# Scaling matrices and counting the perfect matchings in graphs

Fanny Dufossé*, Kamer Kaya†, Ioannis Panagiotas‡, and Bora Uçar§

December 28, 2021

**Abstract**

We investigate efficient randomized methods for approximating the number of perfect matchings in bipartite graphs and general undirected graphs. Our approach is based on assigning probabilities to edges, randomly selecting an edge to be in a perfect matching, and discarding edges that cannot be put in a perfect matching. The probabilities are chosen according to the entries in the doubly stochastically scaled version of the adjacency matrix of the given graph. The experimental analysis on random and real-life graphs shows improvements in the approximation over previous and similar methods from the literature.

**Keywords:** permanent, perfect matching, doubly stochastic matrix.

## 1 Introduction

We investigate efficient randomized methods for approximating the number of perfect matchings in bipartite graphs and general undirected graphs. Our main tool is a sparse matrix scaling algorithm which we apply to the adjacency matrix of the given graph (bipartite or general) to assign probabilities to the edges and base random choices to these probabilities. Previously, we have shown that such scaling algorithms are useful in approximating the maximum cardinality of matchings in bipartite graphs [6] and general undirected graphs [7].

A given bipartite graph with $n$ vertices on both sides has an $n \times n$ unsymmetric adjacency matrix $\mathbf{A}$, where $a_{ij} = 1$ if the vertex $i$ on the first part and the vertex $j$ on the second part are connected, and $a_{ij} = 0$ otherwise. Counting perfect matchings in a bipartite graph is equivalent to computing the permanent of its adjacency matrix, which is defined as $Per(\mathbf{A}) = \sum_\sigma \prod_i a_{i,\sigma(i)}$, where the summation runs over all permutations $\sigma$ of $1, \ldots, n$. A given graph with $n$ vertices has an $n \times n$ symmetric adjacency matrix, where $a_{ij} = a_{ji} = 1$ if there is an edge between vertices $i$ and $j$. We use this adjacency matrix representation when dealing with graphs. While we focus on the case where $\mathbf{A}$ is a 0-1 matrix, our techniques are applicable to the weighted case with some adaptations.

*Univ. Grenoble Alpes, Inria, CNRS, Grenoble INP, LIG, 38000 Grenoble, France (fanny.dufosse@inria.fr).

†Faculty of Engineering and Natural Sciences, Turkey (kaya@sabanciuniv.edu).

‡ENS Lyon, France (ioannis.panagiotas@ens-lyon.fr).

§Univ Lyon, CNRS, ENS de Lyon, Inria, Université Claude-Bernard Lyon 1, LIP UMR 5668, F-69007 Lyon, France (bora.ucar@ens-lyon.fr).

Approximating the permanent is a well-studied problem. Valiant [26] showed the problem to be #P-Complete. Jerrum et al. [13] discuss an approach using Markov Chains which can provide an $(1+\varepsilon)$-approximation for the permanent in fully polynomial time, with $\tilde{O}(n^{10})$ complexity. Their Markov Chain Monte Carlo (MCMC) approach makes use of the underlying graph being bipartite, and the techniques cannot be generalized easily to the general graph case. Štefankovic et al. [24] take the MCMC approach and highlight the difficulties that arise. They also propose a Markov Chain for efficiently estimating the number of perfect matchings in graphs from a special class. Gurvits and Samorodnitsky [11] and Linial et al. [19] have used matrix scaling and proposed deterministic approximations with exponential guarantees ($2^n$ and $e^n$ respectively).

Rasmussen [22] proposes a practically efficient way of estimating the permanent of a 0-1 matrix, or the number of perfect matchings in bipartite graphs. This initial work is followed by a series of others [2, 9, 10]. Rasmussen's iterative algorithm uniformly and randomly chooses a nonzero from the first row in the matrix at hand and discards the first row and the column of the chosen nonzero. This step is repeated until the whole matrix is consumed or there is an empty row remaining. In the former case, an estimate of a nonzero value is returned which is based on the number of nonzeros of the first rows at each step; in the latter case zero is returned as an estimate.

We investigate an improvement of Rasmussen's original approach in which edges are selected non-uniformly based on a matrix scaling method, which scales a matrix to be doubly stochastic. We provide an analysis of the stated approach by meticulously tracking the steps taken. The analysis upper bounds the number of repetitions we must run to have quality guarantees on the estimated permanent. To the best of our knowledge, for scaling-based permanent estimation, this work presents the most thorough analysis in the literature with computable bounds. These ideas are also extended to counting the number of perfect matchings in graphs. On the practical side, we present an extensive set of experiments on a variety of random, constructed, and real-life instances, some with known and others with unknown number of perfect matchings. Our experiments show that the scaling-based selection mechanism exhibits a significantly better performance on all instances compared to the known methods. Furthermore, the practical performance of our algorithm seems to be even superior to the expected performance derived from our theoretical analysis, which implies that there exist further avenues to explore.

The rest of the paper is organized as follows. Section 2 introduces the notation and the background on scaling and the permanent, and Section 3 discusses related work in detail. Section 4 introduces the proposed approach and presents the pseudocode of the main algorithm for bipartite graphs. The extension to the general, undirected case is presented in Section 5. Section 6 presents the experimental results and a comparison with existing algorithms. Section 7 concludes the paper.

