Sublinear Algorithms and Lower Bounds for Estimating MST and TSP Cost in General Metrics

Yu Chen
EPFL, Lausanne, Switzerland

Sanjeev Khanna
University of Pennsylvania, Philadelphia, PA, USA

Zihan Tan
DIMACS, Rutgers University, NJ, USA

Abstract

We consider the design of sublinear space and query complexity algorithms for estimating the cost of a minimum spanning tree (MST) and the cost of a minimum traveling salesman (TSP) tour in a metric on n points. We start by exploring this estimation task in the regime of $o(n)$ space, when the input is presented as a stream of all $\binom{n}{2}$ entries of the metric in an arbitrary order (a metric stream). For any $\alpha \geq 2$, we show that both MST and TSP cost can be α-approximated using $\tilde{O}(n/\alpha)$ space, and moreover, $\Omega(n/\alpha^2)$ space is necessary for this task. We further show that even if the streaming algorithm is allowed p passes over a metric stream, it still requires $\tilde{\Omega}(\sqrt{n/\alpha p^2})$ space.

We next consider the well-studied semi-streaming regime. In this regime, it is straightforward to compute MST cost exactly even in the case where the input stream only contains the edges of a weighted graph that induce the underlying metric (a graph stream), and the main challenging problem is to estimate TSP cost to within a factor that is strictly better than 2. We show that in graph streams, for any $\varepsilon > 0$, any one-pass $(2-\varepsilon)$-approximation of TSP cost requires $\Omega(\varepsilon^2 n^2)$ space. On the other hand, we show that there is an $\tilde{O}(n)$ space two-pass algorithm that approximates the TSP cost to within a factor of 1.96.

Finally, we consider the query complexity of estimating metric TSP cost to within a factor that is strictly better than 2 when the algorithm is given access to an $n \times n$ matrix that specifies pairwise distances between n points. The problem of MST cost estimation in this model is well-understood and a $(1+\varepsilon)$-approximation is achievable by $\tilde{O}(n/\varepsilon^{O(1)})$ queries. However, for estimating TSP cost, it is known that an analogous result requires $\Omega(n^2)$ queries even for $(1,2)$-TSP, and for general metrics, no algorithm that achieves a better than 2-approximation with $o(n^2)$ queries is known. We make progress on this task by designing an algorithm that performs $\tilde{O}(n^{1.5})$ distance queries and achieves a strictly better than 2-approximation when either the metric is known to contain a spanning tree supported on weight-1 edges or the algorithm is given access to a minimum spanning tree of the graph. Prior to our work, such results were only known for the special cases of graphic TSP and $(1,2)$-TSP.

In terms of techniques, our algorithms for metric TSP cost estimation in both streaming and query settings rely on estimating the *cover advantage* which intuitively measures the cost needed to turn an MST into an Eulerian graph. One of our main algorithmic contributions is to show that this quantity can be meaningfully estimated by a sublinear number of queries in the query model. On one hand, the fact that a metric stream reveals pairwise distances for all pairs of vertices provably helps algorithmically. On the other hand, it also seems to render useless techniques for proving space lower bounds via reductions from well-known hard communication problems. Our main technical contribution in lower bounds is to identify and characterize the communication complexity of new problems that can serve as canonical starting point for proving metric stream lower bounds.

2012 ACM Subject Classification Theory of computation → Design and analysis of algorithms

Keywords and phrases Minimum spanning tree, travelling salesman problem, streaming algorithms

Digital Object Identifier 10.4230/LIPIcs.ICALP.2023.37

Category Track A: Algorithms, Complexity and Games

Related Version *Full Version*: https://arxiv.org/abs/2203.14798

Funding *Yu Chen*: Supported by ERC Starting Grant 759471.
Sanjeev Khanna: Supported in part by NSF awards CCF-1934876 and CCF-2008305.
Zihan Tan: Supported by a grant to DIMACS from the Simons Foundation (820931).

1 Introduction

The minimum spanning tree (MST) problem and the metric traveling salesman (TSP) problem are among the most well-studied combinatorial optimization problems with a long and rich history. The two problems are intimately connected to one another, as many approximation algorithms for metric TSP use a minimum spanning tree as a starting point for efficiently constructing an approximate solution. In particular, any algorithm for estimating the MST cost to within a factor of α immediately implies an algorithm for estimating the metric TSP cost to within a factor of 2α. In this work, we consider the design of sublinear space and query complexity algorithms for estimating the cost of a minimum spanning tree (MST) and the cost of a minimum metric traveling salesman (TSP) tour in an n-vertex weighted undirected graph G. An equivalent view of both problems is that we are given an $n \times n$ matrix w specifying pairwise distances between them, where the entry $w[u, v]$ corresponds to the weight of the shortest path from u to v in G. It is clear that any algorithm that works with a weighted graph as input also works when the input is presented as the complete metric. However, the converse is not true. For instance, no single-pass streaming algorithm can obtain a finite approximation to the diameter (or even determine the connectivity) of a graph in $o(n)$ space when the graph is presented as a sequence of edges (a *graph stream*). But if instead we are presented a stream of n^2 entries of the metric matrix w (a *metric stream*), there is a trivial $\tilde{O}(1)$ space algorithm for this problem – simply track the largest entry seen.

1.1 Our Results

In the first part of this work, we explore the power and limitations of graph and metric streams for MST and TSP cost estimation. We start by exploring this estimation task in the regime of $o(n)$ space in the streaming model. It is easy to show that no finite approximation to MST/TSP cost is achievable in this regime when the input stream simply contains the edges of a weighted graph that induce the underlying metric (a graph stream). However, we show that this state of affairs changes completely if the input is instead presented as all entries of the shortest-path-distance metric induced by the input graph (a metric stream).

▶ **Theorem 1.** *For any $\alpha > 1$, there is a randomized one-pass α-approximation streaming algorithm for MST cost estimation in metric streams using $\tilde{O}(n/\alpha)$ space.*

Note that this also immediately gives a one-pass $\tilde{O}(n/\alpha)$-space algorithm for TSP cost estimation for any $\alpha \geq 2$ by simply doubling the MST cost estimate. The result above is in sharp contrast to what is achievable in graph streams. Using a simple reduction from the Index problem, we can show the following lower bound for graph streams.

▶ **Theorem 2.** *For any $\alpha > 1$, any randomized p-pass α-approximation streaming algorithm for MST cost estimation in graph streams requires $\tilde{\Omega}(n/p)$ space.*

We next show that there are limits to the power of metric streams, and in particular, any non-trivial approximation of MST cost still requires polynomial space even if we allow multiple passes over the stream.

▶ **Theorem 3.** *For any $\alpha > 1$, any randomized one-pass α-approximation streaming algorithm for MST cost estimation in metric streams requires $\Omega(n/\alpha^2)$ space.*

▶ **Theorem 4.** *For any $\alpha > 1$, any randomized p-pass α-approximation streaming algorithm for MST cost estimation requires $\tilde{\Omega}(\sqrt{n/\alpha p^2})$ space.*

Table 1 summarizes our results for MST (and TSP) cost estimation in the regime of $o(n)$ space.

Table 1 Summary of results for MST-cost estimation streaming algorithms.

