

Query Efficient Weighted Stochastic Matching

Mahsa Derakhshan
Northeastern University

Mohammad Saneian
Northeastern University

In this paper, we study the *weighted stochastic matching* problem. Let $G = (V, E)$ be a given edge-weighted graph and let its realization \mathcal{G} be a random subgraph of G that includes each edge $e \in E$ independently with a known probability p_e . The goal in this problem is to pick a sparse subgraph Q of G without prior knowledge of G 's realization, such that the maximum weight matching among the realized edges of Q (i.e. the subgraph $Q \cap \mathcal{G}$) in expectation approximates the maximum weight matching of the entire realization \mathcal{G} .

It is established by previous work that attaining any constant approximation ratio for this problem requires selecting a subgraph of max-degree $\Omega(1/p)$ where $p = \min_{e \in E} p_e$. On the positive side, there exists a $(1 - \varepsilon)$ -approximation algorithm by Behnezhad and Derakhshan [FOCS'20], albeit at the cost of max-degree having exponential dependence on $1/p$. Within the $\text{poly}(1/p)$ regime, however, the best-known algorithm achieves a 0.536 approximation ratio due to Dughmi, Kalayci, and Patel [ICALP'23] improving over the 0.501 approximation algorithm by Behnezhad, Farhadi, Hajiaghayi, and Reyhani [SODA'19].

In this work, we present a 0.68 approximation algorithm with $O(1/p)$ queries per vertex, which is asymptotically tight. This is even an improvement over the best-known approximation ratio of $2/3$ for unweighted graphs within the $\text{poly}(1/p)$ regime due to Assadi and Bernstein [SOSA'19]. The $2/3$ approximation ratio is proven tight in the presence of a few correlated edges in \mathcal{G} , indicating that surpassing the $2/3$ barrier should rely on the independent realization of edges. Our analysis involves reducing the problem to designing a randomized matching algorithm on a given stochastic graph with some *variance-bounding* properties. Both the reduction and the existence of such algorithms heavily rely on independent edge realizations, allowing us to break the $2/3$ barrier.

1 Introduction

In the *stochastic weighted matching* problem, we are given an n -vertex weighted graph $G = (V, E)$ along with a parameter $p_e \in (0, 1]$ for any edge $e \in E$. A random subgraph \mathcal{G} of G is then generated by independently including (or realizing) each edge $e \in E$ with probability p_e . Here, we refer to G as the base graph and \mathcal{G} as the realized subgraph. The objective of this problem is to select a subgraph Q of the base graph without the knowledge of its realization such that: (1) Q has a small max-degree, namely a constant with respect to n , and (2) The realized edges of Q (i.e., the graph $Q \cap \mathcal{G}$) contain a large weight approximate matching. We define the approximation ratio as the expected weight of the maximum matching among the realized edges of Q over the expected weight of the maximum weighted matching of \mathcal{G} .

One immediate application of the stochastic weighted matching problem is its use as a *matching sparsifier*, which approximates the maximum weighted matching even when random edge failures occur [AB19]. Additionally, it finds various applications in matching markets, including kidney exchange [BDH⁺15], online labor markets [BFHR19, BR18], and dating platforms. In these applications, we are provided with the base graph G , but we are tasked with finding a matching in the realized subgraph \mathcal{G} . To achieve this, an algorithm can *query* each edge of G to determine whether it is realized. However, these queries often involve time-consuming or costly operations, such as conducting candidate interviews or medical exams. Hence, it is crucial to minimize the number of queries. This can be accomplished by non-adaptively querying a subgraph Q with a small degree while still expecting to find a matching with a large approximation ratio among its realized edges.

It is established by previous work [AKL16] that attaining any constant approximation ratio for this problem requires selecting a subgraph of max-degree $\Omega(1/p)$ where $p = \min_{e \in E} p_e$. On the positive side, there exists a $(1 - \epsilon)$ -approximation algorithm by Behnezhad and Derakhshan [BD20], albeit at the cost of max-degree having exponential dependence on $1/p$. Within the $\text{poly}(1/p)$ regime, however, the best-known algorithm achieves a 0.536 approximation ratio due to Dughmi, Kalayci, and Patel [DKP23]. This raises the question: How far can we push the approximation ratio while querying a graph with max-degree $\text{poly}(1/p)$? A natural barrier that emerges here is $2/3$ which cannot be exceeded in the presence of a few correlated edges in \mathcal{G} [AKL16, BDH20]. Existing algorithms that attain better than a $2/3$ approximation ratio, even for unweighted graphs, either require a max-degree exponential in $1/p$ [BDH20] or heavily rely on the bipartiteness of the graph [BBD22].

In this work, we prove the existence of a subgraph Q with max-degree $O(1/p)$ whose realization contains a 0.681 approximate matching. This not only improves the previously known approximation ratio of 0.536 significantly but also break the $2/3$ barrier in the $\text{poly}(1/p)$ regime.

Theorem 1.1. *There exists an algorithm that picks a $O(1/p)$ degree subgraph Q of G such that the expected weight of the max-weight realized matching in Q is at least 0.681 times the expected weight of the max-weight realized matching in G .*

The significance of our result is threefold. First, we improve the best-known approximation ratio for both weighted and unweighted graphs in the $\text{poly}(1/p)$ regime (respectively from 0.536 [DKP23] and $2/3$ [AB19]). Second, it selects a subgraph with max-degree $O(1/p)$ — a bound that is asymptotically tight for achieving any constant approximation ratio. Third, we break the well-established $2/3$ barrier for the approximation ratio within the $\text{poly}(1/p)$ regime. Additionally, we demonstrate that the problem can be reduced to designing approximate matching algorithms with specific properties, which we term *variance-bounding* matching algorithms. This reduction implies that further

exploration of the stochastic matching problem may be focused on the development of such algorithms.

Further Related Work. After the pioneering work of Blum et al. [BDH⁺15], the stochastic matching problem has received considerable attention [BDH⁺15, AKL16, AKL17, YM18a, BR18, BFHR19, AB19, BDF⁺19, BDH20, BD20]. To provide a comprehensive understanding of the research landscape, Table 1 presents a chronological survey of results for the problem considered in this paper.

	Reference	Approx.	Max-degree of Q
Unweighted	Blum, Dickerson, Haghtalab, Procaccia, Sandholm, Sharma [BDH ⁺ 15]	$0.5 - \varepsilon$	$\text{poly}(1/p)$
	Assadi, Khanna & Li [AKL17]	0.5001	$O(\log(1/p)/p)$
	Behnezhad, Farhadi, Hajiaghayi, Reyhani [BFHR19]	0.6568	$O(\log(1/p)/p)$
	Assadi & Bernstein [AB19]	$2/3 - \varepsilon$	$O(\log(1/p)/p)$
	Behnezhad, Derakhshan & Hajiaghayi [BDH20]	$1 - \varepsilon$	$(1/p)^{\text{polylog}(1/p)}$
	Behnezhad, Blum & Derakhshan [BBD22]	0.73 (bipartite)	$O(\log(1/p)/p)$
Weighted	Yamaguchi & Maehara [YM18b]	$0.5 - \varepsilon$	$O(W \log n/p)$
	Behnezhad & Reyhani [BR18]	$0.5 - \varepsilon$	$\text{poly}(1/p)$
	Behnezhad, Farhadi, Hajiaghayi, Reyhani [BFHR19]	0.501	$\text{poly}(1/p)$
	Behnezhad & Derakhshan [BD20]	$1 - \varepsilon$	$\exp(1/p)$
	Dughmi, Kalayci & Patel [DKP23]	0.536	$O(1/p)$
	This work	0.681	$O(1/p)$

Table 1: Survey of known results in chronological order. For simplicity, we have hidden the actual dependence on ε inside the O -notation in some cases.

2 Technical Overview

The algorithm we use to construct Q is a quite simple one which was introduced by [BFHR19] and subsequently studied by [BDH20, BBD22]. Given a parameter t , the algorithm starts by drawing t realizations of G drawn from the same distribution as \mathcal{G} . Let us represent these random subgraphs by $\mathcal{G}_1, \dots, \mathcal{G}_t$. We then let Q be the union of max-weight matchings of these graphs. That is

$$Q := \cup_{i \in [t]} \text{MWM}(\mathcal{G}_i),$$

where $\text{MWM}(\cdot)$ is a deterministic algorithm returning the max-weight matching of a given graph. Since Q is a union of t matchings, it clearly has max-degree t . The challenge, however, is proving that for a t as small as $O(1/p)$, the realization of Q contains a large weight matching. We provide a constructive proof for this. That is, we design an algorithm for finding a matching with a large approximation ratio in Q , the actual realization of Q . Below, we first briefly review the ideas used by the previous work and then discuss the ingredients we add to achieve our desired result.

