Streaming Algorithms for Estimating the Matching Size in Planar Graphs and Beyond

Hossein Esfandiari [*†] Mohammad T Hajiaghayi [*†] Vahid Liaghat [*†]
Morteza Monemizadeh [‡†] Krzysztof Onak [§]

Abstract

We consider the problem of estimating the size of a maximum matching when the edges are revealed in a streaming fashion. Consider a graph $G = (V, E)$ with n vertices and m edges. The *input stream* is a permutation of edges $S = \langle e_1, \ldots, e_m \rangle$ chosen by an adversary. The goal is to output an estimation of the size of a maximum matching. The algorithm is only allowed to use a small amount of memory (much smaller than n).

When the underlying graph is planar, we present a simple and elegant streaming algorithm that with high probability estimates the size of a maximum matching within a constant factor using $\tilde{\mathcal{O}}(n^{2/3})$ space. The approach generalizes to the family of graphs that have bounded arboricity. Graphs with bounded arboricity include, among other families of graphs, graphs with an excluded constant-size minor. To the best of our knowledge, this is the first result for estimating the size of a maximum matching in the *adversarial-order* streaming model (as opposed to the random-order streaming model). We circumvent the barriers inherent in the adversarial-order model by exploiting several structural properties of planar graphs, and more generally, graphs with bounded arboricity. We hope that this approach finds applications in estimating other properties of graphs in the adversarial-order streaming model. We further reduce the required memory size to $\tilde{\mathcal{O}}(\sqrt{n})$ for three restricted settings: (i) when the underlying graph is a forest; (ii) when we have 2-passes over the stream of edges of a graph with bounded arboricity; and (iii) when the edges arrive in random order and the underlying graph has bounded arboricity.

Finally, by introducing a communication complexity problem, we show that the approximation factor of a deterministic algorithm cannot be better than a constant using $o(n)$ space, even if the underlying graph is a collection of paths. We can show that under a plausible conjecture for the hardness of the communication complexity problem, randomized algorithms with $o(\sqrt{n})$ space cannot have an approximation factor better than a fixed constant.

[*]University of Maryland, email: {hossein,hajiagha,vliaghat}@cs.umd.edu.

[†]Supported in part by NSF CAREER award 1053605, NSF grant CCF-1161626, ONR YIP award N000141110662, DARPA/AFOSR grant FA9550-12-1-0423.

[‡]University of Frankfurt, email: monemizadeh@em.uni-frankfurt.de, Supported in part by MO 2200/1-1

[§]IBM TJ Watson, email: krzysztof@onak.pl

1 Introduction

As noted by Lovasz and Plummer in their classic book [20],"*Matching Theory* is a central part of graph theory, not only because of its applications, but also because it is the source of important ideas developed during the rapid growth of combinatorics during the last several decades." In the classical offline model, where we assume we have enough space to store all vertices and edges of a graph $G = (V, E)$, the problem of computing the maximum matching of G has been extensively studied. The best result in this model is the 30-years-old algorithm due to Micali and Vazirani [23] with running time $\mathcal{O}(m\sqrt{n})$, where $n = |V|$ and $m = |E|$.

In contrast, in the *streaming* model, the algorithm has access to a sequence of edges, called a *stream*. The algorithm reads edges in the order in which they appear in the stream. The main goal is to design an algorithm that solves a given problem, using as little space as possible. In particular, we wish that the amount of space be sublinear in the size of the input.

Note that the size of a maximum matching in a graph can be as large as $\Omega(n)$. Hence there is little hope to solve the problem exactly or even with a relatively good approximation in $o(n)$ space, since the algorithm has to remember the labels of endpoints for each edge in the matching. This and similar constraints with other graph problems were the main motivation behind the *semi-streaming* model introduced by Feigenbaum, Kannan, McGregor, Suri, and Zhang [7]. In this model, the streaming algorithm is allowed to use $\tilde{\mathcal{O}}(n)$ space (the $\tilde{\mathcal{O}}$ notation hides logarithmic dependencies). The semi-streaming model allows for finding a *maximal* matching (a 2-approximation for the maximum matching) using $\tilde{\mathcal{O}}(n)$ space in a greedy manner. For every new edge (u, v), we add it to the current matching if both u and v are unmatched; otherwise we discard it.

Unfortunately, in many real-world applications where the data is modeled by a massive graph, we may not be able to store all vertices in main memory. Hence we investigate the following natural question: *Given the stream of edges of a massive graph G, how well can the maximum matching size be approximated in $o(n)$ space?* We stress again that we focus on approximating the *size* of the maximum matching, not computing a large matching, which may be difficult.

The problem of estimating the size of a maximum matching of a graph has been studied relatively well in the case of *sublinear-time algorithms* [30, 25, 31, 13, 26]. Surprisingly, very little is known about this problem in the streaming model which is one of the most fundamental models of the area of algorithms for big data. The only known result is a recent algorithm by Kapralov, Khanna, and Sudan [16], which computes an estimate within a factor of $\mathcal{O}(\text{polylog}(n))$ in the *random-order* streaming model using $\mathcal{O}(\text{polylog}(n))$ space. In the random-order model, the input stream is assumed to be chosen uniformly at random from the set of all possible permutations of the edges. However, to the best of our knowledge, if the algorithm is required to provide a good approximation with high constant probability for any ordering of edges (we refer to this setting as the *adversarial-order* model), nothing is known for the problem of estimating the size of a maximum matching using $o(n)$ space.

1.1 Our Results

In this paper, we mainly focus on the family of graphs with bounded arboricity. A graph $G = (V, E)$ has arboricity c if

$$c = \max_{U \subseteq V} \left\lceil \frac{|E(U)|}{|U| - 1} \right\rceil,$$

where $E(U)$ is the subset of edges with both endpoints in U.[1] Several important families of graphs have constant arboricity. Examples include planar graphs, bounded genus graphs, bounded

[1] Equivalently, the arboricity of a graph can be defined as the minimum number of forests into which its edges can be partitioned.

treewidth graphs, and more generally, graphs that exclude a fixed minor.[2]

The main result of our paper is a simple and elegant streaming algorithm with a constant approximation factor in the adversarial-model for estimating the size of a maximum matching in graphs with bounded arboricity using $\tilde{\mathcal{O}}(n^{2/3})$ space (see Theorem 1.2). The problem is non-trivial even when the underlying graph is a tree. For the case of trees (or forests), we give a $2(1+\epsilon)$-approximation algorithm that uses $\tilde{\mathcal{O}}(\sqrt{n})$ space, where ϵ is an arbitrarily small constant.

We show that under a plausible communication complexity conjecture, no algorithm with $\tilde{o}(\sqrt{n})$ space can estimate the size of a maximum matching in trees within a factor better than 1.5 (see Section 4 for a formal discussion of the matter).

1.2 Unsuccessful Approaches

As mentioned before, little is known about the adversarial-order version of the problem. Before sketching our main ideas, we briefly mention main difficulties in applying successful techniques from related areas of algorithms. In the area of sublinear-time algorithms [30, 25, 31, 13, 26], the technique of choice has been *local exploration*. A number of dynamic algorithms for matchings [14, 27, 2, 24, 10] have applied various *partitioning* techniques. The polylog(n)-approximation algorithm for random-order streams [16] employs a combination of both partitioning and exploration.

Local exploration is applied by sublinear-time algorithms for the maximum matching size. They locally apply greedy techniques for constructing a good matching. Given random access to a graph $G = (V, E)$ with degree bounded by d, they sample a small number of vertices and explore their local neighborhoods using breadth-first search to bounded depth. Using the information gathered from the exploration, they estimate the fraction of vertices in a locally constructed large matching. By exploring this approach one can obtain an algorithm that approximates the maximum matching size within an additive error of $\epsilon \cdot n$ in time $d^{\mathcal{O}(1/\epsilon^2)}$ [31].

The partitioning approach used by some dynamic algorithms for maintaining a large matching [27, 2] is based on a hierarchical decomposition of the vertex set into a logarithmic number of layers. Depending on a specific approach taken, the decomposition can be a complicated function of the graph structure and the history of operations on the graph. In general, high degree vertices tend to be included in higher layers and low degree vertices tend to be included in lower layers. The exact placement of a vertex depends, however, on its connections to other vertices. Equipped with additional information, such a decomposition allows for easy access to a large matching, which is a result of combining together large matchings involving each layer.

Kapralov, Khanna, and Sudan [16] show that one can get $\mathcal{O}(\text{polylog}(n))$-approximation to the size of a maximum matching in the *random-order* model, using $\mathcal{O}(\text{polylog}(n))$ space. The intuition behind their approach can be interpreted as a combination of the local exploration and partitioning approaches. In the random-order variant, for any arbitrary edge e, we may assume that a large fraction of edges adjacent[3] to e comes after e. Because of that one can get samples from neighborhoods of vertices and edges in the graph. This turns out to be enough to construct a recursive exploration procedure for estimating the number of vertices in a specific layer of the decomposition.

These approaches do not seem to be effective in the adversarial-order model. Local exploration may be made difficult by placing edges in order unsuitable for following sequences of edges. This idea is used in two works [8, 11] to show lower bounds for the amount of space necessary for building a BFS tree from a specific node and for verifying if two nodes are at small distance, even with a few passes over the stream allowed.

[2] It can be shown that for an H-minor-free graph, the arboricity number is $\mathcal{O}(h\sqrt{h})$ where h is the number of vertices of H. [19]

[3] Two edges are adjacent if they share an endpoint.

The partitioning approach is also difficult to implement due to the relations between vertices. Identifying them also require some amount of exploration. For instance, the algorithm of Kapralov et al. [16] for random-order streams uses a recursive procedure for determining which layer a given vertex belongs to. Executing the procedure requires exploration from low-degree to high-degree vertices and sampling random neighbors of each visited vertex. In a random-order stream, after reaching a high-degree vertex, the algorithm is still likely to see sufficiently many incident edges. This is not guaranteed for adversarial-order streams.

1.3 Our Techniques

Our approach consists of two main parts. As discussed before, it is inherently hard to probe a certain neighborhood of the graph in the adversarial-order streaming model. Therefore in the first part, we present combinatorial parameters of bounded-arboricity graphs (including graphs that exclude a fixed-minor) that (i) provide a constant approximation for the size of a maximum matching; and more crucially (ii) computing the parameters do not involve a neighborhood search. In the second part, we design estimators for these parameters in the streaming model by sampling subgraphs with certain properties and carefully bounding the positive correlation between the samples. We hope that this simple approach together with the structural properties provided in this paper may help in estimating other graph properties in the streaming model as well.

For a vertex $v \in V$, let $\deg(v)$ denote the degree of vertex v in G. Let β_G denote an upper bound on the *average* degree of *every* subgraph of G. Recall that for planar graphs $\beta_G \leq 6$. In fact, if the arboricity of a graph is ν, it is easy to verify that $\beta_G \leq 2\nu$. Thus for the family of graphs with constant arboricity (including graphs that exclude a fixed minor), we may assume that β_G is constant. We us the following intuition behind the structural properties. Suppose we sample a small subset of vertices $V' \subseteq V$. Let $G[V']$ denote the subgraph induced by V'. If the maximum degree of V' is constant, then every edge is adjacent to at most a constant number of other edges. Thus the size of a maximum matching in $G[V']$ can be in fact approximated by simply the number of edges in $G[V']$. On the other hand, if the degrees of vertices are large, most of the edges fall between V' and $\overline{V'}$, since the subgraph $G[V']$ is sparse. Therefore in this case one may hope to find a large matching between V' and $\overline{V'}$ that covers most of V'. Hence the size of such a matching can be approximated by $|V'|$. This intuition can be formalized as follows.

Let $\mu = \lceil \beta \rceil + 3$ denote a constant threshold. A vertex v is *light*, if $\deg(v) \leq \mu$; otherwise, the vertex is *heavy*. An edge is *shallow*, if both of its endpoints are light. Throughout the paper, let h_G and s_G denote the number of heavy vertices and the number of shallow edges in G, respectively. We may drop the index G when it is clear from the context. The crux of our algorithm is the following two-fold structural property of graphs.