Stream Type	MST estimation		
	# of passes	Approximation ratio	Upper or Lower bounds
Metric Stream	1	1	$\tilde{O}(n)$ (trivial)
	1	α	$\tilde{O}(n/\alpha)$ (Theorem 1), $\tilde{\Omega}(n/\alpha^2)$ (Theorem 3)
	p	α	$\tilde{\Omega}(\sqrt{n/\alpha p^2})$ (Theorem 4)
Graph Stream	1	1	$\tilde{\Theta}(n)$ (trivial)
	p	any	$\Omega(n/p)$ (Theorem 2)

We next consider the well-studied semi-streaming regime when the streaming algorithm is allowed to use $\tilde{O}(n)$ space. In this regime, it is straightforward to design a deterministic one-pass streaming algorithm to compute MST cost exactly even in graph streams, and this in turn, immediately gives an $\tilde{O}(n)$ space algorithm to estimate TSP cost to within a factor of 2. Thus in the semi-streaming regime, the key challenging problem is to estimate TSP cost to within a factor that is strictly better than 2. A special case of this problem, *graphic TSP cost* estimation, where the input metric corresponds to the shortest-path distances induced by an unweighted undirected graph, was studied in [4], and the authors gave an $\tilde{O}(n)$ space randomized one-pass streaming algorithm that achieves an (11/6)-approximation even in the setting of graph streams. This ratio was recently improved[1] by Behnezhad, Roghani, Rubinstein, and Saberi to 1.83 [2]. However, no analogous result is known for general TSP. We show that there is in fact a good reason for this state of affairs:

▶ **Theorem 5.** *For any $0 < \epsilon < 1$, any randomized one-pass $(2-\varepsilon)$-approximation streaming algorithm for TSP cost estimation in graph streams requires $\Omega(\epsilon^2 n^2)$ space.*

However, we show that the situation changes considerably once we allow two passes and indeed there is now a deterministic $\tilde{O}(n)$ space algorithm that achieves better than a 2-approximation to TSP cost.

▶ **Theorem 6.** *There is a deterministic two-pass 1.96-approximation algorithm for TSP cost estimation in graph streams using $\tilde{O}(n)$ space.*

We note that an interesting remaining question here is if a similar result is achievable using one pass when the input is a metric stream. As a step towards understanding the power of metric streams in semi-streaming regime, we show that any one-pass algorithm that computes TSP cost exactly requires $\Omega(n^2)$ space. Table 2 summarizes our results for TSP cost estimation in the regime of semi-streaming space.

[1] In their paper, they give an $\tilde{O}(n)$-time 1.83-approximation algorithm, which can be easily turned into a one-pass streaming algorithm with space $\tilde{O}(n)$ with the same approximation ratio.

Table 2 Summary of results for TSP-cost estimation streaming algorithms. The statements and proofs for entries marked by (∗) is deferred to the full version.

Stream Type	TSP estimation		
	# of passes	Approximation ratio	Upper or Lower bounds
Metric Stream	1	1	$\Omega(n^2)$*
	1	$2-\varepsilon$	Open
Graph Stream	1	2	$\tilde{\Theta}(n)$ (trivial)
	1	$2-\varepsilon$	$\tilde{\Omega}(\varepsilon^2 n^2)$ (Theorem 5)
	2	1.96	$\tilde{O}(n)$ (Theorem 6)

The second part of our paper focuses on the design of sublinear query complexity algorithms for TSP cost estimation. The related problem of estimating the MST cost using sublinear queries was first studied in the graph adjacency-list model by Chazelle, Rubinfeld, and Trevisan [3]. The authors gave an $\tilde{O}(dW/\varepsilon^2)$-time algorithm to estimate the MST cost to within a factor of $(1+\varepsilon)$ in a graph where the average degree is d, and all edge costs are integers in $\{1,\ldots,W\}$. For certain parameter regimes this gives a sublinear time algorithm for estimating the MST cost, but in general, this run-time need not be sublinear. In fact, it is not difficult to show that in general, even checking if a graph is connected requires $\Omega(n^2)$ queries in the graph adjacency-list model, and hence no finite approximation to MST cost can be achieved in $o(n^2)$ queries. However, the situation changes if one restricts attention to the *metric* MST problem where the edge weights satisfy the triangle inequality, and the algorithm is given access to an $n \times n$ matrix w specifying pairwise distances between vertices. Czumaj and Sohler [6] showed that for any $\varepsilon > 0$, there exists an $\tilde{O}(n/\varepsilon^{O(1)})$ query algorithm that returns a $(1+\varepsilon)$-approximate estimate of the metric MST cost. This result immediately implies an $\tilde{O}(n/\varepsilon^{O(1)})$ time algorithm to estimate the TSP cost to within a factor of $(2+\varepsilon)$ for any $\varepsilon > 0$. In sharp contrast to this result, so far no $o(n^2)$ query algorithms are known to approximate metric TSP cost to a factor that is strictly better than 2. In this work, we consider sublinear query algorithms for TSP cost when the algorithm is given query access to the $n \times n$ distance matrix w. *We will assume throughout the paper that all entries of w are positive integers.*

For the special case of graphic TSP, where the metric corresponds to shortest path distances of some underlying connected unweighted graph, the algorithm of Chen, Kannan, and Khanna [4] combined with the recent result of Behnezhad [1] (which builds on the work of Yoshida et al. [8] and Onak et al. [7]), gives an $\tilde{O}(n)$-query (27/14)-approximation algorithm for estimating graphic TSP cost. The authors in [4] also show that there exists an $\varepsilon_0 > 0$, such that any algorithm that estimates the cost of graphic TSP (or even $(1,2)$-TSP) to within a $(1+\varepsilon_0)$-factor, necessarily requires $\Omega(n^2)$ queries. Later on, Behnezhad, Roghani, Rubinstein, and Saberi [2] improved the graphic TSP result by giving an $\tilde{O}(n)$-query 1.83-approximation, and they also gave an $\tilde{O}(n)$-query $(1.5+\varepsilon)$-approximation algorithm for $(1,2)$-TSP. This leaves open the following question: Is there an $o(n^2)$ query algorithm to estimate TSP cost to a factor strictly better than 2 when the metric is *arbitrary*?

We make progress on this question by designing an $\tilde{O}(n^{1.5})$-query algorithm that achieves a strictly better than 2-approximation when either the metric is known to contain a spanning tree supported on weight-1 edges or the algorithm is given access to a minimum spanning tree of the graph. Prior to our work, such results were only known for the special cases of graphic TSP and $(1,2)$-TSP.

▶ **Theorem 7.** *There is a randomized algorithm, that, given access to an n-point metric w with the promise that w contains a minimum spanning tree supported only on weight-1 edges, estimates with high probability the metric TSP cost to within a factor of $(2-\varepsilon_0)$ for some universal constant $\varepsilon_0 > 0$, by performing $\tilde{O}(n^{1.5})$ queries to w.*

We note that the setting of Theorem 7 captures as a special case graphic TSP but is considerably more general, and hence difficult.

▶ **Theorem 8.** *There is a randomized algorithm, that, given access to an n-point metric w and an arbitrary minimum spanning tree of the complete graph with edge weights given by w, estimates with high probability the metric TSP cost to within a factor of $(2-\varepsilon_0)$ for some universal constant $\varepsilon_0 > 0$, by performing $\tilde{O}(n^{1.5})$ queries to w.*

In what follows, we give an overview of the techniques underlying our results.

1.2 Technical Overview

1.2.1 Overview of Algorithmic Techniques

Our streaming algorithm for MST estimation (Theorem 1) utilizes a rather natural idea. We sample $O(n/\alpha)$ vertices and maintain a MST T' over them. For the remaining vertices, we maintain an estimate of the cost of connecting them to the nearest vertex in T'. We show that these estimates can be suitably combined to obtain an α-approximation of MST cost. In this subsection we focus on providing a high-level overview of the algorithms for TSP estimation.