Crucial/Non-crucial Edge Decomposition The framework utilized to analyze the aforementioned algorithm involves partitioning the edges into two categories: *crucial* and *non-crucial*. Separate arguments are then presented to demonstrate how these edges can be integrated to construct a large weight enough matching in Q . Let x_e denote the probability of edge e being part of the

optimal solution, i.e.,

$$x_e = \Pr[e \in \text{MWM}(\mathcal{G})].$$

We define the set of crucial edges, denoted by C , and the set of non-crucial edges, denoted by N , as follows:

$$C = \{e \in E : x_e \geq \tau\} \quad \text{and} \quad N = \{e \in E : x_e < \tau\},$$

where τ is a fixed threshold in the order of p . Note that by choosing a sufficiently large value for $t = O(1/p)$, we can ensure that Q contains nearly all of the crucial edges. To establish the existence of a large weight matching in \mathcal{G} , the first step is to construct a matching M_c exclusively on the crucial edges which is an α -approximation with respect to the contribution of the crucial edges to the optimal solution. (M_c should satisfy some other useful properties which we will discuss later.) The next step is constructing a fractional matching \mathbf{f} on the subgraph of non-crucial edges whose endpoints are unmatched in M_c . This fractional matching should satisfy the two following properties: first, for any edge $e \in N$, it holds that $\mathbf{E}[f_e] \simeq x_e$; second, the values of f_e should be small enough to ensure that \mathbf{f} has almost no integrality gap. By combining these steps, the framework constructs a matching with weight almost $\alpha \times \mathbf{W}(\text{MWM}(\mathcal{G}) \cap C) + \mathbf{W}(\text{MWM}(\mathcal{G}) \cap N)$. Here $\mathbf{W}(\cdot)$ is a function returning the weight of a given matching.

All papers utilizing this analysis framework require the algorithm used for constructing M_c to match the endpoints of any non-crucial edge e independently. Otherwise, the edge is discarded. This requirement is the main reason why Behnezhad and Derakhshan [BD20] need to take an exponential number of edges per vertex. To achieve this property, they employ a distributed LOCAL algorithm for constructing M_c , which can lead to a vertex being dependent on the vertices within its $\Omega(\log(\Delta))$ radius ball, where Δ denotes the crucial degree of a vertex. Since potentially $(1/p)^{\log(1/p)}$ non-crucial edges may be discarded for each vertex, these edges need to have small x_e values. Consequently, a small threshold τ and a large t must be chosen. Due to known lower bounds for matching in the LOCAL model [KMW16], one cannot hope to prove desirable approximation ratios for a Q of max-degree $\text{poly}(1/p)$ following this approach.

In this work, we demonstrate that it is possible to relax the requirement regarding the independent matching of endpoints of any non-crucial edge in M_c . Instead, we replace it with an upper bound on the variance of a parameter related to the neighborhood of each vertex. Specifically, it should be possible to pick a subset A of the vertices unmatched by M_c such that:

1. Any non-crucial edge $e = (u, v)$ satisfies $\Pr[\{u, v\} \subset A] \geq \delta$ for a fixed constant $\delta > 0$.
2. Let us define random variable Z_v related to the neighborhood of any vertex v as

$$Z_v = \sum_{e=(u,v) \in N} \frac{x_e}{\Pr[\{u, v\} \subset A]} \mathbb{1}_{u \in A}. \quad (1)$$

Note that the randomization here is due to A and M_c being random variables themselves. We require the variance of this random variable to be upper-bounded as follows: $\text{Var}(Z_v) \leq \frac{10\tau}{\delta^2}$.

We define an algorithm for finding M_c and A to be a *variance-bounding* matching algorithm (see definition 5.1) if it satisfies the above-mentioned property (and a few others). We then provide a reduction demonstrating that if M_c is an α -approximation with respect to the contribution of crucial edges to the optimal solution, then it is possible to find an almost $\frac{1}{2-\alpha}$ -approximate solution on the realized edges of Q . Our proof strongly relies on independent edge realizations hence enabling us to break the $2/3$ barrier.

The second step of our analysis involves showing the existence of an $\frac{8}{15}$ -approximate variance-bounding matching algorithm. (See Algorithm 3.) We utilize a randomized algorithm designed by Fu et al. [FTW⁺21] for a variant of online stochastic matching and prove that, with slight modifications, it satisfies our desired property. Our analysis begins by identifying a set of independent random variables that determine the algorithm's outcome. We then utilize a method called the Efron-Stein inequality to establish the desired upper bound on $\text{Var}(Z_v)$. Since this inequality is not very commonly used in theoretical computer science, we hope for our work to serve as an example of using this powerful tool in the analysis of randomized algorithms.

3 Preliminaries

3.1 Notation

In the stochastic weighted matching problem, the input is an n -vertex graph $G = (V, E)$, a vector of weights $\mathbf{w} = \{w_e : e \in E\}$ and a probability vector $\mathbf{p} = \{p_e : p_e \in E\}$. Subgraph \mathcal{G} is a random subgraph of G which contains each edge independently with probability p_e . The goal in this problem is to pick a subgraph Q of G without the knowledge of its realization such that: (1) Q has a small max-degree, namely a constant with respect to n , and (2) The realized edges of Q (i.e., the graph $Q \cap \mathcal{G}$) contain a large weight approximate matching. We define the approximation ratio as

$$\frac{\mathbf{E}[\mathbf{W}(\text{MWM}(\mathcal{Q}))]}{\mathbf{E}[\mathbf{W}(\text{MWM}(\mathcal{G}))]},$$

where $\mathcal{Q} = \mathcal{G} \cap Q$ is the realization of Q and $\text{MWM}(\cdot)$ is a deterministic algorithm returning a maximum weighted matching of a given graph, and $\mathbf{W}(M) = \sum_{e \in M} w_e$ is a function returning the weight of a given matching M . We will use $\text{OPT} = \text{MWM}(\mathcal{G})$ to refer to the maximum matching of the actual realization. We may sometimes abuse notation and use OPT to refer to its expected weight when it is clear from the context. Note that while OPT is a random variable $\mathbf{E}[\mathbf{W}(\text{OPT})]$ is just a number. For any edge $e \in E$, we define

$$x_e = \Pr[e \in \text{OPT}],$$

where the probability is taken over the randomization in \mathcal{G} . Similarly, for any vertex $v \in V$ we let $x_v = \Pr[v \in \text{OPT}]$ be the probability that v is matched in OPT . By the definition stated below, \mathbf{x} is a fractional matching as each vertex joins OPT w.p. at most one.

Definition 3.1 (Fractional matching). *A fractional matching \mathbf{x} of a graph $G = (V, E)$ is an assignment $\{x_e\}_{e \in E}$ to the edges, where $x_e \in [0, 1]$ and for each vertex $v \in V$, $x_v := \sum_{e \ni v} x_e \leq 1$. We use $|\mathbf{x}| := \sum_e x_e$ to denote the size of a fractional matching, and for any subset $E' \subseteq E$, use $\mathbf{x}(E')$ to denote $\sum_{e \in E'} x_e$.*

Throughout the paper, we use the notation $O_\varepsilon(f(n))$ which means we have assumed ε to be a constant to calculate the complexity of $f(n)$. The max-degree of subgraph Q we find in this paper depends on the smallest p_e amongst all edges, which we refer to as p . In other words

$$p = \min_{e \in E} p_e.$$

In the following table, we list a set of variables and their values, which we will use throughout the paper. Values are defined as functions of $\varepsilon \in (0, 1)$, which is a sufficiently small constant, and $\delta \in (0, 1)$, which we will introduce in Definition 5.1.

Variable	τ	η	β	γ	c
Value	$20p\varepsilon^5\delta^2$	$\varepsilon/10$	$\varepsilon^2/100$	$\frac{1-\varepsilon^2}{1+3\eta}$	$10/\varepsilon$

Table 2: Value of the parameters used throughout the paper

3.2 Concentration Inequalities and Probabilistic Tools

In this section, we state the concentration inequalities and some of the probabilistic tools that will be used throughout the paper.

Proposition 3.2 (The Efron–Stein Inequality [Ste86]). *Suppose $X_1, \dots, X_n, X'_1, \dots, X'_n$ are independent random variables with X'_i and X_i having the same distribution for all i . Let $X = (X_1, \dots, X_n)$ and $X^{(i)} = (X_1, \dots, X_{i-1}, X'_i, \dots, X_{i+1}, \dots, X_n)$. Then:*

$$\text{Var}(f(X)) \leq \frac{1}{2} \sum_{i=1}^n \mathbf{E} \left[\left(f(X) - f(X^{(i)}) \right)^2 \right]$$

Proposition 3.3 (Chebyshev’s Inequality). *Let X be a random variable with finite non-zero standard deviation s , (and thus finite expected value μ .) Then for any real number $c > 0$, we have*

$$\Pr[|X - \mu| \geq cs] \leq \frac{1}{c^2}.$$

Proposition 3.4 (Law of Total Variance). *Let X be a random variable and Y be a random variable with respect to the same sample space. Then, the variance of X can be expressed as*

$$\text{Var}(X) = \mathbf{E}[\text{Var}(X | Y)] + \text{Var}(\mathbf{E}[X | Y]).$$

Definition 3.5 (Negative Association). *A set of random variables X_1, \dots, X_n is said to be negatively associated if for any two disjoint index sets $i, j \subseteq [n]$ and two functions f, g both monotone increasing or both monotone decreasing it holds:*

$$\mathbf{E}[f(X_i : i \in I) \cdot g(X_j : j \in J)] \leq \mathbf{E}[f(X_i : i \in I)] \cdot \mathbf{E}[g(X_j : j \in J)]$$

4 The Algorithm for Selecting Q

In this section, we present a formal statement of the algorithm employed to construct the subgraph Q . We then explain how we can use the tools we provide later in the paper to show querying Q proves Theorem 1.1 (the main theorem).

In summary, for a given parameter $t = O_\varepsilon(1/p)$, we draw t matchings from the same distribution as OPT (the optimal solution) and define Q as the union of these matchings.