Lemma 1.1. *Let G be a graph with maximum matching size M^*. We have the following bounds:*

- *Upper bound: $M^* \leq h_G + s_G$; and*
- *Lower bound: $M^* \geq \frac{\max\{h_G, s_G\}}{\eta}$*

where $\eta = 1.25\mu + 0.75$. For graphs with bounded arboricity ν, the constant μ is at most $\lceil 2\nu + 3 \rceil$.

It is not hard to verify the upper bound of Lemma 1.1. Observe that every edge of the maximum matching is either shallow, or saturates at least one heavy vertex. Thus the size of a matching cannot be more than the combined number of shallow edges and heavy vertices. On the other hand, proving the lower bound turns out to be quite non-trivial. In Section 3, we use the sparsity of graphs with bounded arboricity together with an extension of Hall's Theorem to prove the lower bound on M^*.

By Lemma 1.1, estimating h_G and s_G leads to a constant-factor approximation for the size of a maximum matching. Thus, for the second part of our work, we use random sampling to estimate these parameters. For estimating h_G, we sample a set of vertices and we find the number of heavy vertices among them. We show that a sample of size $\mathcal{O}(\sqrt{n})$ is enough for estimating h_G. For estimating s_G, a major difficulty is that in the adversarial-order setting one does not hope to maintain information about independent (or negatively correlated) samples of edges: when an edge arrives, we may have already seen all the edges adjacent to it! Therefore instead of sampling edges, we sample a set of vertices V' and maintain the *shallow-subgraph* induced by V'. We then show that although the probability of sampling edges is positively correlated, the degree of dependency is *constant* and thus the variance can be bounded; showing that the output of the estimator is highly concentrated. However, to obtain a a good estimator for s_G, we need to maintain a shallow-subgraph of size $\mathcal{O}(n^{2/3})$, which is indeed the space bottleneck.

Theorem 1.2. *Let G be a graph with arboricity ν and $n = \omega(\nu^2)$ vertices. Let $\epsilon, \delta \in (0, 1]$ be two arbitrary positive values less than one. With probability at least $(1-\delta)$, our algorithm estimates the size of the maximum matching in G within a $((5\nu+9)(1+\epsilon)^2)$-factor in the streaming model using $\tilde{\mathcal{O}}(\nu\epsilon^{-2}\log\left(\delta^{-1}\right)n^{2/3})$ space. Both the update time and final processing time are $\mathcal{O}(\log(\delta^{-1}))$.*

In particular, for planar graphs, by choosing $\delta = n^{-1}$ and ϵ as a small constant, the output of our algorithm is within 25-approximation of the size of the maximum matching with probability at least $1 - \frac{1}{n}$ using at most $\tilde{\mathcal{O}}(n^{2/3})$ space.

We further study the limits of the approximation factor by considering the simple but still non-trivial case when the graph is a tree, or more generally, a forest with no isolated vertex. We improve the approximation factor of our general result to (roughly) 2 for the special case of trees while reducing the required memory size to $\tilde{\mathcal{O}}(\sqrt{n})$. In fact, in Section 4 we show strong evidence that further improvement may not be possible: under a plausible conjecture for the hardness of a communication complexity problem, using $\tilde{o}(\sqrt{n})$ space one cannot estimate the size of a maximum matching in trees with an approximation factor better than 1.5. We believe that the communication problem introduced in Section 4 may be of its own interest in proving lower bounds for other estimation problems.

Theorem 1.3. *Let F be a forest with no isolated vertices. Let $\delta \in (0, 1]$ and $\epsilon \in (0, 1/5]$ be arbitrary. With probability at least $(1-\delta)$, our algorithm estimates the maximum matching size in F within a factor of $2(1+3\epsilon)$ in the streaming model using $\tilde{\mathcal{O}}(\epsilon^{-2}\log\left(\delta^{-1}\right)\sqrt{n})$ space. Both the update time and final processing time are $\mathcal{O}(\log(\delta^{-1}))$.*

Improving to $\tilde{\mathcal{O}}(\sqrt{n})$ space. We further show that by relaxing the streaming model, one can improve the required space of our algorithm. In particular, given either (i) a *random-order stream* ; or (ii) *two passes over an adversarial-order stream*, we can $(1+\epsilon)$-approximate the number of shallow edges *using only* $\tilde{\mathcal{O}}(\sqrt{n})$ space. This in turn gives $\tilde{\mathcal{O}}(\sqrt{n})$-space algorithm that approximates the size of the maximum matching within $\mathcal{O}(\beta)$ in these models.

In the 2-pass streaming model, we first sample $\tilde{\mathcal{O}}(\sqrt{n})$ edges uniformly at random. In the second pass, we extract the set of shallow edges out of this sample set, leading to an estimation for s_G. If the number of shallow edges is at least $\tilde{\Omega}(\sqrt{n})$, this estimator gives a $(1+\epsilon)$-approximation factor. Otherwise if the number of shallow edges is small, one can argue that maintaining a small maximal matching and estimating h_G is enough to obtain a constant factor approximation factor for the size of the maximum matching. See Section D.1 for a detailed discussion. For a (one-pass) random-order stream, we can indeed use a similar technique. The idea is that the first $\tilde{\mathcal{O}}(\sqrt{n})$ edges of the stream is in fact a random sample set. Therefore we maintain the first $\tilde{\mathcal{O}}(\sqrt{n})$ edges

and we use the rest of the stream to find out how many of them are shallow. This again gives an estimation for the number of shallow edges which leads to an $\mathcal{O}(\beta)$-approximation factor (see Section D.2 for more details).

Table 1.3 summarizes the known results for estimating the size of a maximum matching.

Reference	Graph class	Stream ordering	Approximation factor	Space
Folklore	General	Adversarial	$\sqrt{n}$	$\mathcal{O}(\sqrt{n})$
Greedy	General	Adversarial	2	$\mathcal{O}(n)$
[16]	General	Random	$\mathcal{O}(\text{polylog}(n))$	$\mathcal{O}(\text{polylog}(n))$
This work	Planar	Adversarial	25	$\tilde{\mathcal{O}}(n^{2/3})$
This work	Tree	Adversarial	2	$\tilde{\mathcal{O}}(\sqrt{n})$
This work	Bounded Arboricity	Adversarial	$\mathcal{O}(1)$	$\tilde{\mathcal{O}}(n^{2/3})$
This work	Bounded Arboricity	Random	$\mathcal{O}(1)$	$\tilde{\mathcal{O}}(\sqrt{n})$
This work	Bounded Arboricity	2-Pass Adver.	$\mathcal{O}(1)$	$\tilde{\mathcal{O}}(\sqrt{n})$

Table 1: Known results for estimating the size of a maximum matching in data streams.

1.4 Further Related Streaming Work

The question of approximating the maximum cardinality matching has been extensively studied in the streaming model. An $\mathcal{O}(n)$-space greedy algorithm trivially obtains a maximal matching, which is a 2-approximation for the maximum cardinality matching [7]. A natural question is whether one can beat the approximation factor of the greedy algorithm with $\tilde{\mathcal{O}}(n)$ space. Very recently, it was shown that obtaining an approximation factor better than $\frac{e}{e-1} \simeq 1.58$ in one pass requires $n^{1+\Omega(1/\log\log n)}$ space [9, 15], even in bipartite graphs and in the *vertex-arrival* model.[4]

Closing the gap between the upper bound of 2 and the lower bound of $\frac{e}{e-1}$ remains one of the most appealing open problems in the graph streaming area (see [28]). The factor of 2 can be improved on if one either considers the random-order model or allows for two passes [18]. By allowing even more passes, the approximation factor can be improved to multiplicative $(1-\epsilon)$-approximation via finding and applying augmenting paths with successive passes [21, 22, 4, 5, 1].

Another line of research [7, 21, 32, 6] has explored the question of approximating the maximum-weight matching in one pass and $\tilde{\mathcal{O}}(n)$ space. Currently, the best known approximation factor equals $4.9108 + \epsilon$ (for any positive constant ϵ) [6].

1.5 Preliminaries

The Streaming Model. Let S be a stream of edges of an underlying graph $G = (V, E)$. We assume that the vertex set V is fixed and given, and that $|V| = n$. We assume that there is a unique numbering for the vertices in V so that we can treat $v \in V$ as a unique number v for $1 \leq v \leq n$. We denote an undirected edge in E with two endpoints $u, v \in V$ by (u, v). The graph G can have at most $\binom{n}{2} = n(n-1)/2$ edges. Thus, each edge can also be thought of as referring to a unique number between 1 and $\binom{n}{2}$.

[4] In the vertex-arrival model, the vertices arrive in the stream together with all their incident edges. This setting has also been studied in the context of *online algorithms*, where each arriving vertex has to be either matched or discarded irrevocably upon arrival. Seminal work due to Karp, Vazirani and Vazirani [17] gives an online algorithm with $\frac{e}{e-1}$ approximation factor in the online model.

Adversarial Order. We work in the most-popular *adversarial-order model*. In this model, a streaming algorithm has to compute a satisfying solution with satisfying probability for *every* ordering of items in the input stream. This should be contrasted with the easier *random-order model*, where an algorithm has to provide a satisfying solution with satisfying probability for an input stream permuted uniformly at random.

Approximation Factor. Throughout the paper, let M^* denote the *size* of a maximum matching in G. Our task is to output M, an estimate of M^*, after receiving all the edges. A (randomized) algorithm has *approximation factor* α with probability $(1-\delta)$ if for every permutation of input edges, with probability at least $(1-\delta)$, we have $M \leq M^* \leq \alpha M$.

2 Algorithm

To estimate the size of a maximum matching, by Lemma 1.1, it is sufficient to estimate the two parameters h and s, if one is willing to loose a constant (2η) in the approximation factor. Our algorithm consists of three parallel subroutines. Each subroutine reports an estimated lower bound for the size of a maximum matching. The joined result is the maximum of the three estimated values. Since the probabilities of success for the second and the third subroutines are small, we boost the success probabilities by independently running a logarithmic number of copies of these subroutines. The final output is the median of the values reported by all copies. In what follows we describe these subroutines (see Algorithm 1).

Since we are restricted to sublinear space, it is natural to use sampling for estimating h and s. However, the first obstacle occurs in the instances where both h and s are small. Thus we do not hope to capture a heavy vertex (or a shallow edge) using sublinear samples. We overcome these instances by maintaining a *maximal* matching of the graph, up to a sublinear size. Therefore for instances where $M^* = \tilde{\mathcal{O}}(n^{2/3})$, we can estimate M^* up to a factor of two (see Subroutine 1).

We now focus on the instances where M^* is relatively large, and thus we cannot keep a maximal matching in the memory. By the upper bound of Lemma 1.1, at least one of h or s is large. If the number of heavy vertices is large, we can estimate h using a *vertex-set sample*. We sample $\tilde{\mathcal{O}}(n^{2/3})$ random vertices and we maintain their degrees throughout the execution of the algorithm. We note that in a vertex-set sample, we hit roughly $\Omega(\frac{h}{n^{1/3}})$ heavy vertices. Therefore h would be proportional to the fraction of sampled vertices that are heavy (see Subroutine 2).

Now we are left with the most subtle scenario that h is small, but the number of shallow edges is large. In this case we face a new obstacle inherent of the adversarial setting; it is often not possible to *independently* sample edges and maintain information about their neighborhood. At the time that an edge arrives, we may have already seen all the adjacent edges and thus we will not be able to distinguish whether an edge has a desired property. Recall that an edge is shallow if both its endpoints are light. Therefore instead of sampling edges, we maintain a *shallow-subgraph* sample.

