Weighted Matching in the Random-Order Streaming and Robust Communication Models

Diba Hashemi
EPFL, Lausanne, Switzerland

Weronika Wrzos-Kaminska[1]
EPFL, Lausanne, Switzerland

Abstract

We study the maximum weight matching problem in the random-order semi-streaming model and in the robust communication model. Unlike many other sublinear models, in these two frameworks, there is a large gap between the guarantees of the best known algorithms for the unweighted and weighted versions of the problem.

In the random-order semi-streaming setting, the edges of an n-vertex graph arrive in a stream in a random order. The goal is to compute an approximate maximum weight matching with a single pass over the stream using $O(n \operatorname{polylog} n)$ space. Our main result is a $(2/3 - \epsilon)$-approximation algorithm for maximum weight matching in random-order streams, using space $O(n \log n \log R)$, where R is the ratio between the heaviest and the lightest edge in the graph. Our result nearly matches the best known unweighted $(2/3 + \epsilon_0)$-approximation (where $\epsilon_0 \sim 10^{-14}$ is a small constant) achieved by Assadi and Behnezhad [6], and significantly improves upon previous weighted results. Our techniques also extend to the related robust communication model, in which the edges of a graph are partitioned randomly between Alice and Bob. Alice sends a single message of size $O(n \operatorname{polylog} n)$ to Bob, who must compute an approximate maximum weight matching. We achieve a $(5/6 - \epsilon)$-approximation using $O(n \log n \log R)$ words of communication, matching the results of Azarmehr and Behnezhad [20] for unweighted graphs.

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

Keywords and phrases Maximum Weight Matching, Streaming, Random-Order Streaming, Robust Communication Complexity

Digital Object Identifier 10.4230/LIPIcs.APPROX/RANDOM.2024.16

Category APPROX

Related Version *Full Version*: `https://arxiv.org/abs/2408.15434`

Acknowledgements We thank Ola Svensson and Michael Kapralov for helpful discussions. We additionally thank Michael Kapralov for useful comments on the manuscript.

1 Introduction

The maximum matching problem is a fundamental problem in graph algorithms. In the unweighted version of the problem, we are interested in computing a maximum *cardinality* matching, i.e. to maximize the total number of edges in the matching. In the weighted version, we are interested in computing a maximum *weight* matching, i.e. to maximize the sum of the edge weights in the matching.

In this paper, we study matchings in the semi-streaming model. The semi-streaming model, originally introduced in [42], is motivated by the rise of massive graphs where the data is too large to be stored in memory, and has received extensive attention (see among others

[1] Corresponding author

[69, 38, 48, 78, 58, 35, 73, 47, 59]). In this model, the edges of a graph arrive sequentially as a stream. The algorithm typically makes a single pass over the stream using space $O(n \operatorname{polylog} n)$, and must output an approximate maximum matching at the end of the stream. If the graph is unweighted, the greedy algorithm trivially gives a 1/2-approximation, which is the best known for adversarially ordered streams. On the hardness side, it is known that a 0.59-approximation is not possible [59] (see also [48, 58]). Closing the gap between these upper and lower bounds is one of the major open problems in the graph streaming literature. There has also been a long line of work on the weighted problem [42, 69, 38, 78, 35, 73, 47], culminating in a $(1/2 - \epsilon)$-approximation using space $O(n)$ [73, 47].

Recently, there has been a wide interest in the random-order version of this problem, in which the arrival order of the edges is chosen uniformly at random. This problem has been extensively studied in the unweighted setting [64, 63, 5, 45, 41, 22, 6, 18]. Notably, Bernstein [22] gave a 2/3-approximation, and Assadi and Behnezhad [6] improved it to $(2/3 + \epsilon_0)$ for a small constant $\epsilon_0 \sim 10^{-14}$.

Progress on the weighted version of the problem lags behind. Gamlath et al. [45] broke the barrier of 1/2 in weighted graphs by obtaining a $(1/2 + \delta)$-approximation for a small constant $\delta \sim 10^{-17}$. More recently, Huang and Sellier [54] gave a $\frac{1}{2-1/(2W)}$-approximation under the assumption that the weights take integral values in $[W]$. This leaves a considerable gap between the best known results for the unweighted and weighted versions of the problem. In contrast, in other sublinear contexts, such as adversarially ordered streams or the dynamic graph setting, the weighted/unweighted gap has largely been closed [23]. The challenge of closing the gap in random-order streams remains an open problem, and has been highlighted explicitly in [22] and [23].

In this paper, we give a $(\frac{2}{3} - \epsilon)$-approximation algorithm for the weighted setting. Our result almost matches the best known $(\frac{2}{3} + \epsilon_0)$-guarantee for the unweighted setting, and improves significantly upon the previous results for the weighted setting.

▶ **Theorem 1.1.** *Given any constant $\epsilon > 0$, there exists a deterministic single-pass streaming algorithm that with high probability computes a $(\frac{2}{3} - \epsilon)$-approximate maximum weight matching if the edges arrive in a uniformly random order. The space usage of the algorithm is $O(n \log n \log R)$, where R is the ratio between the heaviest and the lightest edge weight in the graph.*

We also consider the two-player communication complexity model [77], and in particular the one-way communication complexity of matching, which was first studied in [48]. Here, the edge-set is partitioned between two parties Alice and Bob. Alice sends a single message to Bob, who must output an approximate maximum matching. Typically, we are interested in protocols with communication complexity $O(n \operatorname{polylog} n)$.

If the edges are partitioned adversarially between Alice and Bob, the right answer turns out to be 2/3. A 2/3-approximation can be achieved using $O(n)$ communication for both bipartite unweighted [48], general unweighted [9] and general weighted [23] graphs. Going beyond a 2/3-approximation requires $n^{1+1/(\log \log n)} \gg n \operatorname{polylog} n$ communication even for unweighted bipartite graphs [48].

If instead the edges are partitioned randomly between the two parties, the answer is less clear. Recently, Azarmehr and Behnezhad [20] gave a 5/6-approximation algorithm for unweighted graphs, improving upon a previous result of Assadi and Behnezhad [7]. To the best of our knowledge, prior to our work there were no results for weighted graphs (besides the 2/3-approximation implied by adversarial protocols). We match the unweighted guarantees of Azarmehr and Behnezhad [20], thus closing weighted/unweighted gap in the robust communication complexity model.

▶ **Theorem 1.2.** *Given any constant $\epsilon > 0$, there exists a protocol that with high probability computes a $(\frac{5}{6} - \epsilon)$-approximate maximum weight matching in the two-party robust communication model using $O(n \log n \log R)$ words of communication, where R is the ratio between the heaviest and the lightest edge weight in the graph.*

More generally, we match the results of Azarmehr and Behnezhad [20] for unweighted k-party robust communication, thus closing the unweighted/weighted gap also in this model.

▶ **Theorem 1.3.** *Given any $k \geq 2$ and any constant $\epsilon > 0$, there exists a protocol that with high probability computes a $(\frac{2}{3} + \frac{1}{3k} - \epsilon)$-approximate maximum weight matching in the k-party one-way robust communication model using $O(n \log n \log R)$ words of communication, where R is the ratio between the heaviest and the lightest edge weight in the graph.*

1.1 Related Work

The maximum matching problem is one of the most studied problems in the streaming setting, with numerous lines of work. This includes among others single-pass algorithms [42, 69, 38, 48, 78, 73, 47, 58, 35, 59, 8], multi-pass algorithms using 2 or 3 passes [64, 40, 56, 63, 65, 43, 3, 67, 66], and $(1 - \epsilon)$-approximation using a higher number of passes [69, 37, 1, 53, 2, 75, 45, 14, 10, 44, 18, 55, 4]. Garg et al. considered matching in a robust random-order streaming model with adversarial noise [46]. There are many results on dynamic streams, where edges can be deleted [33, 62, 12, 32, 11, 36, 17]. A different line of work considers estimating matching size, either in random-order streams [60, 28, 72, 61, 19] or in adversarially ordered streams [28, 70, 11, 34, 39, 71, 27, 13, 15]. Finally, there have also been several works on exact matching [42, 32, 53, 16, 30, 10].

2 Technical Overview

In this paper, we are interested in the random-order streaming model. The maximum cardinality matching problem has gained significant attention within this framework [64, 63, 45, 5, 41, 22, 6]. Bernstein [22] gave a 2/3-approximation algorithm by adapting the "matching sparsifier" Edge-Degree Constrained Subgraph (EDCS) to the streaming context. Subsequent work by Assadi and Behnezhad [6] improved upon this, achieving a $(2/3 + \epsilon_0)$-approximation by simultaneously running Bernstein's algorithm while identifying short augmenting paths. One of the motivations for studying the random-order setting, is that real-world data is rarely ordered adversarially. Rather, in most practical applications, it is reasonable to assume that the data is drawn from some distribution. However, assuming uniform randomness is often too strong of an assumption, since data correlations are prevalent in many real-world settings. This raises the question:

How robust are random-order streaming algorithms to correlations in the arrival order?

The robustness of random-order streaming algorithms to various types of adversarial distortions has already been studied previously, among others in the context of maximum matching and submodular maximization [46], rank selection [49, 50, 51], clustering problems [68] and component collection and counting [31]. In this paper, we focus on matchings. Our first contribution, is showing that existing algorithms for unweighted matching in random-order streams are in fact robust to correlations in the arrival order.

Bernstein's $\left(\frac{2}{3} - \epsilon\right)$-approximation algorithm is resilient to (limited) adversarial correlations in the arrival order.

Surprisingly, this immediately gives a reduction from weighted matching in random-order streams.

In *adversarially* ordered streams, Bernstein, Dudeja and Langley [23] gave a reduction from maximum weight matching to maximum cardinality matching. Progress in *random-order streams* has been comparatively limited. Gamlath et al. [45] achieved a $(1/2+\delta)$-approximation, where $\delta \sim 10^{-17}$ is a small constant. More recently, Huang and Sellier [54] gave a $\frac{1}{2-1/(2W)}$-approximation under the assumption that the weights take integral values in $[W]$, improving upon the result of Gamlath et al. [45] for small weights. They generalized the definition of EDCS to weighted graphs, which enabled them to adapt Bernstein's algorithm [22] to weighted graphs. However, their generalized notion of EDCS has weaker guarantees compared to the unweighted version, resulting in a significant loss in the approximation ratio.

Our second contribution is to nearly close the gap between weighted and unweighted maximum matching in random-order streams. We show that the reduction of Bernstein, Dudeja and Langley can be applied to random-order streaming algorithms which are resilient to specific correlations in the arrival order. This, together with the fact that Bernstein's algorithm [22] is robust to the appropriate correlations, gives a 2/3-approximation algorithm for weighted bipartite graphs. We are also able to extend the guarantees to non-bipartite graphs.

2.1 Reduction in Adversarial Streams

First, we review the reduction of Bernstein, Dudeja and Langley [23] for adversarial streams. It is based on a technique called graph unfolding by Kao, Lam, Sung and Ting [57].

▶ **Definition 2.1** (Graph Unfolding [57]). *Let $G=(V,E,w)$ be a graph with non-negative integral edge weights. The unfolded graph $\phi(G)$ is an unweighted graph created as follows. For each vertex $u \in V$, let $W_u = \max_{e \ni u} w_e$ be the maximum edge weight incident on u. There are W_u copies of u in $\phi(G)$, denoted by $u^1, \ldots, u^{W_u}$. For each edge $e=(u,v)$ in G, there are w_e edges $\{(u^i, v^{w_e-i+1})\}_{i\in[w_e]}$ in $\phi(G)$. See Figure 1 for an illustration.*

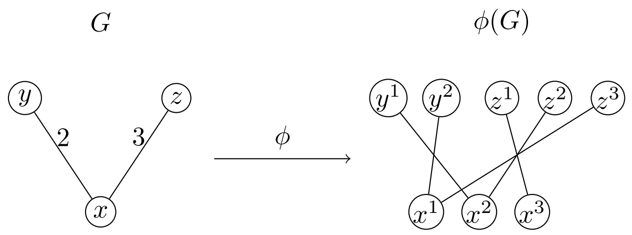

Figure 1 An example of a weighted graph G and its unfolding $\phi(G)$.

One can also do a reverse operation of unfolding to bring a subgraph back to G.