Algorithm 1. The algorithm for constructing Q

```

1  $Q \leftarrow \emptyset$ 
2 for  $i$  from 1 to  $t$  do
3   Let  $G_i$  be a random realization of  $G$  containing each edge independently w.p.  $p_e$ .
4   Set  $M_i = \mu(G_i)$ 
5    $Q \leftarrow Q \cup M_i$ 
6 return  $Q$ 
```

Let us define subsets of crucial and noncrucial edges as follows.

$$C = \{e \in E : x_e \geq \tau\} \quad \text{and} \quad N = \{e \in E : x_e < \tau\}, \quad (2)$$

where $\tau = \theta(\frac{1}{t})$ and $t = \frac{1}{\tau\varepsilon}$ for a sufficiently small $\varepsilon \in (0, 1)$. (The actual value of τ and the other variables used in the paper are presented in Table 2.) Note that in the above algorithm, matching M_1, \dots, M_t are independent from each other and come from the same distribution as OPT. This means that for any edge e and $i \in [t]$ we have $\Pr[e \in M_i] = \Pr[e \in \text{OPT}] = x_e$. As a result, the subgraph outputted by this algorithm contains almost all the crucial edges. Moreover, it picks any non-crucial edge $e \in N$ with a large enough probability as a function of their x_e . We formally state these properties below in Claim 4.1 and Claim 4.2. While the proofs are pretty straightforward, we include them in Section 9 for the sake of completeness.

Claim 4.1. *Given constant numbers $\tau, \varepsilon \in (0, 1)$, Let Q be the output of Algorithm 1 with parameter $t \geq \frac{1}{\tau\varepsilon}$. Any crucial edge $e \in C$ with $x_e \geq \tau$ is present in Q with probability at least $1 - \varepsilon$.*

Claim 4.2. *Any edge $e \in E$ is present in Q with probability at least $\min(1/3, tx_e/3)$.*

4.1 Proof of the Main Theorem.

As discussed previously in Section 2, to prove Theorem 1.1, we will show that Q contains a 0.681-approximate matching. Since Q is Union of $t = O_\varepsilon(1/p)$ matchings, this proves our main result. In Definition 5.1 we define *variance-bounding* matching algorithms. In Lemma 5.2, we prove that for any $\alpha \in (0, 1)$ and a small enough constant $\varepsilon > 0$ existence of an α -approximate variance-bounding algorithm implies Q contains a $(\frac{1}{2-\alpha} - \varepsilon)$ -approximate matching. In Lemma 6.1, we prove the existence of a $\frac{8}{15}$ -approximate variance-bounding matching implying that Q contains a matching with an approximation ratio of

$$\frac{1}{2 - 8/15} - \varepsilon = \frac{15}{22} - \varepsilon \geq 0.6818 - \varepsilon.$$

By picking a small enough $\varepsilon \leq 0.0008$ this gives us an approximation ratio of 0.681.

5 The Reduction

In this section, we first introduce *variance-bounding* matching algorithms and then show that the existence of an α -approximation variance-bounding matching algorithm implies that it is possible to find a $(\frac{1}{2-\alpha} - \varepsilon)$ -approximate matching with $O_\varepsilon(1/p)$ queries per vertex.

Definition 5.1 (Variance-bounding (VB) matching algorithm). *We call a matching algorithm \mathcal{VB} an α -approximation variance-bounding algorithm if it has the following properties. It takes as input (1) a graph $H = (V, E)$ whose edges are realized independently, each with a given probability p_e forming subgraph \mathcal{H} , and (2) a matching M_O of \mathcal{H} found by an arbitrary (potentially randomized) algorithm. The algorithm then outputs a matching M_c of \mathcal{H} and a subset A of vertices that are unmatched in M_c ¹ such that:*

- (i) M_c is in expectation an α -approximate matching with respect to M_O .

¹ A is just a subset of unmatched vertices, so some unmatched vertices may not be in A .

- (ii) For any vertex $v \in V$, $\Pr[v \in A] \geq \Pr[v \notin M_{\mathcal{O}}]$.
- (iii) For any two vertices u, v that do not have an edge in H the following holds. $\Pr[\{u, v\} \subset A] \geq \delta$ for a constant $\delta \in (0, 1)$.
- (iv) Given a parameter $\tau \in (0, 1)$, let \mathbf{x} be a fractional matching on $\overline{H} = (V, \overline{E})$ (the complement of H) with $x_e \leq \tau$ for any $e \in \overline{E}$. For any vertex v variable Z_v , defined below, satisfies $\text{Var}(Z_v) \leq \frac{6\tau}{\delta^2}$.

$$Z_v = \sum_{e=(u,v) \in \overline{E}} \frac{x_e}{\Pr[\{u, v\} \subset A]} \mathbb{1}_{u \in A}.$$

Let us briefly explain why we need a variance-bounding matching algorithm. We will use this algorithm on all the realized crucial edges (i.e., $H = (V, C)$) and define $M_{\mathcal{O}}$ to be a matching with the expected weight the same as the contribution of the crucial edges to the optimal solution. We formally define these inputs in Definition 5.3. This gives us a matching M_c on the crucial edges and a subset A of vertices unmatched in M_c . We will then construct a fractional matching \mathbf{f} with a small integrity gap exclusively using the (queried and realized) non-crucial edges between vertices in A . We need Property (i) to ensure that M_c is large with respect to the contribution of crucial edges to the optimal solution. Property (ii) ensures that each vertex is available in A with a large enough probability for its non-crucial edges to be able to contribute to \mathbf{f} almost as much as their contribution to OPT. We need Property (iii) to ensure that each edge is available for potential contribution to \mathbf{f} with a large enough probability. Finally, we will use Property (iv) to prove that constructing \mathbf{f} in a particular way does not result in fractional degrees of vertices exceeding one too often. For more details about the importance of this property, see Section 5.2.

Lemma 5.2 (The Reduction). *For constant numbers $\alpha \in (0.5, 1)$ and $\varepsilon \in (0, 0.1)$, existence of an α -approximation variance-bounding algorithm \mathcal{VB} (from Definition 5.1) implies a $(\frac{1}{2-\alpha} - \varepsilon)$ approximation algorithm for the weighted stochastic matching problem with $O_\varepsilon(1/p)$ queries per vertex.*

We will prove that the existence of an α -approximation variance-bounding algorithm implies that querying the subgraph Q outputted by Algorithm 1 with parameter $t = O_\varepsilon(1/p)$ gives us a $(\frac{1}{2-\alpha} - \varepsilon)$ -approximate solution. Before formally proving this in Section 5.3, we need to prove a series of other claims and provide some definitions. Below, we give a brief overview of the proof.

The first step of the proof is using the variance-bounding algorithm on the subgraph of all the crucial edges. Recall that \mathcal{VB} takes as input (1) a graph H whose edges are realized independently, each with a given probability p_e forming subgraph \mathcal{H} , and (2) a matching $M_{\mathcal{O}}$ of \mathcal{H} found by an arbitrary (potentially randomized) algorithm. Below, we detail the values assign to these parameters in our reduction.

Definition 5.3 (H and $M_{\mathcal{O}}$). *In our reduction we choose the following values for \mathcal{H} and $M_{\mathcal{O}}$:*

1. We set H to be the subgraph of all the crucial edges C . In other words, $H = (V, C)$.
2. We set $M_{\mathcal{O}} = \text{MWM}(\mathcal{H} \cup \mathcal{N}^*) \cap \mathcal{H}$, where $\mathcal{H} = \mathcal{G} \cap H$ is the actual realization of all the crucial edges, and \mathcal{N}^* is a random hallucination of the non-crucial edges containing each edge independently with probability p_e . Note that $M_{\mathcal{O}}$ is a matching only on the realized crucial edges.

In the remainder of this paper, when referring to a variance-bounding algorithm without specifying the input, we assume that the variables H and $M_{\mathcal{O}}$ are defined according to Definition 5.3.

Executing the variance-bounding algorithm \mathcal{VB} with these predefined inputs gives us a matching M_c on the critical edges and a subset of unmatched vertices denoted as A . Using Property (i), we prove (in Claim 5.4) that M_c is an α approximation with respect to the contribution of the crucial edges to the optimal solution. Since due to Claim 4.1, \mathcal{Q} contains any crucial edge with probability at least $1 - \varepsilon$, this implies that $\mathcal{Q} \cup M_c$ weights, in expectation, at least $(1 - \varepsilon)\alpha$ times the contribution of crucial edges to OPT . It is important to note that we apply the algorithm \mathcal{VB} to all realized critical edges, not exclusively those within \mathcal{Q} (the queried ones). This approach ensures that the output of \mathcal{VB} , consisting of M_c and set A , is independent of the choice of \mathcal{Q} , as outlined in Claim 5.4.

The next step of the reduction is using the non-crucial edges among vertices in A to construct a fractional matching \mathbf{f} . In Lemma 5.11, we use properties of \mathcal{VB} to ensure that the expected contribution of any non-crucial edge to \mathbf{f} is almost the same as its contribution to the optimal solution. We then use the fact that these edges are non-crucial (hence have small f_e s) to prove in Lemma 5.6 that \mathbf{f} has almost no integrity gap. Putting these pieces together, we prove that either union of this rounded matching and M_c is an $(\frac{1}{2-\alpha} - \varepsilon)$ approximate solution, or simply only using the crucial edges in \mathcal{Q} gives us this approximation.

In the following claim, we prove two basic properties about M_c and set A and their relation to the set of non-crucial edges in \mathcal{Q} .

