Sublinear Algorithms and Lower Bounds for Metric TSP Cost Estimation

Yu Chen
Department of Computer and Information Science, University of Pennsylvania
chenyu2@cis.upenn.edu

Sampath Kannan
Department of Computer and Information Science, University of Pennsylvania
kannan@cis.upenn.edu

Sanjeev Khanna
Department of Computer and Information Science, University of Pennsylvania
sanjeev@cis.upenn.edu

Abstract

We consider the problem of designing sublinear time algorithms for estimating the cost of minimum metric traveling salesman (TSP) tour. Specifically, given access to a $n \times n$ distance matrix D that specifies pairwise distances between n points, the goal is to estimate the TSP cost by performing only sublinear (in the size of D) queries. For the closely related problem of estimating the weight of a metric minimum spanning tree (MST), it is known that for any $\varepsilon > 0$, there exists an $\tilde{O}(n/\varepsilon^{O(1)})$ time algorithm that returns a $(1+\varepsilon)$-approximate estimate of the MST cost. This result immediately implies an $\tilde{O}(n/\varepsilon^{O(1)})$ time algorithm to estimate the TSP cost to within a $(2+\varepsilon)$ factor for any $\varepsilon > 0$. However, no $o(n^2)$ time algorithms are known to approximate metric TSP to a factor that is strictly better than 2. On the other hand, there were also no known barriers that rule out existence of $(1+\varepsilon)$-approximate estimation algorithms for metric TSP with $\tilde{O}(n)$ time for any fixed $\varepsilon > 0$. In this paper, we make progress on both algorithms and lower bounds for estimating metric TSP cost.

On the algorithmic side, we first consider the graphic TSP problem where the metric D corresponds to shortest path distances in a connected unweighted undirected graph. We show that there exists an $\tilde{O}(n)$ time algorithm that estimates the cost of graphic TSP to within a factor of $(2-\varepsilon_0)$ for some $\varepsilon_0 > 0$. This is the first sublinear cost estimation algorithm for graphic TSP that achieves an approximation factor less than 2. We also consider another well-studied special case of metric TSP, namely, $(1,2)$-TSP where all distances are either 1 or 2, and give an $\tilde{O}(n^{1.5})$ time algorithm to estimate optimal cost to within a factor of 1.625. Our estimation algorithms for graphic TSP as well as for $(1,2)$-TSP naturally lend themselves to $\tilde{O}(n)$ space streaming algorithms that give an 11/6-approximation for graphic TSP and a 1.625-approximation for $(1,2)$-TSP. These results motivate the natural question if analogously to metric MST, for any $\varepsilon > 0$, $(1+\varepsilon)$-approximate estimates can be obtained for graphic TSP and $(1,2)$-TSP using $\tilde{O}(n)$ queries. We answer this question in the negative – there exists an $\varepsilon_0 > 0$, such that any algorithm that estimates the cost of graphic TSP ($(1,2)$-TSP) to within a $(1+\varepsilon_0)$-factor, necessarily requires $\Omega(n^2)$ queries. This lower bound result highlights a sharp separation between the metric MST and metric TSP problems.

Similarly to many classical approximation algorithms for TSP, our sublinear time estimation algorithms utilize subroutines for estimating the size of a maximum matching in the underlying graph. We show that this is not merely an artifact of our approach, and that for any $\varepsilon > 0$, any algorithm that estimates the cost of graphic TSP or $(1,2)$-TSP to within a $(1+\varepsilon)$-factor, can also be used to estimate the size of a maximum matching in a bipartite graph to within an εn additive error. This connection allows us to translate known lower bounds for matching size estimation in various models to similar lower bounds for metric TSP cost estimation.

2012 ACM Subject Classification Theory of computation → Streaming, sublinear and near linear time algorithms

Keywords and phrases sublinear algorithms, TSP, streaming algorithms, query complexity.

Digital Object Identifier 10.4230/LIPIcs.ICALP.2020.30

1 Introduction

In the metric traveling salesman problem (TSP), we are given n points in an arbitrary metric space with an $n \times n$ matrix D specifying pairwise distances between them. The goal is to find a simple cycle (a TSP tour) of minimum cost that visits all n points. An equivalent view of the problem is that we are given a complete weighted undirected graph $G(V, E)$ where the weights satisfy triangle inequality, and the goal is to find a Hamiltonian cycle of minimum weight. The study of metric TSP is intimately connected to many algorithmic developments, and the poly-time approximability of metric TSP and its many natural variants are a subject of extensive ongoing research (see, for instance, [3, 13, 18, 20, 22, 29–33] and references within for some relatively recent developments). In this paper, we consider the following question: can one design sublinear algorithms that can be used to obtain good estimates of the cost of an optimal TSP tour? Since the complete description of the input metric is of size $\Theta(n^2)$, the phrase sublinear here refers to algorithms that run in $o(n^2)$ time.

A standard approach to estimating the metric TSP cost is to compute the cost of a minimum spanning tree, and output two times this cost as the estimate of the TSP cost (since any spanning tree can be used to create a spanning simple cycle by at most doubling the cost). The problem of approximating the cost of the minimum spanning tree in sublinear time was first studied in the graph adjacency-list model by Chazelle, Rubinfeld, and Trevisan [8]. The authors gave an $\tilde{O}(dW/\varepsilon^2)$-time algorithm to estimate the MST cost to within a $(1+\varepsilon)$-factor in a graph where average degree is d, and all edge costs are integers in $[1..W]$. 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. Subsequently, in an identical setting as ours, Czumaj and Sohler [11] showed that for any $\varepsilon > 0$, there exists an $\tilde{O}(n/\varepsilon^{O(1)})$ time algorithm that returns a $(1+\varepsilon)$-approximate estimate of the MST cost when the input is an n-point metric. This result immediately implies an $\tilde{O}(n/\varepsilon^{O(1)})$ time algorithm to estimate the TSP cost to within a $(2+\varepsilon)$ factor for any $\varepsilon > 0$. However, no $o(n^2)$ query algorithms are known to approximate metric TSP to a factor that is strictly better than 2. On the other hand, there are also no known barriers that rule out existence of $(1+\varepsilon)$-approximate estimation algorithms for metric TSP with $\tilde{O}(n)$ queries for any fixed $\varepsilon > 0$. In this paper, we make progress on both algorithms and lower bounds for estimating metric TSP cost.

On the algorithmic side, we first consider the *graphic TSP* problem, an important case of metric TSP that has been extensively studied in the classical setting – the metric D corresponds to the shortest path distances in a connected unweighted undirected graph [22, 23, 30]. We give the first $\tilde{O}(n)$ time algorithm for graphic TSP that achieves an approximation factor *strictly better* than 2.

▶ **Theorem 1.1.** *There is an $\tilde{O}(n)$ time randomized algorithm that estimates the cost of graphic TSP to within a factor of $2 - \varepsilon_0$ for some constant $\varepsilon_0 > 0$.*

On the other hand, if we are willing to allow a higher sublinear time, we can get a better approximation ratio.

▶ **Theorem 1.2.** *There is an $\tilde{O}(n^{1.5})$ time randomized algorithm that estimates the cost of graphic TSP to within a factor of $(27/14)$.*

At a high-level, our algorithm is based on showing the following: if a graph G either lacks a matching of size $\Omega(n)$ or has $\Omega(n)$ biconnected components (blocks), then the optimal TSP cost is not too much better than $2n$. Conversely, if the graph G has both a large matching and not too many blocks, then we can show that the optimal TSP cost is distinctly better than $2n$. Since we do not know an efficient sublinear algorithm to estimate the number of

blocks in a graph G, we work with another quantity that serves as a proxy for this and can be estimated in $\tilde{O}(n)$ time. The main remaining algorithmic challenge then is to estimate sufficiently well the size of a largest matching. This problem is very important by itself, and has received much attention [17,24,25,28,36]; please see a detailed discussion of this problem, and relevant recent developments towards the end of this section. Our $\tilde{O}(n)$ query results utilize the recent result of Kapralov *et al.* [17] who give an algorithm to approximate the size of maximum matching to within a constant factor (for some very large constant) in $\tilde{O}(n)$ time in the *pair query* model (is there an edge between a given pair of vertices?). We also show that matching size can be estimated to within a factor of 2 in $\tilde{O}(n^{1.5})$ time, crucial to obtaining the approximation guarantee in Theorem 1.2.

Our approach for estimating graphic TSP cost in sublinear time also lends itself to an $\tilde{O}(n)$ space streaming algorithm that can obtain an even better estimate of the cost. To our knowledge, no estimate better than a 2-approximation was known previously. In the streaming model, we assume that the input to graphic TSP is presented as a sequence of edges of the underlying graph G. Any algorithm for this model, clearly also works if instead the entries of the distance matrix are presented in the stream – an entry that is 1 corresponds to an edge of G, and it can be ignored otherwise as a non-edge.

▶ **Theorem 1.3.** *There is an $O(n)$ space randomized streaming algorithm that estimates the cost of graphic TSP to within a factor of $(11/6)$ in insertion-only streams.*

We next consider another well-studied special case of metric TSP, namely, $(1,2)$-TSP where all distances are either 1 or 2 [2,6,27], and obtain the following result.

▶ **Theorem 1.4.** *There is an $\tilde{O}(n^{1.5})$ time randomized algorithm that estimates the cost of $(1,2)$-TSP to within a factor of 1.625.*

Throughout the paper, whenever we refer to a graph associated with a $(1,2)$-TSP instance, it refers to the graph G induced by edges of distance 1 in our $\{1,2\}$-metric. At a high-level, the idea underlying our algorithm is to analyze the structure of the graph G induced by edges of distance 1. We design an algorithm to estimate the size of a maximal "matching pair" of G which is defined to be a pair of edge-disjoint matchings. We show that whenever the size of a matching pair is large in a graph G, the TSP cost is distinctly smaller than $2n$, and conversely, if this quantity is not large, the TSP cost is close to $2n$. The main remaining algorithm challenge then is to estimate sufficiently well the size of a maximal matching pair, and we show that this can be done in $\tilde{O}(n^{1.5})$ time.

