Decremental Matching in General Graphs

Sepehr Assadi
Department of Computer Science, Rutgers University, Piscataway, NJ, USA

Aaron Bernstein
Department of Computer Science, Rutgers University, Piscataway, NJ, USA

Aditi Dudeja
Department of Computer Science, Rutgers University, Piscataway, NJ, USA

Abstract

We consider the problem of maintaining an approximate maximum integral matching in a dynamic graph G, while the adversary makes changes to the edges of the graph. The goal is to maintain a $(1+\varepsilon)$-approximate maximum matching for constant $\varepsilon > 0$, while minimizing the update time. In the fully dynamic setting, where both edge insertion and deletions are allowed, Gupta and Peng (see [29]) gave an algorithm for this problem with an update time of $O(\sqrt{m}/\varepsilon^2)$.

Motivated by the fact that the $O_\varepsilon(\sqrt{m})$ barrier is hard to overcome (see Henzinger, Krinninger, Nanongkai, and Saranurak [30]; Kopelowitz, Pettie, and Porat [34]), we study this problem in the *decremental* model, where the adversary is only allowed to delete edges. Recently, Bernstein, Probst-Gutenberg, and Saranurak (see [9]) gave an $O(\text{poly}(\log n/\varepsilon))$ update time decremental algorithm for this problem in *bipartite graphs*. However, beating $O(\sqrt{m})$ update time remained an open problem for *general graphs*.

In this paper, we bridge the gap between bipartite and general graphs, by giving an $O_\varepsilon(\text{poly}(\log n))$ update time algorithm that maintains a $(1+\varepsilon)$-approximate maximum integral matching under adversarial deletions. Our algorithm is randomized, but works against an adaptive adversary. Together with the work of Grandoni, Leonardi, Sankowski, Schwiegelshohn, and Solomon [26] who give an $O_\varepsilon(1)$ update time algorithm for general graphs in the *incremental* (insertion-only) model, our result essentially completes the picture for partially dynamic matching.

2012 ACM Subject Classification Theory of computation → Dynamic graph algorithms

Keywords and phrases Dynamic algorithms, matching, primal-dual algorithms

Digital Object Identifier 10.4230/LIPIcs.ICALP.2022.11

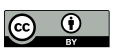
Category Track A: Algorithms, Complexity and Games

Funding *Sepehr Assadi*: Supported in part by an NSF CAREER Grant CCF-2047061, and a gift from Google Research.
Aaron Bernstein: Funded by NSF CAREER Grant 1942010.

1 Introduction

In dynamic graph algorithms, the main goal is to maintain a key property of the graph while an adversary makes changes to the edges of the graph. An algorithm is called *incremental* if it handles only insertions, *decremental* if it handles only deletions and *fully dynamic* if it handles both insertions as well as deletions. The goal is to minimize the update time of the algorithm, which is the time taken by the algorithm to adapt to a single adversarial edge insertion or deletion and output accordingly. For incremental/decremental algorithms, one typically seeks to minimize the *total update time*, which is the aggregate sum of update times over the *entire* sequence of edge insertions/deletions.

We consider the problem of maintaining a $(1+\varepsilon)$-approximation to the maximum matching in a dynamic graph. In the fully dynamic setting, the best known update time for this problem is $O(\sqrt{m}/\varepsilon^2)$ (see Gupta and Peng [29]), and the conditional lower bounds proved in the works




of Henzinger, Krinninger, Nanongkai, and Saranurak (see [30]) and Kopelowitz, Pettie, and Porat (see [34]) suggest that $O(\sqrt{m})$ is a hard barrier to break through. For this reason, several relaxations of this problem have been studied. For example, one line of research has shown that we can get considerably faster update times if we settle for large approximation factors (see for example [13, 14, 16, 11, 41, 37, 6, 33, 27, 5, 39, 15, 8, 12, 10, 11, 7]). Another research direction has been to consider the more relaxed incremental or decremental models. In the incremental (insertion-only) setting, there have been a series of upper and lower bound results (see [17, 19, 28]), culminating in the result of Grandoni, Leonardi, Sankowski, Schwiegelshohn, and Solomon (see [26]), who gave an optimal $O_\varepsilon(m)$ *total* update time (amortized $O_\varepsilon(1)$) for $(1+\varepsilon)$-approximate maximum matching.

The decremental (deletion-only) setting requires an entirely different set of techniques. In fact, even for the special case of bipartite graphs, $O_\varepsilon(m\sqrt{m})$ total time ($O_\varepsilon(\sqrt{m})$ update time) remained the best known until recently, when Bernstein, Probst-Gutenberg, and Saranurak (see [9]) gave an $\text{poly}(\log n/\varepsilon)$ amortized update time algorithm for the case of bipartite graphs. However, achieving a similar result for general graphs remained an open problem. Our main theorem essentially closes the gap between bipartite and general graphs.

▶ **Theorem 1.** *Let G be an unweighted graph and $\varepsilon \in (0,1)$. There is a decremental algorithm with total update time $O_\varepsilon(m \cdot \text{poly}(\log n))$ (amortized $O_\varepsilon(\text{poly}(\log n))$) that maintains a matching M of size at least $(1-\varepsilon)\cdot\mu(G)$ with high probability. Here G refers to the current version of the graph and $\mu(G)$ is the size of the maximum matching of G. The algorithm is randomized but works against an adaptive adversary. The dependence on ε is $2^{O(1/\varepsilon^2)}$.*

The guarantees of our algorithm hold against an *adaptive adversary*: which is allowed to choose an update sequence adaptively. This is in contrast to an *oblivious adversary* which cannot decide its updates based on the algorithm's output. Deterministic algorithms are more desirable because they are robust against such updates, which allows them to be used as a black-box in other static/dynamic algorithms. This property doesn't hold when we have the weaker oblivious adversary assumption. Thus, even though our algorithm is *randomized* (Monte Carlo), it has has the same power as a deterministic algorithm. We refer the reader to Section 1 of [41] and the references therein for a detailed discussion on this.

Our result largely completes the picture for partially dynamic matching by showing that in general graphs one can achieve $\text{poly}(\log n)$ update time in both incremental and decremental settings. But there are a few secondary considerations that remain. Firstly, our update time is $O_\varepsilon(\text{poly}(\log n))$, rather than the $O_\varepsilon(1)$ for the incremental setting (see [26]). Secondly, both the incremental result of [26] and our decremental result for general graphs have an exponential dependence on $1/\varepsilon$, whereas incremental/decremental algorithms for bipartite graphs have a polynomial dependence on $1/\varepsilon$ (see [26, 28]). Optimizing the dependence on ε and $\text{poly}(\log(n))$ factors thus remains an interesting open question for future work.

In algorithms literature, it has been the case that efficient matching algorithms for bipartite graphs do not easily extend to general graphs. Existence of blossoms (among other things), poses a technical challenge to obtaining analogous results for the general case. Consider the polynomial time algorithms for maximum matching for bipartite graphs, the most efficient algorithm, using alternating BFS, was discovered by Hopcroft and Karp; Karzanov (see [31, 32]) in 1973. However, several new structural facts and algorithmic insights were used by Micali and Vazirani to get the same runtime for general graphs (see [36]). This is also a feature of recent work in different models, such as the streaming (see [1, 22] and [24]), fully dynamic (see [10, 11] and [14, 7]), and parallel models (see [23, 40] and [35, 3]). We refer the reader to Section 1.3 of [2] for a detailed discussion of this phenomenon.

2 High-Level Overview

Our algorithm for Theorem 1 follows the high-level framework of *congestion balancing* introduced in [9]. They used it to solve approximate decremental matching in bipartite graphs, and also to solve more general flow problems. But the framework as they used it was entirely limited to cut/flow problems. As we discuss below, extending this framework to general graphs introduces significant technical challenges. Moreover, our result shows how the key subroutine of congestion balancing is naturally amenable to a primal-dual analysis, which we hope can pave the way for this technique to be applied to other decremental problems. Throughout this paper, we will use $\tilde{O}(\cdot)$ to hide $\text{poly}(\log n)$ factors in big-oh notations.

2.1 Previous Techniques

A fractional matching is a non-negative vector $\vec{x} \in \mathbb{R}_{\geqslant 0}$ satisfying fractional matching constraints: for all $v \in V$, $\sum_{e \ni v} x(e) \leqslant 1$. The starting point of [9] is that it is sufficient to develop an $\tilde{O}_\varepsilon(m)$ algorithm that does the following: it either maintains a *fractional matching* of size at least $(1-2\varepsilon) \cdot \mu(G)$ or certifies that $\mu(G)$ has dropped by a $(1-\varepsilon)$ factor because of adversarial deletions. Since a result of [41] enables us to round any bipartite fractional matching to an integral matching of almost the same value, such a fractional algorithm yields an algorithm to maintain an integral matching of size at least $(1-3\varepsilon) \cdot \mu(G)$ in $\tilde{O}_\varepsilon(m)$ total time under adversarial deletions. To motivate why [9] consider computing a fractional matching, consider the following "lazy" algorithm that works with an integral matching: compute an $(1+\varepsilon)$ approximate integral matching M of G using a static $O(m/\varepsilon)$ algorithm, wait for $\varepsilon \cdot \mu(G)$ edges of M to be deleted and then recompute the matching. Since we assume an adaptive adversary, the update time could be as large as $\Omega(m^2/\varepsilon \cdot n)$; this is because the adversary could proceed by only deleting edges of M. As a result, the goal should be to maintain a *robust* matching that can survive many deletions. Thus, the algorithm in [9] maintains a "balanced" fractional matching $\vec{x}$ that attempts to put a low value on every edge. In order to reduce the value of $\vec{x}$ by $\varepsilon \cdot \mu(G)$, the adversary will have to delete a lot of edges from every large matching.

Balanced Fractional Matching in Bipartite Graphs. In order to ensure that the fractional matching is spread out and robust, the algorithm in [9] imposes a capacity function κ on the edges of the graph (initially, all edges have low capacity) and compute a fractional matching obeying these capacities. The main ingredient of the algorithm is the subroutine M-OR-E*(G, ε, κ) which returns one of the following in $\tilde{O}_\varepsilon(m)$ time:

1. A fractional matching $\vec{x}$ with $\sum_{e \in E} x(e) \geqslant (1-\varepsilon) \cdot \mu(G)$, $x(e) \leqslant \kappa(e)$ for all $e \in E$, or,
2. A set of edges E^* with the following two properties.
   a. The total capacity through E^* must be small: $\kappa(E^*) = O(\mu(G) \log n)$ and,
   b. For all $|M| \geqslant (1-3\varepsilon) \cdot \mu(G)$, $|M \cap E^*| \geqslant \varepsilon \cdot \mu(G)$.