Claim 5.4. *Let M_c and A be the outputs of an α -approximation variance-bounding algorithm which takes as input the subgraph $H = (V, C)$ and matching $M_{\mathcal{Q}}$ defined in 5.3. We have the followings for M_c and A :*

1. *The expected weight of matching M_c is at least α times the weight of matching $\text{OPT} \cap C$ (i.e., the contribution of the crucial edges to the optimal solution).*
2. *Let \mathcal{Q} be the subgraph of edges we choose to query. For any non-crucial edge $e \in N$, the event $e \in \mathcal{Q}$ is independent of both M_c and A .*

Proof. **TOPROVE 0** □

5.1 A Fractional Matching on the Non-crucial Edges

In this section, we will construct a fractional matching on the non-crucial edges to augment the matching we get from running a variance-bounding matching on the crucial edges. Given a variance-bounding algorithm \mathcal{VB} , let M_c and A be the output of \mathcal{VB} with inputs given according to Definition 5.3. To begin, let us define a variable g_e for any non-crucial edge as follows:

$$g_e = \frac{x_e}{\Pr[e \in \mathcal{Q}] \Pr[u, v \in A]}, \quad (3)$$

where x_e is defined as

$$x_e = \Pr[e \in \text{OPT}].$$

Ideally, for constructing our fractional matching, we would assign a fractional value of g_e to edge e whenever $e \in \mathcal{Q}$ and both of its endpoints are in A . Since these events are independent due to Claim 5.4, their joint probability is $\Pr[e \in \mathcal{Q}] \Pr[u, v \in A]$. By constructing a fractional matching in this manner, we achieve $\mathbf{E}[f_e] = x_e$ for any edge e and $\mathbf{E}[\mathbf{f} \cdot \mathbf{w}] = \mathbf{E}[\mathbf{W}(\text{OPT})]$.

However, the challenge lies in the fact that constructing \mathbf{f} in this way may result in it not being a valid fractional matching, as certain vertices may have a fractional degree greater than one. In other words, $\sum_{(u,v) \in N} f_{(u,v)} > 1$ may occur for some vertices $v \in V$. To address this issue, we first scale down the fractional values by a small amount. Subsequently, we discard any vertex whose fractional degree still exceeds one. The challenge then becomes demonstrating that this event does not significantly reduce the expected size of the fractional matching. We formally state the algorithm for constructing a fractional matching on the non-crucial edges in Algorithm 2.

Algorithm 2. A fractional matching on the realized non-crucial edges

```

1 Let  $M_c$  and  $A$  be the outputs of an  $\alpha$ -approximation variance-bounding matching algorithm
  with inputs given according to Definition 5.3.
2 Let  $\mathbf{f}$  be an empty fractional matching on the subgraph of non-crucial edges.
3 Let  $\varepsilon \in (0, 1)$  be a small given constant (the same as the one used in Table 2).
4 Set  $\gamma = (1 - \varepsilon^2)/(1 + \frac{3\varepsilon}{10})$ .
5 for  $e = (u, v) \in N$  do
6   Let  $g_e = x_e / \Pr[e \in \mathcal{Q} \text{ and } \{u, v\} \subset A]$ 
7   if  $e \in \mathcal{Q}$  and both of its end-points are in  $A$  then
8     Set  $f_e = g_e \gamma$ .
9   else
10    Set  $f_e = 0$ .
11 If the fractional degree of a vertex  $v$  in  $\mathbf{f}$  exceeds one (i.e.,  $\sum_{e \ni v} f_e > 1$ ), zero out the
    fractional value of its edges.
12 return  $\mathbf{f}$ 

```

Since our ultimate goal is to demonstrate the existence of a large weight integral matching on \mathcal{Q} rather than a fractional one, let us first address the integrality gap of the fractional matching produced by this algorithm. We first prove an upper bound of ε^3 for g_e of non-crucial edges in Claim 5.5. We then use this in Lemma 5.6 to prove that the output of Algorithm 2 has a small integrality gap. To help with the flow of the paper, both proofs are deferred to Section 9.

Claim 5.5. *By choosing a sufficiently small constant $\varepsilon > 0$ in Algorithm 2, we get $g_e \leq \varepsilon^3$ for all non-crucial edges.*

Lemma 5.6. *Consider the fractional matching \mathbf{f} produced by Algorithm 2. There exists an integral matching on the non-crucial edges of \mathcal{Q} between vertices in A with weight at least $(1 - \varepsilon/2)\mathbf{f} \cdot \mathbf{w}$.*

Survival of vertices and non-crucial edges For any vertex $v \in V$, we say v *survives* Algorithm 2 iff it is in set A , and it is not killed in Line 11 of the algorithm (i.e., its fractional degree is not reduced to one). We also say an edge e survives the algorithm iff both its endpoints survive (regardless of whether it is in \mathcal{Q} or not).

5.2 Expected Weight of the Fractional Matching

Let \mathbf{f}' denote the value of \mathbf{f} constructed by Algorithm 2 before it zeroes out certain f_e values in Line 11. As discussed earlier in this section, it is evident that $\mathbf{E}[f'_e] = \gamma x_e$ for any edge in $e \in N$. Since γ deviates from one by a small constant, the expected weight of \mathbf{f}' is a sufficiently large

approximation relative to the contribution of the non-crucial edges to the optimal solution. Thus, the primary challenge lies in proving that we do not incur a substantial loss by zeroing out certain f_e values in Line 11. Roughly speaking, we only have the opportunity to use an edge $e = (u, v)$ whenever it is in Q and its endpoints are in A (i.e., $f'_e \neq 0$), and we lose this opportunity if at least one of its endpoints does not survive Algorithm 2. That is, we have:

$$\Pr[f_e \neq 0] = \Pr[f'_e \neq 0] - \Pr[u \text{ or } v \text{ does not survive} \mid f'_e \neq 0].$$

To quantify the amount of loss per edge, we need to upper-bound $\Pr[u \text{ or } v \text{ do not survive} \mid f'_e \neq 0]$ and show that it is significantly smaller compared to $\Pr[f'_e \neq 0]$. Note that here, $f'_e \neq 0$ is not independent of e 's end-points surviving since it contributes to their fractional degree. Furthermore, $e \in Q$ is correlated, albeit negatively (see Claim 7.1), with the existence of its neighboring non-crucial edges in Q , which may also impact the fractional degrees of u and v in \mathbf{f}' . However, it is still helpful to first upper-bound the probability of u and v surviving without conditioning on $f'_e \neq 0$. The intuition behind this is that since f'_e is very small (i.e., upper-bounded by ε^3 due to Claim 5.5), its impact on the fractional degree of each endpoint is insignificant. Moreover, since $e \in Q$ is negatively associated with $e' \in Q$ for any non-crucial $e' \neq e$ connected to u or v , conditioning on $e \in Q$ does not increase their $f'_{e'}$. To upper-bound $\Pr[v \text{ does not survive}]$ for any vertex v , let us define

$$Y_v = \sum_{e=(u,v) \in N} g_e \cdot \mathbf{1}_{u \in A} \cdot \mathbf{1}_{e \in Q}. \quad (4)$$

Since we set $f'_e = g_e \gamma$ iff $e \in Q$ and $\{u, v\} \subset A$, whenever vertex v is present in A we have

$$Y_v / \gamma = \sum_{e=(u,v) \in N} f'_e.$$

Hence, vertex v survive Algorithm 2 iff $Y_v / \gamma \leq 1$. In Lemma 5.7, we prove that random variable Y_v is concentrated around its mean for any vertex. This analysis crucially relies on Property (iv) of variance-bounding algorithms. This would have been enough if we knew $\mathbf{E}[Y_v]$ is sufficiently close to one. While we do not exactly have this, we can use the second property of the variance-bounding algorithms to show $\mathbf{E}[Y_v | v \in A] \leq 1$. This is only doable thanks to Property (ii). Combining all these together, we are able to finally prove in Lemma 5.8 that Y_v is sufficiently close to one, with a sufficiently large portability.

Since both Lemma 5.7 and Lemma 5.8 have lengthy and technical proofs, we respectively allocate Section 7 and Section 8 to present detailed proofs for them. Finally, we put the pieces together in Lemma 5.11 to demonstrate that $\mathbf{E}[f_e]$ is sufficiently large compared to x_e (the contribution of e to the optimal solution).

Lemma 5.7. *For any vertex $v \in V$ define random variable*

$$Y_v = \sum_{e=(u,v) \in N} g_e \cdot \mathbf{1}_{u \in A} \mathbf{1}_{e \in Q},$$

where $g_e = \frac{x_e}{\Pr[e \in Q] \cdot \Pr[\{u, v\} \subset A]}$. The following inequality holds for these random variables.

$$\Pr \left[|Y_v - \mathbf{E}[Y_v]| \geq \eta \right] \leq \beta$$

for $\beta = \frac{\varepsilon^2}{100}$ and $\eta = \frac{\varepsilon}{10}$.

Lemma 5.8. For any vertex $v \in V$ we have:

$$\Pr[Y_v \geq 1 + 3\eta] \leq \beta$$

Definition 5.9. For a vertex u' we define $Y_v(-u')$ to be the summation that we have for Y_v except for the edge $e' = (u', v)$. Formally:

$$Y_v = \sum_{e=(u,v) \in N, u \neq u'} g_e \cdot \mathbb{1}_{u \in A} \mathbb{1}_{e \in \mathcal{Q}},$$

Lemma 5.10. For every edge $e' = (v, u')$ and constant $\lambda \in (0, 1)$ we have:

$$\Pr[Y_v(-u') > \lambda] \geq \Pr[Y_v(-u') > \lambda \mid e' \in \mathcal{Q}]$$

Proof. **TOPROVE 1** □

Lemma 5.11. For every non-crucial edge $e = (u, v)$ we have $\mathbf{E}[f_e] \geq (1 - \varepsilon/2) \cdot x_e$.