▶ **Definition 2.2** (Refolding [23]). *Let $G=(V,E)$ be a weighted graph and let $H \subseteq \phi(G)$. The refolded graph $\mathcal{R}(H)$ has vertex set V and edge set $E(\mathcal{R}(H)) := \{e=(u,v) \in G : (u^i, v^{w_e-i+1}) \in H \text{ for some } i \in [w_e]\}$. See Figure 2 for an illustration.*

Figure 1 illustrates the unfolding operation and Figure 2 illustrates the refolding operation. The key property of refolding is that it preserves the matching size in bipartite graphs.

▶ **Lemma 2.3** (Refolding preserves matching size in bipartite graphs [23]). *Let G be a weighted bipartite graph, and let $H \subseteq \phi(G)$ be a subgraph of its unfolding. Then $\mu_w(\mathcal{R}(H)) \geq \mu(H)$.*

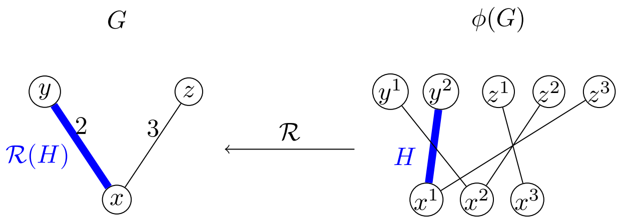

Figure 2 An example of a subgraph $H \subseteq \phi(G)$ and its refolding $\mathcal{R}(H) \subseteq G$. In this example, $H = \{(u^1, v^2)\}$. Then $\mathcal{R}(H) = \{(u, v)\}$.

This leads to a reduction from maximum weight bipartite matching to maximum cardinality bipartite matching in *adversarially ordered streams*: Upon arrival of each weighted edge $e \in G$, unfold e and pass the corresponding unweighted edges $\phi(e)$ into an unweighted streaming algorithm. At the end of the stream we obtain an unweighted matching in $\phi(G)$, which we can refold to obtain a weighted matching in G.

In *random-order streams*, this reduction breaks for the following reason: For each weighted edge $e \in G$, the unweighted edges $\phi(e)$ will necessarily arrive together. This introduces correlations in the arrival order of the edges, so the guarantees of random-order streaming algorithms do not apply. To address this, we consider a new streaming model, the b-batch random-order stream model, which is similar to the hidden-batch model introduced in [31]. This model allows us to capture the edge-correlations that arise from graph unfolding.

▶ **Definition 2.4** (b-batch random-order stream model)**.** *In the b-batch random-order stream model the edge set of the input graph $G = (V, E)$ is presented as follows: An adversary partitions the edge set E into batches $\mathcal{B} = \{B_1, ..., B_q\}$ with $|B_i| \leq b$ for all i. The arrival order of the batches $(B_{i_1}, ..., B_{i_q})$ is then chosen uniformly at random among all the permutations of $\mathcal{B}$. The edges in each batch arrive simultaneously.*

Graph unfolding gives a reduction from weighted bipartite random-order streams to unweighted bipartite b-batch random-order streams. Each batch corresponds to one weighted edge, so given a weighted graph G, we can simply run a b-batch random-order stream algorithm on $\phi(G)$ with batches $\mathcal{B} = \{\phi(e) : e \in G\}$.

2.2 Bernstein's Algorithm for Unweighted Random-Order Streams

We now review Bernstein's algorithm for unweighted random-order streams [22]. The algorithm proceeds in two Phases. Let $\beta = O(\text{poly}(\epsilon^{-1}))$ be a parameter. Phase 1 constructs a subgraph H such that for all $(u, v) \in H$,

$$\deg_H(u) + \deg_H(v) \leq \beta. \qquad (1)$$

Given a subgraph H, we will say that an edge $(u, v) \in G$ is *underfull* if $\deg_H(u) + \deg_H(v) \leq \beta - 2$, otherwise say that (u, v) is non-underfull.

The algorithm constructs H by adding underfull edges in a greedy manner, and then removing any edges that violate Equation 1. Phase 1 terminates when $\approx \text{poly}(\epsilon)\frac{m}{n}$ non-underfull edges arrive in a row, and the algorithm then moves on to Phase 2. Bernstein [22] showed that it is only possible to make at most $n\beta^2$ modifications to H. Since Phase 1 terminates when we see $\approx \text{poly}(\epsilon)\frac{m}{n}$ edges in a row without modifying H, the Phase must terminate within the first $\approx n\beta^2 \cdot \text{poly}(\epsilon)\frac{m}{n} \approx \epsilon m$ edges. This argument also holds in the b-batch random-order stream model.

Then, in Phase 2, the algorithm simply collects all underfull edges into a separate set U (without modifying the graph H). Let G_{late} denote the edges that arrive in Phase 2. Bernstein [22] proved the following structural result about $H \cup U$, which holds regardless of the assumptions on the arrival order:

$$\mu(H \cup U) \geq \left(\frac{2}{3} - \epsilon\right) \mu(G_{late}). \tag{2}$$

Since Phase 2 contains at least a $(1-\epsilon)$ fraction of the edges, and since the stream is uniformly at random, it follows from the Chernoff bound that $\mu(G_{late}) \geq (1-2\epsilon)\mu(G)$. Consequently, by Equation 2, it holds that

$$\mu(H \cup U) \geq \left(\frac{2}{3} - 3\epsilon\right) \mu(G).$$

For the space analysis, observe that H contains at most $n\beta = O(n)$ edges. Let us now consider U. Recall that U is the set of all underfull edges that arrive after the termination of Phase 1, and that Phase 1 terminates when we see $\approx \frac{m}{n}$ non-underfull edges in a row. So the only way for U to become too large, is if we draw $\approx \frac{m}{n}$ non-underfull edges in a row when there are more than $C \cdot n \log n$ underfull edges left in the stream, for some constant C. The probability of this event can be upper-bounded by

$$\left(1 - \frac{C \cdot n \log n}{m}\right)^{m/n} \leq n^{-C},$$

so with high probability, the algorithm stores at most $O(n \log n)$ edges. Note that the space analysis breaks down in the b-batch random-order stream model, due to the correlated arrival orders.

2.3 Applying the Algorithm to Batch Arrivals

We now sketch why Bernstein's algorithm can be adapted to work under batch arrivals. Let b denote the upper-bound on the batch-size, and let q denote the total number of batches in the stream. Recall that in the reduction from weighted random-order streams, b corresponds to the maximum weight in the graph and q corresponds to the number of edges in the weighted graph. We will now describe how to obtain an algorithm with a polynomial space dependence on b. We will later discuss how to remove this dependence in the reduction from weighted random-order streams.

We will say that a batch is *underfull* if it contains at least one underfull edge. Otherwise, if it does not contain any underfull edges, say that it is non-underfull.

We terminate Phase 1 when $\approx \text{poly}(\epsilon)\frac{q}{bn}$ non-underfull batches arrive in a row. This ensures that Phase 1 terminates within the first $\approx n\beta^2 \cdot \text{poly}(\epsilon)\frac{q}{bn} \approx \frac{\epsilon}{b}q$ batches. Since each batch contains at most b edges, and since the arrival order of the batches is uniformly at random, it follows from Chernoff bounds that

$$\mu(G_{late}) \geq (1-2\epsilon)\mu(G).$$

Combining with Equation 2 we obtain

$$\mu(H \cup U) \geq \left(\frac{2}{3} - 3\epsilon\right) \mu(G).$$

The only way for the space usage to become too large, is if $\approx \text{poly}(\epsilon)\frac{q}{b\cdot n}$ non-underfull batches arrive in a row when there are more than $C \cdot n \log n \, \text{poly}(\frac{b}{\epsilon})$ underfull batches left in the stream, for some large constant C. The probability of this event can be upper-bounded by

$$\left(1 - C \cdot \frac{n \log n}{q} \text{poly}\left(\frac{b}{\epsilon}\right)\right)^{\text{poly}(\epsilon/b)q/n} \leq n^{-C},$$

so with high probability, the algorithm stores at most $O(n \log n \, \text{poly}(b))$ edges.

In our reduction, the parameter b corresponds to the maximum edge weight W in the graph. This means that we would incur a polynomial dependence on W in the space usage. However, a reduction due to Gupta and Peng [52] allows us to offset this space dependence. Gupta and Peng [52] devised a scheme for bucketing together edges according to their weight, which gives a reduction from general (possibly non-integral) weights, to integral bounded weights. Combining with this reduction, our algorithm uses space $O(n \log n \log R)$, where R is the ratio between the heaviest and the lightest edge in the graph, and it can handle any (possibly non-integral) edge weights. In particular, the space usage is $O(n \, \text{polylog} \, n)$ as long as the weights are polynomial in n.

2.4 Non-bipartite graphs

In general, the reduction of Bernstein, Dudeja and Langley only holds for bipartite graphs. For non-bipartite graphs, it is no longer true that refolding preserves the matching size, since refolding a matching in $\phi(G)$ can incur an additional 2/3 loss in the approximation ratio. Indeed, consider for example a weighted triangle with all edges of weight 2 (see Figure 3).

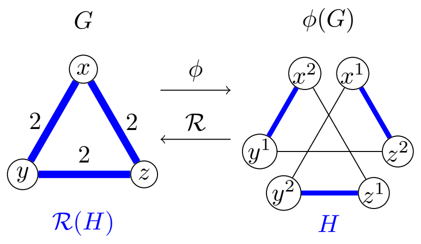

Figure 3 Refolding does not in general preserve matching size in non-bipartite graphs. Consider for example the blue subgraph $H = \{(x^1, z^2), (z^1, y^2), (y^1, x^2)\} \subseteq \phi(G)$ shown in the diagram. Then $\mu(H) = 3$, but $\mu_w(\mathcal{R}(H)) = 2$.

We prove that the subgraph $H \cup U$ computed by Bernstein's algorithm still satisfies $\mu_w(\mathcal{R}(H \cup U)) \geq (2/3 - \epsilon)\mu_w(G)$, even for non-bipartite graphs. This allows us to apply the unfolding reduction without any loss in the approximation ratio. We achieve this by reducing to the bipartite case: We show that for every weighted graph G, there exists a bipartite subgraph $\widetilde{G} \subseteq G$ such that $\mu((H \cup U) \cap \phi(\widetilde{G})) \geq (2/3 - \epsilon)\mu_w(G)$. We can then apply Lemma 2.3 to the bipartite graph $\widetilde{G}$ to get the result.

In order to "bipartify" the graph, we use the following lemma from [23], which says that there exists a bipartite subgraph in which the degrees to H concentrate well (See Lemma 4.10 for the formal statement).

▶ **Lemma 2.5** (Informal version of Lemma 5.7 in [23]). *Let G be a weighted graph and let M^* be a maximum weight matching in G. Suppose that $H \subseteq \phi(G)$ satisfies Equation 1. Then there exists a bipartite subgraph $\widetilde{G} \subseteq G$ such that $\widetilde{G}$ contains M^*, and, setting $\widetilde{H} := H \cap \widetilde{G}$, it holds that*

$$\deg_{\widetilde{H}}(v) \approx \frac{\deg_H(v)}{2} \qquad \forall v \in V.$$

Using this, we will show that $(H \cup U) \cap \phi(\widetilde{G})$ contains an EDCS, and therefore also contains a large matching.

▶ **Definition 2.6** (EDCS [24]). *Let $G = (V, E)$ be an unweighted graph, and $H = (V, E_H)$ a subgraph of G. Given parameters $\beta \geq 2$ and $\lambda < 1$, we say that H is a (β, λ)-EDCS of G if H satisfies the following properties:*

- *(Property P1:) For all edges $(u, v) \in H$, it holds that $\deg_H(u) + \deg_H(v) \leq \beta$.*
- *(Property P2:) For all edges $(u, v) \in G \setminus H$, it holds that $\deg_H(u) + \deg_H(v) \geq \beta(1 - \lambda)$.*

The crucial property of EDCS is that it contains a 2/3-approximate maximum cardinality matching. This was first proved in [24] for bipartite graphs and in [25] for general graphs. See also Lemma 3.2 in [9] for a simpler proof with improved parameters.