## 2 Notation and background

Matrices are shown in bold upper case letters, e.g., $\mathbf{A}$. With $\mathbf{A}_{ij}$ we denote the submatrix of matrix $\mathbf{A}$ obtained by deleting the $i$th row and the $j$th column. The entries in the matrix are shown with lower-case letters and subscripts, e.g., $a_{i,j}$ denotes the entry of the matrix $\mathbf{A}$ at the $i$th row and $j$th column. The column ids of the nonzeros in the $i$th row of $\mathbf{A}$ is represented as $\mathbf{A}(i,:)$. Vectors are shown with bold, lower-case roman letters, e.g., $\mathbf{v}$. Components of a vector are shown with the same letter and subscripts, e.g., $v_i$. For a random variable $X$, we use $\mathbb{E}[X]$ to denote its expectation. The base of the natural logarithm is shown with $e$.

Consider a certain quantity $P$ that we want to measure. In our case, $P$ is the permanent or

the number of perfect matchings. Assume that we access to an unbiased randomized procedure (or estimator as it is more formally referred to) $X$ such that $\mathbb{E}[X] = P$. A technique for measuring $P$ based on $X$ is achieved by combining the output of $N$ copies of $X$. More specifically, assuming $X_i$ denotes the outcome of the $i$th estimator, $X' = \sum_{i=1}^{N} \frac{X_i}{N}$ is the combined estimate for $P$.

The mean estimate $X'$ for a quantity $P$ achieves $(\varepsilon, \delta)$-approximation whenever $\Pr(|X' - P| \leq \varepsilon P) \geq 1-\delta$. In general, an $(\varepsilon, \delta)$-approximation can be achieved by simulating $O\left(\frac{\mathbb{E}[X^2]}{\mathbb{E}[X]^2} \cdot \frac{1}{\varepsilon^2} \cdot \log(\frac{1}{\delta})\right)$ trials. For this reason, the fraction $\frac{\mathbb{E}[X^2]}{\mathbb{E}[X]^2}$ is called the critical ratio, as it is the key factor in determining whether the approximation scheme runs in polynomial time or not.

An $n \times n$ matrix $\mathbf{A} \neq 0$ is said to have *support* if there is a perfect matching in the associated bipartite graph. Furthermore, if each nonzero can be put in at least one perfect matching, then the matrix is said to have *total support*. Any nonnegative matrix $\mathbf{A}$ with total support can be scaled with two positive diagonal matrices $\mathbf{R}$ and $\mathbf{C}$ such that $\mathbf{RAC}$ is doubly stochastic (that is, the sum of entries in any row and in any column is one). For this reason, matrices with total support are also called *scalable*. The Sinkhorn-Knopp algorithm [23] is a well-known method for scaling matrices to doubly stochastic form. This algorithm generates a sequence of matrices (whose limit is doubly stochastic) by normalizing the columns and the rows of the sequence of matrices alternately. If $\mathbf{A}$ is symmetric and scalable, then $\mathbf{S} = \mathbf{RAR}$ is doubly stochastic. While the Sinkhorn-Knopp algorithm obtains this symmetric, doubly stochastic matrix in the limit, there are other iterative algorithms that maintain symmetry all along the way [17, 18].

If an $n \times n$ nonnegative matrix $\mathbf{A}$ is scalable to a doubly stochastic matrix $\mathbf{S} = \mathbf{RAC}$ with positive diagonal matrices $\mathbf{R}$ and $\mathbf{C}$ then the function

$$g_{\mathbf{A}}(\mathbf{x}, \mathbf{y}) = \mathbf{x}^T \mathbf{A} \mathbf{y} - \sum_{i=1}^{n} \ln x_i - \sum_{j=1}^{n} \ln y_j \tag{1}$$

attains its minimum value for positive $x_i$ and $y_i$ at $\mathbf{x} = \operatorname{diag}(\mathbf{R})$ and $\mathbf{y} = \operatorname{diag}(\mathbf{C})$ [14, Proposition 2]. In particular, for an $n \times n$ 0-1 matrix $\mathbf{A}$, we have $n \leq g_{\mathbf{A}}(\mathbf{x}, \mathbf{y}) \leq n + n \ln n$, where the lower bound is met by a permutation matrix and the upper bound is met by the matrix of ones; in both cases $\mathbf{x}^T \mathbf{A} \mathbf{y} = n$.

Idel [12] gives a comprehensive survey of known results for computing doubly stochastic scalings. Recently, a tighter analysis of the Sinkhorn-Knopp algorithm [3] has been carried out, and other efficient algorithms based on convex optimization have been proposed [1, 4].

## 3 Related work

Rasmussen [22] proposes a practically efficient randomized approach for estimating the permanent of a 0-1 matrix, or for counting perfect matchings in bipartite graphs. Rasmussen's algorithm visits the first row of a matrix and among the nonzeros, chooses one with uniform probability. The first row and the column of the selected nonzero are then removed from the matrix and a smaller matrix is obtained. This process is repeated until either there exists an empty row, in which case zero is returned; or all rows of the original matrix were able to choose a column, in which case a nonzero estimator of the permanent $X_{Ra}$ is returned. This estimator is the product of the number of nonzeros of the first rows for which the random selections are done. While Rasmussen's estimator returns zero in most of the cases, it is an unbiased estimator, and its expected value is equal to

the permanent. Rasmussen additionally shows that $\mathbb{E}[X_{Ra}^2] \leq Per(\mathbf{A})^2 n!$, for an $n \times n$ matrix $\mathbf{A}$, although one can reduce $Per(\mathbf{A})^2$ to $Per(\mathbf{A})$. As stated in Section 2, the value of $\mathbb{E}[X_{Ra}^2]$ is important, since it expresses the number of samples required to ensure approximation guarantees. Our estimator $X_{\mathbf{A}}$ works along the same lines. It also generates a perfect matching step-by-step, but the uniform selection of columns is replaced with a more effective weighted sampling mechanism. The weights used in sampling are obtained with a numerical algorithm used for scaling matrices to a doubly stochastic form.