It is well-known that $\mathsf{MST} \leq \mathsf{TSP} \leq 2 \cdot \mathsf{MST}$ holds for any graph/metric, since we can construct a TSP-tour by doubling all edges of a MST (and then shortcut the obtained walk into a tour). Since the MST cost of a graph/metric can be exactly computed by a one-pass $\tilde{O}(n)$ space algorithm (the greedy algorithm) in the streaming model, and can be approximated to within a factor of $(1+\varepsilon)$ by performing $\tilde{O}(n)$ queries in the query model [6], to obtain a factor $(2-\varepsilon)$ approximation for TSP, it suffices to establish either $\mathsf{TSP} \geq (1+\varepsilon) \cdot \mathsf{MST}$ or $\mathsf{TSP} \leq (2-\varepsilon) \cdot \mathsf{MST}$ holds. From the approach due to [5], the minimum weight of a perfect matching on the set of all odd-degree vertices in an MST can immediately gives us the answer. However, obtaining a good approximation to the minimum weight of such a perfect matching appears hard to do, both for semi-streaming algorithms and for a query algorithm that performs $o(n^2)$ queries, even if we are given an MST at the start. To get around this issue, we consider an alternative measure, called the *cover advantage*, that turns out to be more tractable in both models.

Cover Advantage. Let T be a MST of the input graph/metric. For an edge $f \in E(T)$ and an edge $e \notin E(T)$, we say that f is *covered* by e, iff f belongs to the unique tree-path in T connecting the endpoints of e. For a set E' of edges, we denote by $\mathsf{cov}(E', T)$ the set of all edges in $E(T)$ that are covered by at least one edge in E'. The *cover advantage* of E', denoted by $\mathsf{adv}(E')$, is defined to be the total weight of all edges in $\mathsf{cov}(E', T)$ minus the total weight of all edges in E'. Intuitively, if a single-edge set $\{e\}$ where $e = (u, v)$ has cover advantage c, then we can construct a tour by starting from some Euler-tour of T and replacing the segment corresponding to the tree path of T connecting u to v by the single edge e, and thereby "saving a cost of c" from $2 \cdot \mathsf{MST}$, the cost of the Euler-tour obtained by doubling MST edges. Generalizing this idea, we show that if there exists a set E' with cover advantage bounded away from 0 (at least $\varepsilon \cdot \mathsf{MST}$), then $\mathsf{TSP} \leq (2-\varepsilon/2) \cdot \mathsf{MST}$. Conversely, if there does not exist any set E' with cover advantage close to $\mathsf{MST}/2$ (say

at least $(1/2 - \varepsilon/2) \cdot \mathsf{MST}$), then $\mathsf{TSP} \geq (1 + \varepsilon) \cdot \mathsf{MST}$. In fact, we show that the same hold for a more restricted notion called *special cover advantage*, which is defined to be the maximum cover advantage of any subset E' of edges that have at least one endpoint being a special vertex in T (a vertex v is called a *special vertex* of T iff $\deg_T(v) \neq 2$). Therefore, to obtain a better-than-factor-2 approximation for TSP, it suffices to obtain a constant-factor approximation for the maximum cover advantage or the maximum special cover advantage.

Estimating maximum cover advantage in the streaming setting. We construct a one-pass streaming algorithm $O(1)$-estimating the maximum cover advantage, which leads to a two-pass algorithm in Theorem 6 where in the first pass we only compute an MST of the input graph. We store edges with substantial cover advantage with respect to the MST in a greedy manner. Since all edges appear in the stream, it can be shown that, if we end up not discovering a large cover advantage, then the real maximum cover advantage is indeed small (bounded away from $\mathsf{MST}/2$).

Estimating maximum cover advantage in the query model. The task of obtaining a constant-factor approximation to maximum cover advantage turns out to be distinctly more challenging in the query model, even if we are given explicit access to an MST of the metric. The design of sublinear query algorithms for estimating cover advantage is indeed our central algorithmic contribution. We design an $\tilde{O}(n^{1.5})$-query algorithm for estimating the maximum cover advantage when either an MST is explicitly given or we can assume that the MST is supported on weight-1 edges. Note that the latter case generalizes graphic TSP studied in [4].

The algorithms for these two cases share several similarities. To illustrate the ideas behind them, it might be instructive to consider the following two examples. In the first example, we are given an MST T on V that has at most $O(\sqrt{n})$ leaves. We can simply query the distances between all pairs $u, v \in V$ where u is a special vertex of T, and then use the obtained information to compute the maximum special cover advantage, which takes $\tilde{O}(n^{1.5})$ queries since there can be at most $O(\sqrt{n})$ special vertices (or in fact we can even query the distances between all pairs of special vertices in T and compute the minimum weight perfect matching on them). In the second example, we are given an MST T on V which is a star graph centered at a vertex $r \in V$, and all edges have weight 1. Note that, since all edge weights are integers, in this case the distances between every pair of vertices in $V \setminus \{r\}$ is either 1 or 2, and it is not hard to see that the maximum cover advantage is exactly the size of a maximum weight-1 matching on $V \setminus \{r\}$. Therefore, we can adapt the algorithm from [1] to obtain an $O(1)$-approximation of the maximum weight-1 matching size, using $\tilde{O}(n)$ queries. Note that, in this case we obtain an estimate of the maximum cover advantage without computing a set of edges that achieves it.

Taking a step back, we observe that, in the first example where the number of special vertices is small, the cover advantage can be computed in a local and exhaustive manner, while in the second example where the number of special vertices is large, the cover advantage has to be estimated in a global and "superficial" manner. Intuitively, our query algorithms interpolate between these two approaches in an organic manner.

We now provide more details of our query algorithms in the two special cases. We first consider the special case where we are given the structure of an MST.

When MST is given. We root the given MST T at an arbitrary vertex. For each vertex $v \in V$, we say that it is *light* iff the subtree of T rooted at v, denoted by T_v, contains at most $\sqrt{n}$ vertices, and we call T_v a *light subtree* of T. On the one hand, the cover advantages

that are local at some light subtree (achieved by edges with both endpoints in the same light subtree) can be efficiently estimated in an exhaustive manner. On the other hand, if we peel off all light subtrees from T, then the remaining subtree, that we denote by T', contains at most $\sqrt{n}$ leaves, and therefore the special cover advantage achieved by any set of edges with at least one endpoint being a special vertex of T' can also be computed in an exhaustive manner. The only type of cover advantages that is not yet computed are the one achieved by edges with endpoints in different light subtrees. We then observe that the light subtrees hanged at T are similar to the edges of a star graph hanged at its root, and eventually manage to adapt the algorithm from [1] in a delicate way to estimate the cover advantage by edges of this type in a global manner.

When MST consists of only weight-1 edges. This special case appears trickier since we do not know the structure of an MST at the start, and there may not even be a unique MST. To circumvent this, we need to utilize the following technical result of [4]: Let G_1 be the graph on V induced by all weight-1 edges in the given metric, then if G_1 contains a size-s matching consisting of only edges in 2-edge-connected components of G_1, then $\mathsf{TSP} \leq 2n - \Omega(s)$. This result allows us to construct a local procedure that explores some neighborhood of the unknown graph G_1 up to a certain size, such that in the end we either reconstruct a size-$\sqrt{n}$ subgraph of the (locally) unique MST, or certify that a set of $\Omega(\sqrt{n})$ vertices belong to some 2-edge-connected components of G_1, which will be later collected to estimate the maximum weight-1 matching size.

We then use this local procedure on a set of vertices randomly sampled from V. Let T be an MST. Intuitively, if the total size of light subtrees of T is non-negligible, then with high probability some of the sampled vertices will lie in light subtrees of T, and we can obtain an estimate of the local cover advantage within subtrees. If the total size of light subtrees is negligible, then T', the subtree obtained from T by peeling off all light subtrees, has roughly the same size as T, which means that T is close to the first instructive example mentioned before – a tree with only $O(\sqrt{n})$ special vertices. Then we can apply the local procedure to $\Omega(\sqrt{n})$ sampled vertices, to almost reconstruct the whole tree T, and the rest of the algorithm is similar to the algorithm in the first special case.

1.2.2 Overview of Lower Bound Techniques

As our algorithmic results illustrate that metric streams are more powerful than graph streams, it is perhaps not surprising that proving space lower bounds for metrics streams turns out to be a more challenging task that requires new tools. To illustrate this point, it might be instructive to consider the following simplified versions of metric and graph streams. Let G be a graph.

