New Deterministic Approximation Algorithms for Fully Dynamic Matching

[Extended Abstract]

Sayan Bhattacharya
IMSc, Chennai
bsayan@imsc.res.in

Monika Henzinger*
University of Vienna
monika.henzinger@univie.ac.at

Danupon Nanongkai†
KTH, Stockhom
danupon@kth.se

ABSTRACT

We present two deterministic dynamic algorithms for the maximum matching problem. (1) An algorithm that maintains a $(2+\epsilon)$-approximate maximum matching in general graphs with $O(\text{poly}(\log n, 1/\epsilon))$ update time. (2) An algorithm that maintains an α_K approximation of the *value* of the maximum matching with $O(n^{2/K})$ update time in bipartite graphs, for every sufficiently large constant positive integer K. Here, $1 \leq \alpha_K < 2$ is a constant determined by the value of K. Result (1) is the first deterministic algorithm that can maintain an $o(\log n)$-approximate maximum matching with polylogarithmic update time, improving the seminal result of Onak et al. [STOC 2010]. Its approximation guarantee almost matches the guarantee of the best *randomized* polylogarithmic update time algorithm [Baswana et al. FOCS 2011]. Result (2) achieves a better-than-two approximation with *arbitrarily small polynomial* update time on bipartite graphs. Previously the best update time for this problem was $O(m^{1/4})$ [Bernstein et al. ICALP 2015], where m is the current number of edges in the graph.

Categories and Subject Descriptors

F.2 [**Analysis of Algorithms and Problem Complexity**]: Miscellaneous

General Terms

Algorithms, Theory

Keywords

Data Structures, Dynamic Graph Algorithms

*The research leading to this work has received funding from the European Union's Seventh Framework Programme (FP7/2007-2013) under grant agreement number 317532 and from the European Research Council under the European Union's Seventh Framework Programme (FP7/2007-2013)/ERC grant agreement number 340506.

†Support by Swedish Research Council grant 2015-04659.

1. INTRODUCTION

In this paper, we consider the *dynamic maximum cardinality matching* problem. In this problem an algorithm has to quickly maintain an (integral) maximum-cardinality matching or its approximation, when the n-node input graph is undergoing edge insertions and deletions. We consider two versions of this problem: In the *matching version*, the algorithm has to output the change in the (approximate) matching, if any, after each edge insertion and deletion. In the *value version*, the algorithm only has to output the value of the matching. (Note that an algorithm for the matching version can be used to solve the value version within the same time.) When stating the running time below, we give the time *per update*[1]. If not stated otherwise, these results hold for both versions.

The state of the art for maintaining an *exact* solution for the value version of this problem is a randomized $O(n^{1.495})$-time algorithm [16]. This is complemented by various hardness results which rules out polylogarithmic update time [1, 8, 11]. As it is desirable for dynamic algorithms to have polylogarithmic update time, the recent work has focused on achieving this goal by allowing *approximate solutions*. The first paper that achieved this is by Onak and Rubinfeld [13], which gave a randomized $O(1)$-approximation $O(\log^2 n)$-time algorithm and a deterministic $O(\log n)$ approximation $O(\log^2 n)$-time algorithm. As stated in the two open problems in [13], this seminal paper opened up the doors for two research directions:

1. Designing a (possibly randomized) *polylogarithmic time* algorithm with smallest approximation ratio.

2. Designing a *deterministic polylogarithmic time* algorithm with constant approximation ratio.

The second question is motivated by the fact that *randomized* dynamic approximation algorithms only fulfill their approximation guarantee when used by an *oblivious* adversary, i.e., an adversary that gives the next update *without* knowing the outputs of the algorithm resulting from earlier updates. This limits the usefulness of randomized dynamic algorithms. In contrast, *deterministic* dynamic algorithms fulfill their approximation guarantee against *any* adversary, even non-oblivous ones. Thus, they can be used, for example, as a "black box" by any other (potentially static)

[1]In this discussion, we ignore whether the update time is amortized or worst-case as this is not the focus of this paper. The update time of our algorithm is amortized.

algorithm, while this is not generally the case for *randomized* dynamic algorithms. This motivates the search for deterministic fully dynamic approximation algorithms, even though a randomized algorithm with the same approximation guarantee might exists.

(1) Up to date, the best answer to the first question is the randomized 2 approximation $O(\log n)$ update time algorithm from [2]. It remains elusive to design a *better-than-two* approximation factor with polylogarithmic update time. Some recent works have focused on achieving such approximation factor with lowest update time possible. The current best update time is $O(m^{1/4}/\epsilon^{2.5})$ [4, 3], which is deterministic and guarantees a $(3/2+\epsilon)$ approximation factor.

(2) For the second question, deterministic polylogarithmic-time $(1+\epsilon)$-approximation algorithms were known for the special case of low arboricity graphs [12, 11, 15]. On general graphs, the paper [5] achieved a deterministic $(3+\epsilon)$-approximation polylogarithmic-time algorithm by maintaining a fractional matching; this algorithm however works only for the value version. *No* deterministic $o(\log n)$ approximation algorithm with polylogarithmic update time was known for the matching version. (There were many deterministic constant approximation algorithms with $o(m)$ update time for the matching version (e.g. [12, 5, 7, 4, 3]). The fastest among them requires $O(m^{1/4}/\epsilon^{2.5})$ update time [4].)

Our Results. We make progress on both versions of the problem as stated in Theorems 1 and 2.

THEOREM 1. *For every $\epsilon \in (0,1)$, there is a deterministic algorithm that maintains a $(2+\epsilon)$-approximate maximum matching in a graph in $O(poly(\log n, 1/\epsilon))$ update time, where n denotes the number of nodes in the graph.*

Theorem 1 answers Onak and Rubinfeld's second question positively. In fact, our approximation guarantee almost matches the best (2-approximation) one provided by a randomized algorithm [2].[2] Our algorithm for Theorem 1 is obtained by combining previous techniques [5, 7, 15] with two new ideas that concern fractional matchings. First, we dynamize the *degree splitting process* previously used in the parallel and distributed algorithms literature [9] and use it to reduce the size of the support of the fractional matching maintained by the algorithm of [5]. This helps us maintain an approximate integral matching cheaply using the result in [7]. This idea alone already leads to a $(3+\epsilon)$-approximation deterministic algorithm. Second, we improve the approximation guarantee further to $(2+\epsilon)$ by proving a new structural lemma that concerns the ratio between (i) the maximum (integral) matching in the support of a maximal fractional matching and (ii) the maximum (integral) matching in the whole graph. It was known that this ratio is at least 1/3. We can improve this ratio to 1/2 with a fairly simple proof (using Vizing's theorem [17]). We note that this lemma can be used to improve the analysis of an algorithm in [5] to get the following result: There is a deterministic algorithm that maintains a $(2+\epsilon)$ approximation to the size of the maximum matching in a general graph in $O(m^{1/3}/\epsilon^2)$ amortized update time.

[2]By combining our result with the techniques of [6] in a standard way, we also obtain a deterministic $(4+\epsilon)$-approximation $O(\text{poly}\log n \text{poly}(1/\epsilon)\log W)$-time for the dynamic maximum-weight matching problem, where W is the ratio between the largest and smallest edge weights.

THEOREM 2. *For every sufficiently large positive integral constant K, we can maintain an α_K-approximation to the value of the maximum matching[3] in a bipartite graph $G=(V,E)$, where $1 \le \alpha_K < 2$. The algorithm is deterministic and has an amortized update time of $O(n^{2/K})$.*

We consider Theorem 2 to be a step towards achieving a polylogarithmic time (randomized or deterministic) fully dynamic algorithm with an approximation ratio less than 2, i.e., towards answering Onak and Rubinfeld's first question. This is because, firstly, it shows that on bipartite graphs the better-than-two approximation factor can be achieved with arbitrarily small polynomial update time, as opposed to the previous best $O(m^{1/4})$ time of [3]. Secondly, it rules out a natural form of hardness result and thus suggests that a polylogarithmic-time algorithm with better-than-two approximation factor exists on bipartite graphs. More precisely, the known hardness results (e.g. those in [14, 1, 11, 8]) that rule out a polylogarithmic-time α-approximation algorithm (for any $\alpha > 0$) are usually in the form *"assuming some conjecture, there exists a constant $\delta > 0$ such that for any constant $\epsilon > 0$, there is no $(1-\epsilon)\alpha$-approximation algorithm that has $n^{\delta-\epsilon}$ update time"*; for example, for dynamically 2-*approximating* graph's diameter, this statement was proved for $\alpha = 2$ and $\delta = 1/2$ in [8], implying that any better-than-two approximation algorithm for this problem will require an update time close to $n^{1/2}$. Our result in 2 implies that a similar statement cannot be proved for $\alpha = 2$ for the bipartite matching problem since, for any constant $\delta > 0$, there is a $(2-\epsilon)$-approximation algorithm with update time, say, $O(n^{\delta/2})$ for some $\epsilon > 0$.

To derive an algorithm for Theorem 2, we use the fact that in a bipartite graph the size of the maximum fractional matching is the same as the size of the maximum integral matching. Accordingly, a *maximal fractional matching* (which gives 2-approximation) can be *augmented* by a fractional *b-matching*, for a carefully chosen capacity vector b, to obtain a better-than-two approximate fractional matching. The idea of "augmenting a bad solution" that we use here is inspired by the approach in the streaming setting by Konrad et al. [10]. But the way it is implemented is different as [10] focuses on using augmenting paths while we use fractional b-matchings.

Organization. In Section 1.1, we define some basic concepts and notations that will be used throughout the rest of this paper. In Section 2, we give an overview of our algorithm for Theorem 1. In Section 3, we highlight the main ideas behind our algorithm for Theorem 2. Finally, we conclude with some open problems in Section 4. All the missing details can be found in the full version of the paper.

1.1 Notations and preliminaries

Let $n = |V|$ and $m = |E|$ respectively denote the number of nodes and edges in the input graph $G = (V, E)$. Note that m changes with time, but n remains fixed. Let $\deg_v(E')$ denote the number of edges in a subset $E' \subseteq E$ that are incident upon a node $v \in V$. An (integral) matching $M \subseteq E$ is a subset of edges that do not share any common endpoints. The *size* of a matching is the number of edges contained in it. We are also interested in the concept of a *fractional*

[3]We can actually maintain an approximate *fractional* matching with the same performance bounds.

matching. Towards this end, we first define the notion of a *fractional assignment*. A fractional assignment w assigns a weight $w(e) \geq 0$ to every edge $e \in E$. We let $W_v(w) = \sum_{(u,v)\in E} w(u,v)$ denote the total weight received by a node $v \in V$ under w from its incident edges. Further, the *support* of w is defined to be the subset of edges $e \in E$ with $w(e) > 0$. Given two fractional assignments w, w', we define their addition $(w + w')$ to be a new fractional assignment that assigns a weight $(w + w')(e) = w(e) + w'(e)$ to every edge $e \in E$. We say that a fractional assignment w forms a *fractional matching* iff we have $W_v(w) \leq 1$ for all nodes $v \in V$. Given any subset of edges $E' \subseteq E$, we define $w(E') = \sum_{e\in E'} w(e)$. We define the *size* of a fractional matching w to be $w(E)$. Given any subset of edges $E' \subseteq E$, we let $\text{Opt}_f(E')$ (resp. $\text{Opt}(E')$) denote the maximum possible size of a fractional matching with support E' (resp. the maximum possible size of an integral matching $M' \subseteq E'$). Theorem 3 follows from the half-integrality of the matching polytope in general graphs and its total unimodularity in bipartite graphs.

THEOREM 3. *Consider any subset of edges $E' \subseteq E$ in the graph $G = (V, E)$. We have: $Opt(E') \leq Opt_f(E') \leq (3/2) \cdot Opt(E')$. Further, if the graph G is bipartite, then we have: $Opt_f(E') = Opt(E')$.*

Next, we recall that Gupta and Peng [7] gave a dynamic algorithm that maintains a $(1 + \epsilon)$-approximate maximum matching in $O(\sqrt{m}/\epsilon^2)$ update time. A simple modification of their algorithm gives the following result.

THEOREM 4. *[7] If the maximum degree in a dynamic graph never exceeds some threshold d, then we can maintain a $(1 + \epsilon)$-approximate maximum matching in $O(d/\epsilon^2)$ update time.*

Finally, we say that a fractional matching w is *α-maximal*, for $\alpha \geq 1$, iff $W_u(w) + W_v(w) \geq 1/\alpha$ for every edge $(u, v) \in E$. Using LP-duality and complementary slackness conditions, one can show the following result.