Beichl and Sullivan [2] also investigate the same mechanism as us and perform some preliminary analysis. Their bound for $\mathbb{E}[X_{\mathbf{A}}^2]$ uses a notation which denotes the average value of the selection probabilities of all perfect matchings. Their analysis is valuable in that it shows that the scaling helps, but it does not yield computable bounds. We follow up on their work by providing some new theoretical insights as well as a more detailed experimental analysis. More specifically, we provide an analysis (with the main result in Theorem 2) which yields efficiently computable bounds on $\mathbb{E}[X_{\mathbf{A}}^2]$ depending on the scaling factors $\mathbf{R}$ and $\mathbf{C}$ for $\mathbf{A}$.

For estimating the number of perfect matchings in general undirected graphs Fürer and Kasiviswanathan [10] discuss three randomized algorithms, called Simple, REP, and Greedy. The one called Simple is a direct adaptation of Rasmussen's for graphs, and selects neighbours with uniform probability. REP extends Simple such that at certain points during the execution, it creates a number copies where each copy selects its own neighbor, and the procedure continues in a similar way in each copy. The results of each copy are later combined. They also discuss a similar algorithm for the bipartite case [9]. Greedy attempts to assign probabilities in a better way by selecting each node with probability inversely proportional to its degree minus one. Fürer and Kasiviswanathan conclude that Simple is easy to analyze but has high worst case bound; Greedy looks good on many graphs but difficult to analyze; and REP has the best theoretical guarantees for its performance on random (Erdös-Renyi) graphs although its worst-case bounds on other graphs can be high. Our approach can be seen as a more sophisticated variant of Greedy. If Greedy were to choose a random neighbor with probability equal to the degree of the neighbor, then it would have been equivalent to applying a single step of Sinkhorn–Knopp. However, the analysis of our variant is simpler and reveals more insights thanks to the global minimization properties of the doubly stochastic scaling, see the function (1) associated with doubly stochastic scaling.

## 4 The proposed algorithm and its analysis

Before discussing our algorithm, we first motivate the use of scaling in estimating the value of the permanent. The following lemma highlights the close connection of the scaling factors and the permanents of a matrix and its submatrices.

**Lemma 1.** *Let* $\mathbf{A}$ *be an* $n \times n$ *0-1 matrix with total support. Let* $\mathbf{R}$ *and* $\mathbf{C}$ *be the diagonal scaling matrices such that* $\mathbf{S} = \mathbf{RAC}$ *is a double stochastic matrix Then,*

$$Per(\mathbf{S}) = Per(\mathbf{A}) \cdot \prod_{i=1}^{n} r_i \cdot c_i.$$

*In addition,*

$$r_i \cdot c_j = \frac{Per(\mathbf{A}_{ij})}{Per(\mathbf{A})} \cdot \frac{Per(\mathbf{S})}{Per(\mathbf{S}_{ij})}.$$

*Proof.* Since $\mathbf{R}$ and $\mathbf{C}$ are diagonal matrices, we have $Per(\mathbf{S}) = Per(\mathbf{A}) \prod r_i \prod c_j$. The equality $Per(\mathbf{S}_{ij}) = Per(\mathbf{A}_{ij}) \prod_{k \neq i} r_k \prod_{\ell \neq j} c_\ell$ holds, as all diagonal products in $\mathbf{S}_{ij}$ have the same value $\prod_{k \neq i} r_k \prod_{\ell \neq j} c_\ell$. Dividing the two equalities side by side yields the second result. □

### 4.1 The algorithm

The proposed algorithm to estimate the permanent is shown in Algorithm 1. The algorithm takes an $n \times n$, 0-1 matrix $\mathbf{A}$ with a nonzero permanent, produces a random variable denoted as $X_{\mathbf{A}}$ and a perfect matching (this is for the analysis). Initially $X_{\mathbf{A}}$ is equal to one. The algorithm proceeds in $n$ steps. At every step, the algorithm adds a nonzero entry to a matching, thereby obtaining a perfect matching at the end. At step $i$, a nonzero entry in the $i$th row of $\mathbf{A}$ is chosen among those columns $\mathbf{A}$ which have not been matched yet. The nonzero entry chosen at row $i$ defines the matched column. Since the $i$th row is the first row at step $i$, we use $\sigma_1^{(i)}$ to denote the column chosen for $i$, and $\mathbf{A}^{(i)}$ to denote the remaining matrix. The nonzeros are selected according to the values of the entries in a doubly stochastically scaled version of the remaining matrix. The random variable $X_{\mathbf{A}}$ is multiplied by the reciprocal of the value of the chosen nonzero. For the algorithm to work, we discard the nonzeros in $\mathbf{A}^{(i)}$ that cannot be put into a perfect matching. This is achieved by applying the Dulmage-Mendelsohn [8, 21] decomposition, and filtering out the entries that fall into the off-diagonal blocks in a fine decomposition [21].