▶ **Theorem 2.7** (EDCS contain a 2/3-approximate matching [9]). *Let G be an unweighted graph and let $\epsilon < 1/2$ be a parameter. Let λ, β be parameters with $\lambda \leq \frac{\epsilon}{64}, \beta \geq 8\lambda^{-2}\log(1/\lambda)$. Then, for any (β, λ)-EDCS H of G, we have that $\mu(H) \geq (\frac{2}{3} - \epsilon)\mu(G)$.*

Now consider the weighted input graph G. Fix a maximum weight matching M^* in G and let H be the graph computed by Phase 1 of Bernstein's algorithm on input $\phi(G)$. Let $\widetilde{G} \subseteq G$ be the bipartite subgraph from Lemma 2.5. Ideally, we would like to show that $(H \cup U) \cap \phi(\widetilde{G})$ is an EDCS. However, this is not true in general, since the degrees to U can be arbitrarily large (consider for example the case when U is a star, see Figure 4 for an illustration), so $\deg_{(H \cup U) \cap \phi(\widetilde{G})}$ cannot be upper-bound by a constant. Instead, we will sparsify U, so that its contribution to the degrees becomes insignificant. Let $\widetilde{H} = H \cap \phi(\widetilde{G})$ and let $\widetilde{U} = U \cap \phi(M^*)$ (see Figure 4). This idea is similar to Bernstein's original analysis [22], except that now we perform this sparsification in the unfolded and "bipartified" graph.

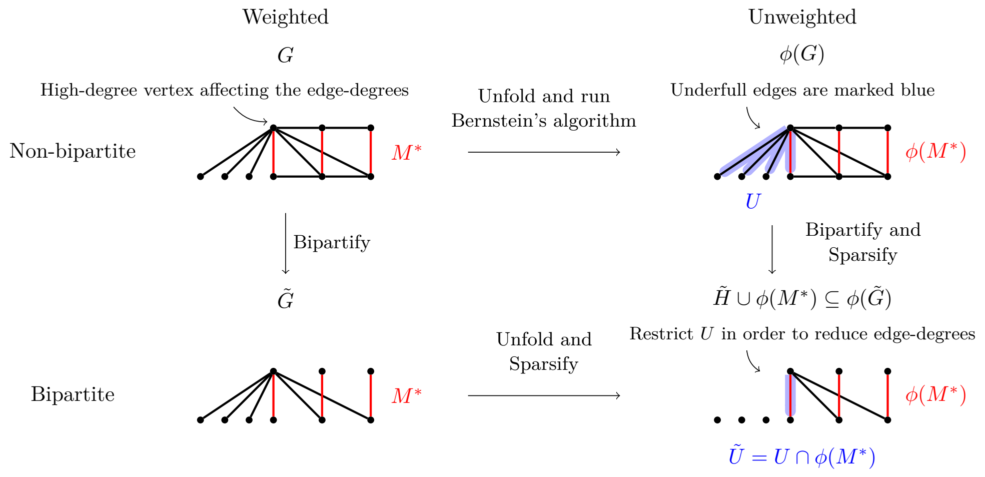
Weighted
G
High-degree vertex affecting the edge-degrees
Non-bipartite
M^*
Unfold and run Bernstein's algorithm
Unweighted
$\phi(G)$
Underfull edges are marked blue
$\phi(M^*)$
U
Bipartify
Bipartify and Sparsify
$\tilde{G}$
$\tilde{H} \cup \phi(M^*) \subseteq \phi(\tilde{G})$
Restrict U in order to reduce edge-degrees
Bipartite
M^*
Unfold and Sparsify
$\phi(M^*)$
$\tilde{U} = U \cap \phi(M^*)$

Figure 4 Illustration of the reduction to the bipartite case. We show that $\widetilde{H} \cup \widetilde{U}$ contains a matching of size at least $\left(\frac{2}{3} - \epsilon\right) \mu_w(G)$. Since $\widetilde{G}$ is bipartite, we can refold $\widetilde{H} \cup \widetilde{U}$ without reducing the matching size.

Now $\widetilde{U}$ is a matching, so for all $v \in V$, we have $\deg_{\widetilde{H} \cup \widetilde{U}}(v) \in \{\deg_{\widetilde{H}}(v), \deg_{\widetilde{H}}(v) + 1\}$. So

$$\deg_{\widetilde{H} \cup \widetilde{U}}(v) \approx \deg_{\widetilde{H}}(v) \approx \frac{1}{2}\deg_H(v).$$

In particular,

$$\forall (u, v) \in \widetilde{H} \cup \widetilde{U}, \qquad \deg_{\widetilde{H} \cup \widetilde{U}}(u) + \deg_{\widetilde{H} \cup \widetilde{U}}(v) \approx \frac{1}{2}\deg_H(u) + \frac{1}{2}\deg_H(v) \leq \frac{\beta}{2},$$

and

$$\forall (u,v) \in \phi(M^*) \setminus (\widetilde{H} \cup \widetilde{U}), \qquad \deg_{\widetilde{H}\cup\widetilde{U}}(u) + \deg_{\widetilde{H}\cup\widetilde{U}}(v) \approx \frac{1}{2}\deg_H(u) + \frac{1}{2}\deg_H(v) \geq \frac{\beta}{2} - 1.$$

Setting $X = \widetilde{H} \cup \widetilde{U}$, $\beta' \approx \frac{\beta}{2}$, and λ' to be a sufficiently small constant, we can now apply Theorem 2.7 to the graph $\widetilde{H} \cup \phi(M^*)$, and obtain

$$\mu(\widetilde{H} \cup \widetilde{U}) \geq (2/3 - \epsilon)\, \mu(\widetilde{H} \cup \phi(M^*)) \geq (2/3 - \epsilon)\, \mu(\phi(M^*)).$$

Since $\widetilde{H} \cup \widetilde{U} \subseteq \phi(\widetilde{G})$ and since $\widetilde{G}$ is bipartite, we can apply Lemma 2.3 to get the required result

$$\mu_w(\mathcal{R}(H \cup U)) \geq \mu_w(\mathcal{R}(\widetilde{H} \cup \widetilde{U})) = \mu(\widetilde{H} \cup \widetilde{U}) \geq (2/3 - \epsilon)\, \mu(\phi(M^*)) = (2/3 - \epsilon)\, \mu_w(G).$$

In the rest of the paper, we will present the full analysis. In Section 4, we formally present the algorithm and analysis for random-order streams. In Section 5, we prove Theorem 1.2 and Theorem 1.3.

3 Notation and Preliminaries

Given a graph $G = (V, E)$, we will use $n := |V|$ to denote the number of vertices and $m := |E|$ to denote the number of edges in G. If G is weighted, then we will use $w : E \to \mathbb{R}^+$ to denote the edge weights, and $R := \max_{e\in E} w_e / \min_{e\in E} w_e$ to denote the ratio between the heaviest and the lightest edge in G. We use $\mu(G)$ to denote the size of the maximum cardinality matching in G, and $\mu_w(G)$ to denote the weight of the maximum weight matching in G.

Given $\epsilon > 0$, define $\gamma_\epsilon := (4/\epsilon)^{\lceil 1/\epsilon \rceil}$, a large constant which will be incurred in the space usage of our algorithms (instead of a dependence on the maximum weight of the graph). Note that for any fixed ϵ, we have $\gamma_\epsilon = O(1)$.

3.1 Models

Random-order streams In the random-order stream model, the weighted edges of the input graph arrive one-by-one in an order chosen uniformly at random from all possible orderings. The algorithm makes a single pass over the stream and must output an approximate maximum weight matching at the end of the stream.

Robust communication model In the k-party one-way robust communication model, each weighted edge of the input graph is assigned independently and uniformly at random to one of the k parties. The ith party is supplied with its assigned edges and a message m_{i-1} from the $(i-1)$st party, and must send a message m_i to the $(i+1)$st party. The kth party must output a valid weighted matching of the input graph. The communication complexity of a protocol is defined to be $\max_{1\leq i\leq k} |m_i|$, where $|m_i|$ is the number of words in message m_i.

In the case of $k = 2$, we refer to the first party as Alice and to the second party as Bob.

3.2 Graph Unfolding

In addition to the facts already stated in Section 2, we will need the following:

▶ **Theorem 3.1** (Unfolding preserves matching size in bipartite graphs [57]). *If G is a weighted bipartite graph, then $\mu_w(G) = \mu(\phi(G))$.*

▶ **Definition 3.2** (Refolding approximate [23]). *Let G be a weighted graph. A subgraph $H \subseteq \phi(G)$ is α-refolding-approximate if $\mu_w(\mathcal{R}(H)) \geq \alpha \cdot \mu_w(G)$.*

3.3 EDCS

We will use the following guarantee which holds for a relaxed notion of EDCS.

▶ **Definition 3.3** (Bounded edge-degree [22]). *We say that a graph H has bounded edge-degree β if for every edge $(u,v) \in H$, it holds that $\deg_H(u) + \deg_H(v) \leq \beta$.*

▶ **Definition 3.4** (Underfull edge [22]). *Let G be any unweighted graph, and let H be a subgraph of G with bounded edge-degree β. For any parameter $\lambda < 1$, we say that an edge $(u,v) \in G \setminus H$ is (G,H,β,λ)-underfull if $\deg_H(u) + \deg_H(v) < \beta(1-\lambda)$.*

▶ **Lemma 3.5** (Relaxed EDCS contain a 2/3-approximate matching [22]). *Let $\epsilon < \frac{1}{2}$ be a parameter, and let λ, β be parameters with $\lambda \leq \frac{\epsilon}{128}$, $\beta \geq 16\lambda^{-2}\log(1/\lambda)$. Consider any unweighted graph G and any subgraph H with bounded edge-degree β. Let U contain all edges in $G \setminus H$ that are (G,H,β,λ)-underfull. Then $\mu(H \cup U) \geq (2/3 - \epsilon)\mu(G)$.*

3.4 Concentration Inequality

We will use the Chernoff bound for negatively associated random variables (see e.g. the primer in [76]).

▶ **Theorem 3.6.** *Let $X_1, \dots X_n$ be negatively associated random variables taking values in $[0,1]$. Let $X := \sum X_i$ and let $\mu := \mathbb{E}[X]$. Then, for any $0 < \delta < 1$, we have*

$$\Pr[X \leq \mu(1-\delta)] \leq \exp\left(\frac{-\mu\delta^2}{2}\right).$$

4 2/3-Approximation in Random-Order Streams

In this section we prove Theorem 1.1. In Section 4.1, we formally describe the reduction from weighted random-order streams to unweighted b-batch random-order streams, and we prove its correctness. In Section 4.2, we show that Bernstein's 2/3-approximation algorithm [22] for random-order streams still works under batch-arrivals. Finally, in Section 4.3, we show that the obtained weighted random-order streaming algorithm still works for non-bipartite graphs, and we complete the proof of Theorem 1.1.

4.1 Reduction to Unweighted b-batch Random-Order Streams

Gupta and Peng [52] gave a reduction which allows us to assume that the edge weights are integral and bounded above by a large constant.They originally proved the reduction for the dynamic graph model, however it also applies to the streaming and one-way communication models (See Theorem 6.1 and Theorem 6.2 in [23]).

▶ **Theorem 4.1** (Reduction to bounded integral weights [52, 23]). *If there is a random-order streaming algorithm $\mathcal{A}$ to compute an α-approximate maximum weight matching in graphs with edge weights in $[W]$ and using space $S(n,m,W,\alpha)$, then there exists a random-order streaming algorithm $\mathcal{A}'$ to compute a $(1-\epsilon)\alpha$-approximate maximum weight matching with weights in graph with weights $\mathbb{R}^+$ using space $O(S(n,m,\gamma_\epsilon,\alpha)\log R)$.*

Similarly, if there is a one-way robust communication complexity protocol to compute an α-approximate maximum weight matching for graphs with edge weights in $[W]$ using $C(n,m,W,\alpha)$ words of communication, then there exists a protocol to compute a $(1-\epsilon)\alpha$-approximate maximum weight matching in graphs with weights in $\mathbb{R}^+$ using $O(C(n,m,\gamma_\epsilon,\alpha)\log R)$ words of communication.

We would like to use the unfolding technique to reduce to the unweighted problem. As Bernstein, Dudeja and Langley [23] showed, in *adversarially ordered* streams, unfolding immediately gives a reduction for bipartite graphs: Whenever a weighted edge $e \in G$ arrives, we can unfold it and pass the unweighted edges $\phi(e)$ sequentially into an unweighted streaming algorithm while tracking the updates in the weighted stream. In *random-order* streams, there is a subtle issue with this approach. If the edges arrive uniformly at random in G, then the arrival order in $\phi(G)$ will not be uniformly at random, but rather there will be batches of edges which necessarily arrive together. To overcome this issue, we consider the b-batch random-order stream model, restated below.