We first sample a set V' of $\mathcal{O}(n^{2/3})$ vertices. We maintain the degrees of these vertices throughout the execution. At an iteration i (i.e., after receiving the i-th edge), let $L' \subseteq V'$ denote the set of vertices of V' with degree at most μ. We maintain all the edges induced by L'. Observe that vertices may become heavy and leave L', in which case we will ignore their adjacent edges (see Subroutine 3). This guarantees that we only need $\tilde{\mathcal{O}}(\mu n^{2/3})$ space for keeping all the edges in our shallow-subgraph sample. After processing the entire input stream, the maintained edges are exactly the shallow edges induced by V'. Every vertex is sampled with probability (roughly) $n^{-1/3}$, however, the probabilities are not independent. Fortunately since we only have negative correlation, we can still say that a shallow edge falls in the induced subgraph with probability roughly $n^{-2/3}$. However as mentioned before, we do not hope to have an efficient sublinear sampling that independently (or even with negative correlation) samples the edges. Our algorithm is

not an exception either. For two edges that share an endpoint, the probabilities of hitting them in a shallow-subgraph sample is *positively* correlated. In general, one cannot hope for strong concentration bounds at the presence of positive correlation. However, the crux of our analysis is that since the light vertices have bounded degree, the degree of dependency is constant. Therefore we are able to bound the variance of our estimator and show that with a constant probability the estimated value is highly concentrated around its expectation. We believe this approach may be of its own interest for testing properties of (sub-)graphs with bounded degree. Finally to boost the probability of success, we maintain independent estimators of subroutines in our algorithm (see Algorithm 1). The analysis is given in Appendix A.

3 Proof of Lemma 1.1

In a maximum matching, the number of edges that are incident to at least one heavy vertex cannot be more than the number of heavy vertices h_G: two matching edges cannot share a (heavy) vertex. On the other hand, the edges that are not adjacent to a heavy vertex are induced by light vertices, i.e., such an edge is a shallow edge. The number of these edges is at most s_G, and thus, the maximum matching size is at most $h_G + s_G$.

In order to prove the lower bound, we first show that there is a matching of size $\frac{4s_G}{5\mu+3}$ in the induced subgraph of light vertices. This implies $M^* \geq \frac{4s_G}{5\mu+3}$. We then show a matching of size $\frac{\mu-\beta+1}{2\mu}h_G$ in G, which means that $M^* \geq \frac{\mu-\beta+1}{2\mu}h_G$. Overall, this implies that $M^* \geq \max\{\frac{\mu-\beta+1}{2\mu}h_G, \frac{4}{5\mu+3}s_G\}$. If we set $\mu = \beta + 3$, we have

$$\frac{\mu-\beta+1}{2\mu} = \frac{(\beta+3)-\beta+1}{2(\beta+3)} = \frac{4}{\beta+3},$$
$$\frac{4}{5\mu+3} = \frac{4}{5(\beta+3)+3} = \frac{1}{1.25\beta+4.5}.$$

By comparing the above functions we can see $\frac{4}{5\mu+3} \leq \frac{\mu-\beta+1}{2\mu}$. Thus, we have

$$\begin{aligned} M^* &\geq \max\{\frac{\mu-\beta+1}{2\mu}h_G, \frac{4}{5\mu+3}s_G\} \\ &\geq \max\{\frac{4}{5\mu+3}h_G, \frac{4}{5\mu+3}s_G\} \\ &= \frac{\max\{h_G, s_G\}}{1.25\mu+0.75} \end{aligned}$$

as desired.

In this section, we write L to denote the set of light vertices.

Claim 3.1. *The subgraph induced by L contains a matching of size $\frac{4s_G}{5\mu+3}$.*

Proof. It is known that the maximum matching size in a graph H is at least $\frac{4m}{5\Delta+3}$, where m in the number of edges in H and Δ is the maximum degree of H [12]. By definition, each vertex of L has degree at most μ. Hence, we have a matching of size $\frac{4s_g}{5\mu+3}$ in the graph induced by L. □

In order to prove the next claim, we use the following generalization of Hall's theorem.

Lemma 3.2 ([3])**.** *Let $G_{(X,Y)}$ be a bipartite graph with bipartition (X, Y). The number of edges in a maximum matching of $G_{(X,Y)}$ is*

$$|X| - \max_{R \subseteq X}(|R| - |N(R)|),$$

where $N(R)$ is the set of all neighbors of vertices in R.

Claim 3.3. *There is a matching of size $\frac{\mu-\beta+1}{2\mu}h_G$ in H, where H is G excluding the edges with both endpoints in L.*

Proof. Let M_h be the set of vertices covered by a maximal matching in the graph induced by heavy vertices and assume $|M_h| = 2\lambda$. Let U be the set of unmatched heavy vertices. Since, M_h is maximal, there is no edge between vertices in U, i.e., U is an independent set. Let $G_{(U,L)}$ denote the bipartite graph consisting of edges connecting vertices in L and U. In the remainder, we use Lemma 3.2 to show the size of a maximum matching in $G_{(U,L)}$ plus $\frac{|M_h|}{2} = \lambda$ is at least $h_G(\frac{1}{2} - \frac{2\beta-2}{\mu-1})$.

Consider the bipartite graph $G_{(U,L)}$ and let R be an arbitrary subset of U. We bound $|N(R)|$ by double counting the number of edges between R and $N(R)$, namely $E[R, N(R)]$. On the one hand, the degree of each vertex in $N(R)$ is at most μ. Hence, we have $E[R, N(R)] \leq \mu N(R)$. On the other hand, vertices in R are heavy, which means they have a degree of at least μ in graph G. However, vertices in R may have some neighbors in M_h that do not exist in $G_{(U,L)}$. Recall that the average degree of vertices in $R \cup M_h$ is at most β and there is at least λ edges between vertices in M_h. Thus, there is at most $\frac{(|R|+2\lambda)\beta}{2} - \lambda$ edges between R and M_h. Therefore, we have $\mu|R| - \left(\frac{(|R|+2\lambda)\beta}{2} - \lambda\right) \leq E[R, N(R)]$. Thus, we have

$$\mu|R| - \left(\frac{|R|\beta}{2} + \lambda\beta - \lambda\right) \leq \mu N(R),$$

which gives us $|R| - \left(\frac{|R|\beta}{2\mu} + \frac{\lambda\beta-\lambda}{\mu}\right) \leq N(R)$. If we apply Lemma 3.2, size of the maximum matching in $G_{(U,L)}$ is at least

$$\begin{aligned}
|U| - \max_{R\subseteq U}(|R| - |N(R)|) &\geq h_G - 2\lambda - \max_{R\subseteq U}\left(|R| - |R| + \left(\frac{|R|\beta}{2\mu} + \frac{\lambda\beta-\lambda}{\mu}\right)\right) \\
&= h_G - 2\lambda - \frac{\lambda\beta-\lambda}{\mu} - \max_{R\subseteq U}\frac{|R|\beta}{2\mu} \\
&= h_G - 2\lambda - \frac{\lambda\beta-\lambda}{\mu} - \frac{(h_G-2\lambda)\beta}{2\mu} \\
&= h_G - \frac{h_G\beta}{2\mu} - 2\lambda + \frac{\lambda}{\mu}.
\end{aligned}$$

Therefore, the total size of the matching is at least

$$\lambda + h_G - \frac{h_G\beta}{2\mu} - 2\lambda + \frac{\lambda}{\mu} = h_G - \frac{h_G\beta}{2\mu} - \lambda + \frac{\lambda}{\mu}.$$

This quantity is decreases as a function of λ. Thus, it is minimized for $\lambda = \frac{h_G}{2}$. In this case, the size of the matching is at least

$$h_G - \frac{h_G\beta}{2\mu} - \frac{h_G}{2} + \frac{h_G}{2\mu} = \frac{\mu-\beta+1}{2\mu}h_G,$$

which completes the proof. □

4 Hardness Results

In this section, we introduce the Matching Parity problem, a communication complexity problem that leads to hardness results for estimating the size of the maximum matching within a factor better than $3/2 - \epsilon$ for any constant $\epsilon > 0$, even if the graph is a collection of paths. For deterministic algorithms, we show a polynomial lower bound for the required space. Under a plausible conjecture for the hardness of randomized algorithms for the Matching Parity problem, we also show a $\Omega(\sqrt{n})$ lower bound for the memory of any randomized algorithm.

Matching Parity: Alice has a (private) binary string $x \in \{0,1\}^{4n}$. Bob's private information is a *perfect matching* H between numbers in $[4n]$. It is guaranteed that there is a $\theta \in \{0,1\}$ such that for at least $2\beta n$ edges (i,j) in the matching, x_i xor $x_j = \theta$. Alice sends a message to Bob and Bob's task is to output θ, the *parity of the matching*. Here β denotes a given real value in $(\frac{1}{2}, 1]$ and n denotes an integer.

We note that the value of β and the existence of θ is common knowledge to both Alice and Bob. Furthermore, we note that the problem is interesting only when the number of zeros in x is in the range $2n(1 \pm \beta)$ (otherwise $\theta = 0$ is the only possible output). Let $f_{\text{det}}(n, \beta)$ denote the minimum required size of Alice's message for which Bob can always compute the correct θ. Similarly, let $f_{\text{rnd}}(n, \beta)$ denote the minimum size of the message for which Bob can output the correct θ with probability more than $3/4$. Before we discuss the lower bounds for $f_{\text{det}}(.)$ and $f_{\text{rnd}}(.)$, let us give a reduction from the Matching Parity problem to the problem of estimating the size of the maximum matching.

Let x and H denote Alice's and Bob's private information, respectively. Let θ denote the parity of H. Let z denote the number of zeros in x. Without loss of generality, we may assume the number of zeros is more than ones, thus $2n \leq z$. Consider a graph $G_{(x,H)} = (V, E_1 \cup E_2)$ constructed as follows.

- For every $i \in [4n]$, if $x[i] = 0$, we have a vertex v_i.
- For every $i \in [4n]$, if $x[i] = 1$, we have two vertices u_i and v_i with an edge (u_i, v_i) between them. Let E_1 denote the set of such edges.
- For every $(i,j) \in H$, we have an edge (v_i, v_j). Let E_2 denote the set of such edges.

Let q denote the number of zero entries that participate in edges with the correct parity θ in Bob's matching. Now consider an edge $(i,j) \in H$ with parity $\tau = x_i$ xor x_j. If $\tau = 1$, the corresponding edge $e = (v_i, v_j) \in E_2$ connects a single vertex to a single edge; thus creating a copy of P_2.[5] If $\tau = 0$ and $x_i = x_j = 1$, then e connects two single edges; creating a P_3. If $\tau = 0$ and $x_i = x_j = 0$, then e connects two single vertices, hence, creating a new P_1. Therefore we have two cases:

(i) If $\theta = 1$, then $G_{(x,H)}$ comprise $\frac{z-q}{2}$ copies of P_1, q copies of P_2, and $2n - \frac{z+q}{2}$ copies of P_3.

(ii) If $\theta = 0$, then $G_{(x,H)}$ comprise $\frac{q}{2}$ copies of P_1, $z - q$ copies of P_2, and $2n - z + \frac{q}{2}$ copies of P_3.

We use this construction in the proof of the following reduction.

Theorem 4.1. *Let* Alg *denote a streaming algorithm that for some $\delta \in [0, 1/2)$ and $\epsilon > 0$, with probability at least $1-\delta$ computes a $(3/2-\epsilon)$-approximation to the maximum matching size, using Q bits of space, if the input graph is a collection of paths. Then there exists a communication protocol for the Matching Parity problem with $\beta \geq 1 - \epsilon$ such that Alice's message has size $Q + \mathcal{O}(\log n)$ while Bob's probability of outputting the correct θ is at least $1 - \delta$.*

Proof. Consider the graph $G_{(x,H)}$ constructed above. Recall that we focus only on $z \geq 2n$. We have two cases:

(i) If $\theta = 1$, then q is the total number of edges with the correct parity, hence $q \geq 2\beta n$. The size of maximum matching of $G_{(x,H)}$ is *at most*:

$$\frac{z-q}{2} + q + 2(2n - \frac{z+q}{2}) = 4n - \frac{z+q}{2} \leq 4n - \frac{z+2\beta n}{2} = (4-\beta)n - \frac{z}{2}.$$

[5] For a non-negative integer $r \geq 0$, let P_r denote a path with r edges.