**Algorithm 1:** PERMANENT ESTIMATION

**Input:** $n \times n$, 0-1 matrix $\mathbf{A}$

**Output:** Permanent estimate $X_{\mathbf{A}}$; $\sigma$ a perfect matching, where the $i$th entry shows the column chosen for the $i$th row.

1: $X_{\mathbf{A}} \leftarrow 1$
2: $\mathbf{A}^{(1)} \leftarrow \mathbf{A}$
3: **for** $i = 1$ **to** $n$ **do**
4: Filter out those entries of $\mathbf{A}^{(i)}$ that cannot be put into a perfect matching
5: $[\mathbf{R}^{(\sigma,i)}, \mathbf{C}^{(\sigma,i)}] \leftarrow \text{SCALE}(\mathbf{A}^{(i)})$
6: Pick a random nonzero column $j \in \mathbf{A}^{(i)}(1,:)$ by using the probability density function

$$p_j = \frac{s_{1,j}}{\Sigma_{k \in \mathbf{A}^{(i)}(1,:)} s_{1,k}} \text{for all nonzeros } a_{1,j}^{(i)}$$

where $s_{t,k} = r_t^{(\sigma,i)} \cdot c_k^{(\sigma,i)}$ is the corresponding entry in the scaled matrix $\mathbf{S} = \mathbf{R}^{(\sigma,i)} \mathbf{A}^{(i)} \mathbf{C}^{(\sigma,i)}$
7: $X_{\mathbf{A}} \leftarrow X_{\mathbf{A}} / p_j$
8: $\mathbf{A}^{(i+1)} \leftarrow \mathbf{A}_{1j}^{(i)}$ ▶ delete the first row and the $j$th column of $\mathbf{A}^{(i)}$
9: $\sigma(i) \leftarrow j$ ▶ assuming the original numbering

We first comment on the run time complexity of the algorithm. There are $n$ steps, and each step requires computing a Dulmage-Mendelsohn decomposition of a matrix, and scaling a matrix. This can be achieved by $O(\sqrt{n}m)$ time on an $n \times n$ matrix with $m$ nonzeros [21]. There are different algorithms for scaling; see the survey by Idel [12], and more recent papers [3, 1, 4]. The most recent methods based on convex optimization techniques [1, 4] have the smallest run time complexity $\tilde{O}(mn + n^{7/3})$ and $\tilde{O}(m^3/2)$—where the terms involving the deviation from the required

row/column sums and $\log n$ are discarded. The Sinkhorn-Knopp algorithm, which is the easiest to implement, has been shown to have $O(\frac{n^2 \ln n}{\varepsilon^2})$ iterations, each iteration costing $O(m)$ time where $\varepsilon$ is the allowable deviation from one. Other easy to implement variants are given elsewhere [17, 18]. To the best of our knowledge, these algorithms are not yet shown to be of fully polynomial time—there are results using the second singular value of the final matrix; and there are no run time analysis with respect to $\varepsilon$. In our experience [6, 7], Sinkhorn-Knopp kind of algorithms work well for scaling 0-1 matrices for practical purposes; usually, a few iterations suffice to obtain practically well behaving algorithms. In summary, the worst case time complexity of Algorithm 1 is $\tilde{O}(n(\sqrt{n}m + mn^2 \ln n))$ when the Sinkhorn-Knopp algorithm is used for scaling.

The algorithm identifies a perfect matching at the end. Since the scaling factors depend on the identified perfect matching, we use $\mathbf{R}^{(\sigma,i)}$ and $\mathbf{C}^{(\sigma,i)}$ to denote the scaling matrices at the $i$th step of the algorithm, where the random perfect matching $\sigma$ is returned. Recall that $\sigma_1^{(i)}$ denotes the column chosen at the $i$th step. Note that the size of the scaling matrices reduces by one at each step, and that the first entry $r_1^{(\sigma,i)}$ of $\mathbf{R}^{(\sigma,i)}$ is the scaling factor associated for the first row of $\mathbf{A}^{(i)}$. With this notation, we can write

$$X_{\mathbf{A}} = \frac{1}{\prod_{i=1}^{n} r_1^{(\sigma,i)} \cdot c_{\sigma_1^{(i)}}^{(\sigma,i)}} \,. \tag{2}$$

### 4.2 The analysis

Here we give theoretical properties of the proposed algorithm. We start by showing that it obtains an unbiased estimator.

**Theorem 1.** *Let $X_{\mathbf{A}}$ be a random variable returned by Algorithm 1 for the estimate of the permanent of an $n \times n$, 0–1 matrix $\mathbf{A}$. Then $\mathbb{E}[X_{\mathbf{A}}] = Per(\mathbf{A})$.*

*Proof.* We prove the theorem using induction. For the base case where $n = 1$, the argument trivially holds. As the inductive hypothesis, assume that the argument holds for $(n-1) \times (n-1)$ matrices. We then have the following:

$$\begin{aligned}
\mathbb{E}[X_{\mathbf{A}}] &= \sum_{j:a_{1,j}\neq 0} p_j \cdot \frac{1}{p_j} \cdot \mathbb{E}[X_{\mathbf{A}_{1j}}] \\
&= \sum_{j:a_{1,j}\neq 0} \mathbb{E}[X_{\mathbf{A}_{1j}}] \\
&= \sum_{j:a_{1,j}\neq 0} Per(X_{\mathbf{A}_{1j}}) \text{ by the inductive hypothesis} \\
&= Per(\mathbf{A}) \,.
\end{aligned}$$

□

Next, we focus our attention on the analysis of the $\mathbb{E}[X_{\mathbf{A}}^2]$ value. For this purpose, we state and prove the following lemma.