Property 2a ensures that the total capacity increase is small, while Property 2b ensures that we only increase capacity on important edges that are actually needed to form a large matching. The authors of [9] show that M-OR-E*() can be used as a black-box to solve decremental matching: at each step, M-OR-E*() is used to find a large fractional matching $\vec{x}$ (this matching is then rounded using [41] to get an integral matching), or to output the set E^* along which we increase capacities. They are able to show that because of Properties 2a and 2b, the edge capacities remain small on average.

The *congestion balancing* framework of [9] consists of an outer algorithm that uses M-OR-E*() as a subroutine. The outer algorithm for bipartite graphs, with some challenges, carries over to general graphs as well. But, M-OR-E*() is significantly more challenging to

implement for general graphs, so this subroutine will be our focus for the rest of the high level review. For bipartite graphs the algorithm M-OR-E*() is easier to implement because maximum fractional matchings correspond to max flows in bipartite graphs. Hence, existing algorithms for approximate maximum flows can be used to find the approximate maximum fractional matching obeying capacity κ. Moreover, if such a fractional matching is not large, then in bipartite graphs, the set of bottleneck edges E^* is exactly a minimum cut of the graph. For general graphs, due to the odd set constraints, max flow, which was the key analytic and algorithmic tool in [9], no longer corresponds to a maximum fractional matching that avoids the integrality gap.

2.2 Our Contribution: Implementing M-or-E*() in General Graphs

At a high-level, there are several structural and computational challenges to implementing M-OR-E*() in the case of general graphs. We explain what the potential impediments are, and detail how our techniques circumvent these.

Fractional Matchings in General Graphs. In general graphs, not all fractional matchings have a large integral matching in their support and therefore, cannot be rounded to give a large matching. While fractional matchings that obey odd set constraints do avoid the integrality gap, it seems hard to compute such a matching that also obeys capacity function κ. In order to get past this, we define a candidate fractional matching that is both easy to compute as well as contains a large integral matching in its support. More concretely, our fractional matching either puts flow one through an edge, or a flow of value at most ε (technically, we put flow much smaller than ε, but for this discussion, ε is sufficient). It is known that such a fractional matching has an integrality gap of at most $1+\varepsilon$, since it obeys all small odd set constraints. Our main contributions are two structural lemmas which show that we can find our candidate matching efficiently.

1. First, given a graph G with capacity κ, we want to determine if the *value* of the maximum fractional matching obeying κ and odd set constraints (denoted $\mu(G,\kappa)$) is at least $(1-\varepsilon)\cdot\mu(G)$. In general graphs, we do this by giving a sampling theorem: let G_s be the graph created by sampling edge e with probability proportional to $\kappa(e)$, then $\mu(G_s) \geqslant \mu(G,\kappa) - \varepsilon\cdot n$ with high probability. Thus, $\mu(G_s)$ is a good proxy for $\mu(G,\kappa)$ and it can be estimated efficiently by running any integral matching algorithm on G_s.
2. Suppose we have determined at some point that $\mu(G,\kappa)$ is large, we are still left with the task of finding a fractional matching. Our next contribution is a structural theorem that enables us to deploy existing flow algorithms to find such a matching. Let M be an approximate maximum matching of G_s. Let $M_L = \{e \in M \mid \kappa(e) \leqslant \beta\}$ and $M_H = \{e \in M \mid \kappa(e) > \beta\}$, where L and H are for low and high respectively and $\beta = O(1/\text{poly}\log n)$. Let $V_L = V(M_L)$ and $V_H = V(M_H)$. Intuitively, M breaks up our vertex set into two parts: vertices matched by low capacity edges (denoted V_L) and those that are matched by high capacity edges (denoted V_H). By adding some slack to our capacity constraints (we show that some slack can be incorporated in congestion balancing framework), we are able to treat the high-capacity edges as integral and compute a matching on $G[V_H]$ using a black box for integral matching in general graphs. Additionally, we show that the maximum fractional matching on low capacity edges of $G[V_L]$ has value at least as much as $|M_L|$ up to an additive error of $\varepsilon\cdot n$. To compute this fractional matching $\vec{f}$, we show that since we are only considering edges of small capacity, small odd set constraints are automatically satisfied, so we can transform $G[V_L]$ into a *bipartite graph* and then use an existing flow algorithm. We then output $M_H + \vec{f}$, which has the property that either the flow through an edge is 1 or at most ε.

The second obstacle is finding the set E^*. Recall, for bipartite graphs, the max flow-min cut theorem gives us an easy characterization of the bottleneck edges. However, for general graphs, this characterization is unclear. To overcome this, we consider the dual of the matching LP of G_s, and show that the bottleneck edges can be identified by considering the dual constraints associated with the edges. This generalizes the cut-or-matching approach of [9]. We compute an integral matching in G_s, and since the algorithm of [20] is primal-dual, we are also able to compute approximate duals for G_s.

Additionally, there are some secondary technical challenges as well. As mentioned before, our structural theorems only guarantee preservation of matching sizes up to an additive error of $\varepsilon \cdot n$. When $\mu(G) = o(n)$, then the results, applied directly are insufficient for us. To get around this, we use a vertex sparsification technique to get $O_\varepsilon(\log n)$ multigraphs which preserve all large matchings of G, but contain only $O(\mu(G)/\varepsilon)$ vertices (this reduction was first used in [4, 18]). However, we now have to show that all of our ideas work for multigraphs as well. In proving the second structural result, we need to introduce some slack in the capacities (see Definition 19). We show that congestion balancing extends to multigraphs, and is flexible enough to handle slack in capacities. Finally, the rounding scheme of [41] cannot be applied as a black-box to any fractional matching in a general graph. Thus, unlike in the bipartite case, we instead have to embed its techniques into the congestion balancing framework.

▶ Note 2. All our structural theorems (see Lemma 15 and Lemma 26) apply to graphs G with $\mu(G) \geqslant \Omega(\varepsilon \cdot n)$. A reduction used by the authors of [18, 4] allows us to construct $O_\varepsilon(\log n)$ multigraphs $H_1, \cdots, H_\lambda$ with $O(\mu(G)/\varepsilon)$ vertices, such that for every M of G with $|M| \geqslant (1-\varepsilon) \cdot \mu(G)$ at least $(1-\varepsilon) \cdot |M|$ edges of M are present in some H_i. Thus, at a high level, we have reduced our problem to the problem of decremental matching on multigraphs with large matchings. So, it is enough to prove our structural theorems for multigraphs with large matchings. We justify this assumption formally in the full version.

3 Preliminaries

We consider the problem of maintaining an approximate maximum integral matching in a graph G in the decremental setting. Throughout the paper, we will use G to refer to the current version of the graph, and let V and E be the vertex and edge sets of G respectively. Additionally, $\mu(G)$ denotes the size of the maximum integral matching of G. During the course of the algorithm, we will maintain a fractional matching $\vec{x}$. For a set $S \subseteq E$, we let $x(S) = \sum_{e \in S} x(e)$. Given a capacity function $\kappa(e)$, we say that x obeys κ if $x(e) \leqslant \kappa(e)$ for all $e \in E$. For a vector $\vec{x}$, we use $\text{supp}(\vec{x})$ to be set of edges that are in the support of $\vec{x}$. For a fractional matching $\vec{x}$, we say that $\vec{x}$ satisfies odd set constraints if for every odd-sized $B \subseteq V$, $\sum_{e \in G[B]} x(e) \leqslant \frac{|B|-1}{2}$. Given a capacity function κ on the edges of G, we use $\mu(G, \kappa)$ to denote the value of the maximum fractional matching of G that obeys the capacity function κ and the odd set constraints. Throughout this paper, for any $\varepsilon \in (0, 1)$, we will let $\alpha_\varepsilon = \log n \cdot 2^{60/\varepsilon^2}$ and $\rho_\varepsilon = \log n \cdot 2^{40/\varepsilon^2}$. We also need the following algorithm computing $(1+\varepsilon)$-approximate matching in general graphs.

▶ **Lemma 3** ([20, 25]). *There is an $O(m/\varepsilon)$ time algorithm Static-Match() that takes as input a graph G, and returns an integral matching M of G with $|M| \geqslant (1-\varepsilon) \cdot \mu(G)$.*

Roadmap for Extended Abstract. As mentioned in the overview, the main technical contribution of our paper is an algorithm for M-or-E*() in general graphs. The rest of the extended abstract focuses exclusively on this algorithm; the details of how M-or-E*() can be used a black box to attain Theorem 1 are left for the full version.

4 Main Contribution: M-or-E*() In General Graphs

As mentioned in the overview, we need our congestion balancing framework to act on multigraphs. This motivates our next few definitions.

▶ **Definition 4.** *Given a multigraph G, for a pair of vertices u and v, define $D(u,v)$ to be the set of edges between u and v. Similarly, if e is an edge between u, v, then $D(e) \coloneqq D(u,v)$.*

▶ **Definition 5.** *Let G be a multigraph with n vertices and m edges. Let κ be a capacity function on the edges. Suppose $\vec{x}$ is a fractional matching of G ($\vec{x}$ is a vector of length m). Then, we define $\vec{x}^C$ to be a vector of support size at most $\min\left\{m, \binom{n}{2}\right\}$, where for a pair of vertices, u and v, $x^C(u,v) \coloneqq \sum_{e \in D(u,v)} x(e)$. Essentially, if $\vec{x}$ is a fractional matching on a multigraph, then $\vec{x}^C$ is a fractional matching obtained by "collapsing" all edges together. Similarly, if $\vec{y}$ is a vector of support at most $\binom{n}{2}$, then we define $\vec{y}^D$ to be an m length vector such that for every $e \in E$ between a pair of vertices u and v, $y^D(e) \coloneqq \frac{y(u,v)\cdot\kappa(e)}{\kappa(D(e))}$ (D is for distributed, and we distribute the flow among the edges in proportion to their capacity).*

▶ Remark 6. Note that in doing the transformations in Definition 5, the support size of the transformed vector is always at most m. Thus, it doesn't negatively affect our runtime.

We now state our core ingredient, which either finds a balanced fractional matching of the multigraph G, or gives a set of edges E^* along which we can increase capacity.