LEMMA 1. *We have $Opt_f(E) \leq 2\alpha \cdot w(E)$ for every α-maximal fractional matching w in a graph $G = (V, E)$.*

2. GENERAL GRAPHS

We give a dynamic algorithm for maintaining an approximate maximum matching in a general graph. We consider the following dynamic setting. Initially, the input graph is empty. Subsequently, at each time-step, either an edge is inserted into the graph, or an already existing edge is deleted from the graph. The node-set of the graph, however, remains unchanged. Our main result in this section is stated in Theorem 1. Throughout this section, we will use the notations and concepts introduced in Section 1.1.

2.1 Maintaining a large fractional matching

Our algorithm for Theorem 1 builds upon an existing dynamic data structure that maintains a large fractional matching. This data structure was developed in [5], and can be described as follows. Fix a small constant $\epsilon > 0$. Define $L = \lceil \log_{(1+\epsilon)} n \rceil$, and partition the node-set V into $L + 1$ subsets $V_0, \ldots, V_L$. We say that the nodes belonging to the subset V_i are in "level i". We denote the level of a node v by $\ell(v)$, i.e., $v \in V_i$ iff $\ell(v) = i$. We next define the "*level of an edge*" (u, v) to be $\ell(u, v) = \max(\ell(u), \ell(v))$. In other words, the level of an edge is the maximum level of its endpoints. We let $E_i = \{e \in E : \ell(e) = i\}$ denote the set of edges at level i, and define the subgraph $G_i = (V, E_i)$. Thus, note that the edge-set E is partitioned by the subsets $E_0, \ldots, E_L$. For each level $i \in \{0, \ldots, L\}$, we now define a fractional assignment w_i with support E_i. The fractional assignment w_i is uniform, in the sense that it assigns the same weight $w_i(e) = 1/d_i$, where $d_i = (1+\epsilon)^i$, to every edge $e \in E_i$ in its support. In contrast, $w_i(e) = 0$ for every edge $e \in E \setminus E_i$. Throughout the rest of this section, we refer to this structure as a *"hierarchical partition"*.

THEOREM 5. *[5] We can maintain a hierarchical partition dynamically in $O(\log n/\epsilon^2)$ update time. The algorithm ensures that the fractional assignment $w = \sum_{i=0}^{L} w_i$ is a $(1+\epsilon)^2$-maximal matching in $G = (V, E)$. Furthermore, the algorithm ensures that $1/(1+\epsilon)^2 \leq W_v(w) \leq 1$ for all nodes $v \in V$ at levels $\ell(v) > 0$.*

COROLLARY 1. *The fractional matching w in Theorem 5 is a $2(1 + \epsilon)^2$-approximation to $Opt_f(E)$.*

PROOF. Follows from Lemma 1 and Theorem 5. □

COROLLARY 2. *Consider the hierarchical partition in Theorem 5. There, we have $deg_v(E_i) \leq d_i$ for all nodes $v \in V$ and levels $0 \leq i \leq L$.*

PROOF. The corollary holds since $1 \geq W_v(w) \geq W_v(w_i) = \sum_{(u,v)\in E_i} w_i(u, v) = (1/d_i) \cdot \deg_v(E_i)$. □

Accordingly, throughout the rest of this section, we refer to d_i as being the *degree threshold* for level i.

2.2 An overview of our approach

We will now explain the main ideas that are needed to prove Theorem 1. Due to space constraints, we will focus on getting a constant approximation in $O(\text{poly} \log n)$ update time. See the full version of the paper for the complete proof of Theorem 1. First, we maintain a hierarchical partition as per Theorem 5. This gives us a $2(1 + \epsilon)^2$-approximate maximum fractional matching (see Corollary 1). Next, we give a dynamic data structure that deterministically *rounds* this fractional matching into an integral matching without losing too much in the approximation ratio. The main challenge is to ensure that the data structure has $O(\text{poly} \log n)$ update time, for otherwise one could simply use any deterministic rounding algorithm that works well in the static setting.

2.2.1 An ideal skeleton

Our dynamic rounding procedure, when applied on top of the data structure used for Theorem 5, will output a low-degree subgraph that approximately preserves the size of the maximum matching. We will then extract a large integral matching from this subgraph using Theorem 4. To be more specific, recall that w is the fractional matching maintained in Theorem 5. We will maintain a subset of edges $E' \subseteq E$ in $O(\text{poly} \log n)$ update time that satisfies two properties.

$$\text{There is a fractional matching } w' \text{ with support } E' \text{ such that } w(E) \leq c \cdot w'(E') \text{ for some constant } c \geq 1. \quad (1)$$

$$\deg_v(E') = O(\text{poly} \log n) \text{ for all nodes } v \in V. \quad (2)$$

Equation 1, along with Corollary 1 and Theorem 3, guarantees that the subgraph $G' = (V, E')$ preserves the size of the maximum matching in $G = (V, E)$ within a constant factor. Equation 2, along with Theorem 4, guarantees that we can maintain a matching $M' \subseteq E'$ in $O(\text{poly}\log n/\epsilon^2)$ update time such that $\text{Opt}(E') \leq (1+\epsilon)\cdot|M'|$. Setting ϵ to be some small constant (say, 1/3), these two observations together imply that we can maintain a $O(1)$-approximate maximum matching $M' \subseteq E$ in $O(\text{poly}\log n)$ update time.

To carry out this scheme, we note that in the hierarchical partition the degree thresholds $d_i = (1+\epsilon)^i$ get smaller and smaller as we get into lower levels (see Corollary 2). Thus, if most of the value of $w(E)$ is coming from the lower levels (where the maximum degree is already small), then we can easily satisfy equations 1, 2. Specifically, we fix a level $0 \leq L' \leq L$ with degree threshold $d_{L'} = (1+\epsilon)^{L'} = \Theta(\text{poly}\log n)$, and define the edge-set $Y = \bigcup_{j=0}^{L'} E_j$. We also define $w^+ = \sum_{i>L'} w_i$ and $w^- = \sum_{i\leq L'} w_i$. Note that $w(E) = w^+(E) + w^-(E)$. Now, consider two possible cases.

Case 1. $w^-(E) \geq (1/2)\cdot w(E)$. In other words, most of the value of $w(E)$ is coming from the levels $[0, L']$. By Corollary 2, we have $\deg_v(Y) \leq \sum_{j=0}^{L'} d_j \leq (L'+1)\cdot d_{L'} = \Theta(\text{poly}\log n)$ for all nodes $v \in V$. Thus, we can simply set $w' = w^+$ and $E' = Y$ to satisfy equations 1, 2.

Case 2. $w^+(E) > (1/2)\cdot w(E)$. In other words, most of the value of $w(E)$ is coming from the levels $[L'+1, L]$. To deal with this case, we introduce the concept of an *ideal skeleton*. See Definition 1. Basically, this is a subset of edges $X_i \subseteq E_i$ that scales *down* the degree of every node by a factor of $d_i/d_{L'}$. We will later show how to maintain a structure akin to an ideal skeleton in a dynamic setting. Once this is done, we can easily construct a new fractional assignment $\hat{w}_i$ that scales *up* the weights of the surviving edges in X_i by the same factor $d_i/d_{L'}$. Since $w_i(e) = 1/d_i$ for all edges $e \in E_i$, we set $\hat{w}_i(e) = 1/d_{L'}$ for all edges $e \in X_i$. To ensure that X_i is the support of the fractional assignment $\hat{w}_i$, we set $\hat{w}_i(e) = 0$ for all edges $e \in E \setminus X_i$. Let $X = \cup_{i>L'} X_i$ and $\hat{w} = \sum_{i>L'} \hat{w}_i$. It is easy to check that this transformation preserves the weight received by a node under the fractional assignment w^+, that is, we have $W_v(\hat{w}) = W_v(w^+)$ for all nodes $v \in V$. Accordingly, Lemma 2 implies that if we set $w' = \hat{w}$ and $E' = X$, then equations 1, 2 are satisfied.

Definition 1. Consider any level $i > L'$. An *ideal skeleton* at level i is a subset of edges $X_i \subseteq E_i$ such that $\deg(v, X_i) = (d_{L'}/d_i)\cdot\deg(v, E_i)$ for all nodes $v \in V$. Define a fractional assignment $\hat{w}_i$ on support X_i by setting $\hat{w}_i(e) = 1/d_{L'}$ for all $e \in X_i$. For every other edge $e \in E \setminus X_i$, set $\hat{w}_i(e) = 0$. Finally, define the edge-set $X = \bigcup_{i>L'} X_i$ and the fractional assignment $\hat{w} = \sum_{j>L'} \hat{w}_j$.

LEMMA 2. *We have: $\deg_v(X) = O(\text{poly}\log n)$ for all nodes $v \in V$, and $\hat{w}(E) = w^+(E)$. The edge-set X and the fractional assignment $\hat{w}$ are defined as per Definition 1.*

PROOF. Fix any node $v \in V$. Corollary 2, Definition 1 imply that: $\deg_v(X) = \sum_{j>L'} \deg_v(X_j) = \sum_{j>L'}(d_{L'}/d_j) \times \deg_v(E_j) \leq \sum_{j>L'}(d_{L'}/d_j)d_j = (L-L')d_{L'} = O(\text{poly}\log n)$.

To prove the second part, consider any level $i > L'$. Definition 1 implies that $W_v(\hat{w}_i) = (1/d_{L'})\cdot\deg_v(X_i) = (1/d_i)\cdot\deg_v(E_i) = W_v(w_i)$. Accordingly, we infer that: $W_v(\hat{w}) = \sum_{i>L'} W_v(\hat{w}_i) = \sum_{i>L'} W_v(w_i) = W_v(w^+)$. Summing over all the nodes, we get: $\sum_{v\in V} W_v(\hat{w}) = \sum_{v\in V} W_v(w^+)$. It follows that $\hat{w}(E) = w^+(E)$. □

2.2.2 A degree-splitting procedure

It remains to show to how to maintain an ideal skeleton. To gain some intuition, let us first consider the problem in a static setting. Fix any level $i > L'$, and let $\lambda_i = d_i/d_{L'}$. An ideal skeleton at level i is simply a subset of edges $X_i \subseteq E_i$ that scales down the degree of every node (w.r.t. E_i) by a factor λ_i. Can we compute such a subset X_i in $O(|E_i|\cdot\text{poly}\log n)$ time? Unless we manage to solve this problem in the static setting, we cannot expect to get a dynamic data structure for the same problem with $O(\text{poly}\log n)$ update time. The SPLIT(E_i) subroutine described below answers this question in the affirmative, albeit for $\lambda_i = 2$. Specifically, in linear time the subroutine outputs a subset of edges where the degree of each node is halved. If $\lambda_i > 2$, then to get an ideal skeleton we need to repeatedly invoke this subroutine $\log_2 \lambda_i$ times: each invocation of the subroutine reduces the degree of each node by a factor of two, and hence in the final output the degree of each node is reduced by a factor of λ_i.[4] This leads to a total runtime of $O(|E_i|\cdot\log_2\lambda_i) = O(|E_i|\cdot\log n)$ since $\lambda_i = d_i/d_{L'} \leq d_i \leq n$.

The SPLIT($\mathcal{E}$) subroutine, where $\mathcal{E} \subseteq E$. To highlight the main idea, we assume that (1) $\deg_v(\mathcal{E})$ is even for every node $v \in V$, and (2) there are an even number of edges in $\mathcal{E}$. Hence, there exists an Euler tour on $\mathcal{E}$ that visits each edge exactly once. We construct such an Euler tour in $O(|\mathcal{E}|)$ time and then collect alternating edges of this Euler tour in a set $\mathcal{H}$. It follows that (1) $\mathcal{H} \subseteq \mathcal{E}$ with $|\mathcal{H}| = |\mathcal{E}|/2$, and (2) $\deg_v(|\mathcal{H}|) = (1/2)\cdot\deg_v(|\mathcal{E}|)$ for every node $v \in V$. The subroutine returns the set of edges $\mathcal{H}$. In other words, the subroutine runs in $O(|\mathcal{E}|)$ time, and returns a subgraph that halves the degree of every node.

2.3 From ideal to approximate skeleton

We now shift our attention to maintaining an ideal skeleton in a dynamic setting. Specifically, we focus on the following problem: We are given an input graph $G_i = (V, E_i)$, with $|V| = n$, that is undergoing a sequence of edge insertions/deletions. The set E_i corresponds to the level i edges in the hierarchical partition (see Section 2.1). We always have $\deg_v(E_i) \leq d_i$ for all nodes $v \in V$ (see Corollary 2). There is a parameter $1 \leq \lambda_i = d_i/d_{L'} \leq n$. In $O(\text{poly}\log n)$ update time, we want to maintain a subset of edges $X_i \subseteq E_i$ such that $\deg_v(X_i) = (1/\lambda_i)\cdot\deg_v(E_i)$ for all nodes $v \in V$. The basic building block of our dynamic algorithm will be the (static) subroutine SPLIT($\mathcal{E}$) from Section 2.2.2.

