matching. 2020.

[35] Frank Hadlock. Finding a maximum cut of a planar graph in polynomial time. *SIAM Journal on Computing*, 4(3):221–225, 1975.

[36] Bernhard Haeupler, D Ellis Hershkowitz, and Goran Zuzic. Deterministic tree embeddings with copies for algorithms against adaptive adversaries. *arXiv preprint arXiv:2102.05168*, 2021.

[37] Bernhard Haeupler, D Ellis Hershkowitz, and Goran Zuzic. Tree embeddings for hop-constrained network design. In *Proceedings of the 53rd Annual ACM SIGACT Symposium on Theory of Computing*, pages 356–369, 2021.

[38] D Ellis Hershkowitz and Jason Li. $o(1)$ steiner point removal in series-parallel graphs. *arXiv preprint arXiv:2104.00750*, 2021.

[39] D Ellis Hershkowitz, Gregory Kehne, and R Ravi. An optimal rounding for half-integral weighted minimum strongly connected spanning subgraph. *Information Processing Letters*, 167:106067, 2021.

[40] Richard M Karp. A 2k-competitive algorithm for the circle. *Manuscript, August*, 5, 1989.

[41] Adam Kasperski and Paweł Zieliński. On the approximability of robust spanning tree problems. *Theoretical Computer Science*, 412(4-5):365–374, 2011.

[42] Rohit Khandekar, Guy Kortsarz, Vahab Mirrokni, and Mohammad R Salavatipour. Two-stage robust network design with exponential scenarios. pages 589–600. Springer, 2008.

[43] M Reza Khani and Mohammad R Salavatipour. Improved approximations for buy-at-bulk and shallow-light k-steiner trees and (k, 2)-subgraph. pages 20–29. Springer, 2011.

[44] Samir Khuller, Balaji Raghavachari, and Neal Young. Balancing minimum spanning trees and shortest-path trees. *Algorithmica*, 14(4):305–321, 1995.

[45] Philip Klein, Serge A Plotkin, and Satish Rao. Excluded minors, network decomposition, and multicommodity flow. pages 682–690, 1993.

[46] Robert Krauthgamer and Havana Inbal Rika. Refined vertex sparsifiers of planar graphs. *SIAM Journal on Discrete Mathematics*, 34(1):101–129, 2020.

[47] Robert Krauthgamer, Huy L Nguyen, and Tamar Zondiner. Preserving terminal distances using minors. *SIAM Journal on Discrete Mathematics*, 28(1):127–141, 2014.

[48] Bundit Laekhanukit, Shayan Oveis Gharan, and Mohit Singh. A rounding by sampling approach to the minimum size k-arc connected subgraph problem. In *International Colloquium on Automata, Languages, and Programming*, pages 606–616. Springer, 2012.

[49] A Ridha Mahjoub and S Thomas McCormick. Max flow and min cut with bounded-length paths: complexity, algorithms, and approximation. *Mathematical programming*, 124(1):271–284, 2010.

[50] Viswanath Nagarajan and Ramamoorthi Ravi. Approximation algorithms for requirement cut on graphs. In *Approximation, Randomization and Combinatorial Optimization. Algorithms and Techniques*, pages 209–220. Springer, 2005.

[51] Joseph Naor, Debmalya Panigrahi, and Mohit Singh. Online node-weighted steiner tree and related problems. pages 210–219. IEEE, 2011.

[52] Harald Racke. Minimizing congestion in general networks. pages 43–52. IEEE, 2002.

[53] Neil Robertson and Paul D Seymour. Graph minors. xx. wagner's conjecture. *Journal of Combinatorial Theory, Series B*, 92 (2):325–357, 2004.

[54] André Rossi, Alexis Aubry, and Mireille Jacomino. Connectivity-and-hop-constrained design of electricity distribution networks. *European journal of operational research*, 218(1):48–57, 2012.

[55] Adrian Vetta. Approximating the minimum strongly connected subgraph via a matching lower bound. pages 417–426. Citeseer, 2001.

[56] Thomas Victor Wimer. Linear algorithms on k-terminal graphs. *PhD Thesis*, 1987. AAI8803914.

[57] Kathleen A Woolston and Susan L Albin. The design of centralized networks with reliability and availability constraints. *Computers & Operations Research*, 15(3):207–217, 1988.