▶ **Lemma 7.** *Let G be a multi-graph with $\mu(G) \geqslant \varepsilon \cdot n/16$. Let κ be a capacity function on the edges of G. There is an algorithm M-OR-E*(), that takes as input $G, \kappa, \varepsilon \in (0, 1/2)$ and $\mu \geqslant (1-\varepsilon)\cdot\mu(G)$ and in time $O(m \cdot \log n/\varepsilon)$ returns one of the following.*

(a) *A fractional matching $\vec{x}$ of value at least $(1-20\varepsilon)\cdot\mu$ with the following properties.*
- **(i)** *For any $e \in supp(\vec{x})$ with $\kappa(D(e)) > 1/\alpha_\varepsilon^2$, $x(e) = \frac{\kappa(e)}{\kappa(D(e))}$, and $x(D(e)) = 1$.*
- **(ii)** *For any $e \in supp(\vec{x})$ with $\kappa(D(e)) \leqslant 1/\alpha_\varepsilon^2$, $x(e) \leqslant \kappa(e)\cdot\alpha_\varepsilon$ and $x(D(e)) \leqslant \kappa(D(e))\cdot\alpha_\varepsilon$.*

(b) *A set E^* of edges such that $\kappa(E^*) = O(\mu \log n)$ such that for any integral matching M with $|M| \geqslant (1-3\varepsilon)\cdot\mu$, we have $|M \cap E^*| \geqslant \varepsilon\mu$. Moreover, $\kappa(e) < 1$ for all $e \in E^*$. Additionally, for every pair of vertices $u, v \in V$, either $D(u,v) \cap E^* = \emptyset$ or $D(u,v) \subseteq E^*$.*

We give some intuition for Lemma 7. Recall that we need a balanced fractional matching that contains a large integral matching in its support. However, as mentioned before, in the case of general graphs, finding such a balanced fractional matching is not straightforward. In order to get past this obstacle, we define a balanced fractional matching that is easy to find and also avoids the integrality gap. We will explain how to find it in the subsequent sections. For now, we explain at a high-level, why the fractional matching $\vec{x}$ found by Lemma 7 avoids the integrality gap. Consider $\vec{x}^C$. Observe from Lemma 7(a), that for any pair of vertices $u, v \in G$, either $x^C((u,v)) = 1$ or $x^C((u,v)) \leqslant 1/\alpha_\varepsilon \leqslant \varepsilon$. Thus, $\vec{x}$ satisfies odd-set constraints for all odd sets of size at most $1/\varepsilon$. By a folklore lemma, we can then argue that $\vec{x}$ contains an integral matching of size at least $(1+\varepsilon)^{-1} \cdot \sum_{u \neq v} x^C((u,v))$.

As mentioned before M-OR-E*() will be used as a subroutine in our **decremental algorithm**. The fractional matching output by M-OR-E*() will have certain properties, we state these properties now, since they will be helpful in visualizing the fractional matching. We give a proof of these in the full version.

▶ **Property 8.** *In our decremental algorithm, M-OR-E*$(G, \mu, \kappa, \varepsilon)$ outputs a fractional matching $\vec{x}$ with the following property: consider any $u, v \in V$ and let e, e' be edges between u, v (recall that G is a multigraph). Then, $\kappa(e') = \kappa(e)$ at all times during the algorithm.*

▶ **Definition 9.** *Let G be a multigraph, let $\varepsilon \in (0,1)$, let κ be a capacity function on the edges of G and let $\vec{x}$ be a fractional matching obeying κ. Then, we split $\vec{x}$ into two parts, $\vec{x}^f$ and $\vec{x}^i$, where $\vec{x} = \vec{x}^f + \vec{x}^i$, and $\text{supp}(\vec{x}^f) = \{e \in E \mid \kappa(D(e)) \leqslant 1/\alpha_\varepsilon^2\}$ and $\text{supp}(\vec{x}^i) = \{e \in E \mid \kappa(D(e)) > 1/\alpha_\varepsilon^2\}$ ($\vec{x}^f$ stands for fractional and $\vec{x}^i$ stands for integral. Though the edges in $\vec{x}^i$ do not have integral capacity, they are large enough to round them to 1).*

We briefly give the implications of this definition, since it is instructive to state them.

▶ **Property 10.** *Let G be any multigraph and $\vec{x}$ be the matching output by M-OR-E*$(G, \mu, \kappa, \varepsilon)$,*

(a) *For $\vec{x}$, we have $x(e) \leqslant \kappa(e) \cdot \alpha_\varepsilon^2$ for all $e \in E$. This follows immediately from Lemma 7(a).*

(b) *For any pair of vertices u, v, either $D(u,v) \subseteq supp(\vec{x}^i)$ and $D(u,v) \cap supp(\vec{x}^f) = \emptyset$ or, $D(u,v) \subseteq supp(\vec{x}^f)$ and $D(u,v) \cap supp(\vec{x}^i) = \emptyset$.*

(c) *Consider $\vec{z} = \vec{x}^i$. Then $supp(\vec{z}^C)$ is a matching. This is implied by Lemma 7(ai).*

5 Ingredients for Algorithm M-or-E*()

Recall we use $\mu(G, \kappa)$ to denote the value of the maximum fractional matching of G obeying capacity function κ and the odd set constraints. As in the congestion balancing set up of [9], we want to check if $\mu(G, \kappa) \geqslant (1 - \varepsilon) \cdot \mu(G)$. But, unlike in bipartite graphs, where we can use flows to find fractional matching, there is no simple way to check if $\mu(G, \kappa) \geqslant (1 - \varepsilon)\mu(G)$ in general graphs. Our first structural result circumvents this. Let G_s be obtained by sampling every edge e with probability $p(e) = \min\{1, \kappa(e) \cdot \rho_\varepsilon\}$. We show that $\mu(G_s) \geqslant \mu(G, \kappa) - \varepsilon\mu(G)$. Thus, Static-Match$(G_s, \varepsilon)$ is used to estimate $\mu(G, \kappa)$.

At a high level, M-OR-E*() proceeds in three phases. In Phase 1, it creates G_s and computes $\mu(G_s)$. If $\mu(G_s)$ is large, then by the above $\mu(G, \kappa)$ must also be large, so the algorithm proceeds to Phase 2, where it finds a fractional matching satisfying Lemma 7(a). On the other hand, if $\mu(G_s)$ is small, then it proceeds to Phase 3, where it finds the set of edges E^* satisfying Lemma 7(b), along which it increases capacity. In the subsequent sections, we will state the main structural properties we use in each of the phases. Finally, in Section 6, we put together these ingredients to give M-OR-E*(), and prove Lemma 7.

5.1 Phase 1 of M-or-E*()

Before we formally state the main guarantees of Phase 1, we will state some standard results in matching theory, that we will use in our main result for Phase 1.

5.1.1 Some Standard Ingredients For Phase 1

The first ingredient we use is the Tutte-Berge formula.

▶ **Definition 11.** *For any multigraph G and $U \subseteq V$, $odd_G(V \setminus U)$ denotes the number of odd components in $G[V \setminus U]$.*

▶ **Lemma 12** (Tutte-Berge Formula)**.** *[38] The size of a maximum matching in a graph $G = (V, E)$ is equal to $\frac{1}{2} \min_{U \subseteq V} (|U| + |V| - odd_G(V \setminus U))$.*

Additionally, we will use some properties of the matching polytope.

▶ **Lemma 13** ([38])**.** *Let G be any multigraph, let $\vec{x}$ be a fractional matching that in addition to the fractional matching constraints, also satisfies the following for all odd-sized $U \subseteq V$: $\sum_{e \in G[U]} x(e) \leqslant \frac{|U|-1}{2}$. Then, there is an integral matching $M \subseteq supp(x)$ with $|M| = \sum_{e \in E} x(e)$.*

▶ **Definition 14.** *Let G be any multigraph, and let $S, T \subseteq V$, then $\delta_G(S,T)$ is defined as the set of edges that have one endpoint in S and the other in T. Additionally, for $S \subseteq V$, we define $\delta_G(S)$ to be the set of edges that have one end point in S, and the other in $V \setminus S$.*

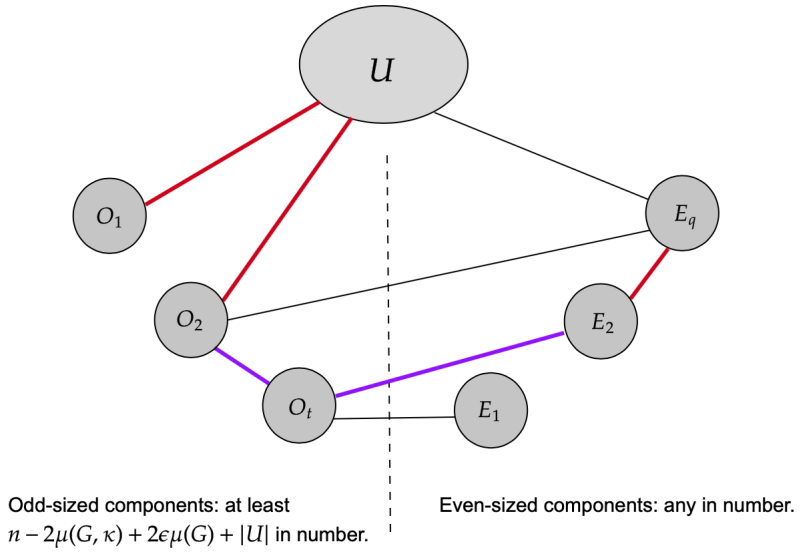


Figure 1 The figure shows the graph G, and a partition $\mathcal{P} = \{U, E_1, \cdots, E_q, O_1, \cdots, O_t\}$ satisfying properties (a) and (b) mentioned in proof of Lemma 15. The thick edges (in red and purple), are the edges in $\text{supp}(\vec{x})$, where $\vec{x}$ is the fractional matching realizing $\mu(G,\kappa)$. The purple edges (edges between the odd components, or between odd and even components) correspond to $E^{\mathcal{P}}_{\text{miss}}$, and $\sum_{e \in E^{\mathcal{P}}_{\text{miss}}} x(e) \geqslant 2 \cdot \varepsilon \cdot \mu(G)$.

5.1.2 Main Lemma for Phase 1

As mentioned earlier, in Phase 1 of M-OR-E*(), we first create a sampled graph G_s. In the following lemma, we show that $\mu(G_s)$ is a good estimate for $\mu(G,\kappa)$ with high probability.