Unfortunately, we will not be able to achieve our initial goal, which was to reduce the degree of *every* node by *exactly* the factor λ_i in a dynamic setting. For one thing, there might be some nodes v with $\deg_v(E_i) < \lambda_i$. It is clearly not possible to reduce their degrees by a factor of λ_i (otherwise their new degrees will be between zero and one). Further, we will need to introduce some *slack* in our data structures if we want to ensure polylogarithmic update time.

We now describe the structures that will be actually maintained by our dynamic algorithm. We maintain a partition

[4]To highlight the main idea, we assume that λ_i is a power of 2.

of the node-set V into two subsets: $B_i \subseteq V$ and $T_i = V \setminus B$. The nodes in B_i (resp. T_i) are called "big" (resp. "tiny"). We also maintain a subset of nodes $S_i \subseteq V$ that are called "spurious". Finally, we maintain a subset of edges $X_i \subseteq E_i$. Fix two parameters $\epsilon, \delta \in (0,1)$. For technical reasons that will become clear later on, we require that:

$$\epsilon = 1/100, \text{ and } \delta = \epsilon^2/L \tag{3}$$

We ensure that the following properties are satisfied.

$$\deg_v(E_i) \geq \epsilon d_i/L \text{ for all nodes } v \in B_i \setminus S_i. \tag{4}$$

$$\deg_v(E_i) \leq 2\epsilon d_i/L \text{ for all nodes } v \in T_i \setminus S_i. \tag{5}$$

$$|S_i| \leq \delta \cdot |B_i| \tag{6}$$

$$\frac{(1-\epsilon)}{\lambda_i} \cdot \deg_v(E_i) \leq \deg_v(X_i) \leq \frac{(1+\epsilon)}{\lambda_i} \cdot \deg_v(E_i) \text{ for all nodes } v \in B_i \setminus S_i. \tag{7}$$

$$\deg_v(X_i) \leq (1/\lambda_i) \cdot (2\epsilon d_i/L) \text{ for all nodes } v \in T_i \setminus S_i. \tag{8}$$

$$\deg_v(X_i) \leq (1/\lambda_i) \cdot d_i \text{ for all nodes } v \in S_i. \tag{9}$$

Equation 4 implies that all the non-spurious big nodes have large degrees in $G_i = (V, E_i)$. On a similar note, equation 5 implies that all the non-spurious tiny nodes have small degrees in G_i. Next, by equation 6, the number of spurious nodes is negligibly small in comparison with the number of big nodes. By equation 7, the degrees of the non-spurious big nodes are scaled by a factor that is very close to λ_i. Thus, the non-spurious big nodes satisfy an approximate version of the degree-splitting property required by Definition 1.

Moving on, by equation 8, the degrees of the non-spurious tiny nodes in X_i are at most $(1/\lambda_i) \cdot (2\epsilon d_i/L) = 2\epsilon d_{L'}/L$. Since each edge in X_i receives weight $1/d_{L'}$ under the assignment $\hat{w}_i$ (see Definition 1), we infer that $W_v(\hat{w}_i) = (1/d_{L'}) \cdot \deg_v(X_i) \leq 2\epsilon/L$ for all nodes $v \in T_i \setminus S_i$. Since there are at most $(L - L')$ relevant levels in a hierarchical partition, we infer that:

$$\sum_{i > L' : v \in T_i \setminus S_i} W_v(\hat{w}_i) \leq L \cdot (2\epsilon/L) = 2\epsilon \tag{10}$$

Since for a non-spurious tiny node $v \in T_i \setminus S_i$ we have $\deg_v(E_i) \leq 2\epsilon d_i/L$ (see equation 5) and $W_v(w_i) = (1/d_i) \cdot \deg_v(E_i) \leq 2\epsilon/L$, an exactly similar argument gives us:

$$\sum_{i > L' : v \in T_i \setminus S_i} W_v(w_i) \leq L \cdot (2\epsilon/L) = 2\epsilon \tag{11}$$

Equations 10, 11 have the following implication: The levels where v is a non-spurious tiny node contribute a negligible amount towards the weights $W_v(w^+)$ and $W_v(\hat{w})$ (see Section 2.2.1). Hence, although we are no longer guaranteed that the degrees of these nodes will be scaled down exactly by the factor λ_i, this should not cause too much of a problem – the sizes of the fractional assignments $w^+(E)$ and $\hat{w}(E)$ should still be close to each other as in Section 2.2.1.

Finally, Corollary 2 states that the degree of a node in E_i is at most d_i. Hence, according to the definition of an ideal skeleton (see Definition 1), the degree of a node in X_i ought not to exceed $(1/\lambda_i) \cdot d_i = d_{L'}$. Equation 9 ensures that the spurious nodes satisfy this property.

If the set of edges X_i satisfies the conditions described above, then we say that we have an *approximate-skeleton* at our disposal. This is formally stated as follows.

Definition 2. Fix any level $i > L'$, and suppose that there is a partition of the node-set V into two subsets $B_i \subseteq V$ and $T_i = V \setminus B_i$. Further, consider another subset of nodes $S_i \subseteq V$ and a subset of edges $X_i \subseteq E_i$. The tuple (B_i, T_i, S_i, X_i) is an *approximate-skeleton* iff it satisfies equations (4) – (9).

One may object at this point that we have deviated from the concept of an ideal skeleton (see Definition 1) so much that it will impact the approximation ratio of our final algorithm. To address this concern, we now state the following theorem whose proof appears in Section 2.5.

THEOREM 6. *For each level $i > L'$, consider an approximate skeleton as per Definition 2. Let $X = \bigcup_{i>L'} X_i$ denote the set of edges from these approximate-skeletons. Let $Y = \bigcup_{i \leq L'} E_i$ denote the set of edges from the remaining levels in the hierarchical partition. Then we have:*

1. *There is a fractional matching w' on support $X \cup Y$ such that $w(E) \leq O(1) \cdot w'(X \cup Y)$. Here, w is the fractional matching given by Theorem 5.*
2. *$deg_v(X \cup Y) = O(poly \log n)$ for all $v \in V$.*

In other words, the set of edges $X \cup Y$ satisfies equations 1, 2.

As per the discussion immediately after equations 1, 2, we infer the following guarantee.

COROLLARY 3. *Suppose that for each level $i > L'$ there is a dynamic algorithm that maintains an approximate-skeleton in $O(poly \log n)$ update time. Then we can also maintain a $O(1)$-approximate maximum matching in the input graph G in $O(poly \log n)$ update time.*

It remains to show how to maintain an approximate skeleton efficiently in a dynamic setting. Accordingly, we state the following theorem whose proof appears in Section 2.4.

THEOREM 7. *Consider any level $i > L'$. In $O(poly \log n)$ update time, we can maintain an approximate-skeleton at level i as per Definition 2.*

Corollary 3 and Theorem 7 imply that we can maintain a $O(1)$-approximate maximum matching in a dynamic graph in $O(\text{poly} \log n)$ update time.

2.4 Maintaing an approximate skeleton: Proof of Theorem 7

Fix a level $i > L'$. We will show how to efficiently maintain an approximate skeleton at level i under the assumption that $\lambda_i = 2$. In the full version of the paper, if $\lambda_i > 2$, then we iteratively apply the algorithm presented here $O(\log_2 \lambda_i) = O(\log_2(d_i/d_{L'})) = O(\log n)$ times, and each iteration reduces the degrees of the nodes by a factor of two. Hence, after the final iteration, we get a subgraph that is an approximate skeleton as per Definition 2.

We maintain the set of edges $E_{B_i} = \{(u,v) \in E_i : \{u,v\} \cap B_i \neq \emptyset\}$ that are incident upon the big nodes. Further, we associate a "status bit" with each node $v \in V$, denoted by $\text{STATUS}[v] \in \{0,1\}$. We ensure that they satisfy two conditions: (1) If $\text{STATUS}[v] = 1$, then $\deg_v(E_i) \geq \epsilon d_i/L$ (which is the threshold for non-spurious big nodes in equation 4). (2) If $\text{STATUS}[v] = 0$, then $\deg_v(E_i) \leq 2\epsilon d_i/L$ (which is the threshold for non-spurious tiny nodes in equation 5).

Whenever an edge incident upon v is inserted into (resp. deleted from) E_i, we update the status bit of v *in a lazy manner* (i.e., we flip the bit only if one of the two conditions is violated). We define an "epoch" of a node v to be the time-interval between any two consecutive flips of the bit STATUS$[v]$. Since there is a gap of $\epsilon d_i/L$ between the thresholds in equations 4, 5, we infer that:

$$\text{In any epoch of a node } v, \text{ at least } \epsilon d_i/L \text{ edge insertions/deletions incident upon } v \text{ takes place in } E_i. \quad (12)$$

Our dynamic algorithm runs in *"phases"*. In the beginning of a phase, there are no spurious nodes, i.e., we have $S_i = \emptyset$. During a phase, we handle the insertion/deletion of an edge in E_i as follows.

2.4.1 Handling the insertion/deletion of an edge

Consider the insertion/deletion of an edge (u, v) in E_i. To handle this event, we first update the set E_i and the status bits of u, v. If $\{u, v\} \cap B_i \neq \emptyset$, then we also update the edge-set E_{B_i}. Next, for every endpoint $x \in \{u, v\} \setminus S_i$, we check if the node x violates any of the equations 4, 5, 7. If yes, then we set $S_i \leftarrow S_i \cup \{x\}$. Finally, we check if $|S_i| > \delta \cdot |B_i|$, and if yes, then we terminate the phase by calling the subroutine TERMINATE-PHASE(.).

2.4.2 The subroutine TERMINATE-PHASE(.)

We scan through the nodes in S_i. For each such node $v \in S_i$, if STATUS$[v] = 1$, then we set $B_i \leftarrow B_i \cup \{v\}$, $T_i \leftarrow T_i \setminus \{v\}$, and ensure that all the edges $(u, v) \in E_i$ incident upon v are included in E_{B_i}. Else if STATUS$[v] = 0$, then we set $T_i \leftarrow T_i \cup \{v\}$, $B_i \leftarrow B_i \setminus \{v\}$, and ensure that all the edges $(u, v) \in E_i$ incident upon v are excluded from E_{B_i}. Finally, we set $S_i \leftarrow \emptyset$ and $X_i \leftarrow$ SPLIT(E_{B_i}) (see Section 2.2.2). From the next edge insertion/deletion in E_i, we begin a new phase.

2.4.3 Correctness.

At the start of a phase, clearly all the properties hold. This fact, along with the observation that an edge is never inserted into X_i during the middle of a phase, implies that equations 8, 9 hold all the time. Whenever a node violates equations 4, 5, 7, we make it spurious. Finally, whenever equation 6 is violated, we terminate the phase. This ensures that all the properties hold all the time.

2.4.4 Analyzing the amortized update time.

Handling an edge insertion/deletion in E_i in the middle of a phase needs $O(1)$ update time. Just before a phase ends, let b and s respectively denote the number of big and spurious nodes. Since a phase ends only when equation 6 is violated, we have $s \geq \delta \cdot b$. In the subroutine TERMINATE-PHASE(.), updating the edge-set E_{B_i} requires $O(s \cdot d_i)$ time, since we need to go through all the s nodes in S, and for each such node, we need to check all the edges incident upon it (and a node can have at most d_i edges incident upon it by Corollary 2). At this stage, the set B_i consists of at most $(s+b)$ nodes, and so the set E_{B_i} consists of at most $(s+b)d_i$ edges. Hence, the call to the subroutine SPLIT(E_{B_i}) takes $O((s+b)d_i)$ time. Accordingly, the total time taken to terminate the phase is $O((s+b)d_i) = O((s+s/\delta)d_i) = O(sd_i/\delta)$. We thus reach the following conclusion: The total time spent on a given phase is equal to $O(sd_i/\delta)$, where s is the number of spurious nodes at the end of the phase. Since $S_i = \emptyset$ in the beginning of the phase, we can also interpret s as being the number of nodes that *becomes spurious* during the phase. Let C denote a counter that is initially set to zero, and is incremented by one each time some node becomes spurious. From the preceding discussion, it follows that the total update time of our algorithm, across all the phases, is at most $O(Cd_i/\delta)$. Let t be the total number of edge insertions/deletions in E_i. We will show that $C = O(tL/(\epsilon^2 d_i))$. This will imply an amortized update time of $O((1/t) \cdot Cd_i/\delta) = O(L/\epsilon^2\delta) = O(\text{poly} \log n)$. The last equality holds due to equation 3.