**Lemma 2.** *Let $\mathbf{A}$ be $n \times n$ matrix such that $a_{1,j} = 1$, $\mathbf{A}_{1j}$ be the $(n-1) \times (n-1)$ principal submatrix, and $\mathbf{B}$ be the $(n-1) \times (n-1)$ submatrix obtained from $\mathbf{A}_{1j}$ after discarding entries that*

*cannot be put in a perfect matching. Let* $\mathbf{R}$, $\mathbf{C}$, $\mathbf{D}$, *and* $\mathbf{F}$ *be the positive diagonal matrices such that* $\mathbf{RAC}$ *and* $\mathbf{DBF}$ *are doubly stochastic. Then,*

$$\prod_{i=2}^{n} r_i \cdot \prod_{i=1, i\neq j}^{n} c_i \leq e^{r_1 \cdot c_j - 1} \prod_{i=1}^{n-1} d_i \cdot f_i.$$

*Proof.* Let $\mathbf{r}'$ and $\mathbf{c}'$ be the vectors $\mathbf{r}' = [r_2, \ldots, r_n]^T$ and $\mathbf{c}' = [c_1, \ldots, c_{j-1}, c_{j+1}, \ldots, c_n]^T$. Observe that for the function (1),

$$g_{\mathbf{B}}(\mathbf{d}, \mathbf{f}) \leq g_{\mathbf{B}}(\mathbf{r}', \mathbf{c}')$$

should hold, since $\mathbf{D}$ and $\mathbf{F}$ scale $\mathbf{B}$ to a doubly stochastic form. Therefore,

$$\mathbf{d}^T\mathbf{B}\mathbf{f} - \sum_{i=1}^{n-1} \ln d_i - \sum_{i=1}^{n-1} \ln f_i \leq \mathbf{r'}^T\mathbf{B}\mathbf{c}' - \sum_{i=1}^{n-1} \ln r'_i - \sum_{i=1}^{n-1} \ln c'_i \,,$$

which we arrange as

$$\mathbf{d}^T\mathbf{B}\mathbf{f} - \mathbf{r'}^T\mathbf{B}\mathbf{c}' \leq \sum_{i=1}^{n-1} \ln d_i + \sum_{i=1}^{n-1} \ln f_i - \sum_{i=1}^{n-1} \ln r'_i - \sum_{i=1}^{n-1} \ln c'_i. \tag{3}$$

The left hand side can be bounded below. Since $\mathbf{D}$ and $\mathbf{F}$ scale $\mathbf{B}$ to a doubly stochastic form,

$$\mathbf{d}^T\mathbf{B}\mathbf{f} = n - 1 \,. \tag{4}$$

Since $\mathbf{B}$ is obtained by discarding some (positive) entries in $\mathbf{A}_{1j}$ we have

$$\mathbf{r'}^T\mathbf{B}\mathbf{c}' \leq \mathbf{r'}^T\mathbf{A}_{1j}\mathbf{c}' = n - 2 + r_1 \cdot c_j. \tag{5}$$

Hence, $\mathbf{d}^T\mathbf{B}\mathbf{f} - \mathbf{r'}^T\mathbf{B}\mathbf{c}' \geq 1 - r_1 \cdot c_j$. Note that this last inequality will be tight, if we do not get rid of any entries from $\mathbf{A}_{1j}$, in which case $\mathbf{B} = \mathbf{A}_{1j}$. Furthermore, since $0 < r_1 \cdot c_j \leq 1$, we see that

$$0 \leq 1 - r_1 \cdot c_j \leq \mathbf{d}^T\mathbf{B}\mathbf{f} - \mathbf{r'}^T\mathbf{B}\mathbf{c}'.$$

By combining this with (3) and exponentiation of all parts, we obtain

$$1 \leq e^{1 - r_1 \cdot c_j} \leq e^{\mathbf{d}^T\mathbf{B}\mathbf{f} - \mathbf{r'}^T\mathbf{B}\mathbf{c}'} \leq \frac{\prod_{i=1}^{n-1} d_i \cdot f_i}{\prod_{i=1}^{n-1} r'_i \cdot c'_i}, \tag{6}$$

and this concludes the proof. □

**Theorem 2.** *Let* $\mathbf{A}$ *be* $n \times n$ *matrix with total support, and* $\mathbf{RAC}$ *be its doubly stochastic scaling. Then* $\mathbb{E}[X_{\mathbf{A}}^2] \leq \dfrac{1}{\prod_i r_i \cdot c_i} \cdot Per(\mathbf{A})$.