▶ **Lemma 15.** *Let G be a multigraph with $\mu(G) \geqslant \varepsilon \cdot n/16$ where $\varepsilon \in (0, 1/2)$. Let κ be a capacity function on the edges of G, and let G_s be obtained by sampling every edge $e \in G$ with probability $p(e) = \min\{1, \kappa(e) \cdot \rho_\varepsilon\}$. Let $\mu(G,\kappa)$ be the value of the maximum fractional matching of G obeying the capacities κ, and the odd set constraints. Then, with high probability, $\mu(G_s) \geqslant \mu(G,\kappa) - \varepsilon \cdot \mu(G)$.*

Proof. We want to show that with high probability, $\mu(G_s) \geqslant \mu(G,\kappa) - \varepsilon \cdot \mu(G)$. In order to do this, by Lemma 12, it is sufficient to show that with high probability, $1 - 1/n^2$, $\frac{1}{2} \min_{U \subseteq V} (|U| + |V| - \text{odd}_{G_s}(V - U)) \geqslant \mu(G,\kappa) - \varepsilon \cdot \mu(G)$. Towards this, we consider a fixed partition $\mathcal{P}$ of V into sets $U, O_1, \cdots, O_t, E_1, \cdots, E_q$ with the following properties (see Figure 1).

(a) We have, $q \geqslant 0$ and $t > n - 2 \cdot \mu(G,\kappa) + 2\varepsilon \cdot \mu(G) + |U|$.

(b) Sets O_i for $i \in [t]$ are odd-sized sets and sets E_l for $l \in [q]$ are even-sized sets.

If $\mu(G_s) < \mu(G,\kappa) - \varepsilon \cdot \mu(G)$, then there is a partition $\mathcal{P} = \{U, O_1, \cdots, O_t, E_1, \cdots, E_q\}$ of G_s (from Lemma 12), satisfying (a) and (b) such that $G_s[V \setminus U]$ is the union of disconnected components $O_1, \cdots, O_t, E_1, \cdots, E_q$. If $G_s[V \setminus U]$ is the union of disconnected components $O_1, \cdots, O_t, E_1, \cdots, E_q$ then $\delta_{G_s}(O_i, O_l) = \emptyset$ for all $i \neq l$ and $\delta_{G_s}(O_i, E_l) = \emptyset$ for all $i \in [t]$,

$l \in [q]$. Thus, to upper bound the probability that $\mu(G_s) < \mu(G,\kappa) - \varepsilon \cdot \mu(G)$, it is sufficient to upper bound the probability that there exists a partition $\mathcal{P} = \{U, O_1, \cdots, O_t, E_1, \cdots, E_q\}$ satisfying (a) and (b), such that *none* of the edges $E^{\mathcal{P}}_{\mathsf{miss}} = \{e \mid e \in \delta_G(O_i, O_l) \text{ for } i \neq l\} \cup \{e \mid e \in \delta_G(O_i, E_l) \text{ for } i \in [t], l \in [q]\}$ are sampled in G_s. In order to bound this probability, we make the following claim.

▷ Claim 16. For a partition $\mathcal{P}$ satisfying (a) and (b), $\kappa(E^{\mathcal{P}}_{\mathsf{miss}}) \geqslant 2 \cdot \varepsilon \cdot \mu(G)$.

Proof. Let $\vec{x}$ be a fractional matching obeying odd set constraints and capacity function κ such that $\sum_{e \in E} x(e) = \mu(G,\kappa)$. We show that if $\kappa(E^{\mathcal{P}}_{\mathsf{miss}}) < 2 \cdot \varepsilon \cdot \mu(G)$, then, $x(E^{\mathcal{P}}_{\mathsf{miss}}) > \kappa(E^{\mathcal{P}}_{\mathsf{miss}})$, which will contradict the fact that $\vec{x}$ is a fractional matching obeying κ.

With this proof strategy in mind, for contradiction assume that $\kappa(E^{\mathcal{P}}_{\mathsf{miss}}) < 2 \cdot \varepsilon \cdot \mu(G) \leqslant n - 2 \cdot \mu(G,\kappa) + 2 \cdot \varepsilon \cdot \mu(G)$. The last inequality follows from the fact that $\mu(G,\kappa)$ corresponds to the value of the maximum fractional matching, so, $\mu(G,\kappa) \leqslant \frac{n}{2}$. Since $\vec{x}$ obeys odd set constraints, $\sum_{l \leqslant t} x(\delta_G(O_l)) \geqslant t$. Note that $\sum_{l \leqslant t} x(\delta_G(O_l, U)) \leqslant |U|$, otherwise for some $v \in U$, $\sum_{e \ni v} x(e) > 1$, violating the fact that $\vec{x}$ is a fractional matching. Next, we observe that $\sum_{l \leqslant t} x(\delta_G(O_l)) = x(E^{\mathcal{P}}_{\mathsf{miss}}) + \sum_{l \leqslant t} x(\delta_G(O_l, U))$. This follows from the fact that all edges emanating out of O_i in G, are incident on O_j for $j \neq i$, or E_k for some $k \in [q]$, or U. We have the following set of inequalities.

$$x(E^{\mathcal{P}}_{\mathsf{miss}}) = \sum_{l \leqslant t} x(\delta_G(O_l)) - \sum_{l \leqslant t} x(\delta_G(O_l, U)) \geqslant t - |U| > n - 2 \cdot \mu(G,\kappa) + 2 \cdot \varepsilon \cdot \mu(G)$$

The last inequality is because $\mathcal{P}$ satisfies (a) and (b). Thus, $x(E^{\mathcal{P}}_{\mathsf{miss}}) > \kappa(E^{\mathcal{P}}_{\mathsf{miss}})$. ◁

We also have a claim which allows us to only focus on $\mathcal{P}$ for which all $e \in E^{\mathcal{P}}_{\mathsf{miss}}$ have $\kappa(e) < 1/\rho_\varepsilon$. Let $\mathcal{H}^{\mathcal{P}}_{\mathsf{miss}}$ denote the event that none of the edges of $E^{\mathcal{P}}_{\mathsf{miss}}$ are sampled.

▶ **Observation 17.** *Suppose $\kappa(e) \geqslant 1/\rho_\varepsilon$ for any $e \in E^{\mathcal{P}}_{miss}$, then,* $\Pr\left(\mathcal{H}^{\mathcal{P}}_{miss}\right) = 0$.

The above observation follows from the fact that an edge e is sampled with probability $\min\{1, \kappa(e) \cdot \rho_\varepsilon\}$. From the above claim, it is sufficient to focus on $E^{\mathcal{P}}_{\mathsf{miss}}$ where all edges e have $\kappa(e) < 1/\rho_\varepsilon$, since these are the only $\mathcal{P}$ that contribute non-zero probability.

$$\begin{aligned}\Pr\left(\mathcal{H}^{\mathcal{P}}_{\mathsf{miss}}\right) &\leqslant \prod_{e \in E^{\mathcal{P}}_{\mathsf{miss}}} (1 - p(e)) \leqslant \exp\left(-\sum_{e \in E^{\mathcal{P}}_{\mathsf{miss}}} p(e)\right) = \exp\left(-\sum_{e \in E^{\mathcal{P}}_{\mathsf{miss}}} \kappa(e) \cdot \rho_\varepsilon\right) \\ &= \exp\left(-\varepsilon \cdot 2^{40/\varepsilon^2} \cdot \mu(G) \cdot \log n\right) \leqslant \exp\left(-2^{39/\varepsilon^2} \cdot \mu(G) \cdot \log n\right).\end{aligned}$$

Note that number of partitions $\mathcal{P}$ satisfying (a) and (b) are upper bounded by the number of ways of partitioning V, and it is known that a set of size n has $2^{n \cdot \log n}$ partitions. Since $n \leqslant 16 \cdot \mu(G)/\varepsilon$, the bound on the number of partitions is at most $2^{16\mu(G)/\varepsilon \cdot \log n}$ (by assumption of Lemma 15).

Thus, applying the equation above and taking a union bound over all the partitions, we know that the with probability $1 - \exp\left(-\mu(G) \cdot \log n/\varepsilon\right)$, in G_s, we have no partition $\mathcal{P} = \{U, O_1, \cdots, O_t, E_1, \cdots, E_q\}$ satisfying (a) and (b) such that $G_s[V \setminus U]$ is a union of disconnected components $O_1, \cdots, O_t, E_1, \cdots, E_q$. Thus, by Lemma 12, with high probability, $\mu(G_s) \geqslant \mu(G,\kappa) - \varepsilon \cdot \mu(G)$. ◀

5.2 Phase 2 of M-or-E*()

The algorithm proceeds to Phase 2 only if the integral matching M_s found in G_s is close to $\mu(G,\kappa)$. But note that although Phase 1 gives us a way to estimate the *value* of $\mu(G,\kappa)$ via $\mu(G_s)$ (by Lemma 15), it is unclear how to actually compute a corresponding fractional matching $\vec{x}$ that obeys capacities and odd set constraints. That is the goal of Phase 2: we show how to compute $\vec{x}$ and show that it is close in value to $\mu(G_s)$ with high probability, and therefore it is close to $\mu(G,\kappa)$ as well (by Lemma 15).

5.2.1 Preliminaries for Phase 2

Phase 2 starts by computing $M_s = \textsc{Static-Match}(G_s, \varepsilon)$ and then uses M_s to compute the desired fractional matching. We will split M_s into low capacity edges and high capacity edges, and as a result split V into vertices matched using high capacity edges, and low capacity edges. We begin by giving a formal definition of low capacity edges.

▶ **Definition 18.** *Let G be any multigraph, and let κ be a capacity function on the edges of G. Let $\varepsilon \in (0, 1/2)$. Define $E_L(G,\kappa) = \{e \in E \mid e \in D(u,v)$ and $\kappa(D(u,v)) \leqslant 1/\alpha_\varepsilon^2\}$. Intuitively, $E_L(G,\kappa)$ is the set of low total capacity edges.*

As mentioned in the high-level overview, in order to prove our probabilistic claims, we will give some slack to the capacities. This motivates our next definition.

▶ **Definition 19.** *Let G be a multigraph, and let κ be a capacity function on the edges of G. Let $\varepsilon \in (0, 1/2)$. We define the capacity function κ^+ as follows: for all $e \in E_L(G,\kappa)$, $\kappa^+(e) = \kappa(e) \cdot \alpha_\varepsilon$ and for all $e \in E \setminus E_L(G,\kappa)$, $\kappa^+(e) = \kappa(e)$.*

To make our analysis easier to follow, we need the following definition of a bipartite double cover of G.

▶ **Definition 20** (Bipartite Double Cover)**.** *Let G be a multigraph and κ be a capacity function on the edges of G. We define the $\textsc{bc}(G)$ to be the following bipartite graph with capacity function κ_{BC}.*

(a) *For every vertex $v \in V(G)$, make two copies v and v' in $V(\textsc{bc}(G))$.*

(b) *If e is an edge between $u, v \in V(G)$, then for each such e we add two edges e' and e'', one between u and v' and the other between v and u'. We let $\kappa_{BC}(e') = \kappa_{BC}(e'') = \kappa(e)$.*

We defer the proof of the following claim, relating $\mu(G)$ and $\mu(\textsc{bc}(G))$ to the full version.