▶ **Definition 2.4** (b-batch random-order stream model)**.** *In the b-batch random-order stream model the edge set of the input graph $G = (V, E)$ is presented as follows: An adversary partitions the edge set E into batches $\mathcal{B} = \{B_1, ..., B_q\}$ with $|B_i| \le b$ for all i. The arrival order of the batches $(B_{i_1}, ..., B_{i_q})$ is then chosen uniformly at random among all the permutations of $\mathcal{B}$. The edges in each batch arrive simultaneously.*

Graph unfolding naturally gives a reduction from weighted random-order streams to unweighted b-batch random-order streams.

▶ **Theorem 4.2** (Reduction to the b-batch model)**.** *If there exists an algorithm $\mathcal{A}_B$ for the unweighted b-batch random-order stream model that computes an α-approximate maximum cardinality matching in bipartite graphs using space $S(n, m, b, \alpha)$, then there exists an algorithm $\mathcal{A}_w$ for weighted random-order streams (with weights in $\mathbb{R}^+$) that computes a $(1-\epsilon)\alpha$-approximate maximum weight matching in bipartite graphs using space $O(S(n\gamma_\epsilon, m\gamma_\epsilon, \gamma_\epsilon, \alpha)\log R)$.*

Moreover, suppose that $\mathcal{A}_B$ computes an α-refolding approximate subgraph whenever the input graph is of the form $\phi(G)$ for some weighted graph G with batches $\mathcal{B} = \{\phi(e) : e \in G\}$. Then the guarantees of $\mathcal{A}_w$ also hold for non-bipartite graphs.

Proof. Let $\mathcal{A}_B$ be the unweighted b-batch random-order streaming algorithm using space $S(n, m, b, \alpha)$. By Theorem 4.1, it suffices to construct an algorithm $\mathcal{A}_W$ that computes an α-approximate maximum weight matching using space $S(Wn, Wm, W, \alpha)$ when the edge weights are in $[W]$. We can obtain the required algorithm $\mathcal{A}_W$ as follows:

Whenever an edge e arrives in the weighted stream, define a batch $B_e := \phi(e)$ consisting of the unfolded edges of e. Feed the batch B_e as an update to the batch algorithm $\mathcal{A}_B$. In other words, $\mathcal{A}_B$ is applied to the graph $\phi(G)$ with batches $\mathcal{B} = \{\phi(e) : e \in G\}$. At the end of the stream, $\mathcal{A}_B$ outputs an α-approximate maximum cardinality matching M of $\phi(G)$. The algorithm $\mathcal{A}_W$ outputs the maximum weight matching in $\mathcal{R}(M)$ (which can easily be computed from M). We have

$$\begin{aligned}
\mu_w(\mathcal{R}(M)) &\ge \mu(M), && \text{by Lemma 2.3}\\
&\ge \alpha \cdot \mu(\phi(G)), && \text{by the assumption on } M\\
&= \alpha \cdot \mu_w(G), && \text{by Theorem 3.1}
\end{aligned}$$

so $\mathcal{A}_W$ outputs an α-approximate maximum weight matching. Since the graph $\phi(G)$ has at most Wn vertices and Wm edges, the space usage of $\mathcal{A}_W$ is at most $S(Wn, Wm, W, \alpha)$, as required.

For the "Moreover"-part, suppose that $\mathcal{A}_B$ computes an α-refolding approximate subgraph H. Define $\mathcal{A}_W$ as before, except that now $\mathcal{A}_W$ should output the maximum weight matching in $\mathcal{R}(H)$. Then

$$\mu_w(\mathcal{R}(H)) \geq \alpha \cdot \mu_w(G), \qquad \text{by Definition 3.2,}$$

so the approximation ratio achieved by $\mathcal{A}_W$ is still α, even for non-bipartite graphs, as required. ◀

▶ Remark 4.3. The argument can easily be adapted to the robust communication model. Consider a b-batch robust communication model, in which an adversary partitions the edges into batches of size at most b, and each batch gets assigned uniformly at random to each of the parties. Then any protocol for the b-batch robust communication model gives a protocol for the weighted robust communication model.

4.2 2/3-Approximation in b-batch Random-Order Streams

In this section, we prove the following proposition.

▶ **Proposition 4.4.** *Given any unweighted graph G and any approximation parameter $0 < \epsilon < 1$, Bernstein's algorithm (Algorithm 1) with high probability computes a $(2/3 - \epsilon)$-approximate maximum cardinality matching in the b-batch random-order stream model. The space complexity of the algorithm is $O(nb^2 \log n \log b \operatorname{poly}(\epsilon^{-1}))$, where b is the upper bound on batch-size.*

▶ **Definition 4.5** (Parameters)**.** *Let $\epsilon < \frac{1}{2}$ be the final approximation parameter we are aiming for, $\lambda = \frac{\epsilon}{512}, \beta = 144\lambda^{-2}\log(2b/\lambda)$; note that $\beta = O(\operatorname{poly}(\epsilon^{-1})\log b)$. Set $\alpha = \frac{\epsilon q}{b(n\beta^2+1)}$ and $\gamma = 7\log n\frac{q}{\alpha} = O(nb\log n\log b\operatorname{poly}(\epsilon^{-1}))$.*

We now describe Bernstein's algorithm [22] adapted to the b-batch model. The algorithm has two Phases. In Phase 1, it computes a subgraph H that is bounded edge-degree β (Definition 3.3). In Phase 2, it stores all the (G, H, β, λ)-underfull edges (Definition 3.4). That way, the algorithm computes a "relaxed" EDCS, which by Lemma 3.5 contains a $(2/3 - \epsilon)$-approximate maximum cardinality matching.

More concretely, the algorithm proceeds as follows: Initialize an empty subgraph H and start Phase 1. Phase 1 is split into epochs, each of which contains exactly α batches. In each epoch, the algorithm processes the batches sequentially. For each edge (u, v) in the batch, if $\deg_H(u) + \deg_H(v) < \beta(1 - \lambda)$, then (u, v) is added to the subgraph H (note that the algorithm changes H over time, so $\deg_H$ always refers to the degree of H at the time when the edge is examined). After adding an edge to H, the algorithm runs procedure RemoveOverfullEdges(H) to ensure that H is always bounded edge-degree β. In each epoch, the algorithm also has a boolean FoundUnderfull, which tracks whether at least one underfull edge arrived in the epoch. If FoundUnderfull is FALSE at the end of an epoch, then the algorithm terminates Phase 1 and moves on to Phase 2. Once Phase 1 terminates, the subgraph H becomes fixed and does not change anymore. Then, in Phase 2, the algorithm simply picks up all the underfull edges into a separate set U. For a formal description, see Algorithm 1. Note that the only difference between Algorithm 1 and Bernstein's original algorithm (Algorithm 1 in [22]) is that the length of each epoch is now determined by the number of batches, rather than the number of edges.

▶ **Definition 4.6.** *Let $\mathcal{B}_{early}$ denote the first $\frac{\epsilon}{b}q$ batches in the stream and let $\mathcal{B}_{late}$ denote the remaining batches. Let $E_{late} := \{e \in E : e \in B \text{ for some } B \in \mathcal{B}_{late}\}$ be the set of edges that arrive as part of the late batches. More generally, let $E_{>i}$ denote the the set of edges that arrive after the first i batches.*

Algorithm 1 Bernstein's Algorithm [22] adapted to the b-batch model.

```
$H \leftarrow \emptyset, U \leftarrow \emptyset$
Procedure Phase 1
    Do until termination
        FoundUnderfull ← FALSE
        for $i = 1, \ldots, \alpha$ do                    // Each epoch has $\alpha$ batches
            Let $B_i$ denote the $i^{th}$ batch in the epoch
            for $(u, v) \in B_i$ do
                if $\deg_H(u) + \deg_H(v) < \beta(1 - \lambda)$ then
                    $H \leftarrow H \cup \{(u, v)\}$
                    FoundUnderfull ← TRUE
                    RemoveUnderfullEdges(H)
        if FoundUnderfull = FALSE then
            Go to Phase 2
Procedure RemoveOverfullEdges(H)
    while there exists $(u, v) \in H$ such that $\deg_H(u) + \deg_H(v) > \beta$ do
        Remove $(u, v)$ from $H$
Procedure Phase 2
    foreach remaining edge $(u, v)$ in the stream do
        if $\deg_H(u) + \deg_H(v) < \beta(1 - \lambda)$ then
            $U \leftarrow U \cup \{(u, v)\}$
    return the maximum matching in $H \cup U$
```

First, we show that we don't loose too many matching edges in the early part of the stream. To this end, we need to assume that the maximum cardinality matching is sufficiently large.

▷ Claim 4.7. We can assume that $\mu(G) \geq 20b^2 \log n\epsilon^{-2}$.

Proof. It is well known that every graph G satisfies $m \leq 2n\mu(G)$. Hence, if $\mu(G) < 20b^2 \log n\epsilon^{-2}$, then $m = O(nb^2 \log n\epsilon^{-2})$, so we can simply store all the edges. ◁

▶ **Lemma 4.8.** *For $\epsilon < b/2$, it holds that* $\Pr[\mu(E_{late}) \geq (1 - 2\epsilon)\mu(G)] \geq 1 - n^{-5}$.

Proof. Fix a maximum cardinality matching in M^* in G, and let $B_1, \ldots, B_k$ be the set of batches containing at least one matching edge from M^*. Each batch B_i contains at most b edges from M^*, so it suffices to show that at most $\frac{2\epsilon\mu(G)}{b}$ of these batches arrive in the early part of the stream. For $1 \leq i \leq k$, let X_i be the indicator that $B_i \in \mathcal{B}_{late}$, and let $X = \sum_{i=1}^{k} X_i$. We will show that with high probability, $X \geq k - \frac{2\epsilon\mu(G)}{b}$. For $1 \leq i \leq k$, we have $\mathbb{E}[X_i] = (1 - \frac{\epsilon}{b})$, and so $\mathbb{E}[X] = k(1 - \frac{\epsilon}{b})$.

The X_is are negatively associated, since these variables correspond to sampling $(1 - \epsilon)q$ batches uniformly at random without replacement, which is known to be negatively associated (see the primer [76]). Applying Theorem 3.6 gives

$$\begin{aligned}\Pr\left[X \geq k - \frac{2\epsilon\mu(G)}{b}\right] &= 1 - \Pr\left[X - \mathbb{E}[X] < -\left(\frac{2\epsilon\mu(G)}{b} - \frac{\epsilon k}{b}\right)\right] \\ &\geq 1 - \exp\left(\frac{-\epsilon^2(2\mu - k)^2}{4b^2 k}\right) \\ &\geq 1 - n^{-5}.\end{aligned}$$

The last inequality follows because $\mu(G) \geq k$ and $\mu(G) \geq 20b^2\epsilon^{-2} \log n$, by Claim 4.7. ◀

▶ **Lemma 4.9.** *Phase 1 satisfies the following properties:*

1. *Phase 1 terminates within the first $\frac{\epsilon q}{b}$ batches of the stream.*
2. *Phase 1 constructs a subgraph $H \subseteq G$ with bounded edge-degree β. As a corollary, H has at most $O(n\beta)$ edges.*
3. *When Phase 1 terminates after processing some batch B_l, with probability at least $1 - n^{-5}$, the total number of $(E_{>l}, H, \beta, \lambda)$-underfull edges in $E_{>l} \setminus H$ is at most $b\gamma$.*

The proof of Lemma 4.9 proceeds similarly to the proof of Lemma 4.1 in [22]. We will use the following result from the original analysis.

▶ **Lemma 4.10** ([22]). *Fix any parameter $\beta > 2$. Let $H = (V, E_H)$ be a graph, with E_H initially empty. Say that an adversary adds and removes edges from H using an arbitrary sequence of two possible moves.*

- *(Deletion move) Remove an edge (u, v) from H for which $\deg_H(u) + \deg_H(v) > \beta$.*
- *(Insertion move) Add an edge (u, v) to H for some pair $u, v \in V$ for which $\deg_H(u) + \deg_H(v) < \beta - 1$.*

Then, after $n\beta^2$ moves, no legal move remains.