*Proof.* We prove the theorem by induction. The base case $n = 1$ holds trivially. Assume that the theorem holds for $(n-1) \times (n-1)$ matrices. We then have

$$\mathbb{E}[X_{\mathbf{A}}^2] = \sum_{a_{1,j} \neq 0} r_1 \cdot c_j \cdot \left( \frac{1}{r_1^2 \cdot c_j^2} \cdot \mathbb{E}[X_{\mathbf{A}_{1j}}^2] \right)$$

$$\mathbb{E}[X_{\mathbf{A}}^2] = \sum_{a_{1,j} \neq 0} \frac{1}{r_1 \cdot c_j} \cdot \mathbb{E}[X_{\mathbf{A}_{1j}}^2]$$

$$\mathbb{E}[X_{\mathbf{A}}^2] \leq \sum_{a_{1,j} \neq 0} \frac{1}{r_1 \cdot c_j} \cdot \frac{1}{\prod_z d_z \cdot f_z} \cdot Per(\mathbf{A}_{1j})$$

by the inductive hypothesis, where $\mathbf{D}$ and $\mathbf{F}$ scale $\mathbf{A}_{1j}$

$$\mathbb{E}[X_{\mathbf{A}}^2] \leq \sum_{a_{1,j} \neq 0} \frac{1}{r_1 \cdot c_j} \cdot \frac{1}{\prod_{z=2}^{n} r_z \cdot \prod_{z=1, z \neq j}^{n} c_z} \cdot Per(\mathbf{A}_{1j}) \quad \text{by Lemma 2,}$$

$$\mathbb{E}[X_{\mathbf{A}}^2] \leq \frac{1}{\prod_i r_i \cdot c_i} \cdot \sum_{a_{1,j} \neq 0} Per(\mathbf{A}_{1j})$$

$$\mathbb{E}[X_{\mathbf{A}}^2] \leq \frac{1}{\prod_i r_i \cdot c_i} \cdot Per(\mathbf{A}) \,.$$

□

In the above proof, we made use of a weakened version for Lemma 2. In the original lemma, $\frac{\prod_i d_i \cdot f_i}{\prod_{i \neq 1} r_i \prod_{i \neq j} c_i} \leq e^{r_1 \cdot c_j - 1} \leq 1$ for the submatrix $\mathbf{B} = \mathbf{A}_{1j}$ with scaling coefficients $\mathbf{d}, \mathbf{f}$. Because each possible $j$ has a different exponent term associated with it, we had to replace all exponent terms by the common upper bound of one, so as to obtain a bound over all perfect matchings. These exponential terms can often be significantly less than one, and this replacement can lead to a loose upper bound for $\mathbb{E}[X_{\mathbf{A}}^2]$. We proceed to state two theorems in which the exponent term is handled differently in order to obtain more accurate bounds.

**Theorem 3.** $\mathbb{E}[X_{\mathbf{A}}^2] \leq Per(\mathbf{A}) \cdot mean \left( \sum_\sigma e^{\sum_j r_1^{(\sigma,j)} c_{\sigma_1^{(j)}}^{(\sigma,j)}} \right) \cdot \frac{e^{-n}}{\prod_i r_i \cdot c_i}$ *where the sum is over all perfect matchings* $\sigma$*.*

*Proof.* Consider any perfect matching $\sigma$ of $\mathbf{A}$. Let $X_{\mathbf{A}_\sigma} = \frac{1}{\prod_{i=1}^{n} r_1^{(\sigma,i)} c_{\sigma_1^{(i)}}^{(\sigma,i)}}$, which is equivalent to the value of the random variable $X_{\mathbf{A}}$ should the matching $\sigma$ be returned, as shown in (2).

We use a bottom-up induction to prove the following: Let $1 \leq i \leq n$. Then

$$\prod_{j=i}^{n} \frac{1}{r_1^{(\sigma,j)} c_{\sigma_1^{(j)}}^{(\sigma,j)}} \leq \frac{e^{\left( \sum_{j=i}^{n} r_1^{(\sigma,j)} c_{\sigma_1^{(j)}}^{(\sigma,j)} \right) - n - 1 + i}}{\prod_{j=i}^{n} r_{j-i+1}^{(\sigma,i)} c_{\sigma_1^{(j)}}^{(\sigma,i)}} \,. \tag{7}$$

That is we provide a relation for the scaling matrices $\mathbf{R}^{(\sigma,i)}$ and $\mathbf{C}^{(\sigma,i)}$ of the $i$th step and the multiplication of all $r_1^{(\sigma,j)} c_{\sigma_1^{(j)}}^{(\sigma,j)}$ where $j \geq i$. We use the index $j - i + 1$ to refer to rows with index greater than $i \geq 1$ because in the reshaped matrix of the $i$th step, row $i$ occupies the first position.

The idea resembles the proof of Theorem 2 except that here a tighter upper bound is provided by having the numerator less than one.

In the base case $i = n$, and the relation holds with equality. Assume it holds until $i$ where $i > 1$. Then, for $i - 1$

$$\prod_{j=i-1}^{n} \frac{1}{r_1^{(\sigma,j)} c_{\sigma_1^{(j)}}^{(\sigma,j)}} \leq \frac{e^{\left(\sum_{j=i}^{n} r_1{}^{(\sigma,j)} c_{\sigma_1^{(j)}}^{(\sigma,j)}\right) - n - 1 + i}}{\prod_{j=i}^{n} r_{j-i+1}^{(\sigma,i)} c_{\sigma_1^{(j)}}^{(\sigma,i)}} \cdot \frac{1}{r_1^{(\sigma,i-1)} c_{\sigma_1^{(i-1)}}^{(\sigma,i-1)}}$$

by the induction hypothesis holding for value $i$.