(ii) If $\theta = 0$, then $z - q$ is the number of edges with the wrong parity, hence $z - q \leq (1-\beta)2n$. Therefore the size of maximum matching is *at least*:

$$\begin{aligned}\frac{q}{2} + (z-q) + 2(2n - z + \frac{q}{2}) &= 4n - z + \frac{q}{2} \\ &\geq 4n - z + \frac{z - (1-\beta)2n}{2} = 4n - \frac{z}{2} - n + \beta n \\ &\geq (3+\beta)n - \frac{z}{2}.\end{aligned}$$

Observe that since $\beta > 1/2$, the size of the maximum matching is greater in Case (ii). Suppose we further assume that $\beta \geq 1 - \frac{4\epsilon}{5-2\epsilon}$. The ratio between the sizes of matchings between the two cases is at least

$$\begin{aligned}\frac{(3+\beta)n - z/2}{(4-\beta)n - z/2} &\geq \frac{(3+\beta)n - (2n)/2}{(4-\beta)n - (2n)/2} && \text{for } a \geq b, \frac{a-1}{b-1} \geq \frac{a}{b} \\ &= \frac{2+\beta}{3-\beta} \\ &\geq 1.5 - \epsilon && \beta \geq 1 - \frac{4\epsilon}{5-2\epsilon}\end{aligned}$$

Now suppose we run Alg on graph G with the input order $\langle E_1 \cdot E_2 \rangle$, i.e., we first feed all the edges of E_1 (in an arbitrary order) to Alg, then we feed the edges of E_2 (in an arbitrary order). Since Alg has an approximation factor strictly less than $1.5 - \epsilon$, the algorithm has to distinguish between Case (i) and Case (ii), with probability at least $1 - \delta$. Furthermore, at any point of time, Alg keeps at most Q bits of memory. Let $R \in \{0,1\}^Q$ denote the memory of Alg after processing all the edges of E_1. Note that if Alg is randomized, R is a random variable.

The crucial fact is that Alice knows E_1, and Bob knows E_2. Consider the following communication protocol. Given Alg as a black-box, Alice simulates the execution of Alg over the edges of E_1. Let R denote the memory of the algorithm after processing E_1. Alice transfers R (and the size of z) to Bob. Next, Bob continues the simulation by starting with memory R and processing the edges of E_2. If the final output of Alg is more than $(4-\beta)n - z/2$, Bob outputs $\theta = 0$, otherwise he outputs $\theta = 1$. As discussed above, the output is correct with probability at least $1 - \delta$ while the length of the message is $Q + \mathcal{O}(\log n)$ bits. □

Theorem 4.1 immediately leads to the following:

Corollary 4.2. *A randomized streaming algorithm that estimates the size of maximum matching within $(1.5-\epsilon)$-approximation with probability more than $3/4$, requires $f_{\mathrm{rnd}}(\lfloor n/6 \rfloor, 1-\epsilon)$ space. Similarly, a deterministic one requires $f_{\mathrm{det}}(\lfloor n/6 \rfloor, 1-\epsilon)$ space. These hold even if the input graph is restricted to a collection of paths. Here ϵ is an arbitrary positive value and n is the number of vertices of the input graph.*

In Appendix C, we establish a linear lower bound for the deterministic variant of Matching Parity by proving the following theorem.

Theorem 4.3. *For a deterministic protocol in which Bob cannot err, Alice has to send a message of length $\Omega((1-\beta)^2 n)$, i.e., $f_{\mathrm{det}}(n, \beta) = \Omega((1-\beta)^2 n)$.*

We conjecture that the following bounds for the Matching Parity problem are tight. We show evidence supporting the randomized variant in Appendix C.

Conjecture 4.4. *For the Matching Parity problem, $f_{\mathrm{det}}(n, 1) = \Omega(n)$ and $f_{\mathrm{rnd}}(n, 1) = \Omega(\sqrt{n})$.*

References

[1] K. J. Ahn and S. Guha. Linear programming in the semi-streaming model with application to the maximum matching problem. In *ICALP (2)*, pages 526–538, 2011.

[2] S. Baswana, M. Gupta, and S. Sen. Fully dynamic maximal matching in O($\log n$) update time. In *Proceedings of the 52nd IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 383–392, 2011.

[3] J. A. Bondy and U. S. R. Murty. *Graph theory with applications*, volume 6. Macmillan London, 1976.

[4] S. Eggert, L. Kliemann, P. Munstermann, and A. Srivastav. Bipartite graph matchings in the semi-streaming model. *Algorithmica*, 63(1-2):490–508, 2012.

[5] S. Eggert, L. Kliemann, P. Munstermann, and A. Srivastav. Bipartite matching in the semi-streaming model. *Algorithmica*, 63(1-2):490–508, 2012.

[6] L. Epstein, A. Levin, J. Mestre, and D. Segev. Improved approximation guarantees for weighted matching in the semi-streaming model. *SIAM J. Discrete Math.*, 25(3):1251–1265, 2011.

[7] J. Feigenbaum, S. Kannan, A. McGregor, S. Suri, and J. Zhang. On graph problems in a semi-streaming model. *Theoretical Computer Science*, 348(2):207–216, 2005.

[8] J. Feigenbaum, S. Kannan, A. McGregor, S. Suri, and J. Zhang. Graph distances in the data-stream model. *SIAM J. Comput.*, 38(5):1709–1727, 2008.

[9] A. Goel, M. Kapralov, and S. Khanna. On the communication and streaming complexity of maximum bipartite matching. In *SODA*, pages 468–485, 2012.

[10] M. Gupta and R. Peng. Fully dynamic $(1 + \epsilon)$-approximate matchings. In *FOCS*, pages 548–557, 2013.

[11] V. Guruswami and K. Onak. Superlinear lower bounds for multipass graph processing. *Proceedings of the 28th Annual IEEE Conference on Computational Complexity (CCC)*, pages 287–298, 2013.

[12] Y. Han. Matching for graphs of bounded degree. In *Frontiers in Algorithmics*, pages 171–173. Springer, 2008.

[13] A. Hassidim, J. A. Kelner, H. N. Nguyen, and K. Onak. Local graph partitions for approximation and testing. In *Proceedings of the 50th IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 22–31, 2009.

[14] Z. Ivkovic and E. L. Lloyd. Fully dynamic maintenance of vertex cover. In *WG*, pages 99–111, 1993.

[15] M. Kapralov. Better bounds for matchings in the streaming model. *Proceedings of the 23rd Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1679–1697, 2013.

[16] M. Kapralov, S. Khanna, and M. Sudan. Approximating matching size from random streams. *Proceedings of the 25th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 734–751, 2014.

[17] R. M. Karp, U. V. Vazirani, and V. V. Vazirani. An optimal algorithm for on-line bipartite matching. *Proceedings of the 22nd Annual ACM Symposium on Theory of Computing (STOC)*, pages 352–358, 1990.

[18] C. Konrad, F. Magniez, and C. Mathieu. Maximum matching in semi-streaming with few passes. In *APPROX-RANDOM*, pages 231–242, 2012.

[19] A. V. Kostochka. Lower bound of the hadwiger number of graphs by their average degree. *Combinatorica*, 4(4):307–316, 1984.

[20] L. Lovasz and M. Plummer. Matching theory. In *North-Holland, Amsterdam-New York*, 1986.

[21] A. McGregor. Finding graph matchings in data streams. In *Proceedings of the 8th International Workshop on Randomization and Approximation Techniques in Computer Science (RANDOM)*, pages 170–181, 2005.

[22] A. McGregor. Graph mining on streams. In *Encyclopedia of Database Systems*, pages 1271–1275, 2009.

[23] S. Micali and V. V. Vazirani. An $o(\sqrt{|V|}|e|)$ algorithm for finding maximum matching in general graphs. *Proceedings of the 21st IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 17–27, 1980.

[24] O. Neiman and S. Solomon. Simple deterministic algorithms for fully dynamic maximal matching. *Proceedings of the 45th Annual ACM Symposium on Theory of Computing (STOC)*, pages 745–754, 2013.

[25] H. N. Nguyen and K. Onak. Constant-time approximation algorithms via local improvements. In *Proceedings of the 49th IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 327–336, 2008.

[26] K. Onak, D. Ron, M. Rosen, and R. Rubinfeld. A near-optimal sublinear-time algorithm for approximating the minimum vertex cover size. In *SODA*, pages 1123–1131, 2012.

[27] K. Onak and R. Rubinfeld. Maintaining a large matching and a small vertex cover. In *Proceedings of the 42nd Annual ACM Symposium on Theory of Computing (STOC)*, pages 457–464, 2010.

[28] List of open problems in sublinear algorithms: Problem 60. `http://sublinear.info/60`.

[29] A. Panconesi and A. Srinivasan. Randomized distributed edge coloring via an extension of the chernoff–hoeffding bounds. *SIAM Journal on Computing*, 26(2):350–368, 1997.

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

[31] Y. Yoshida, M. Yamamoto, and H. Ito. An improved constant-time approximation algorithm for maximum matchings. *Proceedings of the 41st Annual ACM Symposium on Theory of Computing (STOC)*, pages 225–234, 2009.

[32] M. Zelke. Weighted matching in the semi-streaming model. *Algorithmica*, 62(1-2):1–20, 2012.

A Analysis of Section 2

A.1 Main Subroutines

The following is the main procedure of the algorithm.

Algorithm 1 Maximum Matching Estimator

Input: A stream of edges of the graph $e_1, \ldots, e_m$.
Output: An estimation M of the size of a maximum matching M^*.
Initialization Process:

1: Set $\alpha_1 = 3$, $\alpha_2 = 3\epsilon^{-2}$, $\alpha_3 = 4\epsilon^{-1}$ and $Q = 26\lceil \ln(\frac{1}{\delta}) \rceil + 1$.
2: Set $c_1 = c_2 = c_3 = 2/3$.
3: Initialize Subroutine 1 with factor α_1 and c_1.
4: Initialize Q independent executions of Subroutines 2 and 3, with the corresponding α and c factors.

Update Process, upon the arrival of $e_i = (u, v)$**:**

1: Update each subroutine with the new arrival.

Termination Process:

1: Let R denote the return value of Subroutine 1.
2: For $i \in [Q]$, let $\hat{h}_i$ denote the value returned by the i-th execution of Subroutine 2. Similarly, let $\hat{s}_i$ denote that of Subroutine 3.
3: For $i \in [Q]$, let $M_i = \max\{\alpha_1 n^{2/3}, \hat{h}_i, \hat{s}_i\}$.
4: **if** $R < \alpha_1 n^{2/3}$ **then**
5: Output $M = R$.
6: **else**
7: Let M_{med} denote the median value of M_i's.
8: Output $M = \frac{M_{\text{med}}}{(1+\epsilon)\eta}$.

Subroutine 1 In this subroutine, we greedily construct a maximal matching M''. However, we stop the process if $|M''|$ exceeds the limit $\alpha_1 n^{c_1}$, where the constants α_1 and c_1 are tuned by the algorithm.

Subroutine 1. [Bounded Maximal Matching]

Input: A stream of edges of the graph $e_1, \ldots, e_m$.
Return Value: $R := \min\{|M'|, \alpha_1 n^{c_1}\}$ where M' is a maximal matching.
Initialization Process:

1: Initialize M'' to an empty set.

Update Process, upon the arrival of $e_i = (u, v)$**:**

1: **if** $|M''| < \alpha_1 n^{c_1}$ **and** $M'' \cup \{e_i\}$ is a valid matching **then**
2: $M'' \leftarrow M'' \cup \{e_i\}$.
3: **else**
4: Discard e_i.

Termination Process:

1: Return $|M''|$.

Subroutine 2 In this subroutine, we choose a subset of size $\alpha_2 n^{c_2}$ from the set of vertices uniformly at random. The parameters α_2 and c_2 are tuned by the algorithm. During the execution, we maintain the degrees of sampled vertices.

Subroutine 2. [Heavy Vertices]

Input: A stream of edges of the graph $e_1, \ldots, e_m$.
Return Value: $\hat{h}$, an estimation for the number of heavy vertices in the graph.
Initialization Process:

Initialize $V' \subset V$ as a random subset of size $\alpha_2 n^{c_2}$.
For every $v \in V'$, initialize $\deg(v)$ to zero.

Update Process, upon the arrival of $e_i = (u, v)$:

if $u \in V'$ **then**
    $\deg(u) \leftarrow \deg(u) + 1$.
if $v \in V'$ **then**
    $\deg(v) \leftarrow \deg(v) + 1$.