▷ Claim 21. For any multigraph G, $\mu(\textsc{bc}(G)) \geqslant 2 \cdot \mu(G)$.

Next, we state the following lemma, which follows from standard techniques, and we give a formal proof of it in the full version. The lemma essentially states that a fractional matching which has low flow on all edges has a very small integrality gap.

▶ **Lemma 22.** *Let G be a multigraph, and let $\varepsilon \in (0,1)$. Let κ be a capacity function on the edges of G, with $\kappa(D(e)) \leqslant 1/\alpha_\varepsilon$ for all $e \in E(G)$. Then, $\mu(\textsc{bc}(G), \kappa_{BC}) \leqslant 2 \cdot (1+\varepsilon) \cdot \mu(G,\kappa)$, where $\mu(G,\kappa)$ is the maximum fractional matching of G obeying κ and the odd set constraints, and $\mu(\textsc{bc}(G), \kappa_{BC})$ is the maximum fractional matching of $\textsc{bc}(G)$ obeying κ_{BC}.*

Additionally, we will need the following lemma, which follows as a corollary of Lemma 12.

▶ **Proposition 23** (Extended Hall's Theorem)**.** *Let $G = (L \cup R, E)$ be a bipartite graph with $n = |L| = |R|$, then $\mu(G) = n - \max_{S \subseteq L}(|S| - |N_G(S)|)$, where $N_G(S)$ refers to the neighbourhood of S in G.*

In order to prove Lemma 26, we will use the following version of Chernoff bound.

▶ **Lemma 24** (Chernoff Bound). *[21] Let $X_1, \cdots, X_k$ be negatively correlated random variables, and let X denote their sum, and let $\mu = \mathbb{E}[X]$. Suppose $\mu_{min} \leqslant \mu \leqslant \mu_{max}$, then for all $\delta > 0$,* $\Pr(X \geqslant (1+\delta)\mu_{max}) \leqslant \left(\frac{e^\delta}{(1+\delta)^\delta}\right)^{\mu_{max}}$.

Additionally, we state the following observation. The proof follows from an application of the max-flow min-cut theorem (see [9]).

▶ **Observation 25.** *Let G be any bipartite multigraph, with vertex bipartitions S and T and capacity κ on the edges. Then, for any $C \subseteq S$, and $D \subseteq T$, we have, $|S| - |C| + |D| + \kappa(C, T \setminus D) \geqslant \mu(G, \kappa)$. Moreover, there are sets $C \subseteq S$ and $D \subseteq T$ such that equality holds.*

5.2.2 Main Result for Phase 2

We briefly give some intuition about the statement of next lemma. Recall in the high level review, we mentioned that M, the integral matching of G_s has two parts M_H, which is the high capacity part, and M_L, the low capacity part, and we defined $V_H = V(M_H)$ and $V_L = V(M_L)$. We said that congestion balancing allows us to give slack to capacities (κ^+ in Definition 19), and therefore, we can round up the capacities of M_H to 1. However, we still want to compute a fractional matching of $G[V_L]$. In order to do this, we observe that if the fractional matching in $G[V_L]$ is only on low capacity edges, then we can use flow algorithm on the low capacity edges of the bipartite graph $\text{BC}(G[V_L])$ to compute such a matching. Therefore, our main structural result for Phase 2 states that if $\vec{y}$ is a maximum fractional matching with support on the low capacity edges of $G[V_L]$, then with high probability $\sum_{e \in E} y(e) \geqslant |M_L| - \varepsilon \cdot \mu(G)$. We now state this result formally.

▶ **Lemma 26.** *Let G be a multigraph and let $\varepsilon \in (0, 1/2)$ and suppose $\mu(G) \geqslant \varepsilon \cdot n/16$. Let κ be a capacity function on the edges, and let G_s be the graph obtained from G by sampling each edge e with probability $p(e) = \kappa(e) \cdot \rho_\varepsilon$. Let $E_L := E_L(G, \kappa)$. Then, with high probability, for all $W \subseteq V$, we have $\mu(\text{BC}(G_s[W] \cap E_L)) \leqslant \mu(\text{BC}(G[W] \cap E_L), \kappa^+_{BC}) + 8 \cdot \varepsilon\mu(G)$.*

▶ Remark 27. Note that by definition of E_L, all edges $e \in G[W] \cap E_L$ have $\kappa(e) \leqslant 1/\alpha_\varepsilon^2$. Thus, $\kappa^+_{\text{BC}}(e) = \kappa_{\text{BC}}(e) \cdot \alpha_\varepsilon$ for all $e \in \text{BC}(G[W] \cap E_L)$ (recall Definition 19 and Definition 18).

Before we prove it, we have the following statement as a corollary of Lemma 26.

▶ **Corollary 28.** *Let G be a multi-graph, and let $\varepsilon \in (0, 1)$. Let κ be a capacity function on the edges of G with $\kappa(e) \leqslant 1/\alpha_\varepsilon$ for all $e \in E(G)$. Then, with high probability, **for all** $W \subseteq V$, $\mu(G_s[W] \cap E_L) \leqslant \mu(G[W] \cap E_L, \kappa^+) + 5 \cdot \varepsilon\mu(G)$.*

Proof. The inequality in the statement follows due to the following line of reasoning.

$$
\begin{aligned}
2 \cdot \mu(G_s[W] \cap E_L) &\leqslant \mu(\text{BC}(G_s[W] \cap E_L)) \\
&\quad \text{(since } 2 \cdot \mu(H) \leqslant \mu(\text{BC}(H)), \text{ see Claim 21)} \\
&\leqslant \mu(\text{BC}(G[W] \cap E_L), \kappa^+_{\text{BC}}) + 8 \cdot \varepsilon\mu(G) \\
&\quad \text{(by Lemma 26)} \\
&\leqslant 2 \cdot (1+\varepsilon) \cdot \mu(G[W] \cap E_L, \kappa^+) + 8\varepsilon\mu(G) \\
&\quad \text{(by Lemma 22)} \\
&\leqslant 2 \cdot \mu(G[W] \cap E_L, \kappa^+) + 10 \cdot \varepsilon\mu(G).
\end{aligned}
$$

The second to last inequality follows from the fact that any fractional matching in G obeying κ^+ and odd set constraints is upper bounded by $\mu(G)$ (by Lemma 13). ◀

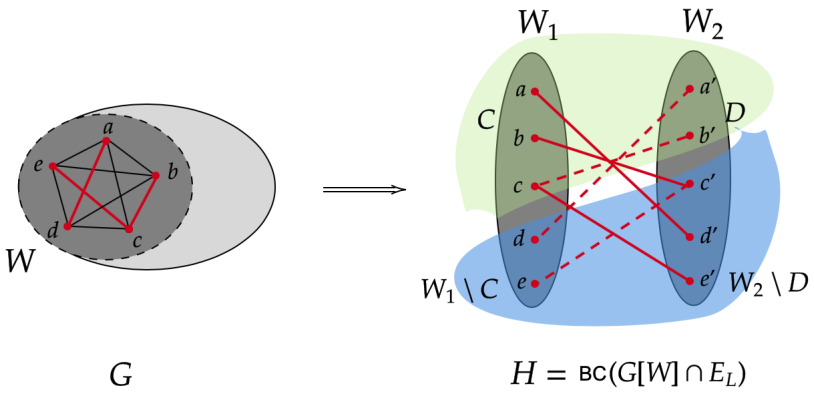


Figure 2 In the left panel, we consider the graph G, and $W = \{a, b, c, d, e\}$. The red edges correspond to the low capacity edges of $G[W]$, denoted $G[W] \cap E_L$. On the right panel, we have the bipartite graph $\text{BC}(G[W] \cap E_L)$, which has bipartitions W_1 and W_2. The solid red edges are the edges going between C and $W_2 \setminus D$, and these edges have capacity $\kappa(C, W_2 \setminus D)$.

Proof of Lemma 26. Throughout this proof we refer the reader to Figure 2. Consider a fixed $W \subseteq V$. From now, we will use H to denote $\text{BC}(G[W] \cap E_L)$ and let H_s denote $\text{BC}(G_s[W] \cap E_L)$. For the bipartite graph H, we will use W_1 and W_2 to denote the two bipartitions of H corresponding to W. We now want to prove that $\mu(H_s) \leqslant \mu(H, \kappa^+_{\text{BC}}) + 8\varepsilon\mu(G)$. To prove this, it is sufficient to show a set $C \subseteq W_1$ such that $|C| - |N_{H_s}(C)| \geqslant |W_1| - \mu(H, \kappa^+_{\text{BC}}) - 8\varepsilon\mu(G)$ with high probability. Then, by Proposition 23, we have the following inequality: $|W_1| - \mu(H, \kappa^+_{\text{BC}}) - 8\varepsilon\mu(G) \leqslant |C| - |N_{H_s}(C)| \leqslant |W_1| - \mu(H_s)$. This would prove our claim. Towards this, we consider the set $C \subseteq W$ satisfying the following equation (applying Observation 25 to H and κ^+_{BC}), and show that this is the required set.

$$|W_1| - |C| + |D| + \kappa^+_{\text{BC}}(C, W_2 \setminus D) = \mu(H, \kappa^+_{\text{BC}}). \tag{1}$$

We want to show that $|C| - |N_{H_s}(C)| \geqslant |W_1| - \mu(H, \kappa^+_{\text{BC}}) - 8\varepsilon\mu(G)$ with high probability. Let L be the set of vertices in $N_{H_s}(C) \cap W_2 \setminus D$. We know that $|N_{H_s}(C)| \leqslant |D| + |L|$. If we show that $|L| \leqslant \kappa^+_{\text{BC}}(C, W_2 \setminus D) + 8 \cdot \varepsilon\mu(G)$, then, $|N_{H_s}(C)| - 8\varepsilon \cdot \mu(G) \leqslant |D| + \kappa^+_{\text{BC}}(C, W_2 \setminus D)$. Substituting in Equation (1), we have $|W_1| - |C| + |N_{H_s}(C)| - 8\varepsilon \cdot \mu(G) \leqslant \mu(H, \kappa^+_{\text{BC}})$, which implies that $|C| - |N_{H_s}(C)| \geqslant |W_1| - 8 \cdot \varepsilon\mu(G) - \mu(H, \kappa^+_{\text{BC}})$. We now focus on bounding $|L|$.