Proof of Lemma 4.9. *Property 1:* By Lemma 4.10, the algorithm can make at most $n\beta^2$ changes to H. Since each epoch that does not terminate Phase 1 must make at least one change to H, there can be at most $n\beta^2 + 1$ epochs in Phase 1. So overall, Phase 1 goes through at most $\alpha(n\beta^2 + 1) = \frac{\epsilon q}{b}$ batches in Phase 1.
Property 2: Holds by construction of the algorithm, since the RemoveOverfull procedure ensures that H is always bounded edge-degree β.
Property 3: Let l be the last batch processed in Phase 1. We will say that a batch B_j with $j > l$ is *underfull* if it contains at least one $(E_{>l}, H, \beta, \lambda)$-underfull edge. We will show that with probability at least $1 - n^{-5}$, the number of underfull batches is at most γ. Since each underfull batch contains at most b underfull edges, this will give the result. The intuition is as follows: Phase 1 terminates only if there is an epoch without a single underfull batch. Since the stream is random, this means that there are relatively few underfull batches left in the stream. More formally, for each epoch $0 \le i \le \frac{\epsilon q}{b}$, say that a batch is *underfull* if it contains at least one (G, H_i, β, λ)-underfull edge, where H_i is the subgraph H constructed by the algorithm at the beginning of epoch i. Let $\mathcal{E}_i$ be the event that no underfull batches appear in epoch i, and let $\mathcal{F}_i$ be the event that there are more than γ underfull batches left in the stream when epoch i begins. Property 3 fails only if $\mathcal{E}_i \wedge \mathcal{F}_i$ happens for some i, so we need to upper bound $\Pr[\cup_{i=1}^{\epsilon q/b} \mathcal{E}_i \wedge \mathcal{F}_i]$. Let $\mathcal{B}_i^r$ denote the set of batches that have not yet appeared at the beginning of epoch i (r for remaining), let $\mathcal{B}_i^e$ denote the set of batches that appear in epoch i (e for epoch) and let $\mathcal{B}_i^u$ denote the set of underfull batches that remain in the stream (u for underfull). With these definitions, we can write $\mathcal{E}_i \wedge \mathcal{F}_i$ as the event $(\mathcal{B}_i^u \cap \mathcal{B}_i^e = \emptyset) \wedge (|\mathcal{B}_i^u| > \gamma)$, since the event $\mathcal{B}_i^u \cap \mathcal{B}_i^e = \emptyset$ ensures that the graph H does not change throughout the epoch. We have

$$\begin{aligned}
\Pr[\mathcal{E}_i \wedge \mathcal{F}_i] &= \Pr[(\mathcal{B}_i^u \cap \mathcal{B}_i^e = \emptyset) \wedge (|\mathcal{B}_i^u| > \gamma)] \\
&\le \Pr\left[\mathcal{B}_i^u \cap \mathcal{B}_i^e = \emptyset \big| |\mathcal{B}_i^u| > \gamma\right] \\
&< \left(1 - \frac{\gamma}{q}\right)^{\alpha} \\
&= \left(1 - \frac{7 \log n}{\alpha}\right)^{\alpha} \\
&\le n^{-7}.
\end{aligned}$$

Here the second inequality follows because $\mathcal{B}_i^e$ is obtained by sampling α batches from $\mathcal{B}_i^r$ uniformly at random without replacement, and since $|\mathcal{B}_i^u| > \gamma$ and $|\mathcal{B}_i^r| \leq q$. There are at most n^2 epochs in total, so taking the union bound over all epochs gives the result. ◀

Finally, we complete the proof of Proposition 4.4.

Proof of Proposition 4.4. Let us first show the approximation guarantee. By Part 2 of Proposition 4.9, Phase 1 computes a subgraph H which has bounded edge-degree β. Moreover, by Part 1 of Proposition 4.9, it holds that $H \subseteq E_{late}$. Phase 2 finds the set U of all $(E_{late}, H, \beta, \lambda)$-underfull edges. So by Lemma 3.5 applied to the graph E_{late}, the algorithm returns a matching of size at least

$$\begin{aligned} \mu(H \cup U) &\geq (2/3 - \epsilon)\mu(E_{late}) && \text{by Lemma 3.5} \\ &\geq (2/3 - \epsilon)(1 - 2\epsilon)\mu(G), && \text{by Lemma 4.8,} \end{aligned}$$

where the last inequality holds with probability at least $1 - n^{-5}$. Re-scaling ϵ gives the approximation ratio.

For the space analysis: By Part 2 of Lemma 4.9, the space usage of Phase 1 is $O(n\beta) = O(n \log b \operatorname{poly}(\epsilon^{-1}))$. By Part 3 of Lemma 4.9, with probability at least $1 - n^{-5}$, the space usage of Phase 2 is at most $O(b\gamma) = O(nb^2 \log n \log b \operatorname{poly}(\epsilon^{-1}))$. So with probability at least $1 - n^{-5}$, the total space usage is at most $O(nb^2 \log n \log b \operatorname{poly}(\epsilon^{-1}))$. By a union bound, the overall success probability of the algorithm is at least $1 - 2n^{-5}$. ◀

4.3 Extension to Non-Bipartite Graphs

In this section, we show that the computed graph $H \cup U$ is $(2/3 - \epsilon)$-refolding approximate. This, together with Proposition 4.4 and Theorem 4.2 will complete the proof of Theorem 1.1.

In [23], it was shown that EDCS are (almost) 2/3-refolding approximate. However, since $H \cup U$ is not actually an EDCS, but rather a relaxed version of an EDCS, this result cannot be applied directly. Instead, we need a more careful argument. We need the following lemma which was proved in [23].

▶ **Lemma 4.11** (Lemma 5.7 in [23]). *Let G be a weighted graph with weights in $[W]$. Let $\delta \in (0, 1/2)$, and let $d \geq 36\delta^{-2} log(W/\delta)$. For any matching M in G and any subgraph H of $\phi(G)$ with maximum degree at most d, there exists a bipartite subgraph $\widetilde{G} = \widetilde{G}(M, H)$ of G such that, setting $\widetilde{H} := H \cap \phi(\widetilde{G})$, it holds that*

1. *$M \subseteq \widetilde{G}$ and*
2. *$|\deg_{\widetilde{H}}(v) - \deg_H(v)/2| \leq \delta d$ for all vertices $v \in V(H)$.*

▶ Remark 4.12. Bernstein, Dudeja and Langley [23] state the lemma for the special case when M is a maximum weight matching in G, however, without changing their argument, the same is true for any arbitrary matching M.

We now prove the main technical lemma, which shows that $H \cup U$ is $(2/3 - \epsilon)$-refolding approximate.

▶ **Lemma 4.13.** *Let G be a (possibly non-bipartite) weighted graph with weights in $[W]$. Let $\epsilon > 0$, $\lambda \leq \frac{\epsilon}{512}$, $\beta \geq \frac{144}{\lambda^2}\log(2W/\lambda)$. Let $G_S \subseteq G$ be any subgraph of G. Consider the unfolded graph $\phi(G)$. Let H be a subgraph of $\phi(G)$ with bounded edge-degree β, and let U be the set of all edges in $\phi(G_S) \setminus H$ that are $(\phi(G_S), H, \beta, \lambda)$-underfull. Then $\mu_w(\mathcal{R}(H \cup U)) \geq (2/3 - \epsilon)\mu_w(G_S)$.*

Proof. Let $\delta = \frac{\lambda}{2}$. Fix a maximum-weight matching M^* of G_S, and let $\widetilde{G} = \widetilde{G}(M^*, H)$ be the bipartite subgraph obtained from Lemma 4.11. Consider the subgraph $\phi(\widetilde{G}) \subseteq \phi(G)$. Let $\widetilde{H} := H \cap \phi(\widetilde{G})$ be the restriction of H to $\phi(\widetilde{G})$, and let $\widetilde{U} := U \cap \phi(M^*)$ be the restriction of U to the matching $\phi(M^*)$. Note that $\widetilde{U}$ is a matching. By Lemma 4.11, we have $\phi(M^*) \subseteq \phi(\widetilde{G})$, and therefore $\widetilde{H} \cup \widetilde{U} \subseteq \phi(\widetilde{G})$. Therefore, we may now apply Lemma 2.3 to the bipartite graph $\widetilde{G}$ and the subgraph $\widetilde{H} \cup \widetilde{U}$ of $\phi(\widetilde{G})$.

$$\begin{aligned} \mu_w(\mathcal{R}(H \cup U)) &\geq \mu_w(\mathcal{R}(\widetilde{H} \cup \widetilde{U})), && \text{since } H \cup U \supseteq \widetilde{H} \cup \widetilde{U} \\ &\geq \mu(\widetilde{H} \cup \widetilde{U}), && \text{by Lemma 2.3.} \end{aligned} \tag{3}$$

Furthermore, recalling that M^* is a maximum weight matching in G_S, we have

$$\mu_w(G_S) = w(M^*) = |\phi(M^*)| \leq \mu(\widetilde{H} \cup \phi(M^*)). \tag{4}$$

We will show that $\mu(\widetilde{H} \cup \widetilde{U}) \geq (\frac{2}{3} - \epsilon)\mu(\widetilde{H} \cup \phi(M^*))$. To this end, we will show that $\widetilde{H} \cup \widetilde{U}$ is an EDCS of $\widetilde{H} \cup \phi(M^*)$.

▷ Claim 4.14. $\widetilde{H} \cup \widetilde{U}$ is a (β', λ')-EDCS of $\widetilde{H} \cup \phi(M^*)$ for $\beta' = \frac{\beta}{2} + \beta\lambda + 2, \lambda' = 8\lambda$.

Proof. Let us start by showing property P1 in Definition 2.6. First note that for all $(u, v) \in \widetilde{H} \cup \widetilde{U}$, it holds that $\deg_H(u) + \deg_H(v) \leq \beta$. Indeed, if $(u, v) \in \widetilde{H}$, then $(u, v) \in H$, so $\deg_H(u) + \deg_H(v) \leq \beta$ since H is bounded edge-degree β. If instead $(u, v) \in \widetilde{U}$, then $(u, v) \in U$, so $\deg_H(u) + \deg_H(v) \leq (1 - \lambda)\beta$, since all elements of U are $(\phi(G), H, \beta, \lambda)$-underfull. Therefore, for all $(u, v) \in \widetilde{H} \cup \widetilde{U}$, it holds that

$$\begin{aligned} \deg_{\widetilde{H} \cup \widetilde{U}}(u) + \deg_{\widetilde{H} \cup \widetilde{U}}(v) &\leq \deg_{\widetilde{H}}(u) + \deg_{\widetilde{H}}(v) + \deg_{\widetilde{U}}(u) + \deg_{\widetilde{U}}(v) \\ &\leq \frac{\deg_H(u) + \deg_H(v)}{2} + 2\delta\beta + \deg_{\widetilde{U}}(u) + \deg_{\widetilde{U}}(v) \\ &\leq \frac{\beta}{2} + \beta\lambda + \deg_{\widetilde{U}}(u) + \deg_{\widetilde{U}}(v) \\ &\leq \frac{\beta}{2} + \beta\lambda + 2 \\ &= \beta'. \end{aligned}$$

The second inequality follows by Lemma 4.11, and the third inequality follows since $\delta = \frac{\lambda}{2}$.

We now show property P2 in Definition 2.6: If $(u, v) \in (\widetilde{H} \cup \phi(M^*)) \setminus (\widetilde{H} \cup \widetilde{U})$, then $(u, v) \in \phi(M^*) \setminus U$, and in particular $\deg_H(u) + \deg_H(v) > (1 - \lambda)\beta$ (by definition of U). Thus,

$$\begin{aligned} \deg_{\widetilde{H} \cup \widetilde{U}}(u) + \deg_{\widetilde{H} \cup \widetilde{U}}(v) &\geq \deg_{\widetilde{H}}(u) + \deg_{\widetilde{H}}(v) \\ &\geq \frac{\deg_H(u) + \deg_H(v)}{2} - 2\delta\beta, && \text{by Lemma 4.11} \\ &\geq \frac{\beta(1 - \lambda)}{2} - \beta\lambda, && \text{since } \delta = \frac{\lambda}{2} \\ &\geq \left(\frac{\beta}{2} + \lambda\beta + 2\right)(1 - 8\lambda) \\ &= \beta'(1 - \lambda'). \end{aligned}$$

The last inequality follows from simple algebraic manipulations, using the fact that $\lambda\beta \geq 2$. ◁

By the choice of parameters, we have $\lambda' \leq \frac{\epsilon}{64}$ and $\beta' \geq 8\lambda'^{-2}\log(1/\lambda')$, so Claim 4.14 together with Theorem 2.7 yields $\mu(\widetilde{H} \cup \widetilde{U}) \geq (2/3-\epsilon)\mu(\widetilde{H} \cup \phi(M^*))$. Combining everything, we get

$$\begin{aligned}
\mu_w(\mathcal{R}(H \cup U)) &\geq \mu(\widetilde{H} \cup \widetilde{U}), && \text{by Equation 3}\\
&\geq (2/3-\epsilon)\mu(\widetilde{H} \cup \phi(M^*))\\
&\geq (2/3-\epsilon)\mu_w(G_S), && \text{by Equation 4.}
\end{aligned}$$ ◀

Finally, we complete the proof of Theorem 1.1.