For $(1,2)$-TSP, an $\tilde{O}(n)$ query algorithm that estimates the cost of $(1,2)$-TSP to within a factor of 1.75 was claimed in [1] but this result is based on the matching size estimation results of [25]. Unfortunately, as confirmed by the authors [26], there is a problem with the proof of one of the statements in the paper — Observation 3.9 — which is crucial for the correctness of the main result. As a result, the $\tilde{O}(d)$ time result in the neighbor query model as well as the $\tilde{O}(n)$ time result in the adjacency matrix, claimed in [25] can no longer be relied upon, and we have chosen to make this paper independent of these results [25]. It is worth mentioning that if the $\tilde{O}(n)$-time matching estimation result of [25] can be shown to hold, then the run-time of both Theorems 1.2 and 1.4 can be improved to $\tilde{O}(n)$ time.

We note that it is easy to show that randomization is crucial to getting better than a 2-approximation in sublinear time for both graphic TSP and $(1,2)$-TSP (we include the proof of these results in the full version of this paper). The algorithms underlying Theorems 1.2 and 1.4, lend themselves to $\tilde{O}(n)$ space single-pass streaming algorithms with identical approximation guarantees. These sublinear time algorithms motivate the natural question if

analogously to metric MST, there exist sublinear time algorithms that for any $\varepsilon > 0$, output a $(1+\varepsilon)$-approximate estimate of TSP cost for graphic TSP and $(1,2)$-TSP in $\tilde{O}(n)$ time. We rule out this possibility in a strong sense for both graphic TSP and $(1,2)$-TSP.

▶ **Theorem 1.5.** *There exists an $\varepsilon_0 > 0$, such that any algorithm that estimates the cost of graphic TSP ($(1,2)$-TSP) to within a $(1+\varepsilon_0)$-factor, necessarily requires $\Omega(n^2)$ queries.*

This lower bound result highlights a sharp separation between the behavior of metric MST and metric TSP problems. At a high-level, our lower bound is inspired by the work of Bogdanov *et al.* [7] who showed that any query algorithm that for any $\varepsilon > 0$ distinguishes between instances of parity equations (mod 2) that are either satisfiable (Yes) or at most $(1/2+\varepsilon)$-satisfiable (No), requires $\Omega(n)$ queries where n denotes the number of variables. However, the query model analyzed in [7] is different from ours (see more details in Section 4). We first show that the lower bound of [7] can be adapted to an $\Omega(n^2)$ lower bound in our model, and then show that instances of parity equations can be converted into instances of graphic TSP (resp. $(1,2)$-TSP) such that for some $\varepsilon_0 > 0$, any $(1+\varepsilon_0)$-approximation algorithm for graphic TSP (resp. $(1,2)$-TSP), can distinguish between the Yes and No instances of the parity equations, giving us the desired result.

Finally, similar to many classical approximation algorithms for TSP, our sublinear time estimation algorithms utilize subroutines for estimating the size of a maximum matching in the underlying graph. We show that this is not merely an artifact of our approach.

▶ **Theorem 1.6.** *For any $\varepsilon \in [0, 1/5)$, any algorithm that estimates the cost of an n-vertex instance of graphic TSP or $(1,2)$-TSP to within a $(1+\varepsilon)$-factor, can also be used to estimate the size of a maximum matching in an n-vertex bipartite graph to within an εn additive error, with an identical query complexity, running time, and space usage.*

This connection allows us to translate known lower bounds for matching size estimation in various models to similar lower bounds for metric TSP cost estimation. In particular, using the results of [5], we can show that there exists an ε_0 such that any randomized single-pass dynamic streaming algorithm for either graphic TSP or $(1,2)$-TSP that estimates the cost to within a factor of $(1+\varepsilon_0)$, necessarily requires $\Omega(n^2)$ space.

We conclude by establishing several additional lower bound results that further clarify the query complexity of approximating TSP cost. For instance, we show that if an algorithm can access an instance of graphic TSP by only querying the edges of the graph (via neighbor and pair queries), then any algorithm that approximates the graphic TSP cost to a factor better than 2, necessarily requires $\Omega(n^2)$ queries. This is in sharp contrast to Theorem 1.1, and shows that working with the distance matrix is crucial to obtaining sublinear time algorithms for graphic TSP. We also show that even in the distance matrix representation, the task of *finding a tour* that is $(2-\varepsilon)$-approximate for any $\varepsilon > 0$, requires $\Omega(n^2)$ queries for both graphic TSP and $(1,2)$-TSP.

Matching Size Estimation: As the problem of matching size estimation is intimately connected to metric TSP cost estimation, we briefly review some relevant work here. This line of research primarily assumes that we are given a graph $G(V,E)$ with maximum degree d, that can be accessed via *neighbor queries* [14]: (a) for any vertex v, we can query its degree, and (b) for any vertex v and an integer i, we can learn the i_{th} neighbor of v.

Parnas and Ron [28] initiated the study of matching size estimation in sublinear time and gave an $d^{O(\log(d/\varepsilon)}$ time algorithm that estimates the matching size to within a constant factor plus an additive εn error for any $\varepsilon > 0$. Nguyen and Onak [24] presented a new estimation algorithm and showed that it can estimate the matching size to within a factor of 2 plus an

additive εn error in $2^{O(d)}/\varepsilon^2$ time. We will refer to this approximation guarantee as a $(2,\varepsilon)$-*approximation* of matching size. Yoshida *et al.* [36] strongly improved upon the performance guarantee obtained in [24], and showed that a $(2,\varepsilon)$-approximation to matching size can be accomplished in $O(d^4/\varepsilon^2)$ time (in fact, they obtain the stronger $(2\pm\varepsilon)$-approximation guarantee). The analysis of [36] was further improved by Onak *et al.* [25] who showed that the state of the art for $(2,\varepsilon)$-approximation of matching size. We note that it is known that any $(O(1),\varepsilon)$-approximate estimate of matching size necessarily requires $\Omega(d)$ queries [28], so the result of [25] is essentially best possible. Unfortunately, as mentioned above, we recently discovered a subtle mistake in the analysis of Onak *et al.* [26]. Consequently, the best known time complexity for obtaining a $(2,\varepsilon)$-approximate estimate is $\tilde{O}(d^2/\varepsilon^2)$); this weaker result also follows from the work of [25], but does not rely on the incorrect observation in [25].

The difference between a linear dependence versus a quadratic dependence on degree d is however huge in the sublinear time applications when the graph is not very sparse. In particular, while an $\tilde{O}(d)$ query result translates into an $\tilde{O}(n)$ time algorithm in the adjacency matrix model, an $\tilde{O}(d^2)$ query result gives only an $\tilde{O}(n^2)$ time algorithm, which is clearly not useful. Very recently, Kapralov *et al.* [17] gave an alternate approach based on a vertex "peeling" strategy (originally proposed in [28]) that yields an $(O(1),\varepsilon)$-approximation of matching size in $\tilde{O}(d/\varepsilon^2)$ time. Unfortunately, the constant hidden in the $O(1)$ notation is very large, and efficiently obtaining a $(2,\varepsilon)$-approximation to matching size remains an important open problem. Meanwhile, by directly building on the work of [36], we obtain an $\tilde{O}(n^{1.5})$ time algorithm for a $(2,\varepsilon)$-approximation to matching size in the adjacency matrix model, and it is this algorithm that is used in the results of Theorem 1.2 and Theorem 1.4.

Other Related Work: We note here that there is an orthogonal line of research that focuses on computing an approximate solution in near-linear time when the input is presented as a weighted undirected graph, and the metric is defined by shortest path distances on this weighted graph. It is known that in this model, for any $\varepsilon > 0$, there is an $\tilde{O}(m/\varepsilon^2 + n^{1.5}/\varepsilon^3)$ time algorithm that computes a $(3/2+\varepsilon)$-approximate solution; here n denotes the number of vertices and m denotes the number of edges [10], and that a $(3/2+\varepsilon)$-approximate estimate of the solution cost can be computed in $\tilde{O}(m/\varepsilon^2)$ time [9]. It is not difficult to show that in this access model, even when the input graph is unweighted (i.e. a graphic TSP instance), any algorithm that outputs better than a 2-approximate estimate of the TSP cost, requires $\Omega(n+m)$ time even when $m = \Omega(n^2)$. Hence this access model does not admit sublinear time algorithms that beat the trivial 2-approximate estimate.

Organization: In Section 2, we present our algorithms for graphic TSP (Theorem 1.1 and Theorem 1.2). In section 3, we present the 1.625-approximation algorithm of $(1,2)$-TSP (Theorem 1.4). In Section 4, we present our lower bound result that rules out possibility of a sublinear-time approximation scheme for both graphic TSP and $(1,2)$-TSP (Theorem 1.5). In Section 5, we present a strong connection between approximating metric TSP cost and estimating matching size (Theorem 1.6). We defer to the full version the proofs of several additional lower bound results on the complexity of approximating graphic TSP and $(1,2)$-TSP cost.