- Unweighted Graph Stream: a sequence that contains all edges of $E(G)$, and the same edge may appear more than once in the stream;
- Unweighted Metric Stream: a sequence that contains, for each pair u, v of $V(G)$, a symbol $f(u, v)$ indicating whether or not the edge (u, v) belongs to $E(G)$.

Note that in unweighted metric streams, the non-edge information between pairs of vertices is also explicitly given (as the edge information), as opposed to being given implicitly in the unweighted graph stream. This seemingly unimportant distinction, unexpectedly, makes proving lower bounds for several problems much harder in unweighted metric streams than unweighted graph streams.

For example, consider the problem of deciding whether the input graph is a clique. On the one hand, to prove a space lower bound for streaming algorithms in unweighted graph streams, we consider the following two-player one-way communication game: Alice is given a

graph G_A and Bob is given a graph G_B on a common vertex set V, and Alice and Bob want to decide if $G_A \cup G_B$ is the complete graph on V. It is easy to show that this communication game has back-and-forth communication complexity $\Omega(n^2)$. In fact, Alice's input graph G_A can be viewed as a vector $x^A \in \{0,1\}^{\binom{V}{2}}$ and Bob's input graph G_B can be viewed as a vector $x^B \in \{0,1\}^{\binom{V}{2}}$, where the coordinate $x^A_{(u,u')}$ indexed by the pair u, u' of vertices in V indicates whether or not the edge (u, u') appears in graph G_A, and similarly $x^B_{(u,u')}$ indicates whether or not the edge (u, u') appears graph G_B. It is then easy to see that the two players need to detect whether or not the bitwise-OR of vectors x^A and x^B is the all-one vector, which requires $\Omega(n^2)$-bits information exchange even in the back-and-forth communication model. On the other hand, in the corresponding two-player one-way communication game for unweighted metric streams, Alice and Bob are each given a set of edge/non-edge information, with the promise that the edge/non-edge information between each pair of vertices appears in at least one of the player's input. There is a one-bit protocol: Alice simply sends to Bob a signal indicating whether or not in her input there is non-edge information between any pair of vertices, and Bob outputs "Not a Clique" iff either he sees Alice's "non-edge" signal or he sees a non-edge information in his input.

The distinction that all non-edge information is explicitly given in the unweighted stream seems to fail all reductions from standard problems (like Disjointness and Index) to prove lower bounds. Therefore, in the lower bound proofs of Theorem 3 and Theorem 4, we identify new "primitive" graph-theoretic problems, prove communication lower bounds for them, and then reduce them to MST-estimation problems. Here we briefly provide some ideas for the proof of Theorem 4.

We consider the special type of metrics, in which the distance between every pair of vertices is either 1 or a large enough real number. Intuitively, the problem of estimating the MST cost is equivalent to the problem of estimating the number of connected components of the graph induced by all weight-1 edges, which is essentially a graph-theoretic problem in unweighted metric streams.

As a first step, we consider the following problem: given an unweighted metric stream, decide whether the underlying graph is a perfect matching or a perfect matching minus one edge. Unlike the previous clique-identification problem, we show that the corresponding two-player communication game for this problem has communication complexity $\Omega(n)$ in the back and forth communication model, even if the complete edge/non-edge information is split between Alice and Bob. The proof is by analyzing the information complexity of any protocol for the problem, We construct several similar input combinations for Alice and Bob, among which some cross-combination lead to different answers, and then lower bound the mutual information between the protocol transcript and the players' inputs.

However, this perfect matching vs perfect matching minus one edge problem is not sufficient for our purpose, since a perfect matching graph on n vertices has $n/2$ connected components, while a perfect matching minus one edge graph on n vertices has $n/2 - 1$ connected components, and the ratio between $n/2$ and $n/2 - 1$ are too small to provide a space lower bound for α-approximation of the number of connected components. To fix this issue, we next consider a generalization of this problem, called the *Clique or Independent Set* problem ($\mathsf{COI}_{a,b}$) parametrized by two integers a, b. In this problem, we are required to decide whether the input graph is the disjoint union of b cliques of size a each (Yes case) or it is a disjoint union of $(b-1)$ cliques of size a each and an independent set of size a (No case). Note that if $a = 2$ and $b = n/2$ then this problem is exactly the perfect matching vs perfect matching minus one edge problem. Now if we let $a \gg b$, then the ratio between numbers of connected components in Yes case and in No case is $(a + b - 1)/b = \Omega(a/b)$, which is

enough for giving a space lower bound for $o(a/b)$-approximation streaming algorithms for MST estimation. For the proof of the communication lower bound of problem $\mathsf{COI}_{a,b}$, we first consider the special case $\mathsf{COI}_{a,2}$ and show that the communication complexity is $\tilde{\Omega}(1)$ via a Hellinger distance analysis on transcript distributions on certain input combinations, and then use a direct sum type argument to show that the communication complexity of $\mathsf{COI}_{a,b}$ is $\tilde{\Omega}(b)$. Both steps use techniques similar to the ones used in the proof of communication lower bound for the perfect matching vs perfect matching minus one edge problem. Now for a given approximation ratio $\alpha > 1$, setting $a = \Theta(\sqrt{\alpha n})$ and $b = \Theta(\sqrt{n/\alpha})$ yields the desired communication lower bound, which then implies the space lower bounds for streaming algorithms.

1.3 Organization

Due to the limit of space, in the remainder of the paper we only present the proof sketches of one of our algorithmic results, which best illustrates the utilization of cover advantage. We first introduce the notion of cover advantage in Section 2. We then we sketch the proof of Theorem 6 in Section 3 and sketch the proof of Theorem 8 in Section 4. The proofs of all other theorems are deferred to the full version (in the appendix).

2 Cover Advantage

In this section, we introduce the notion of cover advantage, which is a key notion that captures the gap between the MST cost and the TSP cost. Our TSP cost estimation algorithms in both streaming and query settings will crucially utilize this notion. Due to the limit of space, the proofs of some lemmas presented in this section are deferred to the full version.

At a high-level, the TSP estimation algorithms in this paper are based on converting an MST T of the input graph/metric into a spanning Eulerian subgraph. A trivial approach is to simply double all edges in T obtaining a 2-approximation. A more clever approach due to Christofides [5] instead makes T Eulerian by adding a minimum weight perfect matching on odd-degree vertices in T, obtaining a 3/2-appproximation. However, computing a good approximation to the minimum weight perfect matching on a set of vertices appears hard to do either in the semi-streaming setting or with sublinear number of queries. We instead identify the more tractable notion, called the cover advantage, that can be efficiently implemented in the semi-streaming and query model.

We say that an edge f of tree T is *covered* by an edge e that may or may not belong to T, iff $f \in E(P_e^T)$; and we say that f is covered by a set E' of edges, iff it is covered by some edge of E'. We denote by $\mathsf{cov}(e)$ the set of all edges of T that are covered by e, and define $\mathsf{cov}(E') = \bigcup_{e \in E'} \mathsf{cov}(e)$.

Let T' be a subtree of T. For each edge $e \notin E(T)$, we define $\mathsf{cov}(e, T') = \mathsf{cov}(e) \cap E(T')$. Similarly, for a set E' of edges, we define $\mathsf{cov}(E', T') = \bigcup_{e \in E'} \mathsf{cov}(e, T')$. Clearly, $\mathsf{cov}(E', T') = \mathsf{cov}(E') \cap E(T')$.