Proof. **TOPROVE 2** □

5.3 Proof of Lemma 5.2 (The Reduction)

In this section, we will put all the pieces together to formally prove Lemma 5.2. Let Q be the subgraph outputted by Algorithm 1. We prove that the existence of an α -approximation variance-bounding matching algorithm means \mathcal{Q} , the realization of Q , contains a $\frac{1}{2-\alpha} - \varepsilon$ approximate solution. Since Q is the union of $t = \frac{1}{\tau\varepsilon}$ matchings, plugging in the value of $\tau = p\varepsilon^5\delta^2$ from Table 2 implies Q has max-degree $O_\varepsilon(1/p)$. Therefore proving that \mathcal{Q} contains a $\frac{1}{2-\alpha} - \varepsilon$ approximate solution proves this lemma.

Let M_c and A be the outputs of the α -approximation variance-bounding algorithm on inputs specified in Definition 5.3. Recall that M_c is a matching on the crucial edges and A is a subset of vertices unmatched in M_c . Let σ be the ratio of the optimal solution that comes from the crucial edges. That is

$$\sigma = \frac{\sum_{e \in C} \Pr[e \in \text{OPT}] w_e}{W(\text{OPT})}.$$

Due to Claim 5.4, we know that the expected weight of M_c is $\alpha\sigma$ fraction of the optimal solution. Furthermore, we showed in Claim 4.1 that any crucial edge belongs to \mathcal{Q} with probability at least $(1 - \varepsilon)$. As a result we have

$$\mathbf{E}[W(M_c \cap \mathcal{Q})] \geq (1 - \varepsilon)\alpha\sigma W(\text{OPT}). \quad (5)$$

The next step is to use the non-crucial edges among vertices in A to augment $M_c \cap \mathcal{Q}$. In Lemma 5.11, we prove that it is possible to construct a fractional matching \mathbf{f} on the non-crucial edges among vertices in A such that for any non-crucial edge

$$\mathbf{E}[f_e] \geq (1 - \varepsilon/2) \Pr[e \in \text{OPT}].$$

Hence, $\mathbf{E}[\mathbf{f}w] \geq (1 - \sigma)(1 - \varepsilon/2)W(\text{OPT})$. As a result of Lemma 5.6 it is possible to round \mathbf{f} and achieve an integral matching M_n such that

$$\mathbf{E}[W(M_n)] \geq (1 - \varepsilon/2)(1 - \sigma)(1 - \varepsilon/2)W(\text{OPT}) \geq (1 - \varepsilon)(1 - \sigma)W(\text{OPT}). \quad (6)$$

Putting equation (5) and equation (6) together implies the existence of a matching in \mathcal{Q} with expected weight at least

$$(1 - \varepsilon)\alpha\sigma W(\text{OPT}) + (1 - \varepsilon)(1 - \sigma)W(\text{OPT}) = (1 - \varepsilon)W(\text{OPT})(1 - \sigma + \alpha\sigma).$$

We claim that the best of this matching and simply taking the max-weight matching among the crucial edges of \mathcal{Q} gives us the desired approximating ratio. Since each crucial edge belongs to \mathcal{Q} w.p. at least $1 - \varepsilon$, its realization contains a matching with expected weight at least $(1 - \varepsilon)$ times the contribution of crucial edges to the optimal solution which is $(1 - \varepsilon)\sigma W(\text{OPT})$. The best of these two solutions gives us the approximation ratio of at least

$$(1 - \varepsilon) \cdot \max(\sigma, 1 - \sigma + \alpha\sigma) \geq (1 - \varepsilon) \frac{1}{2 - \alpha} \geq \frac{1}{2 - \alpha} - \varepsilon.$$

Hence, this implies that the realization of subgraph \mathcal{Q} with max-degree $O_\varepsilon(1/p)$ contains a $(\frac{1}{2 - \alpha} - \varepsilon)$ -approximate solution completing the proof of Lemma 5.2.

6 A Variance-Bounding Matching Algorithm

In this section, we discuss the existence of an $\frac{8}{15}$ -approximation variance-bounding matching algorithm.

Lemma 6.1 (Variance-bounding Matching Lemma). *There exists an $8/15$ -approximation variance-bounding matching algorithm (as defined in Definition 5.1).*

To prove this lemma, we will design an algorithm which achieves the properties of a variance bounding matching algorithm discussed in Definition 5.1. One of the main tools we employ in our design is an algorithm designed for online matching due to Fu, Tang, Wu, Wu, and Zhang [FTW⁺21]. We will refer to this algorithm as FTWWZ (the authors' initials). Below, we briefly review the main components of their work to the extent required for presenting our algorithm and results.

Batched RCRS: In our algorithm, we will use a batched random order contention resolution scheme (RCRS) in the FTWWZ algorithm for the following online matching problem. We are given a graph $G = (V, E)$ along with a fractional matching \mathbf{y} on the graph. The graph is revealed in an online manner as follows. Vertices arrive in a uniformly random order given by a permutation π . Upon the arrival of a vertex v , the status of the edges connecting v to vertices before it (i.e., all vertices u that $\pi_u < \pi_v$) are revealed, namely a *batch* of edges. Then at most one of the edges in the batch becomes *active* such that

$$\Pr[e \text{ becomes active}] = y_e$$

for any edge e in the batch. A batched RCRS decides, upon the arrival of each vertex, irrevocably whether to select the active edge (if any exists). At any point in time, the selected edges must form a matching. Given a parameter α a batched RCRS is called α -selectable if it picks each active element with probability at least α . FTWWZ provides a simple greedy algorithm which is $\frac{8}{15}$ -selectable. The algorithm starts by modifying the graph and then greedily matches any active edge if its endpoints are unmatched. One of these modifications is lowering the active probability of each edge using a function

$$g(y_e) = \frac{3y_e}{3 + 2y_e}. \tag{7}$$

Below is the main result we use from FTWZZ's work.

Proposition 6.2. *If in the above-mentioned setting, each edge becomes active with probability $g(y_e) = \frac{3y_e}{3+2y_e}$ then it is possible to design an RCRS for constructing a matching which selects each edge w.p. at least $\frac{8}{15}y_e$.*

We state our variance-bounding matching algorithm formally in Algorithm 3. The algorithm starts by drawing a random permutation π over the vertices uniformly at random. We then let the vertices arrive in the order given by this permutation. Upon arrival of a vertex v , we look at the realization of its edges to the vertices with smaller π_u . Then, a random process decides which one of its edges (if any) becomes active. We explain this process in Definition 6.3. The process is designed in a way that the probability of each edge becoming active is $g(\Pr[e \in M_O])$ where M_O is the random matching in the statement of Lemma 6.1. Proposition 6.2 implies that using FTWZZ's RCRS we can construct a matching M_c which $\frac{8}{15}$ -approximates M_O . We then let set A be the set of all the vertices that do not have an active edge throughout the algorithm. Note that any vertex in A is also unmatched since only active edges can join the matching. However, there may be some unmatched vertices that are not in A .

Definition 6.3. (*Edge Activation Process*) *The activation probability of the edges in this process comes from matching M_O in the statement of Definition 5.1. Recall that M_O is a matching on the realized crucial edges. See Definition 5.3 for what we set M_O to.*

First, let us define

$$y_e = \Pr[e \in M_O]. \quad (8)$$

Here, the randomization comes from the realization of edges in \mathcal{H} and the algorithm for finding M_O . Note that \mathbf{y} is a fractional matching since each vertex joins M_O with probability at most one. Moreover, define set $E_{v,\pi} = \{u \in E : \pi_u < \pi_v\}$ to be all of v 's edges to vertices u with $\pi_u < \pi_v$. After looking at the realization of all these edges let y'_e be the probability of e being in M_O conditioned on the realization of $E_{v,\pi}$. That is

$$y'_e = \Pr[e \in M_O \mid \text{realization of edges in } E_{v,\pi}]. \quad (9)$$

We activate at most one of the realized edges at random such that the probability of any realized edge e being the active one is $\frac{3y'_e}{3+2y'_e}$. This is possible since y'_e of the realized edges sums up to at most one.

We now have all the required tools to design our algorithm (stated below) which we claim satisfies the properties stated in Lemma 6.1.

Algorithm 3. Variance-bounding Matching on $H = (V, E)$

```

1  $M_c \leftarrow \emptyset$  and  $A \leftarrow \emptyset$ .
2 Let  $\pi$  be a permutation over vertices in  $V$  picked uniformly at random.
3 for  $v \in V$  in the order of  $\pi$  do
4   Let  $E_{v,\pi} = \{u \in E : \pi_u < \pi_v\}$  be all of  $v$ 's edges to vertices  $u$  with  $\pi_u < \pi_v$ .
5   At most one edges  $e \in E_{v,\pi}$  becomes active as described by the edge activation process 6.3
   according to  $M_{\mathcal{O}}$ .
6   if  $e = (v, u)$  becomes active and  $u$  is unmatched in  $M_c$  then
7      $\perp$  FTWZZ's RCRS from Propostion 6.2 decides whether  $e$  joins matching  $M_c$  or not.
8 Kill all the vertices who had at least one active edge at some point in the algorithm.
9 Let  $A$  be the set of remaining alive vertices.
10 return  $M_c$  and  $A$ .

```

Claim 6.4. *For any permutation π in Algorithm 3, the probability of any edge e becoming active is $g(\Pr[e \in M_{\mathcal{O}}])$.*

Proof. **TOPROVE 3** □

Proof of Property (i). Consider matching M_c outputted by Algorithm 3. The first property of Lemma 6.1 requires M_c to be a $\frac{8}{15}$ approximate matching with respect to $M_{\mathcal{O}}$. This is an immediate corollary of Proposition 6.2 since we proved in Claim 6.4 that each edge will be activated with probability $g(\Pr[e \in M_{\mathcal{O}}])$, and Proposition 6.2 states that in this case, FTWZZ's RCRS selects any edge with probability $\frac{8\Pr[e \in M_{\mathcal{O}}]}{15}$. Hence using this RCRS in the algorithm results in satisfying Property (i).