2 Approximation for Graphic TSP Cost

In this section, we exploit well-known properties of biconnected graphs and biconnected components in graphs to give an algorithm that achieves a $(2-\frac{1}{7c_0})$-approximation for graphic TSP if we have an efficient algorithm that approximates the maximum matching size within a factor of c_0. We first relate the cost of the TSP tour in a graph to the costs of

the TSP tours in the biconnected components of the graph. Next we show that if the graph does not have a sufficiently big matching, it does not have a TSP tour whose length is much better than $2n$. We also show that if a graph has too many degree 1 vertices, or vertices of degree 2, both whose incident edges are bridges, then it does not have a TSP tour of cost much better than $2n$. We then establish the converse - a graph that has a good matching and not too many bad vertices (namely, vertices of degree 1 or articulation points of degree 2), then it necessarily has a TSP tour of cost much better than $2n$. We design $\tilde{O}(n)$ time test for the second condition, allowing us to approximate the cost of an optimal graphic TSP tour in sublinear time together with some known techniques for testing the first condition. In what follows, we first present some basic concepts and develop some tools that will play a central role in our algorithms.

2.1 Preliminaries

An unweighted graph $G = (V, E)$, defines a *graphic metric* in V, where the distance between any two vertices u and v is given by the length of the shortest path between u and v. The *graphic TSP* is the Traveling Salesman Problem defined on such a graphic metric. In this paper our goal is to find a non-trivial approximation to the length of the traveling salesman tour in sublinear time in a model where we are allowed to make *distance queries*. In the distance query model, the algorithm can make a query on a pair of vertices (u, v) and get back the answer $d(u, v)$, the distance between u and v in G.

In a connected graph G, an edge e is a *bridge* if the deletion of e would increase the number of connected components of G. A connected graph with no bridge is called a *2-edge-connected graph*. A maximal 2-edge-connected subgraph of G is called a *2-edge-connected component*. The *bridge-block tree* of a graph is a tree such that the vertex set contains the 2-edge-connected components and the edge set contains the bridges in the graph.

A connected graph G is called *2-vertex-connected* or *biconnected* if when any one vertex is removed, the resulting graph remains connected. In a graph which is not biconnected, a vertex v whose removal increases the number of components is called an *articulation point*. It is easy to prove that any biconnected graph with at least 3 vertices does not have degree 1 vertices. A well-known alternate characterization of biconnectedness is that, a graph G is biconnected if and only if for any two distinct edges, there is a simple cycle that contains them.

A *biconnected component* or *block* in a graph is a maximal biconnected subgraph. Any graph G can be decomposed into blocks such that the intersection of any two blocks is either empty, or a single articulation point. Each articulation point belongs to at least two blocks. If a block is a single edge, then we call this block a *trival block*; otherwise it is a *non-trivial block*. A trival block is also a bridge in the graph. The size of a block is the number of vertices in the block. The following lemma shows the relationship between the number of blocks and the sum of the sizes of the blocks.

▶ **Lemma 2.1.** *If a connected graph G has n vertices and k blocks, then the sum of the sizes of the blocks is equal to $n + k - 1$.*

Proof. We prove the lemma by induction on the number k of blocks. The base case is when $k = 1$. In this case, G itself is a block of size n.

For the induction step, we have $k > 1$ and thus the graph has at least one articulation point. Suppose v is an arbitrary articulation point in G. Let $V_1, V_2, \ldots, V_j$ be the set of vertices in the connected components of $G \setminus \{v\}$. We have $\sum_{i=1}^{j} |V_i| = n - 1$. Let $G_1, G_2, \ldots, G_j$ be the subgraphs of G induced by $V_1 \cup \{v\}, V_2 \cup \{v\}, \ldots, V_j \cup \{v\}$. For any

G_i, let k_i be the number of blocks in G_i, we have $\sum_{i=1}^{j} k_i = k$. By induction hypothesis, the sum of the sizes of blocks in G_i is $|V_i| + 1 + k_i - 1 = |V_i| + k_i$. So the sum of the sizes of blocks in G is $\sum_{i=1}^{j} |V_i| + k_i = n - 1 + k$. ∎

The block decomposition of a graph has a close relationship with the cost of graphic TSP of the graph.

▶ **Lemma 2.2** (Lemma 2.1 of [21]). *The cost of the graphic TSP of a connected graph $G = (V, E)$ is equal to the sum of the costs of the graphic TSP of all blocks in the graph.*

Together these two lemmas give us a simple lower bound on the cost of the graphic TSP of a graph G (using the fact that the cost of graphic TSP is at least the number of vertices in the graph).

▶ **Lemma 2.3.** *If a graph G has n vertices and k blocks, then the cost of graphic TSP of G is at least $n + k - 1$.*

An *ear* in a graph is a simple cycle or a simple path. An ear which is a path is also called an *open ear* and it has two endpoints, whereas for a cycle, one vertex is designated as the endpoint. An *ear decomposition* of a graph is a partition of a graph into a sequence of ears such the endpoint(s) of each ear (except for the first) appear on previous ears and the internal points (the points that are not endpoints) are not on previous ears. A graph G is biconnected if and only if G has an ear decomposition such that each ear but the first one is an open ear [34]. An ear is *nontrivial* if it has at least one internal point. The following lemma upper bounds the cost of graphic TSP of a biconnected graph.

▶ **Lemma 2.4** (Lemma 5.3 of [30], also a corollary of Lemma 3.2 of [22]). *Given a 2-vertex-connected graph $G = (V, E)$ and an ear-decomposition of G in which all ears are nontrivial, a graphic TSP tour of cost at most $\frac{4}{3}(|V| - 1) + \frac{2}{3}\pi$ can be found in polynomial-time, where π denotes the number of ears.*

We now prove an important lemma that gives an upper bound on the cost of graphic TSP in a biconnected graph in terms of the size of a matching in the graph.

▶ **Lemma 2.5.** *Suppose G is a biconnected graph with at least $n \geq 3$ vertices. If G has a matching M, then the cost of graphic TSP of G is at most $2n - 2 - \frac{2|M|}{3}$.*

Proof. We first find a spanning biconnected subgraph of G that only contains $2n - 2 - M$ edges, then use Lemma 2.4 to bound the cost of graphic TSP.

We construct a spanning biconnected subgraph $G^\star = P_0 \cup P_1 \cup \dots$ recursively: P_0 contains a single edge in M. If $G_{i-1} = P_0 \cup P_1 \cup \dots \cup P_{i-1}$ is a spanning subgraph of G, let $G^\star = G_{i-1}$ and finish the construction. Otherwise we construct P_i as follows. Let e be an edge in M both whose endpoints are not in G_{i-1}. If there is no such edge, then let e be an arbitrary edge such that at least one of its endpoints is not in G_{i-1}. Let e' be an arbitrary edge in G_{i-1}. By the alternate characterization of biconnectedness, there is a simple cycle C_i that contains both e and e'. Let P_i be the path in C_i that contains e and exactly two vertices in G_{i-1}, which are the endpoints of P_i.

Since P_i contains at least one vertex not in G_{i-1}, the construction always terminates. Note that $P_0 \cup P_1$ is a cycle, and each P_i ($i > 1$) is an open ear of $G^\star$. So, $(P_0 \cup P_1, P_2, \dots)$ is an open ear decomposition of $G^\star$, which means $G^\star$ is biconnected.

Now we prove that the number of edges in $G^\star$ is at most $2n - 2 - M$. Let n_i be the number of vertices in $G_i \backslash G_{i-1}$. Let G_{-1} be the empty graph, so that n_0=2. Let p_i be the

number of edges in P_i and m_i be the number of edges e in M such that $e \cap G_i \neq \emptyset$ and $e \cap G_{i-1} = \emptyset$. (Here we view an edge as a 2-vertex set.) Note that $m_0 = 1$. Suppose $G^\star = G_k$. Then $\sum_{i=1}^{k} n_i = n$, $\sum_{i=1}^{k} p_i$ is the number of edges in $G^\star$ and $\sum_{i=1}^{k} m_i = |M|$. For any $i > 0$, P_i is an open ear whose internal points are not in G_{i-1}. So $n_i = p_i - 1$. If there is an edge $e \in M$ such that $e \cap G_{i-1} = \emptyset$, then P_i contains both endpoints of an edge in M, which means $m_i \leq n_i - 1$. If all edges in M already have an endpoint in G_{i-1}, $m_i = 0 \leq n_i - 1$. So in both cases, $p_i = n_i + 1 = 2n_i - (n_i - 1) \leq 2n_i - m_i$. Also, $p_0 = 1 = 2n_0 - 2 - m_0$. So the number of edges in $G^\star$ is $\sum_{i=0}^{k} p_i \leq 2n_0 - 2 - m_0 + \sum_{i=1}^{k}(2n_i - m_i) = 2n - 2 - |M|$.

As $(P_0 \cup P_1, P_2, P_3, \ldots, P_k)$ is an open ear decomposition of $G^\star$, the number of ears in G is k. On the other hand, $\sum_{i=0}^{k} p_i = 1 + \sum_{i=1}^{k}(n_i + 1) = n - 1 + k$, we have $n - 1 + k \leq 2n - 2 - |M|$, which means $k \leq n - 1 - |M|$. By Lemma 2.4, the cost of graphic TSP of $G^\star$ is at most $\frac{4}{3}(n-1) + \frac{2}{3}k \leq 2(n-1) - \frac{2}{3}|M|$.

Since $G^\star$ is a subgraph of G that contains all the vertices in G, the cost of graphic TSP of G is at most the cost of graphic TSP of $G^\star$, which is at most $2n - 2 - \frac{2}{3}|M|$. ∎

2.2 Approximation Algorithm for Graphic TSP

We now give our sublinear-time algorithm for approximating the cost of graphic TSP of a graph G to within a factor strictly smaller than 2.

We call a vertex v a *bad* vertex if v has degree 1 or is an articulation point with degree 2.

For any given $\delta > 0$, the graphic TSP algorithm performs the following two steps.

1. Obtain an estimate $\hat{\alpha}n$ of the size of maximum matching αn.
2. Obtain an estimate $\hat{\beta}n$ of the number of bad vertices βn.