We define the *cover advantage* of a set E' of edges on a subtree T' of T, denoted by $\mathsf{adv}(E', T')$, to be $\mathsf{adv}(E', T') = w(\mathsf{cov}(E', T')) - w(E')$. The *optimal cover advantage* of a subtree T', denoted by $\mathsf{adv}(T')$, is defined to be the maximum cover advantage of any set E' of edges that have at least one endpoint lying in $V(T')$ on T'. The *optimal special cover advantage* of a subtree T', denoted by $\mathsf{adv}^*(T')$, is defined to be the maximum cover advantage of any set E' of edges that have at least one endpoint being a special vertex of T' (a vertex v is a special vertex of T' iff $\deg_{T'}(v) \neq 2$). Clearly, by definition, $\mathsf{adv}(T') \geq \mathsf{adv}^*(T') \geq 0$.

The next two lemmas show that the optimal cover advantage and the optimal special cover advantage of any subtree can be computed using a small number of queries.

▶ **Lemma 9.** *There is an algorithm, that given a subtree T' of T, computes the optimal cover advantage of T' as well as a set E' of edges achieving the optimal cover advantage of T', by performing at most $O(n \cdot |V(T')|)$ queries.*

▶ **Lemma 10.** *There is an algorithm, that given a subtree T' of T, computes the optimal special cover advantage of T' as well as a set E' of edges achieving the optimal special cover advantage of T', by performing $O(n \cdot k_{T'})$ queries, where $k_{T'}$ is the number of special vertices in T'.*

The following lemma is crucial to our algorithms. It shows that the high cover advantage of edge-disjoint subtrees of an MST translates into a TSP tour whose cost is bounded away from 2 times the MST cost.

▶ **Lemma 11.** *Let T be an MST on a set V of vertices, and let $\mathcal{T}$ be a set of edge-disjoint subtrees of T. Then* $\mathsf{TSP} \leq 2 \cdot \mathsf{MST} - \frac{1}{2} \cdot \sum_{T' \in \mathcal{T}} \mathsf{adv}(T')$.

Proof. We introduce some definitions before providing the proof.

Let E' be a set of edges that do not belong to $E(T)$. We define the multi-graph $H_{T,E'}$ as follows. Its vertex set is $V(H_{T,E'}) = V$. Its edge set is the union of (i) the set E'; and (ii) the set $E_{[T,E']}$ that contains, for each edge $f \in E(T)$, 2 copies of f iff f is covered by an even number of edges in E', 1 copy of f iff f is covered by an odd number of edges in E'. Equivalently, graph $H_{T,E'}$ can be obtained from the following iterative algorithm. Throughout, we maintain a graph $\hat{H}$ on the vertex set V, that initially contains two copies of each edge of $E(T)$. We will maintain the invariant that, over the course of the algorithm, for each edge f of $E(T)$, graph $\hat{H}$ contains either one copy or two copies of f. We then process edges of E' one-by-one (in arbitrary order) as follows. Consider now an edge $e \in E'$ and the tree-path P_e^T. We add one copy of edge e to $\hat{H}$. Then for each edge $f \in E(P_e^T)$, if currently the graph $\hat{H}$ contains 2 copies of f, then we remove one copy of it from $\hat{H}$; if currently the graph $\hat{H}$ contains 1 copy of f, then we add one copy of it into $\hat{H}$. Clearly after each iteration of processing some edge of E', the invariant still holds. It is also easy to see that the resulting graph we obtain after processing all edges of E' is exactly the graph $H_{T,E'}$ defined above.

We prove the following observation.

▶ **Observation 12.** *For any set E', graph $H_{T,E'}$ is Eulerian.*

Proof. Consider the algorithm that produces the graph $H_{T,E'}$. Initially, graph $\hat{H}$ contains 2 copies of each edge of T, and is therefore Eulerian. It is easy to see that, in the iteration of processing the edge $e \in E'$, we only modify the degrees of vertices in the cycle $e \cup P_e^T$. Specifically, for each vertex in the cycle $e \cup P_e^T$, either its degree is increased by 2 (if a copy is added to both of its incident edges in the cycle), or its degree is decreased by 2 (if a copy is removed from both of its incident edges in the cycle), or its degree remains unchanged (if a copy is removed from one of its incident edges, and a copy is added to the other incident edge). Therefore, the graph $\hat{H}$ remains Eulerian after this iteration, and it follows that the resulting graph $H_{T,E'}$ is Eulerian. ◀

We now provide the proof of Lemma 11. Denote $\mathcal{T} = \{T_1, \ldots, T_k\}$. For each index $1 \leq i \leq k$, let E_i^* be the set of edges that achieves the maximum cover advantage on T_i. Denote $E^* = \bigcup_{1 \leq i \leq k} E_i^*$, and then we let E' be the random subset of E^* that includes

each edge of E^* independently with probability $1/2$. We will show that the expected total weight of all edges in $E(H_{T,E'})$ is at most $2 \cdot \mathsf{MST}(G) - \frac{1}{2} \cdot \sum_{1 \le i \le k} \mathsf{adv}(T_i)$. Note that this implies that there exists a subset E^{**} of E^*, such that the weight of graph $H_{T,E^{**}}$ is at most $2 \cdot \mathsf{MST}(G) - \frac{1}{2} \cdot \sum_{1 \le i \le k} \mathsf{adv}(T_i)$. Combined with Observation 12 and the fact that TSP is upper bounded by the total cost of any connected Eulerian graph, this implies $\mathsf{TSP} \le 2 \cdot \mathsf{MST}(G) - \frac{1}{2} \cdot \sum_{1 \le i \le k} \mathsf{adv}(T_i)$, completing the proof of Lemma 11.

We now show that $\mathbb{E}[w(H_{T,E'})] \le 2 \cdot \mathsf{MST}(G) - \frac{1}{2} \cdot \sum_{1 \le i \le k} \mathsf{adv}(T_i)$. From the definition of graph $H_{T,E'}$, $E(H_{T,E'}) = E' \cup E_{[T,E']}$. On one hand, from the construction of set E', $\mathbb{E}[w(E')] = w(E^*)/2$. On the other hand, for each edge $f \in \mathsf{cov}(E^*)$, with probability $1/2$ graph $H_{T,E'}$ contains 1 copy of it, and with probability $1/2$ graph $H_{T,E'}$ contains 2 copies of it. Therefore, $\mathbb{E}[w(E_{[T,E']})] = 2 \cdot w(E(T)) - w(\mathsf{cov}(E^*))/2$. Note that subtrees $\{T_i\}_{1 \le i \le k}$ are edge-disjoint, so the edge sets $\{\mathsf{cov}(E^*, T_i)\}_{1 \le i \le k}$ are mutually disjoint. Altogether,

$$\begin{aligned}
\mathbb{E}[w(H_{T,E'})] &= 2 \cdot \mathsf{MST} - \frac{w(\mathsf{cov}(E^*)) - w(E^*)}{2} \\
&\le 2 \cdot \mathsf{MST} - \frac{1}{2} \cdot \sum_{1 \le i \le k} \Big(w(\mathsf{cov}(E^*, T_i)) - w(E_i^*) \Big) \\
&\le 2 \cdot \mathsf{MST} - \frac{1}{2} \cdot \sum_{1 \le i \le k} \Big(w(\mathsf{cov}(E_i^*, T_i)) - w(E_i^*) \Big) \\
&= 2 \cdot \mathsf{MST} - \frac{1}{2} \cdot \sum_{1 \le i \le k} \mathsf{adv}(T_i).
\end{aligned}$$

◀

Complementing Lemma 11, the next lemma shows that, if the special cover advantage is low, then the TSP cost is close to 2 times the MST cost.

▶ **Lemma 13.** *Let T' be any subtree of an MST T. Then $\mathsf{TSP} \ge 2 \cdot w(T') - 2 \cdot \mathsf{adv}^*(T')$.*

Proof. Let π^* be the optimal TSP-tour that visits all vertices of V, so $\mathsf{TSP} = w(\pi^*)$. Let π be the tour obtained from π^* by deleting all vertices of $T \setminus T'$, so π is a tour that visits all vertices of $V(T')$, and, from triangle inequality, $w(\pi^*) \ge w(\pi)$. We now show that $E(\pi)$ can be partitioned into two subsets $E(\pi) = E_0 \cup E_1$, such that $E(T') \subseteq \mathsf{cov}(E_0)$ and $E(T') \subseteq \mathsf{cov}(E_1)$.