Termination Process:

Let $h' = |\{v \in V' | \deg(v) > \mu\}|$ denote the number of heavy vertices in V'.
Return $\hat{h} = h' \times \frac{n^{1-c_2}}{\alpha_2}$

Subroutine 3 In this subroutine, we again choose a subset of size $\alpha_3 n^{c_3}$ from the set of vertices uniformly at random. However, in addition to the degrees of vertices, we also maintain the edges that the degrees of their endpoints are at most μ. Thus we require at most $\tilde{\mathcal{O}}(\mu\alpha_3 n^{c_3})$ memory space.

Subroutine 3. [Shallow Edges]

Input: A stream of edges of the graph $e_1, \ldots, e_m$.
Return Value: $\hat{s}$, an estimation for the number of shallow edges in the graph.
Initialization Process:

Initialize $V' \subset V$ as a random subset of size $\alpha_3 n^{c_3}$.
For every $v \in V'$, initialize $\deg(v)$ to zero and E' to an empty set.

Update Process, upon the arrival of $e_i = (u, v)$:

if both $u, v \in V'$ **then**
    $E' \leftarrow E' \cup \{e_i\}$.
if $u \in V'$ **then**
    $\deg(u) \leftarrow \deg(u) + 1$.
    If $\deg(u) > \mu$, remove all the edges adjacent to u from E'.
if $v \in V'$ **then**
    $\deg(v) \leftarrow \deg(v) + 1$.
    If $\deg(v) > \mu$, remove all the edges adjacent to v from E'.

Termination Process:

Let $s' = |E'|$ denote the number of shallow edges induced by V'.
Return $\hat{s} = s' \times \frac{n(n-1)}{(|V'|)(|V'|-1)}$

A.2 Analysis

In this section we analyze the approximation ratio and the success probability of our randomized algorithm. We first distinguish between two primary cases, depending on the size of a maximum matching M^*. Recall that the size of a maximal matching is always withing a factor 2 of a maximum matching. The first subroutine of our algorithm, maintains the size R of a maximal matching up to $\alpha_1 n^{2/3}$. Observe that in Algorithm 1, we completely ignore the second and third subroutines if $R < \alpha_1 n^{2/3}$. Therefore for the input scenarios that $R < \alpha_1 n^{2/3}$, we indeed have a maximal matching of size R and thus our algorithm *deterministically* estimates M^* within a two factor. In the rest of this section we focus on the case where $\alpha_1 n^{2/3} \leq R \leq M^*$.

By Lemma 1.1, we know that either h or s is large (respectively, the number of heavy vertices and the number of shallow edges). Consider a single execution of Subroutine 2 (resp. Subroutine 3), that returns an estimation for $\hat{h}$ (resp. $\hat{s}$). In our algorithm, we maintain $\log(\frac{1}{\delta})$ executions of the subroutines to boost the success probability. Therefore the goal is to show that for a single execution, with a constant probability the output of our algorithm is within a constant factor of M^*. Since we output the *maximum* of the two estimations, we need to show two concentration bounds for each estimator: (i) if the target value is large, with constant probability we get a constant approximation; and (ii) if the target value is small, with constant probability the estimation is small too. We prove this for both estimators in the following lemmas.

Lemma A.1. *For Subroutine 2, we have:*

- *If* $h \geq n^{2/3}$, $\Pr\left[\left|\hat{h} - h\right| \geq \epsilon h\right] \leq 2\exp(-n^{1/3})$*; and*
- *If* $h < n^{2/3}$, $\Pr\left[\hat{h} > \alpha_1 n^{2/3}\right] \leq \exp(-n^{1/3})$.

Proof. Recall in the subroutine we select a random sample V' of size $n' = \alpha_2 n^{2/3}$. Let $V' = \{v'_1, \ldots, v'_{n'}\}$. For $i \in [n']$, let X_i denote the random variable where $X_i = 1$ if v'_i is a heavy vertex, and $X_i = 0$ otherwise. Since we have h heavy vertices, we have that $\Pr[X_i = 1] = \frac{h}{n}$ for every $i \in [n']$. Let $h' = \sum_i X_i$. Clearly $\mathrm{E}[h'] = \frac{n' \cdot h}{n}$. Recall that the return value of Subroutine 2 is $\hat{h} = h' \times \frac{n}{n'}$. Thus in expectation, we return the value of h. Now since all X_i's are negatively correlated, we use the extension of Chernoff bound E.1 to prove the first bound of the lemma for $h \geq n^{2/3}$.

$$
\begin{aligned}
\Pr\left[\left|\hat{h} - h\right| \geq \epsilon h\right] &= \Pr\left[\left|\hat{h} - \mathrm{E}\left[\hat{h}\right]\right| \geq \epsilon\, \mathrm{E}\left[\hat{h}\right]\right] \\
&= \Pr\left[\left|h' - \mathrm{E}\left[h'\right]\right| \geq \epsilon\, \mathrm{E}\left[h'\right]\right] && \hat{h} = h' \times \frac{n}{n'} \\
&\leq 2\exp\left(-\frac{\epsilon^2\, \mathrm{E}\left[h'\right]}{3}\right) && \textit{extension of Chernoff bound} \\
&= 2\exp\left(-\frac{\epsilon^2 \cdot n' \cdot h}{3n}\right) && \mathrm{E}\left[h'\right] = \frac{n' \cdot h}{n} \\
&= 2\exp\left(-\frac{h}{n^{1/3}}\right) && n' = \frac{3}{\epsilon^2} \cdot n^{2/3} \\
&\leq 2\exp(-n^{1/3}) && h \geq n^{2/3}
\end{aligned}
$$

We again use the Chernoff bound to prove the lemma for $h < n^{2/3}$.

$$\begin{aligned}
\Pr\left[\hat{h} > \alpha_1 n^{2/3}\right] &\leq \Pr\left[\hat{h} > \left(\frac{\alpha_1 n^{2/3}}{h}\right) \mathrm{E}\left[\hat{h}\right]\right] && \mathrm{E}\left[\hat{h}\right] = h \\
&= \Pr\left[h' > \left(\frac{\alpha_1 n^{2/3}}{h}\right) \mathrm{E}\left[h'\right]\right] && \hat{h} = h' \times \frac{n}{n'} \\
&\leq \Pr\left[h' > \left(1 + \frac{\alpha_1 n^{2/3}}{2h}\right) \mathrm{E}\left[h'\right]\right] \\
&\leq \exp\left(-\frac{\alpha_1^2 \cdot n^{4/3} \cdot \mathrm{E}\left[h'\right]}{12h^2}\right) && \text{extension of Chernoff bound} \\
&= \exp\left(-\frac{\alpha_1^2 \cdot n^{4/3} \cdot n' \cdot h}{12h^2 \cdot n}\right) && \mathrm{E}\left[h'\right] = \frac{n' \cdot h}{n} \\
&= \exp\left(-\frac{\alpha_1^2 \cdot n^{1/3} \cdot n'}{12h}\right) \\
&= \exp\left(-\frac{\alpha_1^2 \cdot \alpha_2 \cdot n}{12h}\right) && n' = \alpha_2 n^{2/3} \\
&\leq \exp\left(-\left(\frac{\alpha_1}{2}\right)^2 n^{1/3}\right) && h < n^{2/3}, \alpha_2 \geq 3
\end{aligned}$$

□

Indeed for the shallow-edge estimator, the problem is more sophisticated. For a shallow edge $e = (u, v)$, the probability of hitting e in a shallow subgraph is positively correlated to that of all edges adjacent to u and v. Thus a normal application of (generalized variants of) the Chernoff bound fails to satisfy the required concentration bounds. Indeed in the next lemma, we use the small dependency degree between the variables to prove the concentration bounds.

Lemma A.2. *For Subroutine 3, we have:*

- *If* $s \geq n^{2/3}$, $\Pr\left[|\hat{s} - s| \geq \epsilon s\right] \leq 1/4$*; and*
- *If* $s < n^{2/3}$, $\Pr\left[\hat{s} > \alpha_1 n^{2/3}\right] \leq \alpha_1^{-2} = 1/9$.

Proof. Recall that in Subroutine 3, we sample a shallow subgraph (V', E') where $|V'| = n' = \alpha_3 n^{2/3}$. Let $e_1, \ldots, e_s$ denote the shallow edges in the original graph. For a shallow edge e_i, let X_i denote the random variable where $X_i = 1$ if $e_i \in E'$ and $X_i = 0$ otherwise. Let $p = \Pr\left[X_i = 1\right]$. We have

$$\mathrm{E}\left[X_i\right] = \Pr\left[X_i = 1\right] = p = \frac{\binom{n'}{2}}{\binom{n}{2}} = \frac{n'(n'-1)}{n(n-1)}$$

We note that this probability is indeed very small $p \in [\frac{\alpha_3^2 n^{-2/3}}{2}, \alpha_3^2 n^{-2/3}]$. Let $s' = \sum_i X_i$; note that $\mathrm{E}\left[s'\right] = sp$. In order to get a concentration bound for s', we would need to bound the variance of our estimator. We know that

$$\mathrm{Var}\left(s'\right) = \sum_i \mathrm{Var}\left(X_i\right) + \sum_{i \neq j} \mathrm{Cov}\left(X_i, X_j\right)$$

Thus we need to calculate $\mathrm{Var}(X_i)$ and $\mathrm{Cov}(X_i, X_j)$. We have

$$\begin{aligned}\mathrm{Var}(X_i) &= \mathrm{E}\left[(X_i - \mathrm{E}[X_i])^2\right] \\ &= \Pr[X_i = 1](1 - \mathrm{E}[X_i])^2 + \Pr[X_i = 0](\mathrm{E}[X_i])^2 \\ &= p(1-p)^2 + (1-p)(p)^2 \\ &= p(1-p) \le p \end{aligned} \tag{1}$$

For the covariance, we have

$$\mathrm{Cov}(X_i, X_j) = \mathrm{E}[X_i X_j] - \mathrm{E}[X_i]\,\mathrm{E}[X_j] = \Pr[X_i = X_j = 1] - p^2$$

Thus we have two cases based on whether the edges corresponding to X_i and X_j share an endpoint: (i) if e_i and e_j share an endpoint, $\Pr[X_i = X_j = 1]$ is the probability of having 3 fixed vertices in V'; and (ii) if e_j and e_j do not share an endpoint, $\Pr[X_i = X_j = 1]$ is the probability of having 4 fixed vertices in V'.

$$\mathrm{Cov}(X_i, X_j) = \begin{cases} \frac{\binom{n-3}{n'-3}}{\binom{n}{n'}} - p^2 = \frac{n'(n'-1)(n'-2)}{n(n-1)(n-2)} - p^2 < p^{3/2} - p^2 \le p^{3/2} & \text{for Case (i)} \\ \frac{\binom{n-4}{n'-4}}{\binom{n}{n'}} - p^2 = \frac{n'\cdots(n'-3)}{n\cdots(n-3)} - p^2 < p^2 - p^2 = 0 & \text{for Case (ii)} \end{cases}$$

This implies that we can completely ignore the terms for Case (ii) since we are interested in an upper bound on the variance. On the other hand, the degree of a light vertex is at most μ. Hence, every shallow edge is adjacent to at most 2μ other edges. Leading to the following upper bound for the variance of s'.

$$\begin{aligned}\mathrm{Var}(s') &= \sum_{i=1}^{s} \mathrm{Var}(X_i) + \sum_{i \ne j} \mathrm{Cov}(X_i, X_j) \\ &\le sp + \sum_{i \ne j} \mathrm{Cov}(X_i, X_j) && \textit{Equation (1)} \\ &\le sp + \sum_{i \ne j : e_i \text{ and } e_j \text{ share an endpoint}} p^{3/2} && \textit{By Cases (i) and (ii)} \\ &\le sp + s(2\mu)(p^{3/2}) \le 2sp && p = \Theta(n^{-2/3}) \text{ and } n \ge (2\mu)^3 \end{aligned}$$