The algorithm then output $\min\{2n, (2 - \frac{2}{7}(\hat{\alpha} - 2\hat{\beta}))n\}$.

To perform the second step in $\tilde{O}(n)$ distance queries and time, we randomly sample $O(\frac{1}{\delta^2})$ vertices. For each sampled vertex, we can obtain the degree with n queries. The following lemma shows that we can also check whether a degree 2 vertex is an articulation point using distance queries in $O(n)$ time. Then by the Chernoff bound, we can approximate the number of bad vertices with additive error $O(\delta n)$ with a high constant probability.

▶ **Lemma 2.6.** *Suppose a vertex v in a connected graph G has only two neighbors u and w. The following three conditions are equivalent:*

1. *v is an articulation point.*
2. *The edges (u, v) and (v, w) are both bridges.*
3. *For any vertex $v' \neq v$, $|d(u, v') - d(w, v')| = 2$.*

Proof. We first prove the first two conditions are equivalent. If v is an articulation point, then v is in two different blocks. So edge (u, v) and (v, w) are in different blocks, which means v has degree 1 in both blocks. So both blocks are trivial, which means (u, v) and (v, w) are both bridges. If (u, v) and (v, w) are both bridges, then deleting either (u, v) or (v, w) will disconnect u and w, which means deleting v will also disconnect u and w.

Next we prove that the third condition is equivalent to the first two. Suppose v is an articulation point. Since v has degree 2, the graph $G \setminus \{v\}$ has only two components, one containing u and the other containing w. For any vertex $v' \neq v$, without loss of generality, suppose v' is in the same component as u in $G \setminus \{v\}$. Since (u, v) and (v, w) are both bridges in G, any path between v' and w contains u and v. So $d(v', w) = d(v', u) + 2$.

If v is not an articulation point, then u and w are connected in $G \setminus \{v\}$. Let $(u = v_0, v_1, v_2, \ldots, v_k = w)$ be the shortest path between u and w in $G \setminus \{v\}$. For any vertex v_i on

the path, the distance between v_i and u (resp. w) in $G \setminus \{v\}$ is i (resp. $k-i$). Consider the shortest path between u and v_i in G. If this path does not contain v, then it is the same as the path in $G \setminus \{v\}$. In this case, $d(u, v_i) = i$. If the shortest path contains v, then v must be the second last vertex on the path and w be the third last one. In this case, $d(u, v_i) = k - i + 2$. So $d(u, v_i) = \min\{i, k - i + 2\}$. Similarly, we also have $d(v_i, w) = \min\{i + 2, k - i\}$. Let $v' = v_{\lfloor k/2 \rfloor}$. Since $|i - (k - i)| \leq 1$, we have $i < k - i + 2$ and $k - i < i + 2$, which means $|d(u, v') - d(w, v')| = |i - (k - i)| \leq 2$. ∎

Next, we prove that if α is small or β is large, the cost of graphic TSP is bounded away from n. The following lemma shows that if the size of maximum matching of a graph is small, then the cost of the graphic TSP is large.

▶ **Lemma 2.7.** *For any $\varepsilon > 0$, if the maximum matching of a graph G has size at most $\frac{(1-\varepsilon)n}{2}$, then the cost of graphic TSP of G is at least $(1 + \varepsilon)n$.*

Proof. Suppose the optimal TSP tour is $(v_0, v_1, \ldots, v_{n-1}, v_n = v_0)$. Since the size of maximum matching in G is at most $\frac{(1-\varepsilon)n}{2}$, there are at most $\frac{(1-\varepsilon)n}{2}$ edges between pairs (v_i, v_{i+1}) where i is even (resp. odd). So there are at least εn pairs of (v_i, v_{i+1}) that have distance at least 2, which means that the optimal cost of TSP tour of G is $\sum_{i=1}^{n-1} d(v_i, v_{i+1}) \geq n + \varepsilon n = (1 + \varepsilon)n$. ∎

The following lemma shows that if β is large, the cost of graphic TSP is large.

▶ **Lemma 2.8.** *For any $\varepsilon > 0$, if a connected graph G has εn bad vertices, then the cost of graph-TSP of G is at least $(1 + \varepsilon)n - 2$.*

Proof. We first prove by induction on the number of vertices that a graph with k bad vertices has $k - 1$ bridges. The base case is when $n = 2$, the graph has $k = 2$ bad vertices and $1 = k - 1$ bridge.

For the induction step, the graph has n vertices with $n \geq 3$. If G has no degree 1 vertices, then the graph has k articulation points with degree 2. By Lemma 2.6, any edge incident on a degree 2 articulation point is a bridge. So each bad vertex is incident on 2 bridges. On the other hand, a bridge is incident on at most 2 vertices. So there are at least $\frac{2k}{2} = k$ bridges in G. Next, suppose G has degree 1 vertices. Let v be an arbitrary such vertex and let u be its neighbor. Since G is connected and $n \geq 3$, u must has degree at least 2, since otherwise u and v are not connected to other vertices in G. Consider the graph $G \setminus \{v\}$, if u is a bad vertex in G, u has degree 1 in $G \setminus \{v\}$ and is still a bad vertex. So the number of bad vertices in $G \setminus \{v\}$ is $k - 1$. By induction hypothesis, $G \setminus \{v\}$ has at least $k - 2$ bridges. G has at least $k - 1$ bridges since (u, v) is also a bridge.

So G has at least $\varepsilon n - 1$ bridges, and the number of blocks in G is at least $\varepsilon n - 1$. By Lemma 2.3, the cost of graph-TSP of G is at least $n + \varepsilon n - 2 = (1 + \varepsilon)n - 2$. ∎

Finally, the following lemma shows that the cost of graphic TSP is at most $(2 - \frac{2}{7}(\hat{\alpha} - 2\beta))n$.

▶ **Lemma 2.9.** *If a graph has a matching M of size $\alpha' n$ and the graph has βn bad vertices, the cost of graphic TSP of G is at most $(2 - \frac{2}{7}(\alpha' - 2\beta))n$.*

Proof. Let $G_1, G_2, \ldots, G_k$ be the block decomposition of G. Let n_i be the size of G_i. If $|n_i| \geq 3$, by Lemma 2.5, the cost of the graphic TSP of G_i is at most $2n_i - 3$ since any non-empty graph has a matching of size at least 1. If $|n_i| = 2$, then the graphic TSP of G_i is exactly $2 = 2n_i - 2$. Suppose G has ℓ non-trivial blocks. Then by Lemma 2.2 the cost of graphic TSP of G is at most $\sum_{i=1}^{k}(2n_i - 2) - \ell$, which equals to $2n - 2 - \ell$ by Lemma 2.1.

Let m_i be the size of maximum matching in G_i if G_i is a non-trivial block, and let $m_i = 0$ if G_i is a trivial block. By Lemma 2.5, the cost of the graphic TSP of G_i is at most $2n_i - 2 - \frac{2m_i}{3}$. For any non-trivial block G_i, $M \cap G_i$ is a matching in G_i. So the size of maximum matching in G_i is at least the number of edges in $M \cap G_i$. So by Lemma 2.2 and Lemma 2.1, the cost of graphic TSP of G is at most $\sum_{i=1}^{k}(2n_i - 2 - \frac{2}{3}m_i) = 2n - 2 - \frac{2}{3}|M'|$, where M' is the set of edges in M that are not bridges in G. Let B be the number of bridges in G. We have $2n - 2 - \frac{2}{3}|M'| \leq 2n - 2 - \frac{2}{3}(|M| - B)$.

So there are two upper bounds of the graphic TSP of G — $2n-2-\ell$ and $2n-2-\frac{2}{3}(|M|-B)$. Which bound is better depends on the number of bridges B.

If $B \leq (\frac{4}{7}\alpha' + \frac{6}{7}\beta)n$, the cost of graphic TSP of G is at most

$$2n - 2 - \frac{2}{3}(|M| - B) \leq 2n - \frac{2}{3}(\frac{3}{7}\alpha' - \frac{6}{7}\beta)n = (2 - \frac{2}{7}(\alpha' - 2\beta))n$$

If $B > (\frac{4}{7}\alpha' + \frac{6}{7}\beta)n$, consider the bridge-block tree T of G. T has at least B edges and at least $B + 1$ vertices. Since T is a tree, there are at least $\frac{B}{2}$ vertices of degree at most 2. For any vertex v_T of degree at most 2 in T, if the vertex v_T represents a single vertex v in G, then v is either a degree 1 vertex or a degree 2 articulation point in G, otherwise v_T represents a 2-edge-connected component of size at least 2 in G. So There are at least $\frac{B}{2} - \beta n \geq (\frac{2}{7}\alpha' - \frac{4}{7}\beta)n$ 2-edge-connected components of size at least 2. Since any 2-edge-connected component of size at least 2 has no bridge, each such component of G contains at least 1 non-trivial block in G, implying that $\ell \geq \frac{2}{7}(\alpha' - 2\beta)n$. So the cost of graphic TSP of G is at most $2n - 2 - \ell \leq (2 - \frac{2}{7}(\alpha' - 2\beta))n$. ∎

We summarize the ideas in this section and prove the following lemma.

▶ **Lemma 2.10.** *For any $c_0 > 1$ and $\delta > 0$, suppose $\hat{\alpha} \leq \alpha \leq c_0\hat{\alpha} + \delta$ and $\hat{\beta} - \delta \leq \beta \leq \hat{\beta}$. Then $(2 - \frac{2}{7}(\hat{\alpha} - 2\hat{\beta}))n$ is an approximation of the size of graphic TSP within a factor of $2 - \frac{1}{7c_0} + \delta$.*

Proof. Let $\hat{T} = (2 - \frac{2}{7}(\hat{\alpha} - 2\hat{\beta}))n$. Since $\hat{\beta} \geq \beta$ and $\hat{\alpha} \leq \alpha$, by Lemma 2.9, $T \leq \hat{T}$.