Let V' be the set of all odd-degree vertices in T', so $|V'|$ is even. Denote $V' = \{v_1, v_2, \ldots, v_{2k}\}$, where the vertices are index according to the order in which they appear in π. For each $1 \le i \le 2k$, we define edge $e_i = (v_i, v_{i+1})$ and define E_π^i to be the set of all edges traversed by π between vertices v_i and v_{i+1}. Clearly, $\mathsf{cov}(e_i) \subseteq \mathsf{cov}(E_\pi^i)$. We define $E_0 = \bigcup_{0 \le i \le k-1} E_\pi^{2i}$ and $E_1 = \bigcup_{0 \le i \le k-1} E_\pi^{2i+1}$.

Consider now the tour π' induced by edges of $e_1, \ldots, e_{2k}$. Clearly, π' is a tour that visits all vertices of V'. We define sets $F_0 = \{e_{2i} \mid 0 \le i \le k-1\}$ and $F_1 = \{e_{2i+1} \mid 0 \le i \le k-1\}$, so $E(\pi') = F_0 \cup F_1$. We now show that $E(T') \subseteq \mathsf{cov}(F_0)$ and $E(T') \subseteq \mathsf{cov}(F_1)$. Note that this implies that $E(T') \subseteq \mathsf{cov}(E_0)$ and $E(T') \subseteq \mathsf{cov}(E_1)$, since

$$\mathsf{cov}(F_0) = \Big(\bigcup_{0 \le i \le k-1} \mathsf{cov}(e_{2i}) \Big) \subseteq \Big(\bigcup_{0 \le i \le k-1} \mathsf{cov}(E_\pi^{2i}) \Big) \subseteq \mathsf{cov}\Big(\bigcup_{0 \le i \le k-1} E_\pi^{2i} \Big) = \mathsf{cov}(E_0),$$

and similarly $\mathsf{cov}(F_1) \subseteq \mathsf{cov}(E_1)$.

In fact, note that F_0 is a perfect matching on V'. Since V' is the set of odd-degree vertices of T', the graph on $V(T')$ induced by edges of $E(T') \cup F_0$ is Eulerian. Therefore, every edge of T' appears in at least two sets of $\{\mathsf{cov}(e') \mid e' \in E(T') \cup M_o\}$. Note that for each $e' \in E(T')$, $\mathsf{cov}(e') = \{e'\}$. Therefore, every edge $e \in E(T')$ appears in at least one set of $\{\mathsf{cov}(e') \mid e' \in F_0\}$, i.e., $E(T') \subseteq \mathsf{cov}(F_0)$. Similarly, we get that $E(T') \subseteq \mathsf{cov}(F_1)$.

From triangle inequality, $w(F_0) + w(F_1) \leq w(E_0) + w(E_1) = w(\pi)$. Note that edges of F_0 and F_1 have both endpoint in V', and moreover, from the definition of V', all vertices of V' are special vertices of T'. Since $E(T') \subseteq \mathsf{cov}(F_0)$, from the definition of $\mathsf{adv}^*(T')$, we get that $\mathsf{adv}^*(T') \geq w(\mathsf{cov}(F_0, T')) - w(F_0) = w(T') - w(F_0)$, and similarly $\mathsf{adv}^*(T') \geq w(T') - w(F_1)$. Therefore, $\mathsf{TSP} = w(\pi^*) \geq w(\pi) = w(E_0) + w(E_1) \geq w(F_0) + w(F_1) \geq 2 \cdot (w(T') - \mathsf{adv}^*(T'))$. ◀

The last two lemmas show that the total cover advantage and the total special cover advantage of a set of edge-disjoint subtrees of an MST can be efficiently and accurately estimated.

▶ **Lemma 14.** *There is an algorithm, that, given a constant $0 < \varepsilon < 1$ and a set $\mathcal{T}$ of edge-disjoint subtrees of T, with high probability, either correctly reports that $\sum_{T' \in \mathcal{T}} \mathsf{adv}(T') \geq \varepsilon \cdot \mathsf{MST}$, or correctly reports that $\sum_{T' \in \mathcal{T}} \mathsf{adv}(T') \leq 2\varepsilon \cdot \mathsf{MST}$, by performing $\tilde{O}((n/\varepsilon^2) \cdot \max\{|V(T')|\}_{T' \in \mathcal{T}})$ queries.*

▶ **Lemma 15.** *There is an algorithm, that, given a constant $0 < \varepsilon < 1$ and a set $\mathcal{T}$ of edge-disjoint subtrees of T, with high probability, either correctly reports that $\sum_{T' \in \mathcal{T}} \mathsf{adv}^*(T') \geq \varepsilon \cdot \mathsf{MST}$, or correctly reports that $\sum_{T' \in \mathcal{T}} \mathsf{adv}^*(T') \leq 2\varepsilon \cdot \mathsf{MST}$, by performing $\tilde{O}((n/\varepsilon^2) \cdot \max\{k_{T'} \mid T' \in \mathcal{T}\})$ queries, where $k_{T'}$ is the number of special vertices in T'.*

3 A Two-Pass Algorithm for TSP Estimation in Graph Streams

In this section, we present a deterministic 2-pass 1.96-approximation algorithm for TSP estimation in graph streams, which uses $\tilde{O}(n)$ space, thus proving Theorem 6. Our algorithm will utilize the notion of cover advantage, introduced in Section 2. This result is in a sharp contrast to Theorem 5 which showed that any single-pass algorithm requires $\Omega(n^2)$ space to obtain a better than 2-approximation.

Algorithm. Let $\alpha, \beta \in (0, 1)$ be two constants whose values will be set later. In the first pass, we simply compute a minimum spanning tree T and its cost $\mathsf{MST} = \sum_{e \in E(T)} w(e)$. Throughout the second pass, we maintain a subset E_{temp} of edges, that is initialized to be $\emptyset$, and will only grow over the course of the algorithm. Upon the arrival of each edge e, we compare $w(e)$ with $w(\mathsf{cov}(e) \setminus \mathsf{cov}(E_{\mathsf{temp}})) = \sum_{f \in \mathsf{cov}(e) \setminus \mathsf{cov}(E_{\mathsf{temp}})} w(f)$. We add the edge e to set E_{temp} iff $w(e) \leq \alpha \cdot w(\mathsf{cov}(e) \setminus \mathsf{cov}(E_{\mathsf{temp}}))$. Let E^* be the set E_{temp} at the end of the algorithm. We then compute $w(\mathsf{cov}(E^*)) = \sum_{e \in \mathsf{cov}(E^*)} w(e)$. If $w(\mathsf{cov}(E^*)) \geq \beta \cdot \mathsf{MST}$, then we output $(2 - \frac{(1-\alpha)\cdot\beta}{2}) \cdot \mathsf{MST}$ as an estimate of TSP; otherwise we output $2 \cdot \mathsf{MST}$. We use the parameters $\alpha = 0.715$ and $\beta = 0.285$, so $(2 - \frac{(1-\alpha)\cdot\beta}{2}) \approx \frac{2}{2\alpha(1-\beta)} \approx 1.96$.

Proof of Correctness. The correctness of the algorithm is guaranteed by the following two claims.

▷ Claim 16. If $w(\mathsf{cov}(E^*)) \geq \beta \cdot \mathsf{MST}$, then $\mathsf{TSP} \leq (2 - \frac{(1-\alpha)\cdot\beta}{2}) \cdot \mathsf{MST}$.