Let $E_H(C, W_2 \setminus D)$ be the set of edges in H between C and $W_2 \setminus D$. Let X be the random variable that denotes the number of edges in $E_H(C, W_2 \setminus D)$ that are sampled in H_s. Note that X is not a sum of independent random variables. Recall that H is a subgraph of $\text{BC}(G)$, and suppose $e \in G[W] \cap E_L$ is included in G_s, then e' and e'' (recall e' and e'' are copies of e in $\text{BC}(G)$, Definition 20) are both included in H_s else both are excluded. Thus, the random variables associated with e' and e'' are correlated with each other. We instead consider an arbitrary subset of $E^*_H(C, W_2 \setminus D)$ of $E_H(C, W_2 \setminus D)$ that satisfies the following properties.

(a) For $e \in G[W]$, if $\{e', e''\} \subset E_H(C, W_2 \setminus D)$, then exactly one of e' or e'' is included in $E^*_H(C, W_2 \setminus D)$.

(b) For $e \in G[W]$, if $\{e', e''\} \cap E_H(C, W_2 \setminus D) = \{e'\}$, then only e' is included in $E^*_H(C, W_2 \setminus D)$ and if $\{e', e''\} \cap E_H(C, W_2 \setminus D) = \{e''\}$, then only e'' is included in $E^*_H(C, W_2 \setminus D)$.

We consider the random variable Y that denotes the number of edges in $E^*_H(C, W_2 \setminus D)$ that are included in H_s. Observe that Y is a sum of independent random variables satisfying the condition of Lemma 24. Moreover, $X \leqslant 2Y$. Thus, it is sufficient to upper bound the value

Y can take with high probability. Note that for any $e \in E_H(C, W_2 \setminus D)$, using the definition of H, $\kappa_{\text{BC}}(e) < 1/\alpha_\varepsilon^2$. Since $\rho_\varepsilon < \alpha_\varepsilon$, $p(e) = \rho_\varepsilon \cdot \kappa_{\text{BC}}(e) < 1$. So, we have,

$$\mathbb{E}[Y] \leqslant \sum_{e \in E^*_H(C, W_2 \setminus D)} p(e) \leqslant \rho_\varepsilon \cdot \kappa_{\text{BC}}(C, W_2 \setminus D) \leqslant \frac{\kappa^+_{\text{BC}}(C, W_2 \setminus D)}{2^{20/\varepsilon^2}}.$$

The last inequality follows from the fact that in H, $\kappa^+_{\text{BC}}(e) = \kappa_{\text{BC}}(e) \cdot \alpha_\varepsilon$ for all $e \in H$ (see Definition 19). This is because by definition of H and Definition 19, for all edges $e \in H$, the corresponding original edge in G is in $E_L(G, \kappa)$. We want to bound the $\Pr\left(Y \geqslant \frac{\kappa^+_{\text{BC}}(C, W_2 \setminus D)}{2^{20/\varepsilon^2}} + 4 \cdot \varepsilon\mu(G)\right)$. Applying Lemma 24 with $\delta = \frac{4 \cdot \varepsilon\mu(G) \cdot 2^{20/\varepsilon^2}}{\kappa^+_{\text{BC}}(C, W_2 \setminus D)}$, we have,

$$\begin{aligned}
\Pr\left(Y \geqslant \frac{\kappa^+_{\text{BC}}(C, W_2 \setminus D)}{2^{20/\varepsilon^2}} + 4 \cdot \varepsilon\mu(G)\right) &= \exp\left(\varepsilon\mu(G) - \varepsilon\mu(G) \log\left(1 + \frac{4 \cdot \varepsilon\mu(G) \cdot 2^{20/\varepsilon^2}}{\kappa^+_{\text{BC}}(C, W_2 \setminus D)}\right)\right) \\
&\leqslant \exp\left(\varepsilon\mu(G) - \varepsilon\mu(G) \log\left(1 + \varepsilon \cdot 2^{20/\varepsilon^2}\right)\right) \\
&\text{(Since } \kappa^+_{\text{BC}}(C, W_2 \setminus D) \leqslant 4 \cdot \mu(G) \text{ as proved below.)} \\
&\leqslant \exp\left(\varepsilon\mu(G) - \varepsilon\mu(G) \log\left(1 + 2^{19/\varepsilon^2}\right)\right) \\
&\text{(Using the fact that } 2^{1/\varepsilon^2} \geqslant 1/\varepsilon) \\
&= \exp\left(\varepsilon\mu(G) - 19\mu(G)/\varepsilon\right) \\
&\text{(Using the fact that } 2^{19/\varepsilon^2} \leqslant 2^{19/\varepsilon^2} + 1).
\end{aligned}$$

To see why $\kappa^+_{\text{BC}}(C, W_2 \setminus D) \leqslant 4 \cdot \mu(G)$, consider Equation (1), $\kappa^+_{\text{BC}}(C, W_2 \setminus D)$ is equal to

$$\begin{aligned}
\mu(H, \kappa^+_{\text{BC}}) - |W_1| + |C| - |D| &\leqslant 2 \cdot (1 + \varepsilon) \cdot \mu(G[W] \cap E_L(G, \kappa), \kappa^+) - |W_1| + |W_1| \\
&\leqslant 4 \cdot \mu(G).
\end{aligned}$$

The first inequality is due to Lemma 22, and the fact that $H = \text{BC}(G[W] \cap E_L(G, \kappa))$, and κ^+ satisfies the hypothesis. Finally, observe that $|L| \leqslant X \leqslant 2Y \leqslant \kappa^+_{\text{BC}}(C, W_2 \setminus D) + 8 \cdot \varepsilon\mu(G)$ with probability at least $\exp(-19\mu(G)/\varepsilon)$. Taking a union bound over all W, which are at most $2^{16 \cdot \mu(G)/\varepsilon}$ many (since by statement of the lemma, $\mu(G) \geqslant \varepsilon \cdot n/16$), we have our bound. ◀

5.3 Phase 3: Finding set E^*

The algorithm M-OR-E*() proceeds to Phase 3 if the matching found in G_s in Phase 1 is small, and hence $\mu(G, \kappa)$ is too small. In this case, we need to find a set E^* satisfying the properties of Lemma 7(b). In particular, we need to find a set E^* with $\kappa(E^*) = O(\mu(G) \log n)$ such that for every large matching M, there are a lot of edges going through E^*. In order to do this, we rely on the properties of the dual variables associated with the matching problem. The algorithm STATIC-MATCH(), luckily for us, solves both the primal as well as the dual solution. We first begin by stating the properties of the dual program, and then we state the properties of dual variables guaranteed by STATIC-MATCH().

▶ **Definition 29** ([38]). *We consider the dual of the matching linear program:*

1. *Every edge $e = (u, v)$ has a dual constraint $yz((u, v)) := y(u) + y(v) + \sum_{\substack{B \in V_{odd}, \\ e \in G[B]}} z(B)$.*
2. *We use $f(y, z) := \sum_{v \in V} y(u) + \sum_{B \subseteq V_{odd}} \frac{|B|-1}{2} z(B)$ to denote the dual objective function.*

We now describe some properties of the STATIC-MATCH(). All of the properties except Lemma 30(c) are given by the algorithm in [20]. We then show how to ensure property (c) as well by modifying the dual; see full version for details.

▶ **Lemma 30.** *There is an $O(m/\varepsilon)$ time algorithm* Static-Match() *that takes as input a graph G with m edges and a parameter $\varepsilon > 0$, and returns a matching M, and dual vectors $\vec{y}$ and $\vec{z}$ that have the following properties.*
(a) *It returns an integral matching M such that $|M| \geqslant (1-\varepsilon)\cdot\mu(G)$.*
(b) *A set Ω of laminar odd-sized sets, such that $\{B \mid z(B) > 0\} \subseteq \Omega$.*
(c) *For all odd-sized B with $|B| \geqslant 1/\varepsilon + 1$, $z(B) = 0$.*
(d) *Each $y(v)$ is a multiple of ε and $z(B)$ is a multiple of ε.*
(e) *For every edge $e \in E$, $yz(e) \geqslant 1-\varepsilon$. We say that e is* approximately covered *by $\vec{y}$ and $\vec{z}$.*
(f) *The value of the dual objective, $f(y,z)$ is at most $(1+\varepsilon)\cdot\mu(G)$.*

5.3.1 Main Guarantees of Phase 3

We now state the following helper lemma is instrumental in proving one of the two main properties of E^*, namely, that every large matching has a lot of edges passing through E^*.

▶ **Lemma 31.** *Suppose G is a graph, and let $H \subset G$ be a subgraph of G. Let $\vec{y}, \vec{z}$ be the dual variables returned on execution of* Static-Match(H, ε). *Let $E_H = \{e \in G \mid yz(e) \geqslant 1-\varepsilon\}$. For any matching M of G, then $|M \cap E \setminus E_H| \geqslant |M| - (1+\varepsilon)^2\cdot\mu(H)$.*

Proof. Suppose we scale up the dual variables $\vec{y}$ and $\vec{z}$ by a factor of $1+\varepsilon$. Then, $\vec{y}$ and $\vec{z}$ is a feasible solution for the dual matching program for the graph E_H. Thus, using weak duality, we have that $\mu(E_H) \leqslant (1+\varepsilon)f(y,z) \leqslant (1+\varepsilon)^2\cdot\mu(H)$ (this inequality follows from Lemma 30(f)). Suppose M is a matching of G with $|M\cap E\setminus E_H| < |M| - (1+\varepsilon)^2\cdot\mu(H)$ then, this implies that $|M \cap E \setminus E_H| < |M| - \mu(E_H)$. Thus, $|M| = |M\cap E_H| + |M\cap E\setminus E_H| < \mu(E_H) + |M| - \mu(E_H) = |M|$, which is a contradiction. ◀

We now define set E^*, and show that it has the property that $\kappa(E^*) = O(\mu(G)\log n)$.