Then we prove that $\hat{T} \leq (2 - \frac{1}{7c_0} + \delta)T$. By Lemma 2.7 and Lemma 2.8, $T \geq \max\{(2 - 2\alpha)n, (1 + \beta)n - 2\}$, which means

$$(2 - \frac{1}{7c_0} + \delta)T \geq (2 - \frac{1}{7c_0})\max\{(2 - 2\alpha)n, (1 + \beta)n\} - 4 + \delta n$$

On the other hand, $\hat{T} \leq (2 - \frac{2}{7}(\frac{\alpha}{c_0} - 2\beta))n + \frac{6}{7}\delta n$ since $c_0\hat{\alpha} + \delta \leq \alpha$ and $\hat{\beta} \leq \beta + \delta$. For sufficient large n, we have $\delta n - 4 \geq \frac{6}{7}\delta n$, so it is sufficient to prove that $\frac{2 - \frac{2}{7}(\frac{\alpha}{c_0} - 2\beta)}{\max\{2-2\alpha, 1+\beta\}} \leq 2 - \frac{1}{7c_0}$ for any $0 \leq \alpha, \beta \leq 1$ and $c_0 \geq 1$.

Let $\gamma = \frac{\alpha}{c_0} - 2\beta$, $1 + \beta = 1 + (\frac{\alpha}{c_0} - \gamma)/2$, so if we fix γ, $\max\{2-2\alpha, 1+\beta\}$ is minimized when $2 - 2\alpha = 1 + (\frac{\alpha}{c_0} - \gamma)/2$. In this case $\alpha = \frac{(2+\gamma)c_0}{4c_0+1}$ and $\max\{2 - 2\alpha, 1 + \beta\} = \frac{4c_0+2}{4c_0+1} - \frac{2c_0}{4c_0+1}\gamma$. If $\gamma \leq \frac{1}{2c_0}$,

$$\begin{aligned}\frac{2 - \frac{2}{7}(\alpha - 2\beta)}{\max\{2 - 2\alpha, 1 + \beta\}} &\leq \frac{2 - \frac{2}{7}\gamma}{\frac{4c_0+2}{4c_0+1} - \frac{2c_0}{4c_0+1}\gamma} = \frac{4c_0 + 1}{7c_0} - \frac{2 - \frac{4c_0+2}{7c_0}}{\frac{4c_0+2}{4c_0+1} - \frac{2c_0}{4c_0+1}\gamma} \\ &\leq \frac{4c_0 + 1}{7c_0} + 2 - \frac{4c_0 + 2}{7c_0} = 2 - \frac{1}{7c_0}\end{aligned}$$

If $\gamma > \frac{1}{2c_0}$, $\frac{2 - \frac{2}{7}(\alpha - 2\beta)}{\max\{2-2\alpha, 1+\beta\}} < \frac{2 - \frac{1}{7c_0}}{1} = 2 - \frac{1}{7c_0}$ since $\beta \geq 0$. So $\hat{T} \leq (2 - \frac{1}{7c_0} + \delta)T$. ∎

By Lemma 2.10, we immediately have the following theorem.

▶ **Theorem 2.11.** *For any $\delta > 0$ and $c_0 \geq 1$. Given a graph G with maximum matching size αn, suppose there is an algorithm that uses pair queries, runs in t time, and with probability at least $2/3$, outputs an estimate of the maximum matching size $\hat{\alpha}n$ such that $\hat{\alpha} \leq \alpha \leq c_0\hat{\alpha}+\delta$. Then there is an algorithm that approximates the cost of graphic TSP of G to within a factor of $2 - \frac{1}{7c_0} + \delta$, using distance queries, in $t + \tilde{O}(n/\delta^2)$ time with probability at least $3/5$.*

Proof. We first use the algorithm in the assumption to obtain an estimate $\hat{\alpha}n$ of the size of maximum matching αn. The following analysis is based on the event that this algorithm is run successfully, which has probability $2/3$.

We then sample $N = \frac{100}{\delta^2}$ vertices. For each sampled vertex v, we first query the distance between v and every vertex in G to obtain the degree of v. If v has degree 2, suppose u and w are the neighbors of v. We query the distance from u and w to every vertex in G. By Lemma 2.6, v is an articulation point if and only if there is no vertex v' such that $|d(u, v') - d(w, v')| \leq 1$. So we can check if v is a bad vertex with $O(n)$ distance queries and time. Suppose there are βn bad vertices in G and $(\hat{\beta} - \delta/2)N$ sampled vertices are bad. By Chernoff bound, the probability that $\left|\beta - \hat{\beta} + \delta/2\right| > \delta/2$ is at most $2e^{\frac{\delta^2 N^2}{16}} < 1/15$. We analyze the performance based on the event that $\beta \leq \hat{\beta} \leq \beta + \delta$.

By Lemma 2.10, $(2 - \frac{2}{7}(\hat{\alpha} - 2\hat{\beta}))$ is a $(2 - \frac{1}{7c_0} + \delta)$ approximation of the size of graphic TSP of G. The probability of failure is at most $1/3 + 1/15 = 2/5$. ∎

Proof of Theorem 1.2: The following theorem whose proof appears in the full version, gives an algorithm for matching size estimation that only uses *pair queries* – given a pair of vertices, is there an edge between them? Note that any pair query can be simulated by a single query to the distance matrix in a graphic TSP instance.

▶ **Theorem 2.12.** *For any $\varepsilon > 0$, there is an algorithm that uses pair queries, runs in $\tilde{O}(n^{1.5}/\varepsilon^2)$ time, and with probability $2/3$, outputs an estimate of the size of a maximal matching within an additive error εn.*

Substituting the above result in Theorem 2.11 and using the fact that a maximum matching has size at most twice the size of a maximal matching (setting $c_0 = 2$, and $\delta = \varepsilon$), we obtain Theorem 1.2.

Proof of Theorem 1.1: Kapralov et al. [17] give an algorithm that uses $\tilde{O}(d)$ queries to approximate the size of maximum matching in a graph with average degree d in the neighbor query model (the approximation ratio is a very large constant). Together with a reduction in [25], this implies a pair query algorithm that uses $\tilde{O}(n)$ queries to estimate matching size to a constant factor. Combined with Theorem 2.11, this implies Theorem 1.1.

2.3 An $O(n)$ Space $(\frac{11}{6})$-Approximate Streaming Algorithm

We show here that our approach above can be extended to the insertion-only streaming model to obtain for any $\varepsilon > 0$, an $(\frac{11}{6} + \varepsilon)$-approximate estimate of the graphic TSP cost using $O(n/\varepsilon^2)$ space. Given a stream containing edges of a graph $G(V, E)$, our algorithm performs the following two tasks simultaneously:

- Find a maximal matching M in G – let αn denote its size.
- Estimate the number of bridges in the maximal matching M, say βn, to within an additive error of εn.

The algorithm outputs $(2 - \frac{2}{3}(\alpha - \beta))n$ as the estimated cost of graphic TSP of G.

In an insertion-only stream, it is easy to compute a maximal stream with $O(n)$ space, we start with M initialized as an empty set, and add a new edge (u, v) into the matching M iff neither u nor v are already in M. It is also easy to check if the edge is a bridge in insertion-only stream with $O(n)$ space. We can maintain a disjoint-set data structure. Whenever an edge arrives (other than e), we merge the connected components of its endpoints. If there is only one component remaining at the end of the stream, then e is not a bridge. Otherwise, e is a bridge.

To estimate the number of bridges in the maximal matching, we sample $N = 100/\varepsilon^2$ edges in the matching, and run in parallel N tests where each test determines whether or not the sampled edge is a bridge. We use $O(n/\varepsilon^2)$ space in total since we sample $N = O(1/\varepsilon^2)$ edges. Suppose there are $\bar{\beta}$ sampled edges are bridges, then by Chernoff bound, $\hat{\beta}n = \frac{\bar{\beta}|M|}{N}$ is an approximation of βn to within additive error εn with probability at least $9/10$.

As stated, this gives us a two-pass algorithm: the first pass for computing the matching M, and the second pass for estimating the number of bridges in M. However, we can do both these tasks simultaneously in a single pass as follows: at the beginning of the stream, we start the process of finding connected components of graph G. Whenever an edge e is added to M, if $|M| < N$, then we create a new instance I_e of the connectivity problem that ignores the edge e. This clearly allows us to test whether or not e is a bridge. Once $|M| > N$, then whenever an edge e is added to M, with probability $\frac{N}{|M|}$, we randomly drop an existing instance, say $I_{e'}$ of connectivity, and create a new instance I_e of connectivity that only ignores edge e (we insert back the edge e' into I_e). Since there are at most N instances of connectivity instance that are running in parallel, the algorithm uses $O(nN) = O(n/\varepsilon^2)$ space.

We defer to the full version the details of the analysis showing that the cost estimate output above is a $(\frac{11}{6} + \varepsilon)$-approximate estimate.

3 (1.625)-Approximation for $(1, 2)$-TSP Cost in $\tilde{O}(n^{1.5})$ Time

In this section, we give an algorithm that for any $\delta > 0$, approximates the cost of the minimum $(1, 2)$-TSP to within a factor of $1.625 + \delta$ with $\tilde{O}(n^{1.5}/\delta^2)$ queries. The idea of the algorithm is to approximate the size of a maximal "matching pair" of G. In a graph G, a *matching pair* (M_1, M_2) is a pair of edge-disjoint matchings. A *maximal* matching pair is a matching pair (M_1, M_2) such that for any edge $e \notin M_1 \cup M_2$, neither $M_1 \cup \{e\}$ nor $M_2 \cup \{e\}$ is a matching. The size of a matching pair (M_1, M_2) is the sum of the sizes of M_1 and M_2. The following lemma shows that the size of any maximal matching pair is lower bounded by the size of maximum matching in the graph.