Note that during a phase a node v becomes spurious because of one of two reasons: (1) It violated equations 4 or 5. In this case, the node's status bit is flipped. Hence, by equation 12, between any two such events, at least $\epsilon d_i/L$ edge insertions/deletions occur incident upon v. (2) It violates equation 7. In this event, note that in the beginning of the phase we had $v \in B_i$, $\deg_v(X_i) = (1/2) \cdot \deg_v(E_i) = (1/\lambda_i) \cdot \deg_v(E_i)$ and $\deg_v(E_i) \geq \epsilon d_i/L$. The former guarantee holds since we set $X_i \leftarrow$ SPLIT(E_{B_i}) at the end of the previous phase, whereas the latter guarantee follows from equation 4. On the other hand, when the node v violates equation 7, we find that $\deg_v(X_i)$ differs from $(1/\lambda_i) \cdot \deg_v(E_i)$ by at least $(\epsilon/\lambda_i) \cdot \deg_v(E_i) = (\epsilon/2) \cdot \deg_v(E_i)$. Accordingly, during this time-interval (that starts at the beginning of the phase and ends when equation 7 is violated), at least $\Omega(\epsilon^2 d_i/L)$ edge insertions/deletions incident upon v must have taken place in E_i. To summarize, for each unit increment in C, we must have $\Omega(\epsilon^2 d_i/L)$ edge insertions/deletions in E_i. Thus, we have $C = O(t/(\epsilon^2 d_i/L)) = O(tL/(\epsilon^2 d_i))$.

2.5 Approximation guarantee from approximate skeletons: Proof of Theorem 6

We devote this section to the complete proof of Theorem 6. At a high level, the main idea behind the proof remains the same as in Section 2.2.1. We will have to overcome several intricate obstacles, however, because now we are dealing with the relaxed notion of an approximate skeleton as defined in Section 2.3.

We start by focussing on the second part of Theorem 6, which states that the degree of every node in $X \cup Y$ is at most $O(\text{poly} \log n)$. This is stated and proved in Lemma 3.

LEMMA 3. *Consider the subsets of edges $X \subseteq E$ and $Y \subseteq E$ as per Theorem 6. Then we have $deg_v(X \cup Y) = O(poly \log n)$ for every node $v \in V$.*

PROOF. We first bound the degree of a node $v \in V$ in X. Towards this end, consider any level $i > L'$. By equation 9, we have that $\deg_v(X_i) \leq (1/\lambda_i) \cdot d_i = d_{L'}$ for all spurious nodes $v \in S_i$. By equation 8, we have $\deg_v(X_i) \leq (1/\lambda_i) \cdot (2\epsilon d_i/L) = 2\epsilon d_{L'}/L \leq d_{L'}$ for all non-spurious tiny nodes $v \in T_i \setminus S_i$. Finally, by equation 7 and Corollary 2, we have that $\deg_v(X_i) \leq ((1+\epsilon)/\lambda_i) \cdot \deg_v(E_i) \leq (1+\epsilon) \cdot (d_i/\lambda_i) = (1+\epsilon) \cdot d_{L'}$ for all non-spurious big nodes $v \in B_i \setminus S_i$. By Definition 2, a node belongs to exactly one of the three subsets – S_i, $T_i \setminus S_i$ and $B_i \setminus S_i$. Combining all these observations, we get: $\deg_v(X_i) \leq (1+\epsilon) \cdot d_{L'} = O(\text{poly} \log n)$ for all nodes $v \in V$. Now, summing over all $i > L'$, we get: $\deg_v(X) = \sum_{i>L'} \deg_v(X_i) \leq (L - L') \cdot O(\text{poly} \log n) = O(\text{poly} \log n)$ for all the nodes $v \in V$.

Next, we bound the degree of a node $v \in V$ in Y. Note that the degree thresholds in the levels $[0, L']$ are all at most

$d_{L'}$. Specifically, for all $i \leq L'$ and $v \in V$, Corollary 2 implies that $\deg_v(E_i) \leq d_i \leq d_{L'} = O(\text{poly}\log n)$. Hence, for every node $v \in V$, we have $\deg_v(Y) = \sum_{i \leq L'} \deg_v(E_i) \leq (L'+1) \cdot O(\text{poly}\log n) = O(\text{poly}\log n)$.

To summarize, the maximum degree of a node in the edge-sets X and Y is $O(\text{poly}\log n)$. Hence, for every node $v \in V$, we have: $\deg_v(X \cup Y) = \deg_v(X) + \deg_v(Y) = O(\text{poly}\log n)$. This concludes the proof of the lemma. □

We now focus on the first part of Theorem 6, which guarantees the existence of a large fractional matching with support $X \cup Y$. This is stated in the lemma below. Note that Lemma 3 and Lemma 4 together imply Theorem 6.

LEMMA 4. *Consider the subsets of edges $X \subseteq E$ and $Y \subseteq E$ as per Theorem 6. Then there exists a fractional matching w' on support $X \cup Y$ such that $w(E) \leq O(1) \cdot w'(E)$. Here, w is the fractional matching given by Theorem 5.*

We devote the rest of this section to the proof of Lemma 4. As in Section 2.2.1, we start by defining two fractional assignments $w^+ = \sum_{i>L'} w_i$ and $w^- = \sum_{i \leq L'} w_i$. In other words, w^+ captures the fractional weights assigned to the edges in levels $[L'+1, L]$ by the hierarchical partition, whereas w^- captures the fractional weights assigned to the edges in the remaining levels $[0, L']$. The fractional assignment w^+ has support $\cup_{i>L'} E_i$, whereas the fractional assignment w^- has support $\cup_{i \leq L'} E_i = Y$. We have $w = w^+ + w^-$ and $w(E) = w^+(E) + w^-(E)$. If at least half of the value of $w(E)$ is coming from the levels $[0, L']$, then there is nothing to prove. Specifically, suppose that $w^-(E) \geq (1/2) \cdot w(E)$. Then we can set $w' = w^-$ and obtain $w(E) \leq 2 \cdot w^-(E) = 2 \cdot w^-(Y) = 2 \cdot w'(Y) = 2 \cdot w'(X \cup Y)$. This concludes the proof of the lemma. Accordingly, from this point onward, we will assume that at least half of the value of $w(E)$ is coming from the levels $i > L'$. Specifically, we have:

$$w(E) \leq 2 \cdot w^+(E) \tag{13}$$

Given equation 13, we will construct a fractional matching with support X whose size is within a constant factor of $w(E)$.[5] We want to follow the argument applied to ideal skeletons in Section 2.2.1 (see Definition 1). Accordingly, for every level $i > L'$ we now define a fractional assignment $\hat{w}_i$ with support X_i.

$$\begin{aligned} \hat{w}_i(e) &= 1/d_{L'} \text{ for all edges } e \in X_i \\ &= 0 \qquad \text{for all edges } e \in E \setminus X_i. \end{aligned} \tag{14}$$

We next define the fractional assignment $\hat{w}$.

$$\hat{w} = \sum_{i>L'} \hat{w}_i \tag{15}$$

In Section 2.2.1 (see Lemma 2), we observed that $\hat{w}$ is a fractional matching with support X whose size is exactly the same as $w^+(E)$. This observation, along with equation 13, would have sufficed to conclude the proof of Lemma 4. The intuition was that at every level $i > L'$, the degree of a node $v \in V$ in X_i is exactly $(1/\lambda_i)$ times its degree in E_i. On the other hand, the weight of an edge $e \in X_i$ under $\hat{w}_i$ is exactly λ_i times its weight under w_i. This ensured that the weight of node remained unchanged as we transitioned from w_i to $\hat{w}_i$, that is, $W_v(w_i) = W_v(\hat{w}_i)$ for all nodes $v \in V$. Unfortunately, this guarantee will no longer hold for approximate-skeletons. It still seems natural, however, to compare the weights a node receives under these two fractional assignments w_i and $\hat{w}_i$. This depends on the status of the node under consideration, depending on whether the node belongs to the set $B_i \setminus S_i$, $T_i \setminus S_i$ or S_i (see Definition 2). Towards this end, we derive Claims 1, 2, 3. The first claim states that the weight of a non-spurious big node under $\hat{w}_i$ is *very close* to its weight under w_i. The second claim states that the weight of a non-spurious tiny node under $\hat{w}_i$ is *very small* (less than $2\epsilon/L$). The third claim states that the weight of a spurious node under $\hat{w}_i$ is at most one.

CLAIM 1. *For all $i > L'$ and $v \in B_i \setminus S_i$, we have:*

$$(1-\epsilon) \cdot W_v(w_i) \leq W_v(\hat{w}_i) \leq (1+\epsilon) \cdot W_v(w_i).$$

PROOF. Fix any level $i > L'$ and any node $v \in B_i \setminus S_i$. The claim follows from equation 7 and the facts below:

(1) $\lambda_i = d_i/d_{L'}$.

(2) $W_v(\hat{w}_i) = (1/d_{L'}) \cdot \deg_v(X_i)$. See equation 14.

(3) $W_v(w_i) = (1/d_i) \cdot \deg_v(E_i)$. See Section 2.1. □

CLAIM 2. *For all levels $i > L'$ and non-spurious tiny nodes $v \in T_i \setminus S_i$, we have $W_v(\hat{w}_i) \leq 2\epsilon/L$.*

PROOF. Fix any level $i > L'$ and any node $v \in T_i \setminus S_i$. The claim follows from equation 8 and the facts below:

(1) $\lambda_i = d_i/d_{L'}$.

(2) $W_v(\hat{w}_i) = (1/d_{L'}) \cdot \deg_v(X_i)$. See equation 14. □

CLAIM 3. *For all $i > L'$ and $v \in S_i$, we have $W_v(\hat{w}_i) \leq 1$.*

PROOF. Fix any level $i > L'$ and any node $v \in S_i$. The claim follows from equation 9 and the facts below:

(1) $\lambda_i = d_i/d_{L'}$.

(2) $W_v(\hat{w}_i) = (1/d_{L'}) \cdot \deg_v(X_i)$. See equation 14. □

Unfortunately, the fractional assignment $\hat{w}$ need not necessarily be a fractional matching, the main reason being that at a level $i > L'$ the new weight $W_v(\hat{w}_i)$ of a spurious node $v \in S_i$ can be much larger than its original weight $W_v(w_i)$. Specifically, Claim 3 permits that $W_v(\hat{w}_i) = 1$ for such a node $v \in S_i$. If there exists a node $v \in V$ that belongs to S_i at every level $i > L'$, then we might have $W_v(\hat{w}) = \sum_{i>L'} W_v(\hat{w}_i) = \sum_{i>L'} 1 = (L - L') >> 1 \geq W_v(w)$.

To address this concern regarding the weights of the spurious nodes, we switch from $\hat{w}$ to a new fractional assignment w'', which is defined as follows. For every level $i > L'$, we construct a fractional assignment w''_i that sets to zero the weight of every edge in X_i that is incident upon a spurious node $v \in S_i$. For every other edge e, the weight $w''_i(e)$ remains the same as $\hat{w}_i(e)$. Then we set $w'' = \sum_{i>L'} w''_i$.

$$\begin{aligned} w''_i(u,v) &= \hat{w}_i(u,v) \text{ if } (u,v) \in X_i \text{ and } \{u,v\} \cap S_i = \emptyset \\ &= 0 \text{ if } (u,v) \in X_i \text{ and } \{u,v\} \cap S_i \neq \emptyset. \\ &= 0 \text{ else if } (u,v) \in E \setminus X_i \end{aligned} \tag{16}$$

$$w'' = \sum_{i>L'} w''_i \tag{17}$$

[5] Recall that w is the fractional matching given by the hierarchical partition. See Section 2.1.

The above transformation guarantees that $W_v(w_i'') = 0$ for every spurious node $v \in S_i$ at level $i > L'$. Thus, the objection raised above regarding the weights of spurious nodes is no longer valid for the fractional assignment w_i''. We now make three claims on the fractional assignments $\hat{w}$ and w''.

Claim 4 bounds the maximum weight of a node under w''. Its proof appears in Section 2.5.1.

CLAIM 4. *We have $W_v(w'') \leq 1 + 3\epsilon$ for all $v \in V$.*

Claim 5 states that the size of w'' is close to the size of $\hat{w}$. Its proof appears in Section 2.5.2.

CLAIM 5. *We have $w''(E) \geq \hat{w}(E) - 4\epsilon \cdot w^+(E)$.*

Claim 6 states that the size of $\hat{w}$ is within a constant factor of the size of w^+. Its proof appears in Section 2.5.3.