▶ **Lemma 32.** *Let G be a graph multi-graph such that $\mu(G) \geqslant \varepsilon\cdot n/16$, and let κ be a capacity function on the edges of the graph. Suppose G_s is the graph obtained by sampling every edge $e \in E$ with probability $p(e) = \min\{1, \kappa(e)\cdot\rho_\varepsilon\}$. Let $\vec{y}, \vec{z}$ be the output of* Static-Match(G_s, ε). *Let $E^* = \{e \in E \mid yz(e) < 1-\varepsilon\}$. Then, for all $e \in E^*$, $\kappa(e) < 1$ and with high probability, $\kappa(E^*) = O(\mu(G)\log n)$.*

Proof. We consider the set $\mathcal{D}$ of assignments $\vec{y}, \vec{z}$ to the vertices and odd components that satisfy the following properties.
(a) For all $v \in V$, $y(v)$ is a multiple of ε, and for all $B \subset V$, $z(B)$ is a multiple of ε.
(b) Let $\Omega = \{B \subset V \mid z(B) > 0\}$, then Ω is laminar.
(c) If $z(B) > 0$ for some $B \subseteq V$, then $|B| \leqslant 1/\varepsilon$.
Observe that,

$$|\mathcal{D}| \leqslant \sum_{i=0}^{2n} \binom{n^{1/\varepsilon}}{i} \cdot \left(\frac{1}{\varepsilon}\right)^n \cdot \left(\frac{1}{\varepsilon}\right)^{2n} \leqslant n \cdot \binom{n^{1/\varepsilon}}{2n} \cdot \left(\frac{1}{\varepsilon}\right)^n \cdot \left(\frac{1}{\varepsilon}\right)^{2n} \leqslant 2^{(4/\varepsilon)\cdot n\log n}.$$

This number follows from the following argument. Since Ω is laminar, it can contain at most $2n$ sets. Moreover, from (c), we deduce that these $2n$ sets are chosen from among $n^{1/\varepsilon}$ sets. Further, from (a), we deduce that each $y(v)$ can be assigned at most $1/\varepsilon$ values, and for every $B \in \Omega$, $z(B)$ can be assigned $1/\varepsilon$ values. Therefore, for a given choice of Ω, there are at most $(1/\varepsilon)^n \cdot (1/\varepsilon)^{2n}$ choices for $\vec{y}$ and $\vec{z}$. Moreover, the above number also upper bounds the number of possible duals that can be a returned by the algorithm Static-Match(), since the duals in $\mathcal{D}$ satisfy a subset of the properties given in Lemma 30. Since $\mu(G) \geqslant \varepsilon\cdot n/16$, we can upper bound $|\mathcal{D}|$ by $2^{(32/\varepsilon^2)\cdot\mu(G)\log n}$.

Now consider a fixed $\vec{y}, \vec{z}$ that satisfies the above-mentioned properties, and let E^* be the set of edges that are not approximately covered by $\vec{y}, \vec{z}$. Note that for any $e \in E^*$, $\kappa(e) \leqslant 1/\rho_\varepsilon$. If not, then, $p(e) = 1$, and then it would be sampled in G_s, and be approximately covered by $\vec{y}, \vec{z}$ (by Lemma 30(e)). Thus, for any $e \in E^*$, $p(e) = \kappa(e) \cdot \rho_\varepsilon$. Now, suppose $\kappa(E^*) > \mu(G) \log n$. Then, observe that if $\vec{y}, \vec{z}$ is output by Static-Match(G_s, ε) (denote this event by $\mathcal{E}_{y,z}$) then none of the edges in E^* were sampled (denote this event by $\mathcal{E}_2$). Thus,

$$\begin{aligned}
\Pr(\mathcal{E}_{y,z}) &\leqslant \Pr(\mathcal{E}_2) \\
&\leqslant \prod_{e \in E^*} (1 - p(e)) \\
&\leqslant \exp\left(- \sum_{e \in E^*} \kappa(e) \cdot \rho_\varepsilon \right) \\
&\leqslant \exp\left(-2^{40/\varepsilon^2} \cdot \mu(G) \log n \right).
\end{aligned}$$

Our lemma follows by taking an upper bound over the set $\mathcal{D}$:

$$\begin{aligned}
\Pr\left(\bigcup_{\vec{y}, \vec{z} \in \mathcal{D}} \mathcal{E}_{y,z} \right) &\leqslant \exp\left(-2^{40/\varepsilon^2} \cdot \mu(G) \log n \right) \cdot 2^{(32/\varepsilon^2) \cdot \mu(G) \cdot \log n} \\
&= \exp\left(-O(2^{1/\varepsilon^2} \cdot \mu(G) \cdot \log n) \right).
\end{aligned}$$

◀

6 Algorithm M-or-E*()

In this section, we give the main subroutine M-or-E*() (see Lemma 7). We first define a few terms, and then state algorithm Frac-Match(), which follows almost directly from known results, with some very minor modifications; see full version for details.

▶ **Definition 33.** *Recall Definition 18, and consider an integral matching M, define $E_L^M(G, \kappa) = E_L(G, \kappa) \cap M$, and let V_L^M be the endpoints of $E_L^M(G, \kappa)$.*

▶ **Lemma 34.** *Given a multigraph G (possibly non-bipartite), with edge capacities κ and $\varepsilon \in (0, 1)$, such that $\kappa(D(e)) \leqslant 1/\alpha_\varepsilon$ for all $e \in E$, then there is an algorithm Frac-Match() that takes as input G, κ and ε, and returns a fractional matching $\vec{x}$ such that $\sum_{e \in E} x(e) \geqslant (1 - \varepsilon) \cdot \mu(G, \kappa)$, obeying the capacities κ and the odd set constraints. The runtime of this algorithm is $O(m \cdot \log n/\varepsilon)$.*

For the purpose of the algorithm recall κ^+ in Definition 19. We now formalize M-or-E*() from Lemma 7 in Algorithm 1 below; recall that the input is a multigraph G with $\mu(G) \geqslant \varepsilon \cdot n/16$.

We now show Lemma 7 holds.

Proof of Lemma 7. We first show the runtime of the algorithm. Graph G_s can be computed in time $O(m)$. Using Lemma 30, we conclude that we can compute E^* in order $O(m/\varepsilon)$ time by running Static-Match(G_s, ε) and Lemma 34 implies that Algorithm 1 takes $O(m \cdot \log n/\varepsilon)$ time.

We show Lemma 7((a)). First observe that V_L^M and $V(M_I)$ are disjoint, and since $\vec{y}$ and $\vec{x}$ are fractional matchings, $\vec{z}$ is also a fractional matching. Note that $|M| \geqslant \mu - 7\varepsilon\mu$. Moreover, $\sum_{e \in E} x(e) \geqslant (1 - \varepsilon) \cdot \mu(G[V_L^M] \cap E_L, \kappa^+)$. This follows from applying Lemma 34

Algorithm 1 M-OR-E*$(G, \kappa, \varepsilon, \mu)$.

Include each $e \in E(G)$ independently with probability $p(e) = \min\{1, \kappa(e) \cdot \rho_\varepsilon\}$ into graph G_s.
Let M and $\vec{y}, \vec{z}$ be the output of STATIC-MATCH(G_s, ε). ▷ Phase 1
if $|M| < \mu - 7\varepsilon\mu$ **then** ▷ Phase 3
  Return $E^* = \{e \in E(G) \mid yz(e) < 1 - \varepsilon\}$.
else ▷ Phase 2
  $M_I \leftarrow M \setminus E_L(G, \kappa)$
  $\vec{y} \leftarrow M_I^D$ ▷ See Definition 5
  $\vec{x} \leftarrow$ FRAC-MATCH$(G[V_L^M] \cap E_L(G, \kappa), \kappa^+, \varepsilon)$
  Return $\vec{z} \leftarrow \vec{y} + \vec{x}$.

to $G[V_L^M] \cap E_L(G, \kappa)$ with capacity function κ^+. Recall Definition 19 and Definition 18 to see that $\kappa^+(D(e)) \leqslant 1/\alpha_\varepsilon$ for $e \in G[V_L^M] \cap E_L(G, \kappa)$, thus satisfying the requirements of Lemma 34. Next, applying Corollary 28, we have, $\mu(G_s[V_L^M] \cap E_L) \leqslant \mu(G[V_L^M] \cap E_L, \kappa^+) + 5 \cdot \varepsilon\mu(G)$. Thus, we have $\sum_{e \in E} x(e) \geqslant (1 - \varepsilon) \cdot \big(\mu(G_s[V_L^M] \cap E_L) - 5\varepsilon\mu\big) \geqslant (1 - \varepsilon) \cdot (|M \setminus M_I| - 5\varepsilon\mu)$. This is because $M \setminus M_I$ is a matching of $G_s[V_L^M] \cap E_L$. So, $\sum_{e \in E} z(e) \geqslant |M_I| + (1 - \varepsilon) \cdot (|M \setminus M_I| - 5\varepsilon\mu) \geqslant (1 - \varepsilon) \cdot (\mu - 12\varepsilon\mu) \geqslant \mu - 13\varepsilon\mu$.

We now show Lemma 7(ai) and (aii). Consider any edge $e \in \text{supp}(\vec{z})$ with $\kappa(D(e)) \leqslant 1/\alpha_\varepsilon^2$, $e \in G[V_L^M] \cap E_L(G, \kappa)$. Thus, $e \in \text{supp}(\vec{x})$, and therefore, from Lemma 34, $z(e) = x(e) \leqslant \kappa^+(e) \leqslant \kappa(e) \cdot \alpha_\varepsilon$ and $z(D(e)) = x(D(e)) \leqslant \kappa(D(e)) \cdot \alpha_\varepsilon$. Similarly, for any $e \in \text{supp}(\vec{z})$ with $\kappa(D(e)) > 1/\alpha_\varepsilon^2$, $e \in \text{supp}(\vec{y})$. By definition of $\vec{y}$, $z(e) = y(e) = \kappa(e)/\kappa(D(e))$ (recall Definition 5) and $z(D(e)) = 1$. This proves our claim.

Next, we show Lemma 7(b). First recall from the assumption of Lemma 7 that $\mu \geqslant (1-\varepsilon) \cdot \mu(G)$. From this fact and Lemma 32, we can conclude that $\kappa(E^*) = O(\mu \log n)$ and that for all $e \in E^*$, $\kappa(e) < 1$. Next, observe that $\mu(G_s) \leqslant (1+\varepsilon) \cdot |M| \leqslant (1-6\varepsilon) \cdot \mu$. Applying Lemma 31 with $H = G_s$, we have, that for any matching M' of G, $|E^* \cap M'| \geqslant |M'| - (1+\varepsilon)^2 \cdot (1-6\varepsilon) \cdot \mu$. If $|M'| \geqslant (1 - 3\varepsilon) \cdot \mu$, then we have $|E^* \cap M'| \geqslant \varepsilon \cdot \mu$. Finally, consider any pair of vertices u, v and let $e', e'' \in D(u, v)$. Then, either both e', e'' are both approximately covered by $\vec{y}, \vec{z}$ or neither of them are. This implies that either $D(u, v) \subseteq E^*$ or $D(u, v) \cap E^* = \emptyset$. ◀

References

1 Kook Jin Ahn and Sudipto Guha. Linear programming in the semi-streaming model with application to the maximum matching problem. In *Proceedings of the 38th International Conference on Automata, Languages and Programming - Volume Part II*, ICALP'11, pages 526–538, Berlin, Heidelberg, 2011. Springer-Verlag.