Now we can indeed use the Chebyshev's inequality E.2 to prove the desired concentration bounds. For $s \ge n^{2/3}$ we have

$$\begin{aligned}\Pr[|\hat{s} - s| \ge \epsilon s] &= \Pr[|\hat{s} - \mathrm{E}[\hat{s}]| \ge \epsilon\,\mathrm{E}[\hat{s}]] \\ &= \Pr\left[\left|s' - \mathrm{E}\left[s'\right]\right| \ge \epsilon\,\mathrm{E}\left[s'\right]\right] && \hat{s} = \frac{s'}{p} \\ &\le \frac{\mathrm{Var}(s')}{\epsilon^2(\mathrm{E}[s'])^2} && \text{Chebyshev's inequality} \\ &\le \frac{2sp}{\epsilon^2(\mathrm{E}[s'])^2} = \frac{2sp}{\epsilon^2(sp)^2} = \frac{2}{\epsilon^2 \cdot s \cdot p} \\ &\le \frac{2}{\epsilon^2 \cdot n^{2/3} \cdot p} && s \ge n^{2/3} \\ &\le \frac{4n^{2/3}}{\epsilon^2 n^{2/3} \alpha_3^2} && p \ge \frac{\alpha_3^2}{2n^{2/3}} \\ &\le 1/4 && \alpha_3 = 4\epsilon^{-1} \end{aligned}$$

One can use a similar argument to prove the concentration bound for $s < n^{2/3}$.

$$\begin{aligned}
\Pr\left[\hat{s} > \alpha_1 n^{2/3}\right] &= \Pr\left[\hat{s} > \left(\frac{\alpha_1 n^{2/3}}{\mathrm{E}\left[\hat{s}\right]}\right)\mathrm{E}\left[\hat{s}\right]\right] \\
&= \Pr\left[s' > \left(\frac{\alpha_1 n^{2/3}}{s}\right)\mathrm{E}\left[s'\right]\right] \\
&\leq \frac{\mathrm{Var}\left(s'\right)}{\left(\mathrm{E}\left[s'\right]\left(\frac{\alpha_1 n^{2/3}}{s} - 1\right)\right)^2} && \text{Chebyshev's Inequality} \\
&\leq \frac{\mathrm{Var}\left(s'\right)}{\left(\mathrm{E}\left[s'\right]\left(\frac{\alpha_1 n^{2/3}}{2s}\right)\right)^2} && s < n^{2/3} \\
&\leq \frac{4s^2\,\mathrm{Var}\left(s'\right)}{s^2 p^2 \alpha_1^2 n^{4/3}} && \mathrm{E}\left[s'\right] = sp \\
&\leq \frac{8s}{p \cdot \alpha_1^2 n^{4/3}} && \mathrm{Var}\left(s'\right) \leq 2sp \\
&\leq \frac{8}{p \cdot \alpha_1^2 n^{2/3}} && s < n^{2/3} \\
&\leq \frac{16 n^{2/3}}{\alpha_3^2 \alpha_1^2 n^{2/3}} && p \geq \frac{\alpha_3^2}{2n^{2/3}} \\
&\leq \alpha_1^{-2} && \alpha_3 \geq 4
\end{aligned}$$

□

We are now ready to prove the main theorem.

Proof of Theorem 1.2. Consider the parameters in Algorithm 1. As discussed before, if $R < \alpha_1 n^{2/3}$, we deterministically output the size of a maximal matching, which is within a 2-approximation of M^*. Thus we focus on the instances where $M^* \geq \alpha_1 n^{2/3}$. In this scenario, the output is proportional to the median of the M_i values where $M_i = \max\{\alpha_1 n^{2/3}, \hat{h}_i, \hat{s}_i\}$. Let $\tau = \max\{\alpha_1 n^{2/3}, h_G, s_G\} = \max\{h_G, s_G\}$. By Lemma 1.1, we know that at least one of h_G and s_G is at least $\frac{M^*}{2} > n^{2/3}$. We say the i^{th} execution of the sampling is *successful*, if $|M_i - \tau| \leq \epsilon\tau$. Let X_i denote the random variable where $X_i = 1$ if the i^{th} sample is successful and $X_i = 0$ otherwise. Let us consider the event $X_i = 1$. We have two cases based on whether $h_G \geq s_g$. One can use the union bound to get a lower bound on the probability of success.

$$\Pr\left[X_i = 1\right] \geq \begin{cases} 1 - \Pr\left[\left|\hat{h} - h_G\right| \geq \epsilon h_G\right] - \Pr\left[\hat{s} > \alpha_1 n^{2/3}\right] & \text{for } \tau = h_G \\ 1 - \Pr\left[|\hat{s} - s_G| \geq \epsilon s_G\right] - \Pr\left[\hat{h} > \alpha_1 n^{2/3}\right] & \text{for } \tau = s_G \end{cases}$$

Now we can use the concentration bounds given by Lemmas A.1 and A.2, to get a constant lower bound for $\Pr\left[X_i = 1\right]$. Note that we can assume $n \geq 125$ since otherwise we can keep the (constant-size) input in memory. Thus we have $n^{1/3} \geq 5$.

$$\begin{aligned}
\Pr[X_i = 1] &\geq \min\{1 - \Pr\left[\left|\hat{h} - h_G\right| \geq \epsilon h_G\right] - \Pr\left[\hat{s} > \alpha_1 n^{2/3}\right] \\
&\quad , 1 - \Pr[|\hat{s} - s_G| \geq \epsilon s_G] - \Pr\left[\hat{h} > \alpha_1 n^{2/3}\right]\} \\
&\geq \min\{1 - 2\exp(-n^{1/3}) - 1/9, 1 - 1/4 - \exp(-n^{1/3})\} && \text{Lemmas A.1 and A.2} \\
&\geq \min\{8/9 - 2\exp(-5), 3/4 - \exp(-5)\} \geq 0.743 && n \geq 125
\end{aligned}$$

Observe if for some $i \in [Q]$, $X_i = 1$, then by Lemma 1.1, $\frac{M_i}{\eta(1+\epsilon)} \leq M^* \leq 2M_i(1+\epsilon)$. However, the output of the algorithm is proportional to median value of all M_i's. In the event that more than half of X_i's are successful, the median is indeed successful. Let $X = \sum_{i\in[Q]} X$ denote the number of successful M_i's. With probability at least $\Pr[X \geq 0.501Q]$, the output of our algorithm is within $2\eta(1+\epsilon)^2$-approximation of M^*. Since the executions are independent, we can use a Chernoff bound to get the success probability of our algorithm.

$$\begin{aligned}
\Pr[X \geq 0.501Q] &\geq 1 - \exp\left(-\frac{(1 - \frac{0.501Q}{\mathrm{E}[X]})^2 \mathrm{E}[X]}{2}\right) && \text{Chernoff Bound} \\
&\geq 1 - \exp\left(-\frac{(1 - 0.501/0.743)^2 \mathrm{E}[X]}{2}\right) && \mathrm{E}[X] = \Pr[X_i = 1] Q \geq 0.743Q \\
&\geq 1 - \exp(-0.053 \mathrm{E}[X]) \geq 1 - \exp(-0.039Q) \\
&\geq 1 - \exp(-\ln(\delta^{-1})) = 1 - \delta && Q \geq 26\ln(\delta^{-1})
\end{aligned}$$

Therefore with probability at least $1 - \delta$ the output of our algorithm is within $(2\eta)(1+\epsilon)^2$-approximation of the size of the maximum matching. It only remains to analyze the running time of our algorithm. Observe that the update process and the termination process of each single subroutine can be implemented to run in $O(1)$ time. Therefore the update time of our algorithm is linear to the number of subroutines $\mathcal{O}(Q) = \mathcal{O}(\ln(\delta^{-1}))$. Furthermore, at the end of the execution of the algorithm, we need to find the median of Q values which can also be implemented in $\mathcal{O}(\ln(\delta^{-}1))$ time.

□

B Maximum Matching in Forests

In this section we approximate the maximum matching size of a Tree within a $2(1+3\epsilon)$ factor. We use some structural property of trees to improve the lower and upper bounds of the maximum matching in Lemma B.1. Later, we generalize this lemma to forests with no isolated vertices and approximate the maximum matching size of these forests by a $2(1+3\epsilon)$ factor. During this section we set $\mu = 1$ i.e. vetices with degree 1 are light and vertices with degree 2 or more are heavy.

Lemma B.1. *Let T be a tree with maximum matching size M^*. We have the following bounds:*

- *Upper bound: $M^* \leq h_T + 1$; and*
- *Lower bound: $M^* \geq \frac{h_T+1}{2}$*

Proof. In a tree with more than two vertices, leaves can not be adjacent. Thus, each matching edge shares at least one vertex with the heavy vertices. Thus, maximum matching size in a tree T with more than two vertices is at most h_T . On the other hand, the maximum matching size in a tree with two vertices is 1. Therefore, for every tree we have $M^* \leq \max(h_T, 1) \leq h_T + 1$.

In order to show the lower bound we show that every tree has a matching with the following two property.

- All of the heavy vertices are matched.
- At least one light vertex (leaf) is matched.

This immediately gives us $M^* \geq \frac{h_T+1}{2}$.

Let M be a maximum matching that matches the maximum possible number of heavy vertices. Assume a heavy vertex v is not matched in M. let P_v be a maximal alternative path starting from v. Since, there is no cycle in the graph, the other end of P_v is a leaf. If we replace the unmatched edges and matched edges in P_v, size of the matching remain the same, but the number of matched heavy vertices increases by one. This contradicts the selection of M. Thus, all of the heavy vertices in M are matched.

Let P be a maximal alternative path in T. We know that both ends of P are leaves. On the other hand, since M is a maximum matching, at least one end of P is matched. This means that, at least one leaf is matched in M which completes the proof. □

Lemma B.1 shows that $\frac{h_F+1}{2}$ approximate the maximum matching size with a factor of 2. Thus, if we estimate the number of heavy vertices with a factor of $1+\epsilon$ we can estimate the maximum matching size with a factor of $2(1+\epsilon)$. Later, we use subroutine 2 to estimate the number of heavy vertices.

Lemma B.2 generalize Lemma B.1 to the forests with no isolated vertices.

Lemma B.2. *Let F be a forest with no isolated vertex and with maximum matching size M^*. We have the following bounds:*

- *Upper bound: $M^* \leq h_F + c_F$; and*
- *Lower bound: $M^* \geq \frac{h_F+c_F}{2}$*
- *Lower bound: $M^* \geq c_F$*

where c_F is the number of connected components in F.

Proof. Let $T_1, T_2, \ldots, T_{c_F}$ be the connected components of F. Let M_i^* be the size of the maximum of T_i. For all $1 \leq i \leq c_F$ we know that $\frac{h_{T_i}+1}{2} \leq M_i^* \leq h_{T_i} + 1$. By summing it up over all i we have $\frac{h_F+c_F}{2} \leq M^* \leq h_F + c_F$. In addition, since we do not have any isolated vertex, each connected component contains at least one edge. This give us $M^* \geq c_F$ which completes the proof. □

The number of connected components in a forest is exactly $n - m$ where n is the number of vertices and m is the number of edges [3]. Thus, similar to the trees, if we estimate the number of heavy vertices within a $1+\epsilon$ factor we can estimate the maximum matching size within a $2(1+\epsilon)$ factor. In the following algorithm we reuse Subroutine 2 and estimate the maximum matching size of a forest with no isolated vertex.

¡¡¡¡¡¡¡ .mine

Lemma B.3. *=======*

Lemma B.4. *¿¿¿¿¿¿¿ .r109 For Subroutine 2 with parameters $c_2 = 0.5$, $\alpha_2 = 6\epsilon^{-2}$ and $\alpha_3 = 2$, we have:*

Algorithm 2 Maximum Matching Estimator For Forest

Input: A stream of edges of the forest $e_1, \ldots, e_m$.

Output: An estimation M of the size of maximum matching M^*.

Initialization Process:

Set $\alpha_1 = 5$, $\alpha_2 = 6\epsilon^{-2}$, $\alpha_3 = 2$ and $Q = 21\lceil \ln(\frac{1}{\delta}) \rceil + 1$.
Set $c_1 = c_2 = 0.5$.
Initialize Subroutine 1 with α_1 and c_1 factors.
Initialize Q independent executions of Subroutines 2, with α_2 and c_2 factors.

Update Process, upon the arrival of $e_i = (u, v)$:

Update each subroutine with the new arrival.