Proof. Let E' be the random subset of E^* that includes each edge of E^* independently with probability $1/2$. We will show that the expected total weight of all edges in graph $E(H_{T,E'})$ is at most $(2 - \frac{(1-\alpha)\cdot\beta}{2}) \cdot \mathsf{MST}$, namely $\mathbb{E}[w(E(H_{T,E'}))] \leq (2 - \frac{(1-\alpha)\cdot\beta}{2}) \cdot \mathsf{MST}$. Note that this implies that there exists a subset E^{**} of E^*, such that $w(E(H_{T,E^{**}})) \leq (2 - \frac{(1-\alpha)\cdot\beta}{2}) \cdot \mathsf{MST}$. Combined with Observation 12, this implies that there is an Eulerian tour of the same cost (using only edges of graph $H_{T,E^{**}}$). Therefore, there is a TSP-tour of at most the same cost, completing the proof of Claim 16.

We now show that $\mathbb{E}[w(E(H_{T,E'}))] \leq (2 - \frac{(1-\alpha)\cdot\beta}{2}) \cdot \mathsf{MST}$. From the definition of graph $H_{T,E'}$, $E(H_{T,E'}) = E' \cup E_{[T,E']}$. On one hand, from the definition of the random subset E', $\mathbb{E}[w(E')] = w(E^*)/2$. On the other hand, for each edge $f \in \mathsf{cov}(E^*)$, with probability $1/2$ graph $H_{T,E'}$ contains 1 copy of it, and with probability $1/2$ graph $H_{T,E'}$ contains 2 copies of it. Therefore, $\mathbb{E}[w(E_{[T,E']})] = 2 \cdot w(E(T)) - w(\mathsf{cov}(E^*))/2 = 2 \cdot \mathsf{MST} - w(\mathsf{cov}(E^*))/2$. Altogether, $\mathbb{E}[w(E(H_{T,E'}))] = 2 \cdot \mathsf{MST} - (w(\mathsf{cov}(E^*)) - w(E^*))/2$. The following observation follows immediately from the algorithm.

▶ **Observation 17.** $w(E^*) \leq \alpha \cdot w(\mathsf{cov}(E^*))$.

From Observation 17,

$$\mathbb{E}[w(E(H_{T,E'}))] \leq 2 \cdot \mathsf{MST} - \frac{(1-\alpha) \cdot w(\mathsf{cov}(E^*))}{2} \leq \left(2 - \frac{(1-\alpha)\cdot\beta}{2}\right) \cdot \mathsf{MST}.$$

This concludes the proof of Claim 16. ◁

We next show that, if we do not find a sufficiently large cover, i.e., the value of $w(\mathsf{cov}(E^*))$ is not sufficiently large compared with MST, then TSP must be bounded away from MST.

▷ Claim 18. If $w(\mathsf{cov}(E^*)) < \beta \cdot \mathsf{MST}$, then $\mathsf{TSP} \geq 2\alpha(1-\beta) \cdot \mathsf{MST}$.

Proof. Recall that set $V_1(T)$ contains all vertices with odd degree in T. Let M be a minimum-cost perfect matching on $V_1(T)$, so $\mathsf{TSP} \geq 2 \cdot w(M)$. We use the following observations. The proof of Observation 19 is straightforward and is deferred to the full version.

▶ **Observation 19.** $\mathsf{cov}(M) = E(T)$.

▶ **Observation 20.** *For each* $e \in M$, $w(\mathsf{cov}(e) \setminus \mathsf{cov}(E^*)) < w(e)/\alpha$.

Proof. We denote by Q_e the shortest-path in G connecting the endpoints of e (where G is the graph underlying the stream). Since Q_e is a subgraph of G, all edges of Q_e will appear in the graph stream of G. Note that $w(Q_e) = w(e)$ and $\mathsf{cov}(e) \subseteq \mathsf{cov}(E(Q_e))$.

We will show that, for every edge $e' \in E(Q_e)$, $w(\mathsf{cov}(e') \setminus \mathsf{cov}(E^*)) < w(e')/\alpha$. Note that the observation follows from this assertion, as

$$w(\mathsf{cov}(e) \setminus \mathsf{cov}(E^*)) \leq w(\mathsf{cov}(E(Q_e)) \setminus \mathsf{cov}(E^*)) \leq \sum_{e' \in E(Q_e)} w(\mathsf{cov}(e') \setminus \mathsf{cov}(E^*)) < \frac{w(Q_e)}{\alpha} = \frac{w(e)}{\alpha}.$$

Consider now any edge $e' \in E(Q_e)$, and assume for contradiction that $w(\mathsf{cov}(e') \setminus \mathsf{cov}(E^*)) \geq w(e')/\alpha$. Note that set E_{temp} only grows over the course of the algorithm that computes set E^*, and so does the set $\mathsf{cov}(E_{\mathsf{temp}})$. Therefore, when e' arrives in the stream, $w(\mathsf{cov}(e') \setminus \mathsf{cov}(E_{\mathsf{temp}})) \geq w(e')/\alpha$ must hold. Then according to the algorithm, the edge e' should be added to E_{temp} right away, which means that edge e' will eventually belong to E^*, leading to $\mathsf{cov}(e') \subseteq \mathsf{cov}(E^*)$ and $w(\mathsf{cov}(e') \setminus \mathsf{cov}(E^*)) = 0$, a contradiction to the assumption that $w(\mathsf{cov}(e') \setminus \mathsf{cov}(E^*)) \geq w(e')/\alpha$. ◀

From Observation 19 and Observation 20, we get that

$$(1-\beta) \cdot \mathsf{MST} \leq \mathsf{MST} - w(\mathsf{cov}(E^*)) = w(\mathsf{cov}(M) \setminus \mathsf{cov}(E^*)) \leq \sum_{e \in M} w(\mathsf{cov}(e) \setminus \mathsf{cov}(E^*)) < w(M)/\alpha.$$

Therefore, $w(M) \geq \alpha(1-\beta) \cdot \mathsf{MST}$. Since $\mathsf{TSP} \geq 2 \cdot w(M)$, we conclude that $\mathsf{TSP} \geq 2\alpha(1-\beta) \cdot \mathsf{MST}$. ◁

4 An $(2-\varepsilon_0)$-Approximation Query Algorithm with a given MST

In this section, we provide a proof sketch of Theorem 8, by showing an algorithm that, given access of a minimum spanning tree of a metric, obtains an $(2-\varepsilon_0)$-approximation of TSP (for $\varepsilon_0 = 2^{-100}$) by performing $\tilde{O}(n^{1.5})$ queries. Note that it suffices for the algorithm to correctly claim either $\mathsf{TSP} \geq (1+\varepsilon_0)\cdot\mathsf{MST}$ or $\mathsf{TSP} \leq (2-\varepsilon_0)\cdot\mathsf{MST}$.

Let T be the input MST and let it be rooted at an arbitrary vertex. We first divide T into a top part and a bottom part as follows. We say that a vertex v is *maximally light*, iff T_v (the subtree of T rooted at v) contains at most $\sqrt{n}$ vertices, but its parent node does not. Let T' be the tree obtained from T by deleting from it, for each maximally light vertex v, all edges and vertices of T_v, and we call T' the top part of T, and call $T \setminus T'$ the bottom part of T. It is easy to show that T' has at most $O(\sqrt{n})$ leaves and therefore at most $O(\sqrt{n})$ special vertices. Moreover, we can efficiently partition T' into a set $\mathcal{P}$ of $O(\sqrt{n})$ vertex-disjoint paths, such that, for each path $P \in \mathcal{P}$, either P contains a single vertex of T, or the total number of vertices in T that has an ancestor in P is at most $\sqrt{n}$. For each path $P \in \mathcal{P}$, we call the subtree of T induced by all vertices of P and all their descendants in T a *segment*. Let $\mathcal{S}$ be the set of all segments.

The algorithm consists of four steps, that are summarized as follows.