Note that $\prod_{j=i}^{n} r_{j-i+1}^{(\sigma,i)} c_{\sigma_1^{(j)}}^{(\sigma,i)}$ is the product of the scaling factors at the $i$th step, and that $\prod_{j=i}^{n} r_{j-(i-1)+1}^{(\sigma,i-1)} c_{\sigma_1^{(j)}}^{(\sigma,i-1)}$ is the product of the scaling factors at the $(i-1)$st step excluding the row scaling entry $r_1^{(\sigma,i-1)}$ and the associated column scaling entry. Therefore, we can replace

$$\frac{1}{\prod_{j=i}^{n} r_{j-i+1}^{(\sigma,i)} c_{\sigma_1^{(j)}}^{(\sigma,i)}} \quad \text{with} \quad \frac{e^{\left(r_1{}^{(\sigma,i-1)} c_{\sigma_1^{(i-1)}}^{(\sigma,i-1)}\right) - 1}}{\prod_{j=i}^{n} r_{j-(i-1)+1}^{(\sigma,i-1)} c_{\sigma_1^{(j)}}^{(\sigma,i-1)}}$$

using Lemma 2 to obtain an upper bound. That is

$$\prod_{j=i-1}^{n} \frac{1}{r_1^{(\sigma,j)} c_{\sigma_1^{(j)}}^{(\sigma,j)}} \leq \frac{e^{\sum_{j=i}^{n} r_1{}^{(\sigma,j)} c_{\sigma_1^{(j)}}^{(\sigma,j)} - n - 1 + i} \cdot e^{r_1{}^{(\sigma,i-1)} c_{\sigma_1^{(i-1)}}^{(\sigma,i-1)} - 1}}{\prod_{j=i}^{n} r_{j-i+2}^{(\sigma,i-1)} c_{\sigma_1^{(j)}}^{(\sigma,i-1)}} \cdot \frac{1}{r_1^{(\sigma,i-1)} c_{\sigma_1^{(i-1)}}^{(\sigma,i-1)}} .$$

We see that in both terms of the fraction we can include $r_1^{(\sigma,i-1)} c_{\sigma_1^{(i-1)}}^{(\sigma,i-1)}$ to its respective aggregator

$$\prod_{j=i-1}^{n} \frac{1}{r_1^{(\sigma,j)} c_{\sigma_1^{(j)}}^{(\sigma,j)}} \leq \frac{e^{\sum_{j=i-1}^{n} r_1{}^{(\sigma,j)} c_{\sigma_1^{(j)}}^{(\sigma,j)} - n - 1 + (i-1)}}{\prod_{j=i-1}^{n} r_{j-i+2}^{(\sigma,i-1)} c_{\sigma_1^{(j)}}^{(\sigma,i-1)}}$$

which concludes the proof of (7), as for $j = i - 1$, we have $j - i + 2 = 1$. Thus for $i = 1$ we get:

$$X_{\mathbf{A}_\sigma}^2 \leq \frac{e^{\sum_j r_1{}^{(\sigma,j)} c_{\sigma_1^{(j)}}^{(\sigma,j)}}}{\prod_i r_i \cdot c_i} \cdot e^{-n}.$$

Since

$$\mathbb{E}[X_{\mathbf{A}}^2] = \sum_\sigma X_{\mathbf{A}_\sigma}^2 \cdot \prod_{i=1}^{n} r_1^{(\sigma,i)} c_{\sigma_1^{(i)}}^{(\sigma,i)}, \quad \text{we get that}$$

$$\mathbb{E}[X_{\mathbf{A}}^2] = \sum_\sigma X_{\mathbf{A}_\sigma}$$

$$\mathbb{E}[X_{\mathbf{A}}^2] \leq \sum_\sigma \frac{e^{\sum_j r_1{}^{(\sigma,j)} c_{\sigma_1^{(j)}}^{(\sigma,j)}}}{\prod_i r_i \cdot c_i} \cdot e^{-n} ,$$

as we have $Per(\mathbf{A})$ permutations each with each own value. Replacing with the mean multiplied by $Per(\mathbf{A})$ concludes the theorem. □

**Theorem 4.** *Let* $\mathbf{A}$ *be* $n \times n$ *0-1 matrix with total support, and* $\mathbf{RAC}$ *be its doubly stochastic scaling. There exists a positive vector* $\mathbf{k}$ *such that* $\mathbb{E}[X_{\mathbf{A}}^2] \leq \frac{e^{v-n}}{\prod_i r_i \cdot c_i} \cdot Per(\mathbf{A})$*, where* $v = \sum_i k_i = \|\mathbf{k}\|_1 \leq n$*.*

*Proof.* The proof follows that of Theorem 2 to build $\mathbf{k}$ by considering the maximum exponent value at each step and is given in Appendix. □

Note that Theorem 4 implies Theorem 2 if we set $k_i = 1$. The theorem is constructive but does not yield a polynomial algorithm. Algorithm 1 can be made to construct a $\mathbf{k}$ associated with a perfect matching $\sigma$ obtained at the end, to show how much $\frac{1}{\prod_{i=1}^{n} r_1^{(\sigma,i)} \cdot c_{\sigma_1^{(i)}}^{(\sigma,i)}}$ deviates from a potential bound that can be obtained by *mean* $\left( \sum_\sigma e^{\sum_j r_1{}^{(\sigma,j)} c_{\sigma_1^{(j)}}^{(\sigma,j)}} \right) \cdot \frac{e^{-n}}{\prod_i r_i \cdot c_i}$ from the previous theorem. Note also that $\sum_j r_1{}^{(\sigma,j)} c_{\sigma_1^{(j)}}^{(\sigma,j)}$ is less than $n$, unless all terms are one, in which case $\mathbf{A}$ is the $n \times n$ identity matrix (and the permanent of value one is obtained exactly). Therefore, the obtained bound shows that the upper bound stated in Theorem 2 is loose by exponents of $e$.