Termination Process:

Let R denote the return value of Subroutine 1.
For $i \in [Q]$, let $\hat{h}_i$ denote the value returned by the i-th execution of Subroutine 2.
Let h_{med} denote the median value of $\hat{h}_i$'s.
Let n donate the number of vertices and m donate the number of edges
if $R < \alpha_1\sqrt{n}$ **then**
    Output $M = R$.
else if $h_{\text{med}} \leq \alpha_3\sqrt{n}$ **then**
    Output $M = n - m$.
else
    Output $M = \frac{h_{\text{med}}+n-m}{2(1+\epsilon)}$.

- *If* $h \geq \sqrt{n}$, $\Pr\left[\left|\hat{h} - h\right| \geq \epsilon h\right] \leq \frac{1}{e}$; *and*
- *If* $h < \sqrt{n}$, $\Pr\left[\hat{h} > \alpha_3\sqrt{n}\right] \leq \frac{1}{e}$.

Proof. Recall in the subroutine we select a random sample V' of size $n' = \alpha_2\sqrt{n}$. Let $V' = \{v'_1, \ldots, v'_n\}$. For $i \in [n']$, let X_i denote the random variable where $X_i = 1$ if v'_i is a heavy vertex, and $X_i = 0$ otherwise. Since we have h heavy vertices, we have that $\Pr[X_i = 1] = \frac{h}{n}$ for every $i \in [n']$. Let $h' = \sum_i X_i$. Clearly $\mathrm{E}[h'] = \frac{n' \cdot h}{n}$. Recall that the return value of Subroutine 2 is $\hat{h} = h' \times \frac{n}{n'}$. Thus in expectation, we return the value of h. Now since all X_i's are negatively correlated, we use the extension of Chernoff bound E.1 to prove the first bound of the lemma for

$h \geq \sqrt{n}$.

$$\begin{aligned}
\Pr\left[\left|\hat{h} - h\right| \geq \epsilon h\right] &= \Pr\left[\left|\hat{h} - \mathrm{E}\left[\hat{h}\right]\right| \geq \epsilon\, \mathrm{E}\left[\hat{h}\right]\right] \\
&= \Pr\left[\left|h' - \mathrm{E}\left[h'\right]\right| \geq \epsilon\, \mathrm{E}\left[h'\right]\right] && \hat{h} = h' \times \frac{n}{n'} \\
&\leq 2\exp\left(-\frac{\epsilon^2\, \mathrm{E}\left[h'\right]}{3}\right) && \text{Chernoff bound extension} \\
&= 2\exp\left(-\frac{\epsilon^2 \cdot n' \cdot h}{3n}\right) && \mathrm{E}\left[h'\right] = \frac{n' \cdot h}{n} \\
&= 2\exp\left(-2\frac{h}{\sqrt{n}}\right) && n' = \frac{6}{\epsilon^2} \cdot \sqrt{n} \\
&\leq \frac{2}{e^2} && h \geq \sqrt{n} \\
&\leq \frac{1}{e}
\end{aligned}$$

We again use the Chernoff bound to prove the lemma for $h < \sqrt{n}$.

$$\begin{aligned}
\Pr\left[\hat{h} > \alpha_3\sqrt{n}\right] &\leq \Pr\left[\hat{h} > \left(\frac{\alpha_3\sqrt{n}}{h}\right)\mathrm{E}\left[\hat{h}\right]\right] && \mathrm{E}\left[\hat{h}\right] = h \\
&= \Pr\left[h' > \left(\frac{\alpha_3\sqrt{n}}{h}\right)\mathrm{E}\left[h'\right]\right] && \hat{h} = h' \times \frac{n}{n'} \\
&\leq \Pr\left[h' > \left(1 + \frac{\alpha_3\sqrt{n}}{2h}\right)\mathrm{E}\left[h'\right]\right] \\
&\leq \exp\left(-\frac{\alpha_3^2 \cdot n \cdot \mathrm{E}\left[h'\right]}{12h^2}\right) && \text{Chernoff bound extension} \\
&= \exp\left(-\frac{\alpha_3^2 \cdot n \cdot n' \cdot h}{12h^2 \cdot n}\right) && \mathrm{E}\left[h'\right] = \frac{n' \cdot h}{n} \\
&= \exp\left(-\frac{\alpha_3^2 \cdot n'}{12h}\right) \\
&= \exp\left(-\frac{\alpha_3^2 \cdot \alpha_2 \cdot \sqrt{n}}{12h}\right) && n' = \alpha_2\sqrt{n} \\
&\leq \exp\left(-\left(\frac{\alpha_3}{2}\right)^2\right) && h < \sqrt{n}, \alpha_2 \geq 3 \\
&= \frac{1}{e} && \alpha_3 = 2
\end{aligned}$$

□

proof of theorem 1.3. When the maximum matching size is less than $\alpha_1\sqrt{n}$, we report the size of a maximal matching of F. This is clearly estimates the maximum matching size within a factor of 2. Thus, we can assume the maximum matching size is at least $\alpha_1\sqrt{n}$. First, we assume the following two claims and prove the theorem in three cases. Later, we provide the proofs of the claims.

Claim B.5. *If* $h < \sqrt{n}$, $\Pr\left[h_{\mathrm{med}} > \alpha_3\sqrt{n}\right] \leq \delta$

Claim B.6. *If* $h \geq \sqrt{n}$, $\Pr\left[|h_{\mathrm{med}} - h| \geq \epsilon h\right] \leq \delta$

Case 1: $\mathbf{h < \sqrt{n}}$. In this case using Claim B.5, with probability $1-\delta$ we have $h_{\text{med}} \leq \alpha_3\sqrt{n}$ and the output is $c_F = n - m$. Thus, with probability $1-\delta$, we have

$$\begin{aligned} M^* &\leq h_F + c_F && \text{Lemma B.2} \\ &\leq \sqrt{n} + c_F && h_F \leq \sqrt{n} \\ &\leq \frac{M^*}{\alpha_1} + c_F. && M^* \geq \alpha_1\sqrt{n}. \end{aligned}$$

This combined with $M^* \geq c_F$, says that the output approximate the maximum matching size within a $\frac{\alpha_1}{\alpha_1 - 1} = \frac{5}{4}$ factor.

Case 2: $\mathbf{h \geq \sqrt{n}}$ and $\mathbf{h_{\text{med}} < \alpha_3\sqrt{n}}$. In this case using Claim B.6, with probability $1-\delta$ we have $h_F - \epsilon h_F \leq h_{\text{med}}$. This together with $h_{\text{med}} < \alpha_3\sqrt{n}$ gives us $h_F - \epsilon h_F < \alpha_3\sqrt{n}$. Therefore, with probability $1-\delta$ we have

$$\begin{aligned} M^* &\leq h_F + c_F && \text{Lemma B.2} \\ &\leq \frac{\alpha_3}{1-\epsilon}\sqrt{n} + c_F && h_F - \epsilon h_F < \alpha_3\sqrt{n} \\ &\leq \frac{\alpha_3 M^*}{\alpha_1(1-\epsilon)} + c_F. && M^* \geq \alpha_1\sqrt{n}. \end{aligned}$$

This combined with $M^* \geq c_F$, says that the output approximate the maximum matching size within a $\frac{\alpha_1(1-\epsilon)}{\alpha_1(1-\epsilon)-\alpha_3} \leq 2$ factor, assuming $\epsilon \leq 1/5$.

Case 3: $\mathbf{h \geq \sqrt{n}}$ and $\mathbf{h_{\text{med}} \geq \alpha_3\sqrt{n}}$. In this case using Claim B.6, with probability $1-\delta$ we have $h_F - \epsilon h_F \leq h_{\text{med}}$. Therefore, with probability $1-\delta$ we have

$$\begin{aligned} M^* &\leq h_F + c_F && \text{Lemma B.2} \\ &\leq \frac{h_{\text{med}}}{1-\epsilon} + c_F \\ &\leq \frac{h_{\text{med}} + c_f}{1-\epsilon} \\ &= \frac{h_{\text{med}} + c_f}{2(1+\epsilon)} \frac{2(1+\epsilon)}{1-\epsilon} \\ &\leq \frac{h_{\text{med}} + c_f}{2(1+\epsilon)} 2(1+3\epsilon) && \epsilon \leq 1/5 \end{aligned}$$

and at the same time we have

$$\begin{aligned} M^* &\geq \frac{h_F + c_F}{2} && \text{Lemma B.2} \\ &\geq \frac{h_{\text{med}}/(1+\epsilon) + c_F}{2} && \text{Claim B.6} \\ &\geq \frac{h_{\text{med}} + c_f}{2(1+\epsilon)}. \end{aligned}$$

This means that the output approximate the maximum matching size within a $2(1+3\epsilon)$ factor, assuming $\epsilon \leq 1/5$.

Proof of Claim B.5. ¡¡¡¡¡¡¡ .mine Let X_i donate the random variable where $X_i = 1$, if $\hat{h}_i > \alpha_3\sqrt{n}$ and $X_i = 0$ otherwise. Form Lemma B.3 we know that the probability of $X_i = 1$ is $1/e$. On the other hand, if $h_{med} > \alpha_3\sqrt{n}$ at least half of the h_i's are greater than $\alpha_3\sqrt{n}$. Thus, we have $\Pr\left[h_{med} > \alpha_3\sqrt{n}\right] \leq \Pr\left[\sum_{i\in Q} X_i \geq 0.5|Q|\right]$. This probability can be bounded by Chernoff bound as follow. ======= Let X_i donate the random variable where $X_i = 1$, if $\hat{h}_i > \alpha_3\sqrt{n}$ and $X_i = 0$ otherwise. Form Lemma B.4 we know that the probability of $X_i = 1$ is $1/e$. On the

other hand, if $h_{\text{med}} > \alpha_3\sqrt{n}$ at least half of the h_i's are greater than $\alpha_3\sqrt{n}$. Thus, we have $\Pr\left[h_{\text{med}} > \alpha_3\sqrt{n}\right] \leq \Pr\left[\sum_{i\in Q} X_i \geq 0.5|Q|\right]$. This probability can be bounded by Chernoff bound as follow. ¿¿¿¿¿¿¿ .r109

$$\Pr\left[h_{\text{med}} > \alpha_3\sqrt{n}\right] \leq \Pr\left[\sum_{i\in Q} X_i \geq 0.5|Q|\right] \leq \exp\left(-\frac{(1-\frac{e}{2})^2\frac{1}{e}21\ln(\frac{1}{\delta})}{3}\right) \leq \delta$$

□

Proof of Claim B.5. ¡¡¡¡¡¡¡ .mine Let X_i donate the random variable where $X_i = 1$, if $|h_i - h| \geq \epsilon h$ and $X_i = 0$ otherwise. Form Lemma B.3 we know that the probability of $X_i = 1$ is $1/e$. Again here, if $|h_{med} - h| \geq \epsilon h$ at least half of the h_i's are greater than $\alpha_3\sqrt{n}$. Hence, again we have ======= Let X_i donate the random variable where $X_i = 1$, if $|h_i - h| \geq \epsilon h$ and $X_i = 0$ otherwise. Form Lemma B.4 we know that the probability of $X_i = 1$ is $1/e$. Again here, if $|h_{\text{med}} - h| \geq \epsilon h$ at least half of the h_i's are greater than $\alpha_3\sqrt{n}$. Hence, again we have ¿¿¿¿¿¿¿ .r109

$$\Pr\left[|h_{\text{med}} - h| \geq \epsilon h\right] \leq \Pr\left[\sum_{i\in Q} X_i \geq 0.5|Q|\right] \leq \exp\left(-\frac{(1-\frac{e}{2})^2\frac{1}{e}21\ln(\frac{1}{\delta})}{3}\right) \leq \delta.$$

□

□

C Lower Bounds for $f_{\text{det}}(.)$ and $f_{\text{rnd}}(.)$

We start by proving a lower bound for the deterministic variant. Next we state our conjecture for the randomized variant. We also show that our conjecture holds for a restricted model of policies.