▶ **Lemma 3.1.** *Suppose M is a matching in a graph G. Then any maximal matching pair (M_1, M_2) in G has size at least $|M|$.*

Proof. Let X_1 be the set of vertices matched in both M and M_1, and X_2 be the set of vertices matched in both M and M_2. We have $|X_1| + |X_2| \leq 2|M_1| + 2|M_2|$ since M_1 and M_2 are both matchings. On the other hand, for any edge $e \in M$, if e is either in M_1 or M_2, then both of its endpoints are in X_1 or X_2. If e is neither in M_1 or M_2, then there are edges $e_1 \in M_1$ and $e_2 \in M_2$ that share an endpoint with e since (M_1, M_2) is a maximal matching pair. So both X_1 and X_2 contain at least one endpoint of e. In both case e's endpoints appear twice in X_1 and X_2. So $|X_1| + |X_2| \geq 2|M|$, which means $|M_1| + |M_2| \geq |M|$. ∎

If a graph has a large sized matching pair, then the cost of $(1, 2)$-TSP is not very large.

▶ **Lemma 3.2.** *If a graph G with n vertices contains a matching pair (M_1, M_2) of size X, then the cost of $(1,2)$-TSP of G is at most $2n - \frac{3}{4}X$.*

Proof. Since M_1 and M_2 are both matchings, $M_1 \cup M_2$ only contains paths and cycles of even length. We delete one edge from each cycle in $M_1 \cup M_2$, resulting in a graph that only contains paths. Since the cycles in $M_1 \cup M_2$ are of even length, the size of any cycle is at least 4. We deleted at most $\frac{1}{4}X$ edges, so G contains a set of vertex disjoint paths (including some of length 0, corresponding to isolated vertices), with total size at least $\frac{3}{4}X$. Construct a TSP tour by ordering the paths arbitrarily, orienting each one, and connecting the end of one path with the start of the next, cyclically. The tour contains at least $\frac{3}{4}X$ edges of weight 1, while the remaining edges are of weight 2. So the cost of the tour is at most $2n - \frac{3}{4}X$. ∎

By Lemma 3.1, the maximum matching size is upper bounded by the size of any maximal matching pair. It follows that if the maximum matching size is small, the cost of $(1,2)$-TSP is large.

▶ **Lemma 3.3.** *For any $\varepsilon > 0$, if the maximum matching of a graph G has size at most $\frac{(1-\varepsilon)n}{2}$, then the cost of $(1,2)$-TSP of G is at least $(1+\varepsilon)n$.*

The proof of Lemma 3.3 is similar to the proof of Lemma 2.7 and we omit it here. By Lemma 3.2 and Lemma 3.3, if we can approximate the size of an arbitrary maximal matching pair, then we will get a good approximation of the cost of the $(1,2)$-TSP.

▶ **Theorem 3.4.** *There is an algorithm that uses pair queries, with probability at least $2/3$, approximates the size of a maximal matching pair with additive error εn using $\tilde{O}(n^{1.5}/\varepsilon^2)$ time.*

The algorithm in Theorem 3.4 is given in the full verstion of this paper. With Theorem 3.4, we can approximate the cost of $(1,2)$-TSP in a graph G by the size of maximal matching pair.

▶ **Theorem 3.5.** *For any $\delta > 0$, there is an algorithm that estimates the cost of $(1,2)$-TSP of a graph G within a factor of $1.625 + \delta$ using $\tilde{O}(n^{1.5}/\delta^2)$ queries with probability at least $2/3$.*

Proof. Let $\varepsilon = \delta/2$. We use the algorithm in Theorem 3.4 that approximates the size of a maximal matching pair. Suppose the output of the algorithm is $\bar{X}$. Then, by Theorem 3.4, there is a maximal matching pair of size X such that $\left|X - \bar{X}\right| \leq \varepsilon n$. We output the cost of the $(1,2)$-TSP of G to be $\bar{T} = 2n - \frac{3}{4}(\bar{X} - \varepsilon n)$. Suppose the optimal $(1,2)$-TSP has cost T. By Lemma 3.2, $T \leq 2n - \frac{3}{4}X \leq 2n - \frac{3}{4}(\bar{X} - \varepsilon n) = \bar{T}$. On the other hand, by Lemma 3.3, the size of maximum matching in G is at least $(2n - T)/2$. So by Lemma 3.1, $X \geq (2n - T)/2$, which means $\bar{X} \geq (2n - T)/2 - \varepsilon n$. So $\bar{T} \leq 2n - \frac{3}{4}(n - T/2 - 2\varepsilon n) < 1.25n + 0.375T + \delta n$. Since T is the cost of $(1,2)$-TSP of G, which is at least n, we have $\bar{T} \leq (1.625 + \delta)T$. ∎

▶ Remark 3.6. The algorithm can be generalized to insertion-only streaming model. In insertion-only streaming model, we can compute a maximal matching pair as follows: we set M_1 and M_2 as empty set before the stream. Whenever an edge e comes, we first check if there is an edge in M_1 that shares an endpoint with e. If not, then we add e into M_1. Otherwise, we check if there is an edge in M_2 that shares and endpoint with e. If not, then we add e into M_2. So we get an algorithm that only uses $O(n)$ space to compute a maximal matching pair. We have the following corollary.

▶ **Corollary 3.7.** *There is an insertion-only streaming algorithm that estimates the cost of $(1,2)$-TSP of a graph G within a factor of 1.625 using $O(n)$ space.*

4 An $\Omega(n^2)$ Query Lower Bound for Approximation Schemes

In this section, we prove that there exists an $\varepsilon_0 > 0$, such that any query algorithm for graphic or $(1,2)$-TSP that returns a $(1+\varepsilon_0)$-approximate estimate of optimal cost, requires $\Omega(n^2)$ queries. In order to prove this, we design a new query model for the 3SAT problem and show an $\Omega(n^2)$ query lower bound for 3SAT in this model. We then use a reduction from 3SAT to $(1,2)$-TSP in [27] to prove the lower bound for $(1,2)$-TSP; with some additional changes, we also get an identical lower bound for graphic TSP.

The idea of proving query lower bound for APX-hard problems by reduction from 3SAT is similar to the idea used in [7], and we follow their general approach. However, in [7], the authors study lower bounds for problems in sparse graphs and hence the query model uses only neighbor queries. So in their query model, the lower bound for 3SAT is $\Omega(n)$. In order to prove an $\Omega(n^2)$ query lower bound in the pair query model, we need to design a new query model for 3SAT.

In the 3SAT problem, we are given a 3CNF instance on n variables, and the goal is to estimate the largest fraction of clauses that can be satisfied by any assignment. The algorithm is allowed to perform only one kind of query: is a variable x present in a clause c? If the answer is yes, then the algorithm is given the full information about all variables that appear in the clause c. The proof of the next theorem is deferred to the full version of this paper.

▶ **Theorem 4.1.** *For any $\varepsilon > 0$, any algorithm that with probability at least $2/3$ distinguishes between satisfiable 3CNF instances and 3CNF instances where at most $(7/8+\varepsilon)$ fraction of clauses can be satisfied, needs $\Omega(n^2)$ queries.*

4.1 Reduction from 3SAT to $(1,2)$-TSP

We will utilize an additional property of the hard instances of 3SAT in Theorem 4.1, namely, each variable occurs the same constant number of times where the constant only depends on ε. We denote the number of variables by n, the number of clauses by m, and the number of occurrences of each variable by k; thus $m = kn/3$.

We use the reduction in [27] to reduce a 3SAT instance to a $(1,2)$-TSP instance. In this reduction, there is a gadget for each variable and for each clause. Each of these gadgets has size at most $L = \Theta(k^2)$. Thus the $(1,2)$-TSP contains N vertices where $N \leq L(n+m) = \frac{L(k+3)n}{3}$. Let G_{x_j} be the gadget of variable x_j and G_{c_i} be the gadget of clause c_i. There is a ground graph which is the same for each 3SAT instance. Each variable gadget is connected with the gadgets for clauses that contain that variable. The reduction satisfies the following property. If the 3SAT instance is satisfiable, then the $(1,2)$-TSP instance contains a Hamilton cycle supported only on the weight 1 edges. On the other hand, if at most $m-\ell$ clauses can be satisfied in the 3SAT instance, the $(1,2)$-TSP cost is at least $N + \lceil \ell/2 \rceil$. Thus there is a constant factor separation between the optimal $(1,2)$-TSP cost in the two cases. However, what remains to be shown is that any query algorithm for $(1,2)$-TSP can also be directly simulated on the underlying 3SAT instance with a similar number of queries. The theorem below now follows by establishing this simulation.

▶ **Theorem 4.2.** *There is a constant ε_0 such that any algorithm that approximates the $(1,2)$-TSP cost to within a factor of $(1+\varepsilon_0)$ needs $\Omega(n^2)$ queries.*

Proof. We consider the following stronger queries for $(1,2)$-TSP: for any query (u,v), if u is in a vertex gadget G_{x_j} and v is in a clause gadget G_{c_i} (or vice versa) and x_j occurs in c_i in

the 3SAT instance, then the algorithm is given all the edges incident on G_{c_i}. Otherwise the algorithm just learns if the there is an edge between u and v.