1. We compute the special cover advantage of T' using Lemma 10. If $\mathsf{adv}^*(T') \geq 10\varepsilon_0\cdot\mathsf{MST}$, then we claim $\mathsf{TSP} \leq (2-\varepsilon_0)\cdot\mathsf{MST}$.
2. We estimate the total cover advantage of all segments using Lemma 14. If the algorithm reports that $\sum_{S\in\mathcal{S}} \mathsf{adv}(S) \geq 10\varepsilon_0\cdot\mathsf{MST}$, then we claim $\mathsf{TSP} \leq (2-\varepsilon_0)\cdot\mathsf{MST}$.
3. (informal) We estimate the cover advantage involving $T\setminus T'$, with the help of the algorithm in [1]. If the estimation is at least $10\varepsilon_0\cdot\mathsf{MST}$, then we claim $\mathsf{TSP} \leq (2-\varepsilon_0)\cdot\mathsf{MST}$.
4. If we did not terminate and claim $\mathsf{TSP} \leq (2-\varepsilon_0)\cdot\mathsf{MST}$ in any of the previous step, then we claim $\mathsf{TSP} \geq (1+\varepsilon_0)\cdot\mathsf{MST}$.

We first count the total number of queries performed by the algorithm. First, the algorithm from Lemma 10 performs $\tilde{O}(n^{1.5})$ queries, since the number of special vertices in T' is at most $O(\sqrt{n})$. Second, in Step 2, the algorithm from Lemma 14 performs in total $\tilde{O}(n^{1.5})$ queries, since each segment contains $O(\sqrt{n})$ vertices. Lastly, in Step 3 the algorithm performs $\tilde{O}(n^{1.5})$ queries. Altogether, the algorithm performs in total $\tilde{O}(n^{1.5})$ queries.

We now show that the algorithm indeed returns a $(2-\varepsilon_0)$-approximation of TSP. That is, the claim made by the algorithm is correct.

- If in Step 1, the algorithm from Lemma 10 reported $\mathsf{adv}^*(T') \geq 10\varepsilon_0\cdot\mathsf{MST}$. Then from Lemma 11, $\mathsf{TSP} \leq 2\cdot\mathsf{MST} - (10\varepsilon_0)\cdot\mathsf{MST}/2 = (2-5\varepsilon_0)\cdot\mathsf{MST}$ and the claim is correct.
- If in Step 2, the algorithm from Lemma 14 reported $\sum_{S\in\mathcal{S}} \mathsf{adv}(S) \geq 10\varepsilon_0\cdot\mathsf{MST}$. From Lemma 11, $\mathsf{TSP} \leq 2\cdot\mathsf{MST} - (10\varepsilon_0)\cdot\mathsf{MST}/2 = (2-5\varepsilon_0)\cdot\mathsf{MST}$ and the claim is correct.
- If in Step 2, the estimate of the cover advavtage in $T\setminus T'$ is at least $10\varepsilon_0\cdot\mathsf{MST}$, from Lemma 11, we can derive that $\mathsf{TSP} \leq (2-5\varepsilon_0)\cdot\mathsf{MST}$ and the claim is correct.

We assume from now on the algorithm did not terminate and claim $\mathsf{TSP} \leq (2-\varepsilon_0)\cdot\mathsf{MST}$ in the first three steps. We will show that in this case, $\mathsf{TSP} \geq (1+\varepsilon_0)\cdot\mathsf{MST}$.

Let π be an optimal TSP-tour, so $\mathsf{TSP} = w(\pi)$. Intuitively, if the tour π "continuously travels within the top part of T", then the report in Step 1 guarantees that its total cost must be bounded away from MST; if the tour π "continuously travels within the same segments", then the report in Step 2 guarantees that its total cost must be bounded away from MST. Therefore, we only need to consider the case where the tour "constantly jumping between the top and the bottom parts of T and across different segments".

Note that, since the cover advantage that we discovered in Steps 1-2 are low, for every edge $e = (u, v)$, the weight $w(e)$ should be rougly equal to the total weight of the unique u-v path in T. Therefore, if we replace each edge of π with the corresponding path in T, then the total weight of the resulting edge set should be close to TSP. But since the tour jumps between the top and the bottom parts of T and across different segments, the "segment-connecting" edges in T must be covered many times by π, and this can be used to show that $w(\pi)$ is bounded away from MST.

Specifically, we let E'_π the obtained edge set after replacing each edge of $E(\pi)$ with edges in the corresponding path in T, so E'_π may contain many copies of the same edge. For each edge e that has more than 2 copies contained in E'_π, if E'_π contains an odd number of copies of e, then we delete all but one copies from E'_π; if E'_π contains an even number of copies of e, then we delete all but two copies from E'_π. Denote by E' the resulting set of edges, so each edge has at most 2 copies contained in E'. It is easy to verify that the graph induced by edges of E' (with multiplicity) is connected and Eulerian.

Let E'' be the subset of E' that contains all bridges in the graph induced by edges of E' (ignoring multiplicities), and it is easy to verify that each edge has two copies contained in E'. Then from the report of Steps 1-2 of the algorithm, we can show that $w(E') - w(E(\pi)) \geq O(\varepsilon_0) \cdot \mathsf{MST}$ and $w(E') \geq \mathsf{MST} + w(E'')$. Therefore, if $w(E'') \geq \Omega(\varepsilon_0) \cdot \mathsf{MST}$ then we are done. If $w(E'') \leq O(\varepsilon_0) \cdot \mathsf{MST}$, then we can show that the total cost of all edges with at least one endpoints in $T \setminus T'$ must be large, and from the report of Step 3 of the algorithm, we can conclude that $w(E(\pi))$ is bounded away from MST.

5 Future Directions

In this work, we studied the problems of MST and TSP cost estimation in the streaming and query settings. For TSP cost estimation, we introduced and utilized a novel notion called *cover advantage* that may prove useful for solving this problem in other computational models also. In the streaming setting, an interesting open problem is to obtain a one-pass $o(n^2)$-space $(2-\varepsilon)$-approximate estimation of TSP cost in the metric stream. In the query model, we believe a major open problem is to obtain an $o(n^2)$-query $(2-\varepsilon)$-approximate estimation of TSP-cost in general metrics.

References

1. Soheil Behnezhad. Time-optimal sublinear algorithms for matching and vertex cover. *arXiv preprint*, 2021. `arXiv:2106.02942`.
2. Soheil Behnezhad, Mohammad Roghani, Aviad Rubinstein, and Amin Saberi. Sublinear algorithms for tsp via path covers. *arXiv preprint*, 2023. `arXiv:2301.05350`.
3. Bernard Chazelle, Ronitt Rubinfeld, and Luca Trevisan. Approximating the minimum spanning tree weight in sublinear time. *SIAM J. Comput.*, 34(6):1370–1379, 2005.
4. Yu Chen, Sampath Kannan, and Sanjeev Khanna. Sublinear algorithms and lower bounds for metric tsp cost estimation. *arXiv preprint*, 2020. `arXiv:2006.05490`.
5. Nicos Christofides. Worst-case analysis of a new heuristic for the travelling salesman problem. Technical report, Carnegie-Mellon Univ Pittsburgh Pa Management Sciences Research Group, 1976.
6. Artur Czumaj and Christian Sohler. Estimating the weight of metric minimum spanning trees in sublinear time. *SIAM Journal on Computing*, 39(3):904–922, 2009.

7 Krzysztof Onak, Dana Ron, Michal Rosen, and Ronitt Rubinfeld. A near-optimal sublinear-time algorithm for approximating the minimum vertex cover size. In *Proceedings of the twenty-third annual ACM-SIAM symposium on Discrete Algorithms*, pages 1123–1131. Society for Industrial and Applied Mathematics, 2012.

8 Yuichi Yoshida, Masaki Yamamoto, and Hiro Ito. Improved constant-time approximation algorithms for maximum matchings and other optimization problems. *SIAM Journal on Computing*, 41(4):1074–1093, 2012.