CLAIM 6. *We have $\hat{w}(E) \geq (1/8) \cdot w^+(E)$.*

COROLLARY 4. *We have $w''(E) \geq (1/8 - 4\epsilon) \cdot w^+(E)$.*

PROOF. Follows from Claims 5 and 6. □

To complete the proof of Lemma 4, we scale down the weights of the edges in w'' by a factor of $(1+3\epsilon)$. Specifically, we define a fractional assignment w' such that:

$$w'(e) \quad = \quad \frac{w''(e)}{(1+3\epsilon)} \text{ for all edges } e \in E.$$

Since w'' has support X, the fractional assignment w' also has support X, that is, $w'(e) = 0$ for all edges $e \in E \setminus X$. Claim 4 implies that $W_v(w') = W_v(w'')/(1+3\epsilon) \leq 1$ for all nodes $v \in V$. Thus, w' is fractional matching on support X. Since the edge-weights are scaled down by a factor of $(1+3\epsilon)$, Corollary 4 implies that:

$$w'(E) = \frac{w''(E)}{(1+3\epsilon)} \geq \frac{(1/8 - 4\epsilon)}{(1+3\epsilon)} \cdot w^+(E). \tag{18}$$

Equations 13 and 18 imply that $w(E) \leq O(1) \cdot w'(E)$. This concludes the proof of Lemma 4.

2.5.1 Proof of Claim 4

Throughout the proof, we fix any given node $v \in V$. We will show that $W_v(w'') \leq 1 + 3\epsilon$. We start by making a simple observation:

$$W_v(w_i'') \leq W_v(\hat{w}_i) \text{ for all levels } i > L'. \tag{19}$$

Equation 19 holds since we get the fractional assignment w_i'' from $\hat{w}_i$ by setting some edge-weights to zero and keeping the remaining edge-weights unchanged (see equation 16).

By Definition 2, at every level $i > L'$ the node v is part of exactly one of the three subsets – $T_i \setminus S_i$, $B_i \setminus S_i$ and S_i. Accordingly, we can classify the levels into three types depending on which of these subsets v belongs to at that level. Further, recall that $W_v(w'') = \sum_{i>L'} W_v(w_i'')$. We will separately bound the contributions from each type of levels towards the node-weight $W_v(w'')$.

We first bound the contribution towards $W_v(w'')$ from all the levels $i > L'$ where $v \in T_i \setminus S_i$.

CLAIM 7. *We have:*

$$\sum_{i>L':v\in T_i\setminus S_i} W_v(w_i'') \leq 2\epsilon.$$

PROOF. Claim 2 implies that:

$$\sum_{i>L':v\in T_i\setminus S_i} W_v(\hat{w}_i) \leq \sum_{i>L':v\in T_i\setminus S_i} (2\epsilon/L) \leq 2\epsilon. \tag{20}$$

The claim follows from equations 19 and 20. □

We next bound the contribution towards $W_v(w'')$ from all the levels $i > L'$ where $v \in B_i \setminus S_i$.

CLAIM 8. *We have:*

$$\sum_{i>L':v\in B_i\setminus S_i} W_v(w_i'') \leq 1 + \epsilon.$$

PROOF. Let LHS $= \sum_{i>L':v\in B_i\setminus S_i} W_v(w_i'')$. We have:

$$\begin{aligned}
\text{LHS} \quad &\leq \quad \sum_{i>L':v\in B_i\setminus S_i} W_v(\hat{w}_i) && (21)\\
&\leq \quad \sum_{i>L':v\in B_i\setminus S_i} (1+\epsilon) \cdot W_v(w_i) && (22)\\
&\leq \quad (1+\epsilon) \cdot \sum_{i=0}^{L} W_v(w_i) = (1+\epsilon) \cdot W_v(w)\\
&\leq \quad (1+\epsilon) && (23)
\end{aligned}$$

Equation 21 holds because of equation 19. Equation 22 follows from Claim 1. Finally, equation 23 holds since w is a fractional matching (see Section 2.1). □

Finally, we bound the contribution towards $W_v(w'')$ from all the levels $i > L'$ where $v \in S_i$.

CLAIM 9. *We have:*

$$\sum_{i>L':v\in S_i} W_v(w_i'') = 0.$$

PROOF. Consider any level $i > L'$ where $v \in S_i$. By equation 16, every edge in X_i incident upon v has zero weight under w_i'', and hence $W_v(w_i'') = 0$. The claim follows. □

Adding up the bounds given by Claims 7, 8 and 9, we get:

$$\begin{aligned}
W_v(w'') \quad &= \quad \sum_{i>L'} W_v(w_i'')\\
&= \quad \sum_{i>L':v\in T_i\setminus S_i} W_v(w_i'') + \sum_{i>L':v\in B_i\setminus S_i} W_v(w_i'')\\
&\quad + \sum_{i>L':v\in S_i} W_v(w_i'')\\
&\leq \quad 2\epsilon + (1+\epsilon) + 0 = 1 + 3\epsilon.
\end{aligned}$$

This concludes the proof of Claim 4.

2.5.2 Proof of Claim 5

For any given fractional assignment, the some of the node-weights is two times the sum of the edge-weights (since each edge has two endpoints). Keeping this in mind, instead of relating the sum of the edge-weights under the fractional assignments $w'', \hat{w}$ and w^+ as stated in Claim 5, we will attempt to relate the sum of the node-weights under w'', $\hat{w}$ and w^+.

As we switch from the fractional assignment $\hat{w}_i$ to the fractional assignment w_i'' at some level $i > L'$, all we do is to set to zero the weight of any edge incident upon a spurious node

in S_i. Hence, intuitively, the difference between the sum of the node-weights under $w'' = \sum_{i>L'} w''_i$ and $\hat{w} = \sum_{i>L'} \hat{w}_i$ should be bounded by the sum of the weights of the spurious nodes across all the levels $i > L'$. This is formally stated in the claim below.

CLAIM 10. *We have:*

$$\sum_{v \in V} W_v(w'') \geq \sum_{v \in V} W_v(\hat{w}) - \sum_{i>L'} \sum_{v \in S_i} 2 \cdot W_v(\hat{w}_i).$$

PROOF. The left hand side (LHS) of the inequality is exactly equal to two times the sum of the edge-weights under w''. Similarly, the first sum in the right hand side (RHS) is exactly equal to two times the sum of the edge-weights under $\hat{w}$. Finally, we can also express the second sum in the RHS as the sum of certain edge-weights under $\hat{w}$.

Consider any edge $(x, y) \in E$. We will show that the contribution of this edge towards the LHS is at least its contribution towards the RHS, thereby proving the claim.

Case 1. $(x, y) \notin X_i$ for all $i > L'$. Then the edge (x, y) contributes zero to the left hand side (LHS) and zero to the right hand side (RHS).

Case 2. $(x, y) \in X_i$ at some level $i > L'$, but none of the endpoints of the edge is spurious, that is, $\{x, y\} \cap S_i = \emptyset$. In this case, by equation 16, the edge (x, y) contributes $2 \cdot w''_i(x, y)$ to the LHS, $2 \cdot \hat{w}_i(x, y)$ to the first sum in the RHS, and zero to the second sum in the RHS. Further, we have $w''_i(x, y) = \hat{w}_i(x, y)$. Hence, the edge makes exactly the same contribution towards the LHS and the RHS.

Case 3. $(x, y) \in X_i$ at some level $i > L'$, and at least one endpoint of the edge is spurious, that is, $\{x, y\} \cap S_i \neq \emptyset$. In this case, by equation 16, the edge (x, y) contributes zero to the LHS, $2 \cdot \hat{w}(x, y)$ to the first sum in the RHS, and at least $2 \cdot \hat{w}(x, y)$ to the second sum in the RHS. Hence, the net contribution towards the RHS is at most zero. In other words, the contribution towards the LHS is at least the contribution towards the RHS. □

At every level $i > L'$, we will now bound the sum of the weights of the spurious nodes $v \in S_i$ under $\hat{w}$ by the sum of the node-weights under w_i. We will use the fact that each spurious node gets weight at most one (see Claim 3), which implies that $\sum_{v \in S_i} W_v(\hat{w}_i) \leq |S_i|$. By equation 6, we will upper bound the number of spurious nodes by the number of non-spurious big nodes. Finally, by equation 7, we will infer that each non-spurious big node has sufficiently large degree in E_i, and hence its weight under w_i is also sufficiently large.

CLAIM 11. *For every level $i > L'$, we have:*

$$\sum_{v \in S_i} W_v(\hat{w}_i) \leq (2\delta L/\epsilon) \cdot \sum_{v \in V} W_v(w_i).$$

PROOF. Fix any level $i > L'$. Claim 3 states that $W_v(\hat{w}_i) \leq 1$ for all nodes $v \in S_i$. Hence, we get:

$$\sum_{v \in S_i} W_v(\hat{w}_i) \leq |S_i| \tag{24}$$

Equation 6 implies that $|S_i| \leq \delta \cdot |B_i| \leq \delta \cdot (|B_i \setminus S_i| + |S_i|)$. Rearranging the terms, we get: $|S_i| \leq \frac{\delta}{1-\delta} \cdot |B_i \setminus S_i|$. Since $\delta < 1/2$ (see equation 3), we have:

$$|S_i| \leq 2\delta \cdot |B_i \setminus S_i| \tag{25}$$

From equations 24 and 25, we get:

$$\sum_{v \in S_i} W_v(\hat{w}_i) \leq 2\delta \cdot |B_i \setminus S_i| \tag{26}$$

Now, equation 4 states that $\deg_v(E_i) \geq (\epsilon d_i/L)$ for all nodes $v \in B_i \setminus S_i$. Further, in the hierarchical partition we have $W_v(w_i) = (1/d_i) \cdot \deg_v(E_i)$ for all nodes $v \in V$ (see Section 2.1). Combining these two observations, we get: $W_v(w_i) \geq \epsilon/L$ for all nodes $v \in B_i \setminus S_i$. Summing over all nodes $v \in V$, we get:

$$\sum_{v \in V} W_v(w_i) \geq \sum_{v \in B_i \setminus S_i} W_v(w_i) \geq (\epsilon/L) \cdot |B_i \setminus S_i| \tag{27}$$

The claim follows from equations 26 and 27. □

COROLLARY 5. *We have:*

$$\sum_{i>L'} \sum_{v \in S_i} W_v(\hat{w}_i) \leq (2\delta L/\epsilon) \cdot \sum_{v \in V} W_v(w^+).$$

PROOF. Follows from summing Claim 11 over all levels $i > L'$, and noting that since $w^+ = \sum_{i>L'} w_i$, we have $W_v(w^+) = \sum_{i>L'} W_v(w_i)$ for all nodes $v \in V$. □

From Claim 10 and Corollary 5, we get:

$$\sum_{v \in V} W_v(w'') \geq \sum_{v \in V} W_v(\hat{w}) - (4\delta L/\epsilon) \cdot \sum_{v \in V} W_v(w^+) \tag{28}$$

Since $\delta = \epsilon^2/L$ (see equation 3) and since the sum of the node-weights in a fractional assignment is exactly two times the sum of the edge-weights, Claim 5 follows from equation 28.

2.5.3 Proof of Claim 6

Every edge $(u, v) \in X = \cup_{i>L'} X_i$ has at least one endpoint at a level $i > L'$ (see Definition 2). In other words, every edge in X has at least one endpoint in the set V^* as defined below.

Definition 3. Define $V^* = \{v \in V : \ell(v) > L'\}$ to be the set of all nodes at levels strictly greater than L'.

Thus, under any given fractional assignment, the sum of the node-weights in V^* is within a factor of 2 of the sum of the edge-weights in X. Since both the fractional assignments $\hat{w}$ and w^+ have support X, we get the following claim.

CLAIM 12. *We have:*

$$2 \cdot w^+(E) \geq \sum_{v \in V^*} W_v(w^+) \geq w^+(E).$$

$$2 \cdot \hat{w}(E) \geq \sum_{v \in V^*} W_v(\hat{w}) \geq \hat{w}(E).$$

Since we want to compare the sums of the edge-weights under $\hat{w}$ and w^+, by Claim 12 it suffices to focus on the sum of the node-weights in V^* instead. Accordingly, we first lower bound the sum $\sum_{v \in V^*} W_v(\hat{w})$ in Claim 13. In the proof, we only use the fact that for each level $i > L'$, the weight of a node $v \in B_i \setminus S_i$ remains roughly the same under the fractional assignments $\hat{w}_i$ and w_i (see Claim 1).