Let $\varepsilon = 1/16$, and let the values of k, L and N correspond to this choice for ε according to the redution in Section 4.1. Let $\varepsilon_0 = \frac{k}{32(k+3)L}$. Consider the $(1,2)$-TSP instance reduced from the 3SAT instance generated by the hard distribution in Theorem 4.1 with $\varepsilon = 1/16$. If the 3SAT instance is perfectly satisfiable, then the $(1,2)$-TSP instance has a Hamilton cycle of cost N. If the 3SAT instance satisfies at most $(15/16)$-fraction of clauses, then each Hamilton cycle in the $(1,2)$-TSP instance has cost at least

$$N + (1/8 - \varepsilon)m/2 = N + (1/8 - \varepsilon)kn/6 \geq (1 + \frac{(1/8 - \varepsilon)k}{2(k+3)L})N = (1 + \varepsilon_0)N$$

For any query (u, v) in the $(1,2)$-TSP instance, we can simulate it by at most one query in the corresponding 3SAT instance as follows: if u is in a vertex gadget G_{x_j} and v is in a clause gadget G_{c_i} (or vice versa), then we make a query of x_j and c_i in the 3SAT instance. If the 3SAT query returns YES and the full information of c_i, then we return all the edges incident on G_{c_i} according to the reduction rule and the full information of c_i. If the 3SAT query returns NO or (u, v) are not in a vertex gadget and a clause gadget respectively, we return YES if (u, v) is an edge in the ground graph and NO otherwise.

By Theorem 4.1, any algorithm that distinguishes a perfectly satisfiable 3SAT instance from an instance where at most $(15/16)$-fraction of the clauses can be satisfied needs $\Omega(n^2)$ queries. So any algorithm that distinguishes a $(1,2)$-TSP instance containing a Hamilton cycle of length N from an instance that has minimum Hamilton cycle of cost $(1+\varepsilon_0)N$ needs $\Omega(n^2)$ queries.

4.2 $\Omega(n^2)$ Lower Bound for Graphic TSP

We can reduce an instance of $(1,2)$-TSP to an instance of graphic TSP by adding a new vertex that is adjacent to all other vertices. By doing so, any pair of vertices in the new graph has a distance at most 2. On the other hand, the cost of graphic TSP in the new graph differs by at most 1 from the cost of $(1,2)$-TSP in the old graph. So the $\Omega(n^2)$ query lower bound for $(1,2)$-TSP also holds for the graphic TSP problem.

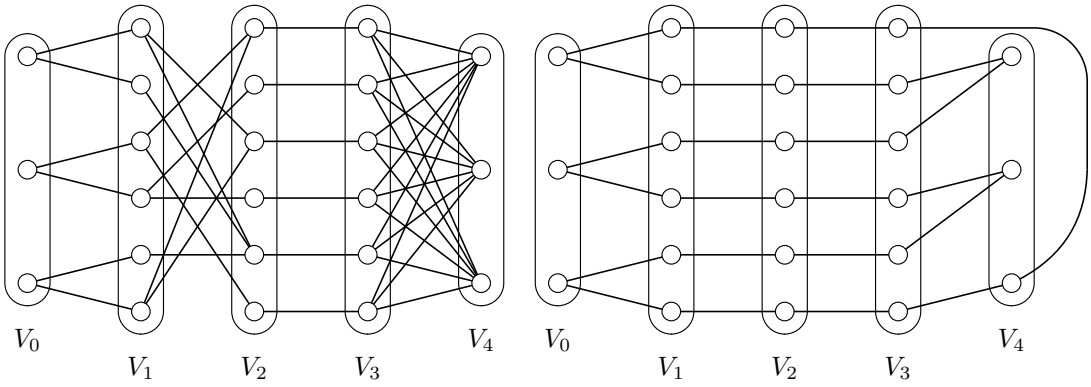
5 A Reduction from Matching Size to TSP Cost Estimation

In this section, we give a reduction from the problem of estimating the maximum matching size in a bipartite graph to the problem of estimating the optimal $(1,2)$-TSP cost. An essentially identical reduction works for graphic TSP cost using the idea described in Section 4.2.

We will denote the size of the largest matching in a graph G by $\alpha(G)$. Given a bipartite graph $G(V, E)$ with n vertices on each side, we construct an instance $G'(V', E')$ of the $(1,2)$-TSP problem on $4n$ vertices such that the optimal TSP cost on G' is $5n - \alpha(G)$. Thus for any $\varepsilon \in [0, 1/5)$, any algorithm that can estimate $(1,2)$-TSP cost to within a $(1+\varepsilon)$-factor, also gives us an estimate of the matching size in G to within an additive error of $5\varepsilon n$.

We will now describe our construction of the graph G'. For clarity of exposition, we will describe G' as the graph that contains edges of cost 1 – all other edges have cost 2. Suppose the vertex set V of G consists of the bipartition $V_1 = \{v_1^1, v_2^1, \ldots, v_n^1\}$ and $V_2 = \{v_1^2, v_2^2, \ldots, v_n^2\}$. We construct the graph G' as follows: we start with the graph G, then add three sets of vertices V_0, V_3 and V_4, such that $V_0 = \{v_1^0, v_2^0, \ldots, v_{n/2}^0\}$ with $n/2$ vertices, $V_3 = \{v_1^3, v_2^3, \ldots, v_n^3\}$ with n vertices, and $V_4 = \{v_1^4, v_2^4, \ldots, v_{n/2}^4\}$ with $n/2$ vertices. The graph G' will only have edges between V_j and V_{j+1} $(j = \{0, 1, 2, 3\})$. We will denote the set

of edges between V_j and V_{j+1} as $E_{j,j+1}$. For any vertex $v_i^0 \in V_0$, it connects to v_{2i-1}^1 and v_{2i}^1 in V_1. $E_{1,2}$ has the same edges as the edges in G. Each vertex $v_i^2 \in V_2$ is connected to vertex v_i^3 in V_3, that is, vertices in V_2 and V_3 induce a perfect matching (identity matching). Finally, each vertex in V_3 is connected to all the vertices in V_4. See Figure 1(a) for an illustration.



(a) The illustration of G'.

(b) The illustration of tour T, where V_2 and V_3 are arranged with order $(v_{f(1)}^2, \ldots, v_{f(6)}^2)$ and $(v_{f(1)}^3, \ldots, v_{f(6)}^3)$.

Figure 1 An illustration of the reduction for $n = 6$.

The lemmas below relate matching size in G to $(1,2)$-TSP cost in G'.

▶ **Lemma 5.1.** *Let M be any matching in G. Then there is a $(1,2)$-TSP tour T in G' of cost at most $5n - |M|$.*

Proof. Let $f : [n] \to [n]$ be any bijection from $[n]$ to $[n]$ such that whenever a vertex v_i^1 is matched to a vertex v_j^2 in M, then $f(i) = j$. Consider the following $(1,2)$-TSP tour T: each vertex $v_i^0 \in V_0$ connects to v_{2i-1}^1 and v_{2i}^1 in T; each vertex $v_i^1 \in V_1$ connects to $v_{\lceil (i+1)/2 \rceil}^0$ and $v_{f(i)}^2$ in T. For any $v_{f(i)}^2 \in V_2$, it connects to v_i^1 and $v_{f(i)}^3$ in T. For any vertex $v_{f(i)}^3 \in V_3$, if $i > 1$, it connects to $v_{f(i)}^2$ and $v_{\lceil i/2 \rceil}^4$ in T; if $i = 1$, it connects to $v_{f(i)}^2$ and $v_{n/2}^4$ in T. See Figure 1(b) as an illustration. T is clearly a TSP-tour.

All edges in T are also edges in G' except for possibly some edges between V_1 and V_2. If v_i^1 is matched in M, then $(v_i^1, v_{f(i)}^2)$ is an edge in G', otherwise it is not in G' and thus has weight 2. So T only has $n - |M|$ weight 2 edges, which means T has cost at most $4n + n - |M| = 5n - |M|$. ∎

▶ **Lemma 5.2.** *For any $(1,2)$-TSP tour T in G, T has cost at least $5n - \alpha(G)$.*

To prove Lemma 5.2, we first prove an auxiliary claim.

▷ Claim 5.3. Suppose $G = (V_1, V_2, E)$ is a bipartite graph which has maximum size $\alpha(G)$. For any 2-degree subgraph H of G, if there are at most X vertices in V_1 has degree 2 in H, then there are at most $\alpha(G) + X$ vertices in V_2 which have degree at least 1 in H. Similarly, if there are at most X vertices in V_2 has degree 2 in H, then there are at most $\alpha(G) + X$ vertices in V_1 which have degree at least 1 in H.

Proof. If there are at most X vertices in V_1 has degree 2 in H. We construct H' by deleting an arbitrary edge on each degree 2 vertex in V_1, then construct H'' by deleing an arbitrary edge on each degree 2 vertex in V_2. Since H'' does not have degree two vertex, it is a matching of G. So the number of degree 1 vertices in V_2 in H'' is at most $\alpha(G)$. On the

other hand, any vertex in V_2 which has degree at least 1 in H' also has degree 1 in H''. So there are at most $\alpha(G)$ vertices of degree at least 1 in V_2 in H'. Furthermore, since there are only X vertices of degree 2 in V_1 in H, we delete at most X edges in H when constructing H'. So H' has at most X more isolate vertices in V_2 than in H, which means H has at most $\alpha(G) + X$ vertices with degree at least 1 in V_2.