Proof of Property (ii). Consider subset A outputted by Algorithm 3. The second property requires $\Pr[v \in A] \geq \Pr[v \notin M_{\mathcal{O}}]$. This claim also holds due to Claim 6.4. A vertex v does not joins set A iff it has an active edge at some point in the algorithm. This happens with probability at most $\Pr[v \in M_{\mathcal{O}}]$ as a result v joins A with probability at least $\Pr[v \notin M_{\mathcal{O}}]$. Hence, we proved Algorithm 3 satisfies the first two properties.

Proof of Property (iii). The third property requires any two vertices u and v that do not share an edge in the input graph to be in A at the same time w.p. at least an absolute constant $\delta \geq 0$. Due to proof of this property being lengthy we provide it in Claim 9.1.

Before discussing the final property, we will define a set of random variables that together can determine the value of Z_v and importantly are independent from each other.

Definition 6.5 (Influential Random Variables $\{X_1, \dots, X_n\}$). *Consider a run of Algorithm 3 with a fixed permutation π . For any vertex u we define random variable X_u as follows. If upon arrival of u , it has an active edge $e = (u, w)$ then $X_u = w$. Otherwise $X_u = \emptyset$.*

Note that knowing the influential random variables uniquely determines set A , as they collectively contain the information regarding which edges become active during the algorithm. A vertex joins set A if it has no active edges during the algorithm. Therefore, by knowing the influential random variables, we have complete knowledge of set A . Importantly, these variables exhibit a crucial feature: they are independent of each other. This independence plays an important role in

the analysis of Property (iv). We provide the proof of this independence in the claim below.

Claim 6.6. *The influential random variables X_1, \dots, X_n defined according to Definition 6.5 for vertices in V are independent.*

Proof. TOPROVE 4 □

Proof of Property (iv). We formally prove this property in Lemma 6.7. However, let us give some intuition before diving into it. Let us first investigate what bound we could get for the variance of Z_v if the membership of vertices in A were independent from each other. In that case, Z_v would be the sum of independent random variables and its variance would be the sum of their variance. Therefore, we would get

$$\begin{aligned} \text{Var}(Z_v) &= \sum_{(u,v) \in \bar{E}} \text{Var}\left(\frac{x_{(u,v)}}{\Pr[\{u,v\} \subset A]} \mathbb{1}_{u \in A}\right) \\ &\leq \sum_{(u,v) \in \bar{E}} \Pr[u \in A] \left(\frac{x_{(u,v)}}{\Pr[\{u,v\} \subset A]}\right)^2 \\ &\leq \frac{\sum_{(u,v) \in \bar{E}} x_{(u,v)}^2}{\delta^2} \leq \frac{1}{\tau} \frac{\tau^2}{\delta^2} \leq \frac{\tau}{\delta^2}. \end{aligned}$$

This implies that our desired upper-bound for $\text{Var}(Z_v)$ is just a constant times the upper-bound we would get under independence which is quite a strong claim. To prove this claim we first write Z_v as a function of the influential random variables. Since they are independent, we can use the Efron–Stein inequality to give an upper bound for $\text{Var}(Z_v)$. This inequality basically requires to show that redrawing any of the influential random variables changes the value of Z_v by a small amount in expectation. One of the main reasons we can achieve this is that redrawing any X_u can only change the membership of three vertices in A . Furthermore, for any pair of vertices u and w we have $\Pr[X_u = w] \leq z'_v$.

Lemma 6.7 (Property (iv)). *Consider set A outputted by Algorithm 3. Given any fractional matching \mathbf{x} on \bar{H} (the complement of H), for any vertex $v \in V$ define*

$$Z_v = \sum_{(u,v) \in \bar{E}} h_{(u,v)} \mathbb{1}_{u \in A},$$

where $h_e = \frac{x_e}{\Pr[\{u,v\} \subset A]}$. If for a parameter $\tau \in (0, 1)$, the fractional matching satisfies $x_e < \tau$, then $\text{Var}(Z_v) \leq \frac{10\tau}{\delta^2}$.

Proof. TOPROVE 5 □

7 Proof of Lemma 5.7

We devote this section to prove Lemma 5.7 due to it being lengthy and technical.

Lemma 5.7 (restated). *For any vertex $v \in V$ define random variable*

$$Y_v = \sum_{e=(u,v) \in N} g_e \cdot \mathbb{1}_{u \in A} \mathbb{1}_{e \in Q},$$

where $g_e = \frac{x_e}{\Pr[e \in \mathcal{Q}] \cdot \Pr[\{u, v\} \subset A]}$. The following inequality holds for these random variables.

$$\Pr \left[|Y_v - \mathbb{E}[Y_v]| \geq \eta \right] \leq \beta$$

for $\beta = \frac{\varepsilon^2}{100}$ and $\eta = \frac{\varepsilon}{10}$.

To prove the desired concentration bound on Y_v we begin by bounding its variance. This will allow us to apply Chebyshev inequality (Proposition 3.3) to prove our desired bound. Let us first examine the random variables that affect Y_v 's value. One collection is the set of variables for presence of vertices after running Algorithm 3 in set A , i.e., $S_A = \{\mathbb{1}_{u \in A} : u \in V\}$ and the second collection is the edges being present in \mathcal{Q} , i.e. $S_Q = \{\mathbb{1}_{e \in \mathcal{Q}} : e = (u, v) \in N\}$.

By the law of total variance (Proposition 3.4) we have:

$$\text{Var}[Y_v] = \mathbb{E}[\text{Var}(Y_v \mid S_A)] + \text{Var}[\mathbb{E}(Y_v \mid S_A)]$$

We will later prove that

$$\mathbb{E}[\text{Var}[Y_v \mid S_A]] \leq 60 \cdot (\varepsilon^6 + \varepsilon^5) \quad (10)$$

To bound the term $\text{Var}[\mathbb{E}(Y_v \mid S_A)]$ let us first examine what $\mathbb{E}(Y_v \mid S_A)$ is.

$$\begin{aligned} \mathbb{E}[Y_v \mid S_A] &= \sum_{e=(u,v) \in N} \mathbb{E}[g_e \cdot \mathbb{1}_{u \in A} \cdot \mathbb{1}_{e \in \mathcal{Q}} \mid S_A] \\ &= \sum_{e=(u,v) \in N} \mathbb{E} \left[\frac{x_e}{\Pr[e \in \mathcal{Q}] \cdot \Pr[\{u, v\} \subset A]} \cdot \mathbb{1}_{u \in A} \cdot \mathbb{1}_{e \in \mathcal{Q}} \mid S_A \right] \\ &= \sum_{e=(u,v) \in N} \mathbb{E} \left[\frac{\mathbb{1}_{e \in \mathcal{Q}}}{\Pr[e \in \mathcal{Q}]} \right] \mathbb{E} \left[\frac{x_e}{\Pr[\{u, v\} \subset A]} \cdot \mathbb{1}_{u \in A} \mid S_A \right] \\ &= \sum_{e=(u,v) \in N} \mathbb{E} \left[\frac{x_e}{\Pr[\{u, v\} \subset A]} \cdot \mathbb{1}_{u \in A} \right] \end{aligned}$$

To go from the second to the third line, we are using the fact that A and \mathcal{Q} are independent due to Claim 5.4. Note that the term in the last line is Z_v in Lemma 6.7. Applying the lemma with the fractional matching x being the edges having $x_e < \tau$ we get that:

$$\text{Var}[\mathbb{E}(Y_v \mid S_A)] \leq \frac{10\tau}{\delta^2}$$

Adding this with what we have from equation (10) and applying law of total variance we get

$$\text{Var}(Y_v) \leq 60 \cdot (\varepsilon^6 + \varepsilon^5) + \frac{10\tau}{\delta^2} \quad (11)$$

Since $\tau = 20p\varepsilon^5\delta^2$ by setting ε to be a small enough constant, we can get the bound $\text{Var}[Y_v] \leq \frac{\varepsilon^4}{10^4}$. This will bound the standard deviation of Y_v by $\frac{\varepsilon^2}{100}$ which is used when applying Chebyshev's Inequality (See Proposition 3.3) on the random variable Y_v . Now that we have $s \leq \frac{\varepsilon^2}{100}$, by applying Chebyshev's Inequality, we get

$$\Pr \left[|Y_v - \mathbb{E}[Y_v]| \geq c \cdot s \right] \leq \frac{1}{c^2} \quad (12)$$

Note that we wanted to bound the probability that Y_v deviates from its mean by η . Now if we have $\eta \geq c \cdot s$, we have

$$\Pr \left[|Y_v - \mathbb{E}[Y_v]| \geq \eta \right] \leq \Pr \left[|Y_v - \mathbb{E}[Y_v]| \geq c \cdot s \right] \quad (13)$$

By replacing value of $\eta = \frac{\varepsilon}{10}$ and the fact that $s \leq \frac{\varepsilon^2}{100}$ we can see that it is enough to set $c = \frac{\varepsilon}{10}$ to satisfy $\eta \geq c \cdot s$. Therefore by combining (12) and (13) we get

$$\begin{aligned} \Pr \left[|Y_v - \mathbb{E}[Y_v]| \geq \eta \right] &\leq \Pr \left[|Y_v - \mathbb{E}[Y_v]| \geq c \cdot s \right] \\ &\leq \frac{1}{c^2} \\ &\leq \frac{\varepsilon^2}{100} = \beta \end{aligned}$$

Now that we proved the statement of the lemma using Equation (10), we prove it which states $\mathbf{E}[\text{Var}[Y_v|S_A]] \leq 60 \cdot (\varepsilon^6 + \varepsilon^5)$. Our first step is to see how random variables in S_Q behave. First of all, random variables in S_Q are not independent since \mathcal{Q} is a collection of matchings, for two incident edges e_1 and e_2 , when e_1 is present in one of the matchings e_2 will not be in that matching. This intuition might make us believe that for edges relevant to S_Q because they all intersect at the vertex v their presence in \mathcal{Q} is pairwise **negatively correlated**. This is in fact true and for proving it we prove a stronger fact about the random variables which is negative association which implies negative correlation. (see Definition 3.5 for definition).