Proof of Theorem 1.1. We apply the reduction in Theorem 4.2 to Algorithm 1. By Proposition 4.4, Algorithm 1 computes a $(2/3-\epsilon)$-approximate maximum cardinality matching using space $O(n \log n \operatorname{poly}(b/\epsilon))$ in the b-batch random-order stream model. It remains to show that if the input graph is of the form $\phi(G)$ for some weighted graph G with batches $\mathcal{B} = \{\phi(e) : e \in G\}$, then $H \cup U$ is $(2/3-\epsilon)$-refolding approximate. Let $G_{late} \subseteq G$ denote the weighted edges corresponding to $\mathcal{B}_{late}$. An application of the Chernoff bound for negatively associated random variables (Theorem 3.6) shows that $\Pr[\mu_w(G_{late}) \geq (1-2\epsilon)\mu_w(G)] \geq 1-n^{-5}$ (the argument is similar to Lemma 4.8). Applying Lemma 4.13 to the graph G and the subgraph $G_S := G_{late}$ yields

$$\begin{aligned}
\mu_w(\mathcal{R}(H \cup U)) &\geq (2/3-\epsilon)\mu_w(G_{late}), && \text{by Lemma 4.13}\\
&\geq (1-2\epsilon)(2/3-\epsilon)\mu_w(G)\\
&\geq (2/3-3\epsilon)\mu_w(G),
\end{aligned}$$

as required. Re-scaling ϵ and applying Theorem 4.2 yields the result. ◀

5 5/6-Approximation in the Robust Communication Model

In this section, we prove Theorem 1.2 and Theorem 1.3. By applying the results from the previous section, we can generalize the protocol of Azarmehr and Behnezhad [20] to the weighted case. By the reduction in Theorem 4.1, we can assume that the edge weights take integral values in $[W]$, for a large constant W. We will now describe the protocol for the two-party model.

Let $\epsilon > 0$ be the final parameter we are aiming for, and let

$$\lambda = \frac{\epsilon}{2048}, \beta = 144\lambda^{-4}\log(2W/\lambda).$$

Let E_A denote the set of edges assigned to Alice and E_B the set of edges assigned to Bob. Alice simulates a random-order stream. She unfolds the edges and runs Algorithm 1 on the corresponding unweighted W-batch random-order stream. That way, she obtains a set $H \subseteq \phi(E_A)$ with bounded edge degree β and a set $U_A \subseteq \phi(E_A)$ consisting of all $(\phi(E_A \setminus E_{early}), H, \beta, \lambda)$-underfull edges, where $E_{early} \subseteq E_A$ denotes the first $\frac{\epsilon}{W}m$ weighted edges in her simulated stream. She communicates $\mathcal{R}(H \cup U_A)$ to Bob. Bob outputs the maximum weight matching in $\mathcal{R}(H \cup U_A) \cup E_B$. See Algorithm 2 for a formal description.

The protocol for k parties is similar, only that now all of the first $k-1$ parties should simulate a random-order stream (we describe the protocol more formally in the proof of Theorem 1.3).

Assume that each edge is assigned to Bob with probability $p \leq \frac{1}{2}$ (this will make the analysis applicable to the k-party setting). Let U_B be the set of all $(\phi(E_B), H, \beta, \lambda)$-underfull edges, i.e. the set of underfull edges assigned to Bob. Let $U := U_A \cup U_B$ denote the set of all

Algorithm 2 Robust Communication Protocol for Weighted Graphs.

```
1. Alice simulates a random-order stream
   by ordering the edges in $E_A$ uniformly at random.
2. Alice obtains $H \cup U_A \subseteq \phi(E_A)$ by running Algorithm 1
   on input $\phi(E_A)$ with batches $\mathcal{B} = \{\phi(e) : e \in E_A\}$.
   She communicates $\mathcal{R}(H \cup U_A)$ to Bob.
3. Bob outputs the maximum weight matching in $\mathcal{R}(H \cup U_A) \cup E_B$.
```

underfull edges. We will define an auxiliary fractional matching x on $\mathcal{R}(H \cup U)$ of weight at least $(2/3 - \epsilon)\mu_w(G)$. We will then extend it to a fractional matching y on $E_B \cup \mathcal{R}(H \cup U_A)$, and show that due to the additional edges in E_B, the fractional matching y has weight at least $(5/6 - \epsilon)\mu_w(G)$.

Let $E_{late} := E \setminus E_{early}$. Fix a maximum weight matching M^* in E_{late}. Define a fractional matching x on $\mathcal{R}(H \cup U)$ as follows:

- Start with $H_1 = H$ and $U_1 = U$.
- For $i = 1, \dots, \lambda\beta$:
  - Let M_i be a maximum weight matching in $\mathcal{R}(H_i \cup U_i)$.
  - Let $H_{i+1} = H_i \setminus \phi(M_i \setminus M^*)$ and $U_{i+1} = U_i \setminus \phi(M_i \setminus M^*)$.
- For every edge e, let $x_e = \frac{|\{i : e \in M_i\}|}{\lambda\beta}$.

In other words, we start with $H \cup U$, and then in each iteration, we find a maximum weight matching M_i in the refolding, and remove the edges in $\phi(M_i \setminus M^*)$ from $H \cup U$.

▶ Remark 5.1. Note that this is a valid fractional matching, since

$$x_u = \sum_{e \ni u} x_e = \sum_{e \ni u} \frac{|\{i : e \in M_i\}|}{\lambda\beta} = \sum_i \frac{|\{e \ni u : e \in M_i\}|}{\lambda\beta} \leq 1.$$

Furthermore, note that $x_e \leq \frac{1}{\lambda\beta}$ whenever $e \notin M^*$. This is because, if $e \in M_i \setminus M^*$ for some i, then $e \notin \mathcal{R}(H_j \cup U_j)$ for all $j > i$.

▶ **Lemma 5.2.** *It holds that $\sum_e w_e x_e \geq (\frac{2}{3} - \epsilon)\mu_w(E_{late})$.*

Proof. For each i, let $G_i := E_{late} \setminus (\cup_{j<i} M_j \setminus M^*)$. We will apply Lemma 4.13 to the graph $G \setminus (\cup_{j<i} M_j \setminus M^*)$ and subgraph $G_S = G_i$. Recall that we obtain H_{i+1} from H_i by removing the edges in $\phi(M_i \setminus M^*)$. Since $\phi(M_i \setminus M^*)$ is a matching, the degree of each edge in $\phi(G)$ will decrease by at most two in each iteration. Therefore, U_i contains all the edges in $G_i \setminus H_i$ that have H_i degree less than $(1 - \lambda)\beta - 2(i - 1) \geq (1 - 3\lambda)\beta$. By Lemma 4.13, we get

$$w(M_i) = \mu_w\left(\mathcal{R}(H_i \cup U_i)\right) \geq \left(\frac{2}{3} - \epsilon\right)\mu_w(G_i). \tag{5}$$

Also, G_i is constructed so that it always contains M^*, so

$$\mu_w(G_i) \geq w(M^*) = \mu_w(E_{late}). \tag{6}$$

Combining, we obtain

$$\begin{aligned}\sum_{e\in\mathcal{R}(H\cup U)} w_e x_e &= \sum_{e\in\mathcal{R}(H\cup U)} w_e \frac{|\{i : e \in M_i\}|}{\lambda\beta}\\ &= \frac{1}{\lambda\beta}\sum_i \sum_{e\in\mathcal{R}(H\cup U)} w_e \mathbb{1}\{e\in M_i\}\\ &= \frac{1}{\lambda\beta}\sum_i w(M_i)\\ &\geq \frac{1}{\lambda\beta}\sum_i \left(\frac{2}{3}-\epsilon\right)\mu_w(G_i). && \text{by Equation 5}\\ &\geq \left(\frac{2}{3}-\epsilon\right)\mu_w(E_{late}), && \text{by Equation 6.}\end{aligned}$$ ◀

Recall that the set of edges that Bob has access to is $E_B \cup \mathcal{R}(H\cup U)$. We need to show that $\mu_w(E_B \cup \mathcal{R}(H\cup U)) \geq (\frac{5}{6}-\epsilon)\mu_w(G)$. We will do this by extending the fractional matching x on $\mathcal{R}(H\cup U)$ to a fractional matching y on $E_B \cup \mathcal{R}(H\cup U)$. In order to describe y, we will condition on the set of early edges E_{early}, thereby fixing $\mathcal{R}(H\cup U)$ and x. For each edge $e \in E_{late}$, we have

$$\Pr[e\in E_B | e\in E_{late}] = \frac{\Pr[e\in E_B \wedge e \in E_{late}]}{\Pr[e\in E_{late}]} = \frac{p}{1-\epsilon/W}.$$

and

$$\Pr[e\in E_A | e\in E_{late}] = 1-\frac{p}{1-\epsilon/W}.$$

Recall that M^* is a fixed maximum weight matching in E_{late}. Let $M_{in} := M^* \cap \mathcal{R}(H\cup U)$ and let $M_{out} := M^* \setminus \mathcal{R}(H\cup U)$. After drawing E_B, define a random matching $M' \subseteq M^*$ as follows:

- Include each edge $e \in M_{in}$ independently with probability p.
- Include each edge $e \in M_{out} \cap E_B$ independently with probability $1-\epsilon/W$.

Conditioned on E_{early}, each edge in M_{out} ends up in M' independently with probability $(1-\epsilon/W)\cdot\frac{p}{1-\epsilon/W} = p$. Each edge in M_{in} also ends up in M' independently with probability p, so overall each edge in M^* ends up in M' independently with probability p.

For any edge $e \notin M^*$, let p_e denote the probability that e is *not* adjacent to any edge in M'. In other words,

$$p_e = \begin{cases}(1-p) & \text{if } e \text{ has exactly one endpoint matched by } M^*,\\ (1-p)^2 & \text{if both of the endpoints of } e \text{ are matched by } M^*.\end{cases}$$

We can now define $\hat{y}$ on $E_B \cup \mathcal{R}(H\cup U)$.

$$\hat{y}_e = \begin{cases}1 & \text{if } e\in M',\\ x_e & \text{if } e \in M^*\setminus M',\\ 0 & \text{if } e\notin M^* \text{ and } e \text{ is adjacent to at least one edge of } M'\\ (1-p)\frac{x_e}{p_e} & \text{if } e \notin M^* \text{ and } e \text{ is not adjacent to } M'.\end{cases}$$

Finally, we scale down $\hat{y}$ and zero out some of the entries in order to obtain a valid fractional matching y.

$$y_{(u,v)} = \begin{cases}0 & \text{if } \hat{y}_u/(1+\epsilon) > 1 \text{ or } \hat{y}_v/(1+\epsilon) > 1\\ \frac{\hat{y}_{(u,v)}}{1+\epsilon} & \text{otherwise.}\end{cases}$$

▶ **Lemma 5.3.** *It holds that*

$$\mathbb{E}\left[\sum_{e\in E} w_e y_e\right] \geq \left(\frac{2}{3}+\frac{p}{3}-4\epsilon\right)\mu_w(G).$$

The proof is similar to Lemma 4.6 in [20], and is included in the full version of the paper. Next, we round y to an integral matching.

▶ **Lemma 5.4.** *There exists a matching of weight at least* $(1-3\epsilon)\sum_{e\in E} w_e y_e$ *in* $E_B \cup \mathcal{R}(H \cup U_A)$.

The proof is similar to Lemma 4.7 in [20] and is included in the full version of the paper. Finally, we show that we have a large matching with high probability, and not just in expectation.

▶ **Lemma 5.5.** *With probability at least* $1-n^{-5}$*, there exists a matching of weight at least* $\left(\frac{2}{3}+\frac{p}{3}-O(\epsilon)\right)\mu_w(G)$ *in* $E_B \cup \mathcal{R}(H \cup U_A)$.

The proof is similar to Lemma 5.2 in [20], and is included in the full version of the paper. We now complete the proofs of Theorem 1.2 and Theorem 1.3.