CLAIM 13. *We have:*

$$\sum_{v \in V^*} W_v(\hat{w}) \geq (1 - \epsilon) \cdot \sum_{v \in V^*} \sum_{i>L' : v \in B_i \setminus S_i} W_v(w_i).$$

PROOF. Fix any node $v \in V^*$. By Claim 1, we have: $W_v(\hat{w}_i) \geq (1-\epsilon) \cdot W_v(w_i)$ at each level $i > L'$ where $v \in B_i \setminus S_i$. Summing over all such levels, we get:

$$\sum_{i>L':v\in B_i\setminus S_i} W_v(\hat{w}_i) \geq (1-\epsilon) \cdot \sum_{i>L':v\in B_i\setminus S_i} W_v(w_i) \quad (29)$$

Since $\hat{w} = \sum_{i>L'} \hat{w}_i$, we have:

$$W_v(\hat{w}) \geq \sum_{i>L':v\in B_i\setminus S_i} W_v(\hat{w}_i).$$

Hence, equation 29 implies that:

$$W_v(\hat{w}) \geq (1-\epsilon) \cdot \sum_{i>L':v\in B_i\setminus S_i} W_v(w_i).$$

We now sum the above inequality over all nodes $v \in V^*$. □

It remains to lower bound the right hand side (RHS) in Claim 13 by $\sum_{v\in V^*} W_v(w^+)$. Say that a level $i > L'$ is of Type I, II or III for a node $v \in V^*$ if v belongs to $B_i \setminus S_i$, S_i or $T_i \setminus S_i$ respectively. By Definition 2, for every node $v \in V^*$, the set of levels $i > L'$ is partitioned into these three types. The sum in the RHS of Claim 13 gives the contribution of the type I levels towards $\sum_{v\in V^*} W_v(w^+)$. In Claims 14 and 15, we respectively show that the type II and type III levels make negligible contributions towards the sum $\sum_{v\in V^*} W_v(w^+)$. Note that the sum of these contributions from the type I, type II and type III levels *exactly* equals $\sum_{v\in V^*} W_v(w^+)$. Specifically, we have:

$$\sum_{v\in V^*} \sum_{i>L':v\in B_i\setminus S_i} W_v(w_i) + \sum_{v\in V^*} \sum_{i>L':v\in S_i} W_v(w_i) + \sum_{v\in V^*} \sum_{i>L':v\in T_i\setminus S_i} W_v(w_i) = \sum_{v\in V^*} W_v(w^+) \quad (30)$$

Hence, equation 30, Claim 14 and Claim 15 lead to the following lower bound on the right hand side of Claim 13.

COROLLARY 6. *We have:*

$$\sum_{v\in V^*} \sum_{i>L':v\in B_i\setminus S_i} W_v(w_i) \geq (1-40\epsilon) \cdot \sum_{v\in V^*} W_v(w^+).$$

From Claim 13, Corollary 6 and equation 3, we get:

$$\sum_{v\in V^*} W_v(\hat{w}) \geq (1/4) \cdot \sum_{v\in V^*} W_v(w^+) \quad (31)$$

Finally, from Claim 12 and equation 31, we infer that:

$$\hat{w}(E) \geq (1/2) \cdot \sum_{v\in V^*} W_v(\hat{w}) \geq (1/8) \cdot \sum_{v\in V^*} W_v(w^+) \geq (1/8) \cdot w^+(E)$$

This concludes the proof of Claim 6. Accordingly, we devote the rest of this section to the proofs of Claims 14 and 15.

CLAIM 14. *We have:*

$$\sum_{v\in V^*} \sum_{i>L':v\in S_i} W_v(w_i) \leq 8\epsilon \cdot \sum_{v\in V^*} W_v(w^+).$$

PROOF. The proof of this claim is very similar to the proof of Claim 11 and Corollary 5. Going through that proof, one can verify the following upper bound on the number of spurious nodes across all levels $i > L'$.

$$\sum_{i>L'} |S_i| \leq (2\delta L/\epsilon) \cdot \sum_{v\in V} W_v(w^+) \quad (32)$$

Since each w_i is a fractional matching (see Section 2.1), we have $W_v(w_i) \leq 1$ for all nodes $v \in V$ and all levels $i > L'$. Hence, we get:

$$\sum_{v\in V^*} \sum_{i>L':v\in S_i} W_v(w_i) \leq \sum_{i>L'} |S_i| \quad (33)$$

From equations 32 and 33, we infer that:

$$\sum_{v\in V^*} \sum_{i>L':v\in S_i} W_v(w_i) \leq (2\delta L/\epsilon) \cdot \sum_{v\in V} W_v(w^+) \quad (34)$$

Since the sum of the node-weights under any fractional assignment is equal to twice the sum of the edge-weights, Claim 12 implies that:

$$\sum_{v\in V} W_v(w^+) = 2 \cdot w^+(E) \leq 4 \cdot \sum_{v\in V^*} W_v(w^+) \quad (35)$$

Claim 14 follows from equations 3, 34 and 35. □

CLAIM 15. *We have:*

$$\sum_{v\in V^*} \sum_{i>L':v\in T_i\setminus S_i} W_v(w_i) \leq 32\epsilon \cdot \sum_{v\in V^*} W_v(w^+).$$

PROOF. Fix any node $v \in V^*$. By equation 5, we have $\deg_v(E_i) \leq (2\epsilon d_i/L)$ at each level $i > L'$ where $v \in T_i \setminus S_i$. Further, the fractional matching w_i assigns a weight $1/d_i$ to every edge in its support E_i (see Section 2.1). Combining these two observations, we get: $W_v(w_i) = (1/d_i) \cdot \deg_v(E_i) \leq 2\epsilon/L$ at each level $i > L'$ where $v \in T_i \setminus S_i$. Summing over all such levels, we get:

$$\sum_{i>L':v\in T_i\setminus S_i} W_v(w_i) \leq 2\epsilon \quad (36)$$

If we sum equation 36 over all $v \in V^*$, then we get:

$$\sum_{v\in V^*} \sum_{i>L':v\in T_i\setminus S_i} W_v(w_i) \leq 2\epsilon \cdot |V^*| \quad (37)$$

A node $v \in V^*$ has level $\ell(v) > L'$. Hence, all the edges incident upon this node also have level at least $L' + 1$. This implies that such a node v receives zero weight from the fractional assignment $w^- = \sum_{i\leq L'} w_i$, for any edge in the support of w^- is at level at most L'. Thus, we have: $W_v(w) = W_v(w^+) + W_v(w^-) = W_v(w^+)$ for such a node v. Now, applying Theorem 5, we get:

$$1/(1+\epsilon)^2 \leq W_v(w^+) \text{ for all nodes } v \in V^*. \quad (38)$$

Summing equation 38 over all nodes $v \in V^*$ and multiplying both sides by $(1+\epsilon)^2$, we get:

$$|V^*| \leq (1+\epsilon)^2 \cdot \sum_{v\in V^*} W_v(w^+) \quad (39)$$

Since $(1+\epsilon)^2 \leq 4$ and $V^* \subseteq V$, equations 37, 39 imply that:

$$\sum_{v\in V^*} \sum_{i>L':v\in T_i\setminus S_i} W_v(w_i) \leq 8\epsilon \cdot \sum_{v\in V} W_v(w^+) \quad (40)$$

The claim follows from equations 35 and 40. □

3. BIPARTITE GRAPHS

Ideally, we would like to present a dynamic algorithm on bipartite graphs that proves Theorem 2. Due to space constraints, however, we will only prove a weaker result stated in Theorem 8 and defer the complete proof of Theorem 2

to the full version. Throughout this section, we will use the notations and concepts introduced in Section 1.1.

THEOREM 8. *There is a randomized dynamic algorithm that maintains a* 1.976 *approximation to the maximum matching in a bipartite graph in* $O(\sqrt{n}\log n)$ *expected update time.*

In Section 3.1, we present a result from [5] which shows how to maintain a $(2+\epsilon)$-approximation to the maximum matching in bipartite graphs in $O(\sqrt{n}/\epsilon^2)$ update time. In Section 3.2, we build upon this result and prove Theorem 8. In Section 3.3, we allude to the extensions that lead us to the proof of Theorem 2 in the full version of the paper.

3.1 $(2+\epsilon)$-approximation in $O(\sqrt{n}/\epsilon^2)$ update time

The first step is to define the concept of a *kernel*. Setting $\epsilon = 0, d = 1$ in Definition 4, we note that the kernel edges in a $(0,1)$-kernel forms a maximal matching – a matching where every unmatched edge has at least one matched endpoint. For general d, we note that the kernel edges in a $(0,d)$-kernel forms a maximal d-matching – which is a maximal subset of edges where each node has degree at most d. In Lemma 5 and Corollary 7, we show that the kernel edges in any (ϵ,d)-kernel preserves the size of the maximum matching within a factor of $2/(1-\epsilon)$. Since d is the maximum degree of a node in an (ϵ,d)-kernel, a $(1+\epsilon)$-approximation to the maximum matching within a kernel can be maintained in $O(d/\epsilon^2)$ update time using Theorem 4. Lemma 6 shows that the set of kernel edges themselves can be maintained in $O(n/(\epsilon d))$ update time. Setting $d = \sqrt{n}$ and combining all these observations, we get our main result in Corollary 8.

Definition 4. Fix any $\epsilon \in (0,1)$, $d \in [1,n]$. Consider any subset of nodes $T \subset V$ in the graph $G = (V,E)$, and any subset of edges $H \subseteq E$. The pair (T,H) is called an (ϵ,d)-kernel of G iff: (1) $\deg_v(H) \leq d$ for all nodes $v \in V$, (2) $\deg_v(H) \geq (1-\epsilon)d$ for all nodes $v \in T$, and (3) every edge $(u,v) \in E$ with both endpoints $u,v \in V \setminus T$ is part of the subset H. We define the set of nodes $T^c = V \setminus T$, and say that the nodes in T (resp. T^c) are "tight" (resp. "non-tight"). The edges in H are called "kernel edges".

LEMMA 5. *Consider any integral matching* $M \subseteq E$ *and let* (T,H) *be any* (ϵ,d)*-kernel of* $G = (V,E)$ *as per Definition 4. Then there is a fractional matching* w'' *in* G *with support* H *such that* $\sum_{v\in V} W_v(w'') \geq (1-\epsilon)\cdot|M|$.

The proof of Lemma 5 appears in Section 3.1.1.

COROLLARY 7. *Consider any* (ϵ,d)*-kernel as per Definition 4. We have* $Opt(H) \geq (1/2)\cdot(1-\epsilon)\cdot Opt(E)$.

PROOF. Let $M \subseteq E$ be a maximum cardinality matching in $G = (V,E)$. Let w'' be a fractional matching with support H as per Lemma 5. Since in a bipartite graph the size of the maximum cardinality matching is the same as the size of the maximum fractional matching (see Theorem 3), we get: $\text{Opt}(H) = \text{Opt}_f(H) \geq w''(H) = (1/2)\cdot\sum_{v\in V} W_v(w'') \geq (1/2)\cdot(1-\epsilon)\cdot|M| = (1/2)\cdot(1-\epsilon)\cdot\text{Opt}(E)$. □

LEMMA 6. *In the dynamic setting, an* (ϵ,d)*-kernel can be maintained in* $O(n/(\epsilon d))$ *amortized update time.*

PROOF. (Sketch) When an edge (u,v) is inserted into the graph, we simply check if both its endpoints are non-tight. If yes, then we insert (u,v) into H. Next, for each endpoint $x \in \{u,v\}$, we check if $\deg_x(H)$ has now become equal to d due to this edge insertion. If yes, then we delete the node x from T^c and insert it into T. All these operations can be performed in constant time.

Now, consider the deletion of an edge (u,v). If both u,v are non-tight, then we have nothing else to do. Otherwise, for each tight endpoint $x \in \{u,v\}$, we check if $\deg_x(H)$ has now fallen below the threshold $(1-\epsilon)d$ due to this edge deletion. If yes, then we might need to change the status of the node x from tight to non-tight. Accordingly, we scan through all the edges in E that are incident upon x, and try to insert as many of them into H as possible. This step takes $\Theta(n)$ time in the worst case since the degree of the node x can be $\Theta(n)$. However, the algorithm ensures that this event occurs only after ϵd edges incident upon x are deleted from E. This is true since we have a *slack* of ϵd between the largest and smallest possible degrees of a tight node. Thus, we get an amortized update time of $O(n/(\epsilon d))$. □

COROLLARY 8. *In a bipartite graph, one can maintain a* $(2+6\epsilon)$*-approximation to the size of the maximum matching in* $O(\sqrt{n}/\epsilon^2)$ *amortized update time.*

PROOF. (Sketch) We set $d = \sqrt{n}$ and maintain an (ϵ,d)-kernel (T,H) as per Lemma 6. This takes $O(\sqrt{n}/\epsilon)$ update time. Next, we note that the maximum degree of a node in H is $d = \sqrt{n}$ (see Definition 4). Accordingly, we can apply Theorem 4 to maintain a $(1+\epsilon)$-approximate maximum matching $M_H \subseteq H$ in $O(\sqrt{n}/\epsilon^2)$ update time. Hence, by Corollary 7, this matching M_H is a $2(1+\epsilon)/(1-\epsilon) \leq (2+6\epsilon)$-approximation to the maximum matching in $G = (V,E)$. □

3.1.1 Proof of Lemma 5

First, define a fractional assignment w as follows. For every edge $(u,v) \in H$ incident on a tight node, we set $w(e) = 1/d$, and for every other edge $(u,v) \in E$, set $w(u,v) = 0$. Since each node $v \in V$ has $\deg_v(H) \leq d$, it is easy to check that $W_v(w) \leq 1$ for all nodes $v \in V$. In other words, w forms a fractional matching in G.