Proof of Theorem 4.3. Let $x, y \in \{0,1\}^{4n}$ be two vectors chosen independently at random. We say x and y are *orthogonal* if for every possible pair of bits (b, b'), for almost $1/4$ of indices i, we have $x_i = b$ and $y_i = b'$. More formally,

$$x \perp y \quad \textit{iff} \quad \forall (b,b') \in \{0,1\}^2 \ : \ \left|\{i \in [4n] : (x_i, y_i) = (b,b')\}\right| \in n(1 \pm (1-\beta))$$

For two orthogonal vectors x and y, there is a matching between the indices for which Bob cannot deterministically output the correct parity: for every pair of bits (b, b'), let $I(b, b') \subset [4n]$ denote the first βn indices that (x, y) agrees with (b, b'). The matching H comprise (i) a one-to-one matching between $I(0, 0)$ and $I(0, 1)$, (ii) a one-to-one matching between $I(1, 0)$ and $I(1, 1)$, and an arbitrary matching between the rest of indices. Observe that the parity of H for x is zero while its parity is one for y. Thus Bob cannot deterministically distinguish between x and y.

Now observe that for every pair of bits (b, b'), the expected number of indices that (x, y) agrees with (b, b') is n. Using the Chernoff bound, for every pair of bits one can bound each of the upper and lower tails by $\exp(-((1-\beta)^2 n)/3)$. Therefore by a union bound we have

$$\Pr[x \perp y] \geq 1 - 8\exp(-((1-\beta)^2 n)/3)$$

Let X denote a maximum-cardinality set of pairwise-orthogonal vectors. Let $p = 8\exp(-((1-\beta)^2 n)/3)$. Since a random pair of vertices are orthogonal with probability $1 - p$, one can apply Turan's theorem[6] to show that $|X| \geq \frac{2}{p} = \frac{\exp(((1-\beta)^2 n)/3)}{4}$. As described above, in the deterministic

[6]Turan's theorem states that a K_{r+1}-free graph with n vertices has at most $(1 - \frac{1}{r}) \cdot \frac{n^2}{2}$ edges.

scenario, Bob cannot distinguish between two orthogonal vectors. Therefore Alice cannot use the same message for two members of X, leading to the lower bound of

$$f_{\text{det}}(n,\beta) \geq \log_2(|X|) \geq \frac{(1-\beta)^2 n}{3\ln(2)} - 2 = \Omega((1-\beta)^2 n)$$

□

Theorem 4.3 together with Corollary 4.2 implies a lower bound of $\Omega(n)$ for the deterministic variant of the problem of estimating the size of the maximum matching. Let us now consider $f_{\text{rnd}}(n,1)$. Suppose that we restrict the communication protocol so that Alice's message simply reveals s bits of x. We note that since $\beta = 1$, this setting is similar to a Birthday Paradox problem: Alice's goal is to hit an edge in H by revealing s bits of x. In fact, the lower bounds are also similar to that of Birthday Paradox.

Lemma C.1. *Suppose Alice reveals s bits of x. Bob cannot always output the correct θ if $s \leq n$. Furthermore, Bob cannot output the correct θ with probability more than $3/4$, if $s \leq \frac{\sqrt{n}}{2}$.*

Proof. Let S_0 and S_1 denote the set of known bits with value 0 and 1, respectively. Without less of generality, we may assume $|S_0| \geq |S_1|$ in all the messages. Let $s' = |S_0|$ and $S = S_0 \cup S_1$.

The claim for the deterministic setting is not hard to verify. Suppose $s' \leq n$. It may happen that none of the edges of H are induced by S. In such a scenario, Bob cannot determine the correct value of θ: all elements of S can be matched to bits with the same parity, or a different parity. Therefore for the success probability of one, it is required that $s' > n$.

We now focus on the randomized setting, in which S is chosen randomly from a distribution π over subsets of size s. We note that by symmetry, if $x[i] = x[j]$ we can assume $\Pr[i \in S] = \Pr[j \in S]$, for every $i, j \in [4n]$. In fact, the same applies for subsets of indices with the same value. For a subset $S \subseteq [4n]$, let r denote the number of elements of S with value 0. By symmetry, the random selection process can be described as following. For every $r \in [0 \dots s]$, a probability π_r is chosen such that $\sum_r \pi_r = 1$. We first draw the value r from the distribution π. We then select a subset S_0 (resp. S_1) of size r (resp. $s-r$) uniformly at random from the set of indices with value 0 (resp. 1). Now if the set $S = S_0 \cup S_1$ contains an edge of the matching H, Bob can easily output the correct θ. Therefore, let us compute the probability of *not hitting an edge*. We assume that $n \geq 4$. If $\theta = 0$, then from every edge between the indices with value 0, we can only choose one endpoint. Hence we have

$$\begin{aligned}
\Pr[\text{hitting no edge} \mid \theta = 0, r = t] &= \frac{2^t \binom{n}{t} 2^{s-t} \binom{n}{s-t}}{\binom{2n}{t}\binom{2n}{s-t}} \\
&\geq \left(\frac{2n-2(t-1)}{2n-(t-1)}\right)^t \left(\frac{2n-2(s-t-1)}{2n-(s-t-1)}\right)^{s-t} \\
&\geq \exp(-\frac{(2t)(t-1)}{2n-t+1}) \exp(-\frac{2(s-t)(s-t-1)}{2n-s+t+1}) \\
&\geq \exp(-\frac{(2t)(t-1)}{2n-s+1}) \exp(-\frac{2(s-t)(s-t-1)}{2n-s+1}) \\
&\geq \exp(-\frac{(2s)(s-1)+2(s)(s-1)}{2n-s+1}) = \exp(-\frac{4s^2}{2n-s+1}) \\
&\geq \exp(-\frac{n}{1.5n}) > 1/2
\end{aligned}$$

On the other hand, if $\theta = 1$, the edges are between indices with different values. Thus we have:

$$\begin{aligned}\Pr[\text{hitting no edge} \mid \theta = 1, r = t] &= \frac{\binom{2n-s+t}{t}}{\binom{2n}{t}} \\ &= \frac{(2n-s+t)\cdots(2n-s+1)}{(2n)\cdots(2n-t+1)} \geq \left(\frac{2n-s+1}{2n-t+1}\right)^t \\ &= \left(1 - \frac{s-t}{2n-t+1}\right)^t \geq \exp(-2t\frac{s-t}{2n-t+1}) \\ &\geq \exp(-\frac{2s^2}{2n-s+1}) \geq \exp(-\frac{n}{2(2n-s)}) > 1/2\end{aligned}$$

Since the lower bound of $1/2$ is independent of the random variable r, this implies that for every distribution π, we have

$$\Pr[\text{hitting no edge} \mid \theta = 0] > 1/2 \qquad \Pr[\text{hitting no edge} \mid \theta = 1] > 1/2$$

Therefore if $s \leq \sqrt{n}/2$, no randomized algorithm can output the correct θ with probability more than $3/4$. □

Indeed we believe that the bounds shown in Lemma C.1 are asymptotically tight for the Matching Parity problem. Therefore we leave Conjecture 4.4 as an open problem.

D 2-Pass and Random Order Streaming Algorithms

Here we approximate the number of shallow edges within $(1 \pm \epsilon)$ factor using $\tilde{O}(\sqrt{n})$ space. Observe that in Lemma A.1 we can replace $h \geq n^{2/3}$ by $h \geq n^{1/2}$ and the lemma works with constant probability. Therefore, we can approximate h and s using $\tilde{O}(\sqrt{n})$ space.

D.1 2-Pass Streaming Algorithm

Subroutine 4. [Shallow Edges using 2 Passes]

Input: *A stream of edges of the graph $e_1, \ldots, e_m$.*
Return Value: *$\hat{s}$, an estimation for the number of shallow edges in the graph.*
First Pass:

1: *Let S be a random sample set of edges chosen with probability $\frac{3\log(4/\delta)}{\epsilon^2\sqrt{n}}$.*

Second Pass:

1: ***for*** *each edge $(u, v) \in S$* ***do***
2: ***if*** *(u, v) is a shallow edge* ***then***
3: *Add (u, v) to set E' which is initialized to zero in the beginning of this pass.*

Termination Process:

1: *Let $s' = |E'|$ denote the number of shallow edges sampled using our algorithm.*
2: *Return $\hat{s} = s' \times \frac{\epsilon^2\sqrt{n}}{3\log(4/\delta)}$.*

Analysis.

Lemma D.1. *For Subroutine 4, we have* $\Pr\left[(1-\epsilon)\cdot|S| \leq \hat{s} \leq (1+\epsilon)\cdot|S|\right] \geq 1-\delta/2$.

Proof. Let $\mathcal{S}$ be the set of shallow edges. Suppose that $|S| \geq \sqrt{n}$. If $|S| \leq \sqrt{n}$, we either maintain a maximal matching if the size of matching is $\tilde{O}(n^{1/2})$ or charge the number of shallow edges to the number of heavy vertices if the number of them is $\tilde{\Omega}(n^{1/2})$. Thus, we do not elaborate on them here. For $i \in [|S|]$, let X_i denote the random variable where $X_i = 1$ if $e_i \in S$, and $X_i = 0$ otherwise. We then have, $\Pr[X_i] = \frac{3\log(4/\delta)}{\epsilon^2\sqrt{n}}$. Let $s' = \sum_{i\in[|S|]} X_i$. Thus, $\mathrm{E}[s'] = \mathrm{E}\left[\sum_{i\in[|S|]} X_i\right] = \frac{3\log(4/\delta)\cdot|S|}{\epsilon^2\sqrt{n}} \geq \frac{3\log(4/\delta)}{\epsilon^2}$. We then use the Chernoff bound to prove that

$$\Pr\left[\left|s' - \mathrm{E}\left[s'\right]\right| \geq \epsilon\cdot\mathrm{E}\left[s'\right]\right] \leq 2\exp\left(-\frac{\epsilon^2\,\mathrm{E}[s']}{3}\right) \leq \delta/2 \ .$$

Let us condition on the event that $|s' - \mathrm{E}[s']| \leq \epsilon\cdot\mathrm{E}[s']$ which happens with probability at least $1-\delta/2$. Since,

$$(1-\epsilon)\cdot\mathrm{E}\left[s'\right]\times\frac{\epsilon^2\sqrt{n}}{3\log(4/\delta)} \leq \hat{s} \leq (1+\epsilon)\cdot\mathrm{E}\left[s'\right]\times\frac{\epsilon^2\sqrt{n}}{3\log(4/\delta)} \ ,$$

we obtain $(1-\epsilon)\cdot|S| \leq \hat{s} \leq (1+\epsilon)\cdot|S|$. □

D.2 Random Order Streams

Subroutine 5. [Shallow Edges in Random-Order Streams]

Input: *A stream of edges of the graph* $e_1,\ldots,e_m$.
Return Value: $\hat{s}$, *an estimation for the number of shallow edges in the graph.*

1: *Let* S *be the first* $\frac{3\log(4/\delta)}{\epsilon^2\sqrt{n}}$ *edges of the stream.*
2: ***for*** *each edge* $(u,v) \in S$ ***do***
3: ***if*** (u,v) *is a shallow edge* ***then***
4: *Add* (u,v) *to set* E' *which is initialized to zero in the beginning of this pass.*

Termination Process:

1: *Let* $s' = |E'|$ *denote the number of shallow edges sampled using our algorithm.*
2: *Return* $\hat{s} = s' \times \frac{\epsilon^2\sqrt{n}}{3\log(4/\delta)}$.

Similar to Lemma D.1 we prove the following lemma.

Lemma D.2. *For Subroutine 5, we have* $\Pr\left[(1-\epsilon)\cdot|S| \leq \hat{s} \leq (1+\epsilon)\cdot|S|\right] \geq 1-\delta/2$.

E Concentration Bounds

We use an extension of the Chernoff-Hoeffding bound for negatively correlated Boolean variables. It was first proved by Panconesi and Srinivasan [29].

Theorem E.1 ([29])**.** *If the Boolean random variables* X_1, X_2, …, X_r *are negatively correlated then for* $X = \sum_{i=1}^{r} X_i$, $\mu = E[X]$, *and* $0 < \delta \leq 1$, *we have* $\Pr[X < (1-\delta)\mu] < e^{-\mu\delta^2/2}$.

We also use Chebyshev's inequality, which we state here for completeness.

Theorem E.2 (Chebyshev's Inequality)**.** *Let* X *be a random variable with finite expected value* μ *and finite non-zero variance* σ^2, *we have* $\Pr[|X-\mu| \geq k\sigma] \leq \frac{1}{k^2}$.