The second part of the claim follows via a similar argument as the first part of the claim. ∎

Proof of Lemma 5.2. Let a_{01} be the number of edges in $T \cap E_{0,1}$, $a_{2,3}$ be the number of edges in $T \cap E_{3,4}$. Let G^X be the intersection graph of G and T. Since the vertices in V_0 only connect to the vertices in V_1 in G', and any vertex in T has degree 2, there are at least $n - a_{01}$ edges incident on V_0 in T are not an edge in G'. On the other hand, since any vertex in V_1 is incident on at at most 1 edge in $E_{0,1}$, there are at least a_{01} vertices in V_1 is connected to a vertex in V_0 in T, which means there are at most $n - a_{01}$ vertices in V_1 has degree 2 in G^X. By Claim 5.3, there are at most $n - a_{01} + \alpha(G)$ vertices in V_2 has edge in G^X. For any isolate vertex in V_2 in T, it has only one edge in G' connecting to V_3, so this vertex must incident on an edge in T which is not in G'. So there are at least $n - (n - a_{01} + \alpha(G)) = \alpha(G) - a_{01}$ edges incident on V_2 in T which is not in G'.

There are $2n$ edges incident on V_3 in T, but among them, there are only a_{23} edges between V_2 and V_3 which is also in G', and there are at most n edges between V_3 and V_4 in T since each vertex has degree only 2. So there are at least $2n - n - a_{23} = n - a_{23}$ edges incident on V_3 which is not in G'. On the other hand, since any vertex in V_2 is incident on at at most 1 edge in $E_{2,3}$, there are at least a_{23} vertices in V_2 is connected to a vertex in V_3 in T, which means there are at most $n - a_{23}$ vertices in V_2 has degree 2 in G^X. By Claim 5.3, there are at most $n - a_{23} + \alpha(G)$ vertices in V_1 has edge in G^X. For any isolate vertex in V_1 in T, it has only one edge in G' connecting to V_0, so this vertex must incident on an edge in T which is not in G'. So there are at least $n - (n - a_{23} + \alpha(G)) = a_{23} - \alpha(G)$ edges incident on V_1 in T which is not in G'.

Since any edge has two endpoints, the number of edges in T but not in G' is at least $((n - a_{01}) + (a_{01} - \alpha(G)) + (n - a_{23}) + (a_{23} - \alpha(G)))/2 = n - \alpha(G)$, which means T has cost at least $4n + n - \alpha(G) = 5n - \alpha(G)$. ∎

▶ **Corollary 5.4.** *For any $\varepsilon \in [0, 1/5)$, any algorithm that can estimate $(1,2)$-TSP cost to within a $(1+\varepsilon)$-factor, can be used to estimate the size of a largest matching in a bipartite graph G on $2n$ vertices to within an additive error of $5\varepsilon n$.*

Proof. We use the reduction above to construct a $(1,2)$-TSP instance G' on $4n$ vertices. By Lemmas 5.1 and 5.2, the optimal TSP cost for G' is $5n - \alpha(G)$. We now run the $(1+\varepsilon)$-approximation algorithm for $(1,2)$-TSP on graph G' (note that the reduction can be simulated in each of neighbor query model, pair query model, and the streaming model without altering the asymptotic number of queries used). Suppose the output is C which satisfies $(1-\varepsilon)(5n - \alpha(G)) \leq C \leq (1+\varepsilon)(5n - \alpha(G))$, which means $5n - \alpha(G) - 5\varepsilon n < C < 5n - \alpha(G) + 5\varepsilon n$. Let $\hat{\alpha} = 5n - C$, we have $\alpha(G) - 5\varepsilon n < \hat{\alpha} < \alpha(G) + 5\varepsilon n$. ∎

Acknowledgements

We thank the anonymous reviewers for their valuable feedback. This work was supported in part by NSF awards CCF-1617851, CCF-1733794, CCF-1763307, CCF-1763514, and CCF-1934876.

References

1. `https://sublinear.info/71`.
2. Anna Adamaszek, Matthias Mnich, and Katarzyna Paluch. New approximation algorithms for (1, 2)-tsp. In *45th International Colloquium on Automata, Languages, and Programming (ICALP 2018)*. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2018.
3. Hyung-Chan An, Robert D. Kleinberg, and David B. Shmoys. Improving christofides' algorithm for the s-t path TSP. *J. ACM*, 62(5):34:1–34:28, 2015.
4. Sepehr Assadi, Yu Chen, and Sanjeev Khanna. Sublinear algorithms for (Δ+ 1) vertex coloring. In *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 767–786. Society for Industrial and Applied Mathematics, 2019.
5. Sepehr Assadi, Sanjeev Khanna, and Yang Li. On estimating maximum matching size in graph streams. In *Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1723–1742. SIAM, 2017.
6. Piotr Berman and Marek Karpinski. 8/7-approximation algorithm for (1, 2)-tsp. In *Proceedings of the seventeenth annual ACM-SIAM symposium on Discrete algorithm*, pages 641–648. Society for Industrial and Applied Mathematics, 2006.
7. Andrej Bogdanov, Kenji Obata, and Luca Trevisan. A lower bound for testing 3-colorability in bounded-degree graphs. In *The 43rd Annual IEEE Symposium on Foundations of Computer Science, 2002. Proceedings.*, pages 93–102. IEEE, 2002.
8. Bernard Chazelle, Ronitt Rubinfeld, and Luca Trevisan. Approximating the minimum spanning tree weight in sublinear time. *SIAM J. Comput.*, 34(6):1370–1379, 2005.
9. Chandra Chekuri and Kent Quanrud. Approximating the held-karp bound for metric TSP in nearly-linear time. In *58th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2017, Berkeley, CA, USA, October 15-17, 2017*, pages 789–800, 2017.
10. Chandra Chekuri and Kent Quanrud. Fast approximations for metric-TSP via linear programming. *CoRR*, abs/1802.01242, 2018.
11. 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.
12. Gabriel Andrew Dirac. Some theorems on abstract graphs. *Proceedings of the London Mathematical Society*, 3(1):69–81, 1952.
13. Zhihan Gao. On the metric s-t path traveling salesman problem. *SIAM Review*, 60(2):409–426, 2018.
14. Oded Goldreich and Dana Ron. Property testing in bounded degree graphs. *Algorithmica*, 32(2):302–343, 2002.
15. VL Goncharov. Some facts from combinatorics. *Izvestia Akad. Nauk. SSSR, Ser. Mat*, 8:3–48, 1944.
16. Johan Håstad. Some optimal inapproximability results. *Journal of the ACM (JACM)*, 48(4):798–859, 2001.
17. Michael Kapralov, Slobodan Mitrović, Ashkan Norouzi-Fard, and Jakab Tardos. Space efficient approximation to maximum matching size from uniform edge samples. In *Proceedings of the Fourteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1753–1772. SIAM, 2020.
18. Marek Karpinski, Michael Lampis, and Richard Schmied. New inapproximability bounds for TSP. *J. Comput. Syst. Sci.*, 81(8):1665–1677, 2015.
19. Ilan Kremer, Noam Nisan, and Dana Ron. On randomized one-round communication complexity. *Computational Complexity*, 8(1):21–49, 1999.
20. Matthias Mnich and Tobias Mömke. Improved integrality gap upper bounds for traveling salesperson problems with distances one and two. *European Journal of Operational Research*, 266(2):436–457, 2018.
21. Tobias Mömke and Ola Svensson. Approximating graphic TSP by matchings. In *IEEE 52nd Annual Symposium on Foundations of Computer Science, FOCS 2011, Palm Springs, CA, USA,*

October 22-25, 2011, pages 560–569, 2011. URL: https://doi.org/10.1109/FOCS.2011.56, doi:10.1109/FOCS.2011.56.

22 Tobias Mömke and Ola Svensson. Removing and adding edges for the traveling salesman problem. *Journal of the ACM (JACM)*, 63(1):2, 2016.

23 Marcin Mucha. $\frac{13}{9}$-approximation for graphic tsp. *Theory of computing systems*, 55(4):640–657, 2014.

24 Huy N. Nguyen and Krzysztof Onak. Constant-time approximation algorithms via local improvements. In *49th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2008, October 25-28, 2008, Philadelphia, PA, USA*, pages 327–336, 2008.

25 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.

26 Krzysztof Onak, Dana Ron, Michal Rosen, and Ronitt Rubinfeld. Personal communication, 2019.

27 Christos H Papadimitriou and Mihalis Yannakakis. The traveling salesman problem with distances one and two. *Mathematics of Operations Research*, 18(1):1–11, 1993.

28 Michal Parnas and Dana Ron. Approximating the minimum vertex cover in sublinear time and a connection to distributed algorithms. *Theor. Comput. Sci.*, 381(1-3):183–196, 2007.

29 András Sebö and Anke van Zuylen. The salesman's improved paths: A 3/2+1/34 approximation. In *IEEE 57th Annual Symposium on Foundations of Computer Science, FOCS 2016, 9-11 October 2016, Hyatt Regency, New Brunswick, New Jersey, USA*, pages 118–127, 2016.

30 András Sebö and Jens Vygen. Shorter tours by nicer ears: 7/5-approximation for the graph-tsp, 3/2 for the path version, and 4/3 for two-edge-connected subgraphs. *Combinatorica*, 34(5):597–629, 2014.

31 Vera Traub and Jens Vygen. Beating the integrality ratio for s-t-tours in graphs. In *59th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2018, Paris, France, October 7-9, 2018*, pages 766–777, 2018.

32 Vera Traub and Jens Vygen. Approaching 3/2 for the s-t-path TSP. *J. ACM*, 66(2):14:1–14:17, 2019. URL: https://doi.org/10.1145/3309715, doi:10.1145/3309715.

33 Jens Vygen. New approximation algorithms for the tsp. 2012.

34 Douglas B. West. *Introduction to graph theory*. Prentice-Hall Inc., 1996.

35 Andrew Chi-Chih Yao. Lower bounds to randomized algorithms for graph properties (extended abstract). In *28th Annual Symposium on Foundations of Computer Science, Los Angeles, California, USA, 27-29 October 1987*, pages 393–400, 1987.

36 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.