Lemma 7.1. *Random variables in S_Q are negatively associated.*

Proof. **TOPROVE 6** □

By definition, negative association implies negative correlation. This means Lemma 7.1 implies that for two edges $e_1 = (u_1, v), e_2 = (u_2, v)$ such that $e_1, e_2 \in N$ we have:

$$\text{Cov}(\mathbf{1}_{e_1 \in \mathcal{Q}}, \mathbf{1}_{e_2 \in \mathcal{Q}}) \leq 0 \quad (14)$$

Let us take an arbitrary realization of variables in S_A and call it \mathbf{A} . Our plan is, given this fixed \mathbf{A} , first upper-bound $\text{Var}[Y_v|\mathbf{A}]$. Then, using that, find an upper bound for $\text{Var}[Y_v]$. At last, we apply Proposition 3.3 to prove the statement of the lemma.

Define the random variable

$$X_u = (g_{(u,v)} \cdot \mathbf{1}_{u \in A} \cdot \mathbf{1}_{e \in \mathcal{Q}} | \mathbf{A}).$$

We can see that if $\mathbf{1}_{u \in A} = 0$, X_u is always equal to zero, and the inequalities discussed further will be trivial for $\text{Var}[X_u]$. In the case that $\mathbf{1}_{u \in A} = 1$, $X_u = (g_{(u,v)} \cdot \mathbf{1}_{e \in \mathcal{Q}})$. We can see that $(Y_v | \mathbf{A}) = \sum_{(u,v) \in N} X_u$. Now, we are ready to bound the variance of Y_v conditioned on \mathbf{A} . The first step is to bound the variance of X_u :

$$\text{Var}[X_u] = \text{Var}[g_e \cdot \mathbf{1}_{u \in A} \cdot \mathbf{1}_{e \in \mathcal{Q}} | \mathbf{A}] \leq \text{Var}[g_e \cdot \mathbf{1}_{e \in \mathcal{Q}} | \mathbf{A}] \quad (15)$$

This is because when we have fixed A , in the case that $\mathbf{1}_{u \in A} = 0$ then variance of X_u is zero and in the case that $\mathbf{1}_{u \in A} = 1$ the bound in Equation (15) holds.

Now we know that $\text{Var}[X_u] = \mathbf{E}[X_u^2] - \mathbf{E}[X_u]^2 \leq \mathbf{E}[X_u^2]$ so from Equation (15) we get:

$$\text{Var}[X_u] \leq \mathbf{E}[X_u^2] \leq \mathbf{E}[(g_e \cdot \mathbf{1}_{u \in \mathcal{Q}} | \mathbf{A})^2] \leq \mathbf{E}[(g_e \cdot \mathbf{1}_{u \in \mathcal{Q}})^2] \leq \Pr[e \in \mathcal{Q}] \cdot g_e^2 \quad (16)$$

Note that we can remove the condition on \mathbf{A} because variables in S_Q and S_A are independent. The last step comes from the fact that with probability $\Pr[e \in \mathcal{Q}]$, $(g_e \cdot \mathbf{1}_{u \in \mathcal{Q}})^2$ equals g_e^2 and it is zero otherwise. To make further progress, we need a bound on $\Pr[e \in \mathcal{Q}]$. The following lemma addresses this.

Expanding g_e in Equation (16), we get:

$$\begin{aligned} \text{Var}[X_u] &\leq \Pr[e \in \mathcal{Q}] \cdot \left(\frac{x_e}{p_e \cdot \Pr[e \in \mathcal{Q}] \cdot \Pr[\{u, v\} \subset A]} \right)^2 \\ &\leq \frac{x_e^2}{p_e \cdot \Pr[e \in \mathcal{Q}] \cdot (\Pr[\{u, v\} \subset A])^2} \\ &\leq \frac{x_e^2}{p_e \cdot \Pr[e \in \mathcal{Q}] \cdot \delta^2} \end{aligned} \quad (17)$$

To go from the first line to the second, first note the distinction between Q and \mathcal{Q} in the equation above. By definition of $e \in \mathcal{Q}$ being $e \in Q \cap e \in \mathcal{G}$ we can see that $\Pr[e \in \mathcal{Q}] = p_e \cdot \Pr[e \in Q]$. This is because $e \in \mathcal{G}$ is independent of $e \in Q$ since Q is constructed on hallucinations of \mathcal{G} . To go from the second line to the third line note that in Lemma 6.1, we showed $\Pr[\{u, v\} \subset A] \geq \delta$.

Moreover, from Claim 4.2 we know that $\Pr[e \in Q] \geq \min(1/3, tx_e/3)$ so we consider two cases:

Case 1: $\Pr[e \in Q] \geq \frac{tx_e}{3}$. Combining this and (17) we get:

$$\begin{aligned} \text{Var}[X_u] &\leq \frac{x_e^2}{p_e \cdot \Pr[e \in Q] \cdot \delta^2} \\ &\leq \frac{3x_e^2}{p_e \cdot t \cdot x_e \cdot \delta^2} \\ &\leq \frac{3x_e}{p_e \cdot t \cdot \delta^2} \end{aligned}$$

Case 2: $\Pr[e \in Q] \geq \frac{1}{3}$. Combining this and (17) we get:

$$\text{Var}[X_u] \leq \frac{x_e^2}{p_e \cdot \Pr[e \in Q] \cdot \delta^2} \leq \frac{3x_e^2}{p_e \cdot \delta^2}$$

Now that we have a bound on all $\text{Var}[X_u]$'s we are ready to bound $\text{Var}[Y | \mathbf{A}]$. The following proposition is what we need.

Proposition 7.2. *Let X be a random variable written as the sum of random variables X_1, \dots, X_n . So we have $X = \sum_{i=1}^n X_i$. Then we have:*

$$\text{Var}[X] = \sum_{i=1}^n \text{Var}[X_i] + 2 \cdot \sum_{i=1}^n \sum_{j>i}^n \text{Cov}(X_i, X_j)$$

In (14) we argued that all variables in S_Q are negatively correlated. Recall the definition of $X_u = g_{(u,v)} \cdot \mathbb{1}_{u \in A} \cdot \mathbb{1}_{e \in Q}$. Because we have fixed \mathbf{A} all X_u 's will be equal to zero or $g_{(u,v)} \cdot \mathbb{1}_{e \in Q}$. Hence we can argue that $\text{Cov}(X_u, X_w) \leq 0$. This is because if at least one of them is equal to zero then $\text{Cov}(X_u, X_w) = 0$. Otherwise, since g_e 's are constants sign of $\text{Cov}(X_u, X_w)$ will be the same as $\text{Cov}(\mathbb{1}_{(u,v) \in Q}, \mathbb{1}_{(w,v) \in Q})$.

Therefore, by applying Proposition 7.2 to all X_u 's and the fact that they are pairwise negatively correlated we get:

$$\text{Var}[Y_v | \mathbf{A}] \leq \sum_u \text{Var}[X_u] \leq \sum_u \max\left(\frac{3x_e}{p_e \cdot t \cdot \delta^2}, \frac{3x_e^2}{p_e \cdot \delta^2}\right) \leq \sum_u \frac{3x_e}{p_e \cdot t \cdot \delta^2} + \sum_u \frac{3x_e^2}{p_e \cdot \delta^2} \quad (18)$$

For brevity, we are writing \sum_u instead of $\sum_{(u,v) \in N}$ for all the equations here. To bound the first sum, note that $t = \frac{1}{20 \cdot \varepsilon^6 \cdot \delta^2 \cdot p}$ and also $\sum_{(u,v) \in N} x_e \leq 1$ therefore we have:

$$\sum_u \frac{3x_e}{p_e \cdot t \cdot \delta^2} \leq \sum_u \frac{3 \cdot 20 \cdot \varepsilon^6 \cdot \delta^2 \cdot p_{\min} \cdot x_e}{p_e \cdot \delta^2} \leq \sum_u 60 \cdot \varepsilon^6 \cdot x_e \leq 60 \cdot \varepsilon^6 \quad (19)$$

To bound the second sum, note that for non-crucial edges, we have $x_e \leq \tau$. Since we have $\tau = 20p_{\min}\varepsilon^5\delta^2$ we get:

$$\sum_u \frac{3x_e^2}{p_e \cdot \delta^2} \leq \sum_u \frac{3 \cdot \tau \cdot x_e}{p_e \cdot \delta^2} \leq \sum_u \frac{3 \cdot 20 \cdot \varepsilon^5 \cdot \delta^2 \cdot p_{\min} \cdot x_e}{p_e \cdot \delta^2} \leq \sum_u 60 \cdot \varepsilon^5 \cdot x_e \leq 60 \cdot \varepsilon^5 \quad (20)$$

Putting things together we get, $\text{Var}[Y_v | \mathbf{A}] \leq 60 \cdot (\varepsilon^6 + \varepsilon^5)$. Now since we have proved this for any arbitrary \mathbf{A} we can remove the condition on \mathbf{A} and get:

$$\mathbf{E}[\text{Var}[Y_v | S_A]] \leq 60 \cdot (\varepsilon^6 + \varepsilon^5) \quad (21)$$

which is exactly Equation (10) so the proof is complete.