Proof of Theorem 1.2. Suppose that the edge weights are in $[W]$. By Proposition 4.4, Protocol 2 uses $O(n \log n \operatorname{poly}(W/\epsilon))$ words of communication with high probability. By Lemma 5.5, the protocol achieves a $(\frac{2}{3}+\frac{p}{3}-O(\epsilon))$-approximation with high probability. So by Theorem 4.1, there exists a protocol that achieves a $(\frac{2}{3}+\frac{p}{3}-O(\epsilon))(1-\epsilon)$-approximation using space $O(n \log n \log R)$ when the edge weights are in $\mathbb{R}^+$. Letting $p=\frac{1}{2}$ and re-scaling ϵ proves the theorem. ◀

Proof of Theorem 1.3. Suppose that the edge weights are in $[W]$. We need to adjust the protocol to the k-party model. The first party simulates the start of a random-order stream by selecting an ordering of their edges uniformly at random. They unfold the edges and run Algorithm 1 on the corresponding unweighted W-batch random-order stream. They pass the memory state of the algorithm to the next party. Each of the next $k-2$ parties will continue to simulate the random-order stream that way. The $(k-1)$st party communicates $\mathcal{R}(H \cup U)$ to the last party, where $H \cup U$ is the unweighted graph computed by Algorithm 1 on the unfolded W-batch stream. Finally, the last party will output the maximum weight matching in the graph consisting of all edges to which they have access. That way, we can set $p=\frac{1}{k}$ and treat the first $k-1$ parties as Alice and the last party as Bob. By Proposition 4.4, the protocol uses $O(n \log n \operatorname{poly}(W/\epsilon))$ words of communication with high probability. By Lemma 5.5, the protocol achieves a $(\frac{2}{3}+\frac{p}{3}-O(\epsilon))$-approximation with high probability. So by Theorem 4.1, there exists a protocol that achieves a $(\frac{2}{3}+\frac{p}{3}-O(\epsilon))(1-\epsilon)$-approximation using space $O(n \log n \log R)$ when the weights are in $\mathbb{R}^+$. Re-scaling ϵ proves the theorem. ◀

References

1 Kook Jin Ahn and Sudipto Guha. Linear programming in the semi-streaming model with application to the maximum matching problem. *Inf. Comput.*, 222:59–79, 2013.

2 Kook Jin Ahn and Sudipto Guha. Access to data and number of iterations: Dual primal algorithms for maximum matching under resource constraints. *ACM Trans. Parallel Comput.*, 4(4):17:1–17:40, 2018.

3 Sepehr Assadi. A two-pass (conditional) lower bound for semi-streaming maximum matching. In *Proceedings of the 2022 ACM-SIAM Symposium on Discrete Algorithms, SODA 2022, Virtual Conference / Alexandria, VA, USA, January 9 – 12, 2022*, pages 708–742. SIAM, 2022.

4 Sepehr Assadi. A simple $(1-\epsilon)$-approximation semi-streaming algorithm for maximum (weighted) matching. In *2024 Symposium on Simplicity in Algorithms, SOSA 2024, Alexandria, VA, USA, January 8-10, 2024*, pages 337–354. SIAM, 2024.

5 Sepehr Assadi, MohammadHossein Bateni, Aaron Bernstein, Vahab S. Mirrokni, and Cliff Stein. Coresets meet EDCS: algorithms for matching and vertex cover on massive graphs. In *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2019, San Diego, CA, USA, January 6-9, 2019*, pages 1616–1635, 2019.

6 Sepehr Assadi and Soheil Behnezhad. Beating Two-Thirds For Random-Order Streaming Matching. In *48th International Colloquium on Automata, Languages, and Programming, ICALP 2021, July 12-16, 2021, Glasgow, Scotland (Virtual Conference)*, volume 198 of *LIPIcs*, pages 19:1–19:13. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2021.

7 Sepehr Assadi and Soheil Behnezhad. On the Robust Communication Complexity of Bipartite Matching. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, APPROX/RANDOM 2021, August 16-18, 2021, University of Washington, Seattle, Washington, USA (Virtual Conference)*, volume 207 of *LIPIcs*, pages 48:1–48:17. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2021.

8 Sepehr Assadi, Soheil Behnezhad, Sanjeev Khanna, and Huan Li. On regularity lemma and barriers in streaming and dynamic matching. In *Proceedings of the 55th Annual ACM Symposium on Theory of Computing, STOC 2023, Orlando, FL, USA, June 20-23, 2023*, pages 131–144. ACM, 2023.

9 Sepehr Assadi and Aaron Bernstein. Towards a unified theory of sparsification for matching problems. In *2nd Symposium on Simplicity in Algorithms, SOSA 2019, January 8-9, 2019, San Diego, CA, USA*, pages 11:1–11:20, 2019.

10 Sepehr Assadi, Arun Jambulapati, Yujia Jin, Aaron Sidford, and Kevin Tian. Semi-streaming bipartite matching in fewer passes and optimal space. In *Proceedings of the 2022 ACM-SIAM Symposium on Discrete Algorithms, SODA 2022, Virtual Conference / Alexandria, VA, USA, January 9 – 12, 2022*, pages 627–669. SIAM, 2022.

11 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, SODA 2017, Barcelona, Spain, Hotel Porta Fira, January 16-19*, pages 1723–1742. SIAM, 2017.

12 Sepehr Assadi, Sanjeev Khanna, Yang Li, and Grigory Yaroslavtsev. Maximum matchings in dynamic graph streams and the simultaneous communication model. In *Proceedings of the Twenty-Seventh Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2016, Arlington, VA, USA, January 10-12, 2016*, pages 1345–1364. SIAM, 2016.

13 Sepehr Assadi, Gillat Kol, Raghuvansh R. Saxena, and Huacheng Yu. Multi-pass graph streaming lower bounds for cycle counting, max-cut, matching size, and other problems. In *61st IEEE Annual Symposium on Foundations of Computer Science, FOCS 2020, Durham, NC, USA, November 16-19, 2020*, pages 354–364. IEEE, 2020.

14 Sepehr Assadi, S. Cliff Liu, and Robert E. Tarjan. An auction algorithm for bipartite matching in streaming and massively parallel computation models. In *4th Symposium on Simplicity in Algorithms, SOSA 2021, Virtual Conference, January 11-12, 2021*, pages 165–171. SIAM, 2021.

15 Sepehr Assadi and Vishvajeet N. Graph streaming lower bounds for parameter estimation and property testing via a streaming XOR lemma. In *STOC '21: 53rd Annual ACM SIGACT Symposium on Theory of Computing, Virtual Event, Italy, June 21-25, 2021*, pages 612–625. ACM, 2021.

16 Sepehr Assadi and Ran Raz. Near-quadratic lower bounds for two-pass graph streaming algorithms. In *61st IEEE Annual Symposium on Foundations of Computer Science, FOCS 2020, Durham, NC, USA, November 16-19, 2020*, pages 342–353. IEEE, 2020.

17 Sepehr Assadi and Vihan Shah. An asymptotically optimal algorithm for maximum matching in dynamic streams. In *13th Innovations in Theoretical Computer Science Conference, ITCS 2022, January 31 – February 3, 2022, Berkeley, CA, USA*, volume 215 of *LIPIcs*, pages 9:1–9:23. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2022.

18 Sepehr Assadi and Janani Sundaresan. Hidden permutations to the rescue: Multi-pass streaming lower bounds for approximate matchings. In *64th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2023, Santa Cruz, CA, USA, November 6-9, 2023*, pages 909–932. IEEE, 2023.

19 Sepehr Assadi and Janani Sundaresan. (Noisy) gap cycle counting strikes back: Random order streaming lower bounds for connected components and beyond. In *Proceedings of the 55th Annual ACM Symposium on Theory of Computing, STOC 2023, Orlando, FL, USA, June 20-23, 2023*, pages 183–195. ACM, 2023.

20 Amir Azarmehr and Soheil Behnezhad. Robust communication complexity of matching: EDCS achieves 5/6 approximation. In *50th International Colloquium on Automata, Languages, and Programming, ICALP 2023, July 10-14, 2023, Paderborn, Germany*, volume 261 of *LIPIcs*, pages 14:1–14:15. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2023.

21 Soheil Behnezhad, Alireza Farhadi, MohammadTaghi Hajiaghayi, and Nima Reyhani. Stochastic matching with few queries: New algorithms and tools. In *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2019, San Diego, CA, USA, January 6-9, 2019*, pages 2855–2874. SIAM, 2019.

22 Aaron Bernstein. Improved bounds for matching in random-order streams. In *47th International Colloquium on Automata, Languages, and Programming, ICALP 2020, July 8-11, 2020, Saarbrücken, Germany (Virtual Conference)*, volume 168 of *LIPIcs*, pages 12:1–12:13. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2020.

23 Aaron Bernstein, Aditi Dudeja, and Zachary Langley. A framework for dynamic matching in weighted graphs. In *STOC '21: 53rd Annual ACM SIGACT Symposium on Theory of Computing, Virtual Event, Italy, June 21-25, 2021*, pages 668–681. ACM, 2021.

24 Aaron Bernstein and Cliff Stein. Fully Dynamic Matching in Bipartite Graphs. In *Automata, Languages, and Programming – 42nd International Colloquium, ICALP 2015, Kyoto, Japan, July 6-10, 2015, Proceedings, Part I*, volume 9134 of *Lecture Notes in Computer Science*, pages 167–179. Springer, 2015.

25 Aaron Bernstein and Cliff Stein. Faster fully dynamic matchings with small approximation ratios. In *Proceedings of the Twenty-Seventh Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2016, Arlington, VA, USA, January 10-12, 2016*, pages 692–711. SIAM, 2016.

26 Stéphane Boucheron, Gábor Lugosi, and Pacal Massart. On concentration of self-bounding functions. *Electronic Journal of Probability*, 14:1884–1899, 2009.

27 Marc Bury, Elena Grigorescu, Andrew McGregor, Morteza Monemizadeh, Chris Schwiegelshohn, Sofya Vorotnikova, and Samson Zhou. Structural results on matching estimation with applications to streaming. *Algorithmica*, 81(1):367–392, 2019.

28 Marc Bury and Chris Schwiegelshohn. Sublinear estimation of weighted matchings in dynamic data streams. In *Algorithms – ESA 2015 – 23rd Annual European Symposium, Patras, Greece, September 14-16, 2015, Proceedings*, volume 9294 of *Lecture Notes in Computer Science*, pages 263–274. Springer, 2015.

29 Amit Chakrabarti, Graham Cormode, and Andrew McGregor. Robust lower bounds for communication and stream computation. In *Proceedings of the 40th Annual ACM Symposium on Theory of Computing, Victoria, British Columbia, Canada, May 17-20, 2008*, STOC '08, pages 641–650. ACM, 2008.

30 Lijie Chen, Gillat Kol, Dmitry Paramonov, Raghuvansh R. Saxena, Zhao Song, and Huacheng Yu. Almost optimal super-constant-pass streaming lower bounds for reachability. In *STOC '21: 53rd Annual ACM SIGACT Symposium on Theory of Computing, Virtual Event, Italy, June 21-25, 2021*, pages 570–583. ACM, 2021.

31 Ashish Chiplunkar, John Kallaugher, Michael Kapralov, and Eric Price. Factorial lower bounds for (almost) random order streams. In *63rd IEEE Annual Symposium on Foundations of Computer Science, FOCS 2022, Denver, CO, USA, October 31 – November 3, 2022*, pages 486–497. IEEE, 2022.

32 Rajesh Chitnis, Graham Cormode, Hossein Esfandiari, MohammadTaghi Hajiaghayi, Andrew McGregor, Morteza Monemizadeh, and Sofya Vorotnikova. Kernelization via sampling with applications to finding matchings and related problems in dynamic graph streams. In *Proceedings of the Twenty-Seventh Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2016, Arlington, VA, USA, January 10-12, 2016*, pages 1326–1344. SIAM, 2016.

33 Rajesh Hemant Chitnis, Graham Cormode, MohammadTaghi Hajiaghayi, and Morteza Monemizadeh. Parameterized streaming: Maximal matching and vertex cover. In *Proceedings of the Twenty-Sixth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2015, San Diego, CA, USA, January 4-6, 2015*, pages 1234–1251. SIAM, 2015.

34 Graham Cormode, Hossein Jowhari, Morteza Monemizadeh, and S. Muthukrishnan. The sparse awakens: Streaming algorithms for matching size estimation in sparse graphs. In *25th Annual European Symposium on Algorithms, ESA 2017, September 4-6, 2017, Vienna, Austria*, volume 87 of *LIPIcs*, pages 29:1–29:15. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2017.