Next, we define another fractional assignment w'. First, for every node $v \in T^c$, we define a "capacity" $b(v) = 1 - W_v(w) \in [0,1]$. Next, for every edge $(u,v) \in H \cap M$ whose both endpoints are non-tight, set $w'(u,v) = \min(b(u),b(v))$. For every other edge $(u,v) \in E$, set $w'(u,v) = 0$.

We finally define $w'' = w + w'$. Clearly, the fractional assignment w'' has support H, since for every edge $(u,v) \in E \setminus H$, we have $w(u,v) = w'(u,v) = 0$. Hence, the lemma follows from Claims 16 and 17.

CLAIM 16. *We have* $W_v(w'') \leq 1$ *for all nodes* $v \in V$*, that is,* w'' *is a fractional matching in* G.

PROOF. If a node v is tight, that is, $v \in T$, then we have $W_v(w'') = W_v(w) + W_v(w') = W_v(w) \leq 1$. Hence, for the rest of the proof, consider any node from the remaining subset $v \in T^c = V \setminus T$. There are two cases to consider here.

Case 1. If v is not matched in M, then we have $W_v(w') = 0$, and hence $W_v(w'') = W_v(w) + W_v(w') = W_v(w) \leq 1$.

Case 2. If v is matched in M, then let u be its mate, i.e., $(u,v) \in M$. Here, we have $W_v(w') = w'(u,v) = \min(1 - W_u(w), 1 - W_v(w)) \leq 1 - W_v(w)$. This implies that $W_v(w'') = W_v(w) + W_v(w') \leq 1$. This concludes the proof. □

CLAIM 17. *We have* $\sum_{v \in V} W_v(w'') \geq (1-\epsilon) \cdot |M|$.

PROOF. Throughout the proof, fix any edge $(u,v) \in M$. We will show that $W_u(w'') + W_v(w'') \geq (1-\epsilon)$. The claim will then follow if we sum over all the edges in M.

Case 1. The edge (u,v) has at least one tight endpoint. Let $u \in T$. In this case, we have $W_u(w'')+W_v(w'') \geq W_u(w'') = W_u(w) + W_u(w') \geq W_u(w) = (1/d) \cdot \deg_u(H) \geq (1-\epsilon)$.

Case 2. Both the endpoints of (u,v) are non-tight. Without any loss of generality, let $W_u(w) \geq W_v(w)$. In this case, we have $W_u(w'') + W_v(w'') \geq W_u(w'') = W_u(w) + W_u(w') = W_u(w) + w'(u,v) = W_u(w) + \min(1-W_u(w), 1-W_v(w)) = W_u(w) + (1-W_u(w)) = 1$. This concludes the proof. □

3.2 Better than 2-approximation

The approximation guarantee derived in Section 3.1 follows from Claim 17. Looking back at the proof of this claim, we observe that we actually proved a stronger statement: Any matching $M \subseteq E$ satisfies the property that $W_u(w'')+W_v(w'') \geq (1-\epsilon)$ for all matched edges $(u,v) \in M$, where w'' is a fractional matching with support H that depends on M. In the right hand side of this inequality, if we replace the term $(1-\epsilon)$ by anything larger than 1, then we will get a better than 2 approximation (see the proof of Corollary 7). The reason it was not possible to do so in Section 3.1 is as follows. Consider a matched edge $(u,v) \in M$ with $u \in T$ and $v \in T^c$. Since u is tight, we have $1-\epsilon \leq W_u(w) = W_v(w'') \leq 1$. Suppose that $W_u(w'') = 1-\epsilon$. In contrast, it might well be the case that $W_v(w)$ is very close to being zero (which will happen if $\deg_v(H)$ is very small). Let $W_v(w) \leq \epsilon$. Also note that $W_v(w'') = W_v(w) + W_v(w') = W_v(w) \leq \epsilon$ since no edge that gets nonzero weight under w' can be incident on v (for v is already incident upon an edge in M whose other endpoint is tight). Hence, in this instance we will have $W_u(w'') + W_v(w'') \leq (1-\epsilon) + \epsilon = 1$, where $(u,v) \in M$ is a matched edge with one tight and one non-tight endpoint.

The above discussion suggests that we ought to "throw in" some additional edges into our kernel – edges whose one endpoint is tight and the other endpoint is non-tight with a very small degree in H. Accordingly, we introduce the notion of *residual edges* in Section 3.2.1. We show that the union of the kernel edges and the residual edges preserves the size of the maximum matching within a factor of strictly less than 2. Throughout the rest of this section, we set the values of two parameters δ, ϵ as follows.

$$\delta = 1/20, \epsilon = 1/2000 \tag{41}$$

3.2.1 The main framework: Residual edges

We maintain an (ϵ, d)-skeleton (T, H) as in Section 3.1. We further partition the set of non-tight nodes $T^c = V \setminus T$ into two subsets: $B \subseteq T^c$ and $S = T^c \setminus B$. The set of nodes in B (resp. S) are called "big" (resp. "small"). They satisfy the following degree-thresholds: (1) $\deg_v(H) \leq 2\delta d/(1-\delta)$ for all small nodes $v \in S$, and (2) $\deg_v(H) \geq (2\delta-\epsilon)d/(1-\delta)$ for all big nodes $v \in B$. Let $E^r = \{(u,v) \in E : u \in T, v \in S\}$ be the subset of edges joining the tight and the small nodes. We maintain a *maximal* subset of edges $M^r \subseteq E^r$ subject to the following constraints: (1) $\deg_v(M^r) \leq 1$ for all tight nodes $v \in T$ and (2) $\deg_v(M^r) \leq 2$ for all small nodes $v \in S$. The edges in M^r are called the "residual edges". The degree of a node in M^r is called its "residual degree". The corollary below follows from the maximality of the set $M^r \subseteq E^r$.

COROLLARY 9. *If an edge* $(u,v) \in E^r$ *with* $u \in T$, $v \in S$ *is not in* M^r, *then either* $deg_v(M^r) = 1$ *or* $deg_u(M^r) = 2$.

LEMMA 7. *We can maintain the set of kernel edges* H *and the residual edges* M^r *in* $O(n \log n/(\epsilon d))$ *update time.*

PROOF. (Sketch) We maintain an (ϵ, d)-kernel as per the proof of Lemma 6. We maintain the node-sets $B, S \subseteq T^c$ and the edge-set E^r in the same lazy manner: A node changes its status only after $\Omega(\epsilon d)$ edges incident upon it are either inserted into or deleted from G (since δ is a constant), and when that happens we might need to make $\Theta(n)$ edge insertions/deletions in E^r. This gives the same amortized update time of $O(n/(\epsilon d))$ for maintaining the edge-set E^r.

In order to maintain the set of residual edges $M^r \subseteq E^r$, we use a simple modification of the dynamic algorithm of Baswana et al. [2] that maintains a maximal matching in $O(\log n)$ update time. This holds since M^r is a *maximal b-matching* in E^r where each small node can have at most two matched edges incident upon it, and each tight node can have at most one matched edge incident upon it. □

LEMMA 8. *Fix any* (ϵ, d) *kernel* (T, H) *as in Section 3.1, any set of residual edges* M^r *as in Section 3.2.1, and any matching* $M \subseteq E$. *Then we have a fractional matching* w'' *on support* $H \cup M^r$ *such that* $\sum_{v \in V} W_v(w'') \geq (1+\delta/4) \cdot |M|$.

Roadmap for the rest of this section. The statement of Lemma 8 above is similar to that of Lemma 5 in Section 3.1. Hence, using a similar argument as in Corollary 7, we infer that the set of edges $M^r \cup H$ preserves the size of the maximum matching within a factor of $2/(1+\delta/4)$. Since $\deg_v(M^r \cup H) = \deg_v(H) + \deg_v(M^r) \leq d+2$ for all nodes $v \in V$ (see Definition 4), we can maintain a $(1+\epsilon)$-approximate maximum matching in $H \cup M^r$ using Theorem 4 in $O(d/\epsilon^2)$ update time. This matching will give a $2(1+\epsilon)/(1+\delta/4) = 1.976$-approximation to the size of maximum matching in G (see equation 41). The total update time is $O(d/\epsilon^2 + n \log n/(\epsilon d))$, which becomes $O(\sqrt{n} \log n)$ if we set $d = \sqrt{n}$ and plug in the value of ϵ. This concludes the proof of Theorem 8.

It remains to prove Lemma 8, which is done in Section 3.2.2.

3.2.2 Proof of Lemma 8

We will define four fractional assignments w, w^r, w', w''. It might be instructive to contrast the definitions of the fractional assignments w, w' and w'' here with Section 3.1.1.

The fractional assignment w: Set $w(e) = (1-\delta)/d$ for every edge $e \in H$ incident upon a tight node. Set $w(e) = 0$ for every other edge $e \in E$. Hence, we have $W_v(w) = ((1-\delta)/d) \cdot \deg_v(H)$ for all nodes $v \in V$. Accordingly, recalling the bounds on $\deg_v(H)$ for various types of nodes (see Definition 4, Section 3.2.1), we get:

$$W_v(w) \leq (1-\delta) \text{ for all nodes } v \in V. \tag{42}$$
$$W_v(w) \leq 2\delta \text{ for all small nodes } v \in S. \tag{43}$$
$$W_v(w) \geq (1-\delta)(1-\epsilon) \text{ for all tight nodes } v \in T. \tag{44}$$
$$W_v(w) \geq 2\delta - \epsilon \text{ for all big nodes } v \in B. \tag{45}$$

The fractional assignment w^r*:* Set $w^r(e) = \delta$ for every residual edge $e \in M^r$. Set $w^r(e) = 0$ for every other edge $e \in E$.

The fractional assignment w'*:* For every node $v \in T^c$, we define a "capacity" $b(v)$ as follows. If $v \in B \subseteq T^c$, then

$b(v) = 1 - W_v(w)$. Else if $v \in S = T^c \setminus B$, then $b(v) = 1 - 2\delta - W_v(w)$. Hence, equations 42, 43 imply that:

$$b(v) \geq \delta \text{ for all big nodes } v \in B. \quad (46)$$

$$b(v) \geq 1 - 4\delta \text{ for all small nodes } v \in S. \quad (47)$$

For every edge $(u, v) \in M$ with $u, v \in T^c = V \setminus T$, we set $w'(u, v) = \min(b(u), b(v))$. For every other edge $e \in E$, we set $w'(e) = 0$. By Definition 4, every edge whose both endpoints are non-tight is a kernel edge. Hence, an edge gets nonzero weight under w' only if it is part of the kernel.

The fractional assignment w'': Define $w'' = w + w^r + w'$.

Roadmap for the rest of the proof. Each of the fractional assignments w, w^r, w' assigns zero weight to every edge $e \in E \setminus (H \cup M^r)$. Hence, the fractional assignment $w'' = w + w^r + w'$ has support $H \cup M^r$. In Claim 18, we show that w'' is a fractional matching in G. Moving on, in Definition 5, we partition the set of matched edges in M into two parts. The subset $M_1 \subseteq M$ consists of those matched edges that have one tight and one small endpoints, and the subset $M_2 = M \setminus M_1$ consists of the remaining edges. In Claims 19 and 20, we relate the node-weights under w, w', w^r with the sizes of the matchings M_1 and M_2. Adding up the bounds from Claims 19 and 20, Corollary 10 lower bounds the sum of the node-weights under w'' by the size of the matching M. Finally, Lemma 8 follows from Claim 18 and Corollary 10.