8 Proof of Lemma 5.8

Recall the random variable Y_v defined as follows for any vertex $v \in V$.

$$Y_v = \sum_{e=(u,v) \in N} g_e \cdot \mathbb{1}_{u \in A} \mathbb{1}_{e \in Q},$$

where

$$g_e = \frac{x_e}{p_e \cdot \Pr[e \in Q] \cdot \Pr[\{u, v\} \subset A]}.$$

In this section, we will prove Lemma 5.8. Recall the statement of the lemma.

Lemma 5.8 (restated) . *For any vertex $v \in V$ we have:*

$$\Pr[Y_v \geq 1 + 3\eta] \leq \beta$$

Claim 8.1. For any vertex $v \in V$ we have

$$\mathbf{E}[Y_v | v \in A] = \frac{x_v}{\Pr[v \in A]}$$

where $x_v = \sum_{e \ni v, e \in N} x_e$.

Proof. **TOPROVE 7** □

Claim 8.2. For any vertex $v \in V$ we have:

$$\mathbf{E}[Y_v] \leq \frac{1}{\delta}$$

Proof. **TOPROVE 8** □

Claim 8.3. Assume we know that $\Pr[|Y_v - \mathbf{E}[Y_v]| \geq \eta] \leq \beta$ then we have the following bound:

$$\mathbf{E}[Y_v] - 2\eta \leq \mathbf{E}[Y_v | v \in A] \leq \mathbf{E}[Y_v] + 2\eta$$

Proof. **TOPROVE 9** □

Now using the claims we proved in this section, we prove Lemma 5.8.

Lemma 5.8 (restated) . For any vertex $v \in V$ we have:

$$\Pr[Y_v \geq 1 + 3\eta] \leq \beta$$

Proof. **TOPROVE 10** □

9 Deferred Proofs

Claim 9.1. Let A be the subset returned by Algorithm 3. For any two vertices u and v that do not share an edge in H we have $\Pr[\{u, v\} \subset A] > \delta$ for a fixed constant $\delta > 0$.

Proof. **TOPROVE 11** □

Claim 4.1 (restated) . Given constant numbers $\tau, \varepsilon \in (0, 1)$, Let Q be the output of Algorithm 1 with parameter $t \geq \frac{1}{\tau\varepsilon}$. Any crucial edge $e \in C$ with $x_e \geq \tau$ is present in Q with probability at least $1 - \varepsilon$.

Proof. **TOPROVE 12** □

Claim 4.2 (restated) . Any edge $e \in E$ is present in Q with probability at least $\min(1/3, tx_e/3)$.

Proof. **TOPROVE 13** □

Claim 5.5 (restated) . By choosing a sufficiently small constant $\varepsilon > 0$ in Algorithm 2, we get $g_e \leq \varepsilon^3$ for all non-crucial edges.

Proof. **TOPROVE 14** □

Lemma 5.6 (restated) . Consider the fractional matching \mathbf{f} produced by Algorithm 2. There exists an integral matching on the non-crucial edges of \mathcal{Q} between vertices in A with weight at least $(1 - \varepsilon/2)\mathbf{f} \cdot \mathbf{w}$.

Proof. **TOPROVE 15** □

References

- [AB19] Sepehr Assadi and Aaron Bernstein. Towards a unified theory of sparsification for matching problems. In Jeremy T. Fineman and Michael Mitzenmacher, editors, *2nd Symposium on Simplicity in Algorithms, SOSA 2019, January 8-9, 2019, San Diego, CA, USA*, volume 69 of *OASICS*, pages 11:1–11:20. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2019.
- [AKL16] Sepehr Assadi, Sanjeev Khanna, and Yang Li. The Stochastic Matching Problem with (Very) Few Queries. In *Proceedings of the 2016 ACM Conference on Economics and Computation, EC '16, Maastricht, The Netherlands, July 24-28, 2016*, pages 43–60, 2016.
- [AKL17] Sepehr Assadi, Sanjeev Khanna, and Yang Li. The Stochastic Matching Problem: Beating Half with a Non-Adaptive Algorithm. In *Proceedings of the 2017 ACM Conference on Economics and Computation, EC '17, Cambridge, MA, USA, June 26-30, 2017*, pages 99–116, 2017.
- [BBD22] Soheil Behnezhad, Avrim Blum, and Mahsa Derakhshan. Stochastic vertex cover with few queries. In Joseph (Seffi) Naor and Niv Buchbinder, editors, *Proceedings of the 2022 ACM-SIAM Symposium on Discrete Algorithms, SODA 2022, Virtual Conference / Alexandria, VA, USA, January 9 - 12, 2022*, pages 1808–1846. SIAM, 2022.
- [BD20] Soheil Behnezhad and Mahsa Derakhshan. Stochastic weighted matching: $(1 - \varepsilon)$ approximation. In *2020 IEEE 61st Annual Symposium on Foundations of Computer Science (FOCS)*, pages 1392–1403. IEEE, 2020.
- [BDF⁺19] Soheil Behnezhad, Mahsa Derakhshan, Alireza Farhadi, MohammadTaghi Hajiaghayi, and Nima Reyhani. Stochastic matching on uniformly sparse graphs. In Dimitris Fotakis and Evangelos Markakis, editors, *Algorithmic Game Theory - 12th International Symposium, SAGT 2019, Athens, Greece, September 30 - October 3, 2019, Proceedings*, volume 11801 of *Lecture Notes in Computer Science*, pages 357–373. Springer, 2019.
- [BDH⁺15] Avrim Blum, John P. Dickerson, Nika Haghtalab, Ariel D. Procaccia, Tuomas Sandholm, and Ankit Sharma. Ignorance is Almost Bliss: Near-Optimal Stochastic Matching With Few Queries. In *Proceedings of the Sixteenth ACM Conference on Economics and Computation, EC '15, Portland, OR, USA, June 15-19, 2015*, pages 325–342, 2015.
- [BDH20] Soheil Behnezhad, Mahsa Derakhshan, and MohammadTaghi Hajiaghayi. Stochastic matching with few queries: $(1 - \varepsilon)$ approximation. In Konstantin Makarychev, Yuri

- Makarychev, Madhur Tulsiani, Gautam Kamath, and Julia Chuzhoy, editors, *Proceedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing, STOC 2020, Chicago, IL, USA, June 22-26, 2020*, pages 1111–1124. ACM, 2020.
- [BFHR19] Soheil Behnezhad, Alireza Farhadi, MohammadTaghi Hajiaghayi, and Nima Reyhani. Stochastic Matching with Few Queries: New Algorithms and Tools. In *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2019, San Diego, California, USA, January 6-9, 2019*, pages 2855–2874, 2019.
- [BR18] Soheil Behnezhad and Nima Reyhani. Almost Optimal Stochastic Weighted Matching with Few Queries. In *Proceedings of the 2018 ACM Conference on Economics and Computation, Ithaca, NY, USA, June 18-22, 2018*, pages 235–249, 2018.
- [DKP23] Shaddin Dughmi, Yusuf Hakan Kalayci, and Neel Patel. On sparsification of stochastic packing problems. In Kousha Etessami, Uriel Feige, and Gabriele Puppis, editors, *50th International Colloquium on Automata, Languages, and Programming, ICALP 2023, July 10-14, 2023, Paderborn, Germany*, volume 261 of *LIPIcs*, pages 51:1–51:17. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2023.
- [DR96] Devdatt P Dubhashi and Desh Ranjan. Balls and bins: A study in negative dependence. *BRICS Report Series*, 3(25), 1996.
- [Edm65] Jack Edmonds. Maximum matching and a polyhedron with 0, 1-vertices. *Journal of research of the National Bureau of Standards B*, 69(125-130):55–56, 1965.
- [FTW⁺21] Hu Fu, Zhihao Gavin Tang, Hongxun Wu, Jinzhao Wu, and Qianfan Zhang. Random order vertex arrival contention resolution schemes for matching, with applications. 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 68:1–68:20. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2021.
- [KLS81] Alam Khursheed and KM Lai Saxena. Positive dependence in multivariate distributions. *Communications in Statistics-Theory and Methods*, 10(12):1183–1196, 1981.
- [KMW16] Fabian Kuhn, Thomas Moscibroda, and Roger Wattenhofer. Local computation: Lower and upper bounds. *Journal of the ACM (JACM)*, 63(2):1–44, 2016.
- [S⁺03] Alexander Schrijver et al. *Combinatorial optimization: polyhedra and efficiency*, volume 24. Springer, 2003.
- [Ste86] J Michael Steele. An efron-stein inequality for nonsymmetric statistics. *The Annals of Statistics*, 14(2):753–758, 1986.
- [YM18a] Yutaro Yamaguchi and Takanori Maehara. Stochastic Packing Integer Programs with Few Queries. In *Proceedings of the Twenty-Ninth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2018, New Orleans, LA, USA, January 7-10, 2018*, pages 293–310, 2018.
- [YM18b] Yutaro Yamaguchi and Takanori Maehara. Stochastic packing integer programs with few queries. In *Proceedings of the Twenty-Ninth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 293–310. SIAM, 2018.