35 Michael S. Crouch and Daniel M. Stubbs. Improved streaming algorithms for weighted matching, via unweighted matching. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, APPROX/RANDOM 2014, September 4-6, 2014, Barcelona, Spain*, volume 28 of *LIPIcs*, pages 96–104. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2014.

36 Jacques Dark and Christian Konrad. Optimal lower bounds for matching and vertex cover in dynamic graph streams. In *35th Computational Complexity Conference, CCC 2020, July 28-31, 2020, Saarbrücken, Germany (Virtual Conference)*, volume 169 of *LIPIcs*, pages 30:1–30:14. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2020.

37 Sebastian Eggert, Lasse Kliemann, Peter Munstermann, and Anand Srivastav. Bipartite matching in the semi-streaming model. *Algorithmica*, 63(1-2):490–508, 2012.

38 Leah Epstein, Asaf Levin, Julián Mestre, and Danny Segev. Improved approximation guarantees for weighted matching in the semi-streaming model. *SIAM J. Discret. Math.*, 25(3):1251–1265, 2011.

39 Hossein Esfandiari, MohammadTaghi Hajiaghayi, Vahid Liaghat, Morteza Monemizadeh, and Krzysztof Onak. Streaming algorithms for estimating the matching size in planar graphs and beyond. *ACM Trans. Algorithms*, 14(4):48:1–48:23, 2018.

40 Hossein Esfandiari, MohammadTaghi Hajiaghayi, and Morteza Monemizadeh. Finding large matchings in semi-streaming. In *IEEE International Conference on Data Mining Workshops, ICDM Workshops 2016, December 12-15, 2016, Barcelona, Spain*, pages 608–614. IEEE Computer Society, 2016.

41 Alireza Farhadi, MohammadTaghi Hajiaghayi, Tung Mah, Anup Rao, and Ryan A. Rossi. Approximate maximum matching in random streams. In *Proceedings of the 2020 ACM-SIAM Symposium on Discrete Algorithms, SODA 2020, Salt Lake City, UT, USA, January 5-8, 2020*, pages 1773–1785. SIAM, 2020.

42 Joan Feigenbaum, Sampath Kannan, Andrew McGregor, Siddharth Suri, and Jian Zhang. On graph problems in a semi-streaming model. *Theoretical Computer Science*, 348(2-3):207–216, 2005.

43 Moran Feldman and Ariel Szarf. Maximum matching sans maximal matching: A new approach for finding maximum matchings in the data stream model. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, APPROX/RANDOM 2022, September 19-21, 2022, University of Illinois, Urbana-Champaign, USA (Virtual Conference)*, volume 245 of *LIPIcs*, pages 33:1–33:24. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2022.

44 Manuela Fischer, Slobodan Mitrovic, and Jara Uitto. Deterministic (1+ϵ)-approximate maximum matching with poly(1/ϵ) passes in the semi-streaming model and beyond. In *STOC '22: 54th Annual ACM SIGACT Symposium on Theory of Computing, Rome, Italy, June 20 – 24, 2022*, pages 248–260. ACM, 2022.

45 Buddhima Gamlath, Sagar Kale, Slobodan Mitrovic, and Ola Svensson. Weighted matchings via unweighted augmentations. In *Proceedings of the 2019 ACM Symposium on Principles of Distributed Computing, PODC 2019, Toronto, ON, Canada, July 29 – August 2, 2019*, pages 491–500. ACM, 2019.

46 Paritosh Garg, Sagar Kale, Lars Rohwedder, and Ola Svensson. Robust Algorithms Under Adversarial Injections. In *47th International Colloquium on Automata, Languages, and Programming, ICALP 2020, July 8-11, 2020, Saarbrücken, Germany (Virtual Conference)*, volume 168 of *LIPIcs*, pages 56:1–56:15. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2020.

47 Mohsen Ghaffari and David Wajc. Simplified and Space-Optimal Semi-Streaming (2+ϵ)-Approximate Matching. In *2nd Symposium on Simplicity in Algorithms, SOSA 2019, January 8-9, 2019, San Diego, CA, USA*, volume 69 of *OASIcs*, pages 13:1–13:8. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2019.

48 Ashish Goel, Michael Kapralov, and Sanjeev Khanna. On the communication and streaming complexity of maximum bipartite matching. In *Proceedings of the Twenty-Third Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2012, Kyoto, Japan, January 17-19, 2012*, pages 468–485. SIAM, 2012.

49 Sudipto Guha and Andrew McGregor. Approximate quantiles and the order of the stream. In *Proceedings of the Twenty-Fifth ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems, June 26-28, 2006, Chicago, Illinois, USA*, pages 273–279. ACM, 2006.

50 Sudipto Guha and Andrew McGregor. Lower bounds for quantile estimation in random-order and multi-pass streaming. In *Automata, Languages and Programming, 34th International Colloquium, ICALP 2007, Wroclaw, Poland, July 9-13, 2007, Proceedings*, volume 4596 of *Lecture Notes in Computer Science*, pages 704–715. Springer, 2007.

51 Sudipto Guha and Andrew McGregor. Stream order and order statistics: Quantile estimation in random-order streams. *SIAM J. Comput.*, 38(5):2044–2059, 2009.

52 Manoj Gupta and Richard Peng. Fully dynamic (1+ ϵ)-approximate matchings. In *54th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2013, 26-29 October, 2013, Berkeley, CA, USA*, pages 548–557. IEEE Computer Society, October 2013.

53 Venkatesan Guruswami and Krzysztof Onak. Superlinear lower bounds for multipass graph processing. *Algorithmica*, 76(3):654–683, 2016.

54 Chien-Chung Huang and François Sellier. Maximum Weight b-Matchings in Random-Order Streams. In *30th Annual European Symposium on Algorithms, ESA 2022, September 5-9, 2022, Berlin/Potsdam, Germany*, volume 244 of *LIPIcs*, pages 68:1–68:14. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2022.

55 Shang-En Huang and Hsin-Hao Su. (1-ϵ)-approximate maximum weighted matching in poly(1/ϵ, log n) time in the distributed and parallel settings. In *Proceedings of the 2023 ACM Symposium on Principles of Distributed Computing, PODC 2023, Orlando, FL, USA, June 19-23, 2023*, pages 44–54. ACM, 2023.

56 Sagar Kale and Sumedh Tirodkar. Maximum matching in two, three, and a few more passes over graph streams. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, APPROX/RANDOM 2017, August 16-18, 2017, Berkeley, CA, USA*, volume 81 of *LIPIcs*, pages 15:1–15:21. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2017.

57 Ming-Yang Kao, Tak-Wah Lam, Wing-Kin Sung, and Hing-Fung Ting. A decomposition theorem for maximum weight bipartite matchings. *SIAM Journal on Computing*, 31(1):18–26, 2001.

58 Michael Kapralov. Better bounds for matchings in the streaming model. In *Proceedings of the Twenty-Fourth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2013, New Orleans, Louisiana, USA, January 6-8, 2013*, pages 1679–1697. SIAM, 2013.

59 Michael Kapralov. Space lower bounds for approximating maximum matching in the edge arrival model. In *Proceedings of the 2021 ACM-SIAM Symposium on Discrete Algorithms, SODA 2021, Virtual Conference, January 10 – 13, 2021*, pages 1874–1893. SIAM, 2021.

60 Michael Kapralov, Sanjeev Khanna, and Madhu Sudan. Approximating matching size from random streams. In *Proceedings of the Twenty-Fifth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2014, Portland, Oregon, USA, January 5-7, 2014*, pages 734–751. SIAM, 2014.

61 Michael Kapralov, Slobodan Mitrovic, Ashkan Norouzi-Fard, and Jakab Tardos. Space efficient approximation to maximum matching size from uniform edge samples. In *Proceedings of the 2020 ACM-SIAM Symposium on Discrete Algorithms, SODA 2020, Salt Lake City, UT, USA, January 5-8, 2020*, pages 1753–1772. SIAM, 2020.

62 Christian Konrad. Maximum matching in turnstile streams. In *Algorithms – ESA 2015 – 23rd Annual European Symposium, Patras, Greece, September 14-16, 2015, Proceedings*, volume 9294 of *Lecture Notes in Computer Science*, pages 840–852. Springer, 2015.

63 Christian Konrad. A Simple Augmentation Method for Matchings with Applications to Streaming Algorithms. In *43rd International Symposium on Mathematical Foundations of Computer Science, MFCS 2018, August 27-31, 2018, Liverpool, UK*, volume 117 of *LIPIcs*, pages 74:1–74:16. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2018.

64 Christian Konrad, Frédéric Magniez, and Claire Mathieu. Maximum matching in semi-streaming with few passes. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques – 15th International Workshop, APPROX 2012, and 16th International Workshop, RANDOM 2012, Cambridge, MA, USA, August 15-17, 2012. Proceedings*, volume 7408 of *Lecture Notes in Computer Science*, pages 231–242. Springer, 2012.

65 Christian Konrad and Kheeran K. Naidu. On two-pass streaming algorithms for maximum bipartite matching. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, APPROX/RANDOM 2021, August 16-18, 2021, University of Washington, Seattle, Washington, USA (Virtual Conference)*, volume 207 of *LIPIcs*, pages 19:1–19:18. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2021.

66 Christian Konrad and Kheeran K. Naidu. An unconditional lower bound for two-pass streaming algorithms for maximum matching approximation. In *Proceedings of the 2024 ACM-SIAM Symposium on Discrete Algorithms, SODA 2024, Alexandria, VA, USA, January 7-10, 2024*, pages 2881–2899. SIAM, 2024.

67 Christian Konrad, Kheeran K. Naidu, and Arun Steward. Maximum matching via maximal matching queries. In *40th International Symposium on Theoretical Aspects of Computer Science, STACS 2023, March 7-9, 2023, Hamburg, Germany*, volume 254 of *LIPIcs*, pages 41:1–41:22. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2023.

68 Harry Lang. Online facility location on semi-random streams. *CoRR*, abs/1711.09384, 2017. `arXiv:1711.09384`.

69 Andrew McGregor. Finding graph matchings in data streams. In *Approximation, Randomization and Combinatorial Optimization, Algorithms and Techniques, 8th International Workshop on Approximation Algorithms for Combinatorial Optimization Problems, APPROX 2005 and 9th InternationalWorkshop on Randomization and Computation, RANDOM 2005, Berkeley, CA, USA, August 22-24, 2005, Proceedings*, volume 3624 of *Lecture Notes in Computer Science*, pages 170–181. Springer, 2005.

70 Andrew McGregor and Sofya Vorotnikova. Planar matching in streams revisited. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, APPROX/RANDOM 2016, September 7-9, 2016, Paris, France*, volume 60 of *LIPIcs*, pages 17:1–17:12. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2016.

71 Andrew McGregor and Sofya Vorotnikova. A simple, space-efficient, streaming algorithm for matchings in low arboricity graphs. In *1st Symposium on Simplicity in Algorithms, SOSA 2018, January 7-10, 2018, New Orleans, LA, USA*, volume 61 of *OASIcs*, pages 14:1–14:4. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2018.

72 Morteza Monemizadeh, S. Muthukrishnan, Pan Peng, and Christian Sohler. Testable bounded degree graph properties are random order streamable. In *44th International Colloquium on Automata, Languages, and Programming, ICALP 2017, July 10-14, 2017, Warsaw, Poland*, volume 80 of *LIPIcs*, pages 131:1–131:14. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2017.

73 Ami Paz and Gregory Schwartzman. A $(2+\epsilon)$-approximation for maximum weight matching in the semi-streaming model. In *Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2017, Barcelona, Spain, January 16-19*, pages 2153–2161. SIAM, 2017.

74 Alexander Schrijver. *Combinatorial optimization: polyhedra and efficiency*, volume 24. Springer, 2003.

75 Sumedh Tirodkar. Deterministic algorithms for maximum matching on general graphs in the semi-streaming model. In *38th IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science, FSTTCS 2018, December 11-13, 2018, Ahmedabad, India*, volume 122 of *LIPIcs*, pages 39:1–39:16. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2018.

76 David Wajc. Negative association-definition, properties, and applications, 2017.

77 Andrew Chi-Chih Yao. Some complexity questions related to distributive computing (preliminary report). In *Proceedings of the 11h Annual ACM Symposium on Theory of Computing, April 30 – May 2, 1979, Atlanta, Georgia, USA*, pages 209–213. ACM, 1979.

78 Mariano Zelke. Weighted matching in the semi-streaming model. *Algorithmica*, 62(1-2):1–20, 2012.