## 5 An estimator for undirected graphs

In this section, we investigate how to extend the proposed approach for undirected graphs. As we will see, this variant shares similar properties with the algorithm for the bipartite case. In particular with $X_G$ showing the random variable and $M(G)$ showing the number of perfect matchings in $G$, it can be shown that

$$\mathbb{E}[X_G] = M(G)$$

and

$$\mathbb{E}[X_G^2] \leq M(G) \frac{1}{\prod_i r_i} ,$$

where $r_i$ is the $i$th diagonal entry of the scaling factor $\mathbf{R}$ of the adjacency matrix of $G$. Note that since the adjacency matrix $\mathbf{A}$ is symmetric, its scaling is in the form $\mathbf{S} = \mathbf{RAR}$, i.e., we do not need separate scaling values for rows and columns. Algorithm 2 shows the pseudocode of the proposed approach.

At the beginning of the iteration $i$, we have a graph $G^{(i)}$ having at least one perfect matching. We then get the adjacency matrix $\mathbf{A}^{(i)}$ of this graph, which is symmetric. We then check if all entries in $\mathbf{A}^{(i)}$ can put into a nonzero diagonal; that is we look at the Dulmage-Mendelsohn decomposition of the bipartite graph associated with $\mathbf{A}^{(i)}$, and discard edges that cannot be in a perfect matching. We discard those entries (symmetrically) and obtain a sparser matrix $\mathbf{A}^{(i)}$ and scale the resulting matrix. This also translates to an updated $G^{(i)}$. Then, for each edge incident on the first vertex of $G^{(i)}$, we test if there is a perfect matching in $G^{(i)}$ containing that edge. Among all the edges that are inside at least one perfect matching, we choose an edge from a probability distribution where the probabilities are proportional to the scaling entries of the allowed edges. Notice that

$$p_k = \frac{s_{1,k}}{\Sigma_{t \in \mathcal{T}} s_{1,t}}$$

**Algorithm 2:** ESTIMATING THE NUMBER OF PERFECT MATCHINGS IN GRAPHS

**Input:** $G = (V, E)$ is an undirected graph with $n = |V|$ vertices, having a perfect matching
**Output:** An estimate $X_G$ of the number of perfect matchings in $G$

1: $X_G \leftarrow 1$
2: $G^{(1)} \leftarrow G$
3: **for** $i = 1$ **to** $n$ by increments of two **do**
4: Let $\mathbf{A}^{(i)}$ be the adjacency matrix of $G^{(i)}$.
5: Filter out those entries of $\mathbf{A}^{(i)}$ that cannot be put into a perfect matching in the bipartite graph corresponding to $\mathbf{A}^{(i)}$, and let $G^{(i)}$ correspond to the graph of $\mathbf{A}^{(i)}$
6: $[\mathbf{R}^{(i)}] \leftarrow \text{scale}(\mathbf{A}^{(i)})$
7: Let $\mathcal{T} = \{j : a_{1,j}^{(i)} \neq 0 \text{ and } G^{(i)} - \{v_1, v_j\} \text{ has a perfect matching}\}$
8: Pick a random nonzero column $j$ from $\mathcal{T}$ by using the probability density function

$$p_k = \frac{s_{1,k}}{\Sigma_{t \in \mathcal{T}} s_{1,t}}, \text{for all nonzero } a_{1,k}^{(i)} \text{ with } k \in \mathcal{T}$$

where $s_{t,k} = r_t^{(i)} \cdot r_k^{(i)}$ is the corresponding entry in the scaled matrix $\mathbf{S} = \mathbf{R}^{(i)}\mathbf{A}^{(i)}\mathbf{R}^{(i)}$
9: $X_G \leftarrow X_G / p_j$
10: $G^{(i+1)} \leftarrow G^{(i)} - \{v_1, v_j\}$ ▶ delete the vertices $v_1$ and $v_j$ from $G^{(i)}$ to obtain $G^{(i+1)}$

where $\mathcal{T} = \{j : a_{1,j}^{(i)} \neq 0 \text{ and } G^{(i)} - \{v_1, v_j\} \text{ has a perfect matching}\}$, $s_{1,t} = r_1^{(i)} \cdot r_t^{(i)}$, and $0 < \Sigma_{t \in \mathcal{T}} s_{1,t} \leq 1$. Therefore, $p_k \geq r_1^{(i)} \cdot r_k^{(i)}$ at Line 8 of Algorithm 2.

By using a proof similar to that of Theorem 1, we can show the following about the expected value of the estimator returned by Algorithm 2.

**Theorem 5.** *Let $X_G$ be a random variable returned by Algorithm 2. Then $\mathbb{E}[X_G] = M(G)$, where $M(G)$ represents the number of perfect matchings in the graph $G$.*

*Proof.* We prove the claim via induction. As the base-case, we consider $n = 2$ and the argument holds where the adjacency matrix is $\mathbf{A} = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$. Assume that the inductive hypothesis holds for $n - 2$. Let $G_{ij}$ be obtained by removing vertices $v_i$ and $v_j$ from $G$, which corresponds to deleting the rows and the columns $i, j$ of the adjacency matrix of $G$