Claim 18. *We have $W_v(w'') \leq 1$ for all nodes $v \in V$.*

Proof. Consider any node $v \in V$. By equation 42, we have: $W_v(w) \leq 1 - \delta$. Also note that $W_v(w^r) \leq \delta$ for all tight nodes $v \in T$, $W_v(w^r) \leq 2\delta$ for all small nodes $v \in S$, and $W_v(w^r) = 0$ for all big nodes $v \in B$. This holds since the degree (among the edges in M^r) of a tight, small and big node is at most one, two and zero respectively. Next, note that for all nodes $v \in T^c$, we have $W_v(w') \leq b(v)$. This holds since there is at most one edge in $M \cap H$ incident upon v (since M is a matching). So at most one edge incident upon v gets a nonzero weight under w', and the weight of this edge is at most $b(v)$. Finally, note that every edge with nonzero weight under w' has both the endpoints in T^c. Hence, we have $W_v(w') = 0$ for all tight nodes $v \in T$. To complete the proof, we now consider three possible cases.

Case 1. $v \in T$. Here, $W_v(w'') = W_v(w) + W_v(w^r) + W_v(w') = W_v(w) + W_v(w^r) \leq W_v(w) + \delta \leq (1 - \delta) + \delta = 1$.

Case 2. $v \in S$. Here, $W_v(w'') = W_v(w) + W_v(w^r) + W_v(w') \leq W_v(w) + 2\delta + b(v) = W_v(w) + 2\delta + (1 - 2\delta - W_v(w)) = 1$.

Case 3. $v \in B$. Here, $W_v(w'') = W_v(w) + W_v(w^r) + W_v(w') = W_v(w) + W_v(w') \leq W_v(w) + b(v) = W_v(w) + (1 - W_v(w)) = 1$. □

Definition 5. Partition the set of edges in M into two parts: $M_1 = \{(u, v) \in M : u \in T, v \in S\}$ and $M_2 = M \setminus M_1$.

Claim 19. *Recall Definition 5. We have:*

$$\sum_{(u,v) \in M_2} W_u(w + w') + W_v(w + w') \geq (1 + \delta/4) \cdot |M_2|.$$

Proof. Fix any edge $(u, v) \in M_2$, and let LHS $= W_u(w + w') + W_v(w + w')$. We will show that LHS $\geq (1 + \delta/4)$. The claim will then follow if we sum over all such edges M_2. We recall equation 41 and consider four possible cases.

Case 1. Both endpoints are tight, that is, $u, v \in T$. Here, from equation 44 we get: LHS $\geq 2 \cdot (1 - \delta - \epsilon + \delta\epsilon) \geq (1 + \delta/4)$.

Case 2. One endpoint is tight, and one endpoint is big, that is, $u \in T$, $v \in B$. Here, from equations 44, 45 we get: LHS $\geq (1 - \delta - \epsilon + \delta\epsilon) + (2\delta - \epsilon) \geq (1 + \delta - 2\epsilon) \geq 1 + \delta/4$.

Case 3. Both endpoints are non-tight, that is, $u, v \in B \cup S$. Without any loss of generality, let $b(u) \geq b(v)$. Note that $(u, v) \in H$ since both $u, v \in T^c$, and hence $(u, v) \cap M_2 \cap H$. Thus, we have $W_u(w') = W_v(w') = w'(u, v) = b(v)$ since at most one matched edge can be incident upon a node. Now, either $v \in B$ or $v \in S$. In the former case, from equation 47 we get: LHS $\geq W_u(w') + W_v(w') = 2 \cdot b(v) \geq 2(1 - 4\delta) \geq (1 + \delta/4)$. In the latter case, from equation 46 we get: LHS $\geq (W_v(w) + W_v(w')) + W_u(w') = (b(v) + W_v(w)) + b(v) = 1 + b(v) \geq 1 + \delta \geq 1 + \delta/4$. □

Claim 20. *Recall Definition 5. We have:*

$$\sum_{(u,v) \in M_1} (W_u(w) + W_v(w)) + \sum_{v \in V} W_v(w^r) \geq (1 + \delta/4) \cdot |M_1|.$$

Proof. Every edge $(u, v) \in M_1$ has one endpoint $u \in T$. Thus, Applying equation 44 we get: $W_u(w) + W_v(w) \geq W_u(w) \geq 1 - \delta - \epsilon$. Summing over all such edges, we get:

$$\sum_{(u,v) \in M_1} W_u(w) + W_v(w) \geq (1 - \delta - \epsilon) \cdot |M_1| \quad (48)$$

Recall that $\deg_u(M^r) \leq 1$ for every tight node $u \in T$. Accordingly, we classify each tight node as being either "full" (in which case $\deg_u(M^r) = 1$) or "deficient" (in which case $\deg_u(M^r) = 0$). Further, recall that each edge $(u, v) \in M_1$ has one tight and one small endpoints. We say that an edge $(x, y) \in M_1$ is *deficient* if the tight endpoint of the edge is deficient. Now, consider any deficient edge $(x, y) \in M_1$ where $x \in T$ and $y \in S$. Since $\deg_x(M^r) = 0$, it follows that $(x, y) \in E^r \setminus M^r$. From the maximality of M^r, we infer that $\deg_y(M^r) = 2$. Accordingly, there must be two edges $(x', y), (x'', y) \in M^r$ with $x', x'' \in T$. It follows that both the nodes x', x'' are full. We say that the tight nodes x', x'' are *conjugates* of the deficient edge $(x, y) \in M_1$. In other words, we have shown that every deficient edge in M_1 has two conjugate tight nodes. Further, the same tight node x' cannot be a conjugate of two different deficient edges in M_1, for otherwise each of those deficient edges will contribute one towards $\deg_{x'}(M^r)$, and we will get $\deg_{x'}(M^r) \geq 2$, which is a contradiction. Thus, a simple counting argument implies that the number of conjugate tight nodes is exactly twice the number of deficient matched edges in M_1. Let $D(M_1), F, C$ respectively denote the set of deficient matched edges in M_1, the set of full tight nodes and the set of conjugate tight nodes. Thus, we get:

$$T \supseteq F \supseteq C, \quad D(M_1) \subseteq M_1, \text{ and } |C| = 2 \cdot |D(M_1)| \quad (49)$$

Now, either $|D(M_1)| \leq (1/3) \cdot |M_1|$ or $|D(M_1)| > (1/3) \cdot |M_1|$. In the former case, at least a $(2/3)^{rd}$ fraction of the edges in M_1 are not deficient, and each such edge has one tight endpoint that is full. Thus, we get $|F| \geq (2/3) \cdot |M_1|$. In the latter case, from equation 49 we get $|F| \geq |C| = 2 \cdot |D(M_1)| > (2/3) \cdot |M_1|$. Thus, in either case we have $|F| \geq (2/3) \cdot |M_1|$. Since each node $v \in F \subseteq T$ has $\deg_v(M^r) = 1$,

and since each edge $e \in M^r$ has weight $w^r(e) = \delta$, it follows that $W_v(w^r) = \delta$ for all nodes $v \in F \subseteq T$. Hence, we get $\sum_{v \in T} W_v(w^r) \geq \delta \cdot |F| \geq (2\delta/3) \cdot |M_1|$. Next, we note that each edge in M^r contributes the same amount δ towards the weights of both its endpoints – one tight and the other small. Thus, we have:

$$\sum_{v \in S} W_v(w^r) = \sum_{v \in T} W_v(w^r) \geq (2\delta/3) \cdot |M_1|.$$

Since $B \cup S \subseteq V$ and $B \cap S = \emptyset$, we get:

$$\sum_{v \in V} W_v(w^r) \geq \sum_{v \in B \cup S} W_v(w^r) \geq (4\delta/3) \cdot |M_1|.$$

This inequality, along with equation 48, gives us:

$$\sum_{(u,v) \in M_1} (W_u(w) + W_v(w)) + \sum_{v \in V} W_v(w^r)$$
$$\geq (1 - \delta - \epsilon) \cdot |M_1| + (4\delta/3) \cdot |M_1| = (1 + \delta/3 - \epsilon) \cdot |M_1|$$
$$\geq (1 + \delta/4) \cdot |M_1|.$$

The last inequality follows from equation 41. □

COROLLARY 10. *We have:*

$$\sum_{v \in V} W_v(w'') \geq (1 + \delta/4) \cdot |M|.$$

PROOF. Since $|M| = |M_1| + |M_2|$, the corollary follows from adding the inequalities stated in Claims 19 and 20, and noting that no node-weight under w'' is counted twice in the left hand side. □

3.3 Extensions

We gave a randomized algorithm for maximum bipartite matching that maintains a better than 2 approximation in $O(\sqrt{n} \log n)$ update time. In the full version of the paper, we derandomize this scheme using the following idea. Instead of applying the randomized maximal matching algorithm from [2] for maintaining the set of residual edges M^r, we maintain a *residual fractional matching* using the deterministic algorithm from [5] (see Theorem 5). To carry out the approximation guarantee analysis, we have to change the proof of Lemma 8 (specifically, the proof of Claim 20).

To get arbitrarily small polynomial update time, we maintain a partition of the node-set into multiple levels. The top level consists of all the tight nodes (see Definition 4). We next consider the subgraph induced by the non-tight nodes. Each edge in this subgraph is a kernel edge (see Definition 4). Intuitively, we split the node-set of this subgraph again into two parts by defining a kernel within this subgraph. The tight nodes we get in this iteration forms the next level in our partition of V. We keep doing this for K levels, where K is a sufficiently large integer. We show that (a) this structure can be maintained in $O(n^{2/K})$ update time, and (b) by combining the fractional matchings from all these levels, we can get an α_K approximate maximum fractional matching in G, where $1 \leq \alpha_K < 2$. By Theorem 3, this gives α_K-approximation to the size of the maximum integral matching in G. See the full version of the paper for the details.

4. OPEN PROBLEMS

In this paper, we presented two deterministic dynamic algorithms for maximum matching. Our first algorithm maintains a $(2+\epsilon)$-approximate maximum matching in a general graph in $O(\text{poly}(\log n, 1/\epsilon))$ update time. The exponent hidden in the polylogorithmic factor of the update time, however, is rather huge. It will be interesting to bring down the update time of this algorithm to $O(\log n/\epsilon^2)$ without increasing the approximation factor. This will match the update time in [5] for maintaining a *fractional* matching.

We also showed how to maintain a better than 2 approximation to the size of the maximum matching on bipartite graphs in $O(n^{2/K})$ update time, for every sufficiently large integer K. The approximation ratio approaches 2 as K becomes large. The main open problem here is to design a dynamic algorithm that gives better than 2 approximation in polylogarithmic update time. This remains open even on bipartite graphs and even if one allows randomization.

5. REFERENCES

[1] A. Abboud and V. Vassilevska Williams. Popular conjectures imply strong lower bounds for dynamic problems. In *FOCS*, 2014.

[2] S. Baswana, M. Gupta, and S. Sen. Fully dynamic maximal matching in $\mathcal{O}(\log n)$ update time. In *FOCS*, 2011.

[3] A. Bernstein and C. Stein. Fully dynamic matching in bipartite graphs. In *ICALP*, 2015.

[4] A. Bernstein and C. Stein. Faster fully dynamic matchings with small approximation ratios. In *SODA*, 2016.

[5] S. Bhattacharya, M. Henzinger, and G. F. Italiano. Deterministic fully dynamic data structures for vertex cover and matching. *CoRR*, abs/1412.1318, 2014. Announced at SODA 2015.

[6] M. Crouch and D. S. Stubbs. Improved streaming algorithms for weighted matching, via unweighted matching. In *APPROX/RANDOM*, 2014.

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

[8] M. Henzinger, S. Krinninger, D. Nanongkai, and T. Saranurak. Unifying and strengthening hardness for dynamic problems via the online matrix-vector multiplication conjecture. In *STOC*, 2015.

[9] A. Israeli and Y. Shiloach. An improved parallel algorithm for maximal matching. *Information Processing Letters*, 22:57–60, 1986.

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

[11] T. Kopelowitz, S. Pettie, and E. Porat. Higher lower bounds from the 3SUM conjecture. In *SODA*, 2016.

[12] O. Neiman and S. Solomon. Simple deterministic algorithms for fully dynamic maximal matching. In *STOC*, 2013.

[13] K. Onak and R. Rubinfeld. Maintaining a large matching and a small vertex cover. In *STOC*, 2010.

[14] M. Patrascu. Towards polynomial lower bounds for dynamic problems. In *STOC*, 2010.

[15] D. Peleg and S. Solomon. Dynamic $(1+\epsilon)$-approximate matchings: A density-sensitive approach. In *SODA*, 2016.

[16] P. Sankowski. Faster dynamic matchings and vertex connectivity. In *SODA*, 2007.

[17] V. G. Vizing. The chromatic class of a multigraph. *Kibernetika*, 3:29–39, 1965.