2 Nima Anari and Vijay V. Vazirani. Matching is as easy as the decision problem, in the nc model. In *ITCS*, 2020.

3 Nima Anari and Vijay V. Vazirani. Planar graph perfect matching is in nc. *J. ACM*, 67(4), May 2020. doi:10.1145/3397504.

4 Sepehr Assadi, Sanjeev Khanna, and Yang Li. The stochastic matching problem with (very) few queries. *ACM Transactions on Economics and Computation (TEAC)*, 7(3):1–19, 2019.

5 Surender Baswana, Manoj Gupta, and Sandeep Sen. Fully dynamic maximal matching in o(log n) update time (corrected version). *SIAM J. Comput.*, 47(3):617–650, 2018. doi:10.1137/16M1106158.

6 Soheil Behnezhad and Sanjeev Khanna. *New Trade-Offs for Fully Dynamic Matching via Hierarchical EDCS*, pages 3529–3566. SIAM, 2022. doi:10.1137/1.9781611977073.140.

7 Soheil Behnezhad, Jakub Łącki, and Vahab Mirrokni. Fully dynamic matching: Beating 2-approximation in δ^{ϵ} update time. In *Proceedings of the Fourteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 2492–2508. SIAM, 2020.

8 Aaron Bernstein, Sebastian Forster, and Monika Henzinger. A deamortization approach for dynamic spanner and dynamic maximal matching. *ACM Trans. Algorithms*, 17(4):29:1–29:51, 2021. doi:10.1145/3469833.

9 Aaron Bernstein, Maximilian Probst Gutenberg, and Thatchaphol Saranurak. Deterministic decremental reachability, scc, and shortest paths via directed expanders and congestion balancing. In *2020 IEEE 61st Annual Symposium on Foundations of Computer Science (FOCS)*, pages 1123–1134. IEEE, 2020.

10 Aaron Bernstein and Cliff Stein. Fully dynamic matching in bipartite graphs. In Magnús M. Halldórsson, Kazuo Iwama, Naoki Kobayashi, and Bettina Speckmann, editors, *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. doi:10.1007/978-3-662-47672-7_14.

11 Aaron Bernstein and Cliff Stein. Faster fully dynamic matchings with small approximation ratios. In Robert Krauthgamer, editor, *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. doi:10.1137/1.9781611974331.ch50.

12 Sayan Bhattacharya, Deeparnab Chakrabarty, and Monika Henzinger. Deterministic dynamic matching in O(1) update time. *Algorithmica*, 82(4):1057–1080, 2020. doi:10.1007/s00453-019-00630-4.

13 Sayan Bhattacharya, Monika Henzinger, and Giuseppe F. Italiano. Deterministic fully dynamic data structures for vertex cover and matching. In *Proceedings of the Twenty-Sixth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '15, pages 785–804, USA, 2015. Society for Industrial and Applied Mathematics.

14 Sayan Bhattacharya, Monika Henzinger, and Danupon Nanongkai. New deterministic approximation algorithms for fully dynamic matching. In *Proceedings of the Forty-Eighth Annual ACM Symposium on Theory of Computing*, STOC '16, pages 398–411, New York, NY, USA, 2016. Association for Computing Machinery. doi:10.1145/2897518.2897568.

15 Sayan Bhattacharya, Monika Henzinger, and Danupon Nanongkai. Fully dynamic approximate maximum matching and minimum vertex cover in $O(\log^3 n)$ worst case update time. In Philip N. Klein, editor, *Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2017, Barcelona, Spain, Hotel Porta Fira, January 16-19*, pages 470–489. SIAM, 2017. doi:10.1137/1.9781611974782.30.

16 Sayan Bhattacharya and Peter Kiss. Deterministic rounding of dynamic fractional matchings. In Nikhil Bansal, Emanuela Merelli, and James Worrell, editors, *48th International Colloquium on Automata, Languages, and Programming, ICALP 2021, July 12-16, 2021, Glasgow, Scotland (Virtual Conference)*, volume 198 of *LIPIcs*, pages 27:1–27:14. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2021. doi:10.4230/LIPIcs.ICALP.2021.27.

17 Bartlomiej Bosek, Dariusz Leniowski, Piotr Sankowski, and Anna Zych. Online bipartite matching in offline time. In *2014 IEEE 55th Annual Symposium on Foundations of Computer Science*, pages 384–393, 2014. doi:10.1109/FOCS.2014.48.

18 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*, pages 1326–1344. SIAM, 2016.

19 Søren Dahlgaard. On the hardness of partially dynamic graph problems and connections to diameter. *CoRR*, abs/1602.06705, 2016. arXiv:1602.06705.

20 Ran Duan and Seth Pettie. Linear-time approximation for maximum weight matching. *Journal of the ACM (JACM)*, 61(1):1–23, 2014.

21 Devdatt P. Dubhashi and Alessandro Panconesi. *Concentration of Measure for the Analysis of Randomized Algorithms*. Cambridge University Press, 2009. URL: http://www.cambridge.org/gb/knowledge/isbn/item2327542/.

22 Sebastian Eggert, Lasse Kliemann, Peter Munstermann, and Anand Srivastav. Bipartite matching in the semi-streaming model. *Algorithmica*, 63:490–508, 2011.

23 Stephen Fenner, Rohit Gurjar, and Thomas Thierauf. Bipartite perfect matching is in quasi-nc. In *Proceedings of the Forty-Eighth Annual ACM Symposium on Theory of Computing*, STOC '16, pages 754–763, New York, NY, USA, 2016. Association for Computing Machinery. doi:10.1145/2897518.2897564.

24 Manuela Fischer, Slobodan Mitrovic, and Jara Uitto. Deterministic $(1+\epsilon)$-approximate maximum matching with poly$(1/\epsilon)$ passes in the semi-streaming model. *CoRR*, abs/2106.04179, 2021. arXiv:2106.04179.

25 Harold N. Gabow and Robert E. Tarjan. Faster scaling algorithms for general graph matching problems. *J. ACM*, 38(4):815–853, October 1991. doi:10.1145/115234.115366.

26 Fabrizio Grandoni, Stefano Leonardi, Piotr Sankowski, Chris Schwiegelshohn, and Shay Solomon. $(1 + \epsilon)$-approximate incremental matching in constant deterministic amortized time. In Timothy M. Chan, editor, *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2019, San Diego, California, USA, January 6-9, 2019*, pages 1886–1898. SIAM, 2019. doi:10.1137/1.9781611975482.114.

27 Fabrizio Grandoni, Chris Schwiegelshohn, Shay Solomon, and Amitai Uzrad. *Maintaining an EDCS in General Graphs: Simpler, Density-Sensitive and with Worst-Case Time Bounds*, pages 12–23. SIAM, 2022. doi:10.1137/1.9781611977066.2.

28 Manoj Gupta. Maintaining approximate maximum matching in an incremental bipartite graph in polylogarithmic update time. In *FSTTCS*, 2014.

29 Manoj Gupta and Richard Peng. Fully dynamic (1+ e)-approximate matchings. *2013 IEEE 54th Annual Symposium on Foundations of Computer Science*, pages 548–557, 2013.

30 Monika Henzinger, Sebastian Krinninger, Danupon Nanongkai, and Thatchaphol Saranurak. Unifying and strengthening hardness for dynamic problems via the online matrix-vector multiplication conjecture. In *Proceedings of the forty-seventh annual ACM symposium on Theory of computing*, pages 21–30, 2015.

31 John E. Hopcroft and Richard M. Karp. An n5/2 algorithm for maximum matchings in bipartite graphs. *SIAM J. Comput.*, 2:225–231, 1973.

32 A. Karzanov. On finding a maximum flow in a network with special structure and some applications. *Matematicheskie Voprosy Upravleniya Proizvodstvom (L.A. Lyusternik, ed.), Moscow State Univ. Press, Moscow, 1973, Issue 5, pp. 81–94, in Russian.*, 1973.

33 Peter Kiss. Deterministic dynamic matching in worst-case update time. In Mark Braverman, editor, *13th Innovations in Theoretical Computer Science Conference, ITCS 2022, January 31 - February 3, 2022, Berkeley, CA, USA*, volume 215 of *LIPIcs*, pages 94:1–94:21. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2022. doi:10.4230/LIPIcs.ITCS.2022.94.

34 Tsvi Kopelowitz, Seth Pettie, and Ely Porat. Higher lower bounds from the 3sum conjecture. In *Proceedings of the twenty-seventh annual ACM-SIAM symposium on Discrete algorithms*, pages 1272–1287. SIAM, 2016.

35 Meena Mahajan and Kasturi R. Varadarajan. A new nc-algorithm for finding a perfect matching in bipartite planar and small genus graphs (extended abstract). In *STOC '00*, 2000.

36 Silvio Micali and Vijay V. Vazirani. An o(sqrt(|v|) |e|) algorithm for finding maximum matching in general graphs. In *21st Annual Symposium on Foundations of Computer Science, Syracuse, New York, USA, 13-15 October 1980*, pages 17–27. IEEE Computer Society, 1980. doi:10.1109/SFCS.1980.12.

37 Mohammad Roghani, Amin Saberi, and David Wajc. Beating the Folklore Algorithm for Dynamic Matching. In Mark Braverman, editor, *13th Innovations in Theoretical Computer Science Conference (ITCS 2022)*, volume 215 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 111:1–111:23, Dagstuhl, Germany, 2022. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.ITCS.2022.111.

38 Alexander Schrijver et al. *Combinatorial optimization: polyhedra and efficiency*, volume 24. Springer, 2003.

39 Shay Solomon. Fully dynamic maximal matching in constant update time. In Irit Dinur, editor, *IEEE 57th Annual Symposium on Foundations of Computer Science, FOCS 2016, 9-11 October 2016, Hyatt Regency, New Brunswick, New Jersey, USA*, pages 325–334. IEEE Computer Society, 2016. doi:10.1109/FOCS.2016.43.

40 Ola Svensson and Jakub Tarnawski. The matching problem in general graphs is in quasi-nc. *2017 IEEE 58th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 696–707, 2017.

41 David Wajc. Rounding dynamic matchings against an adaptive adversary. In *Proceedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing*, pages 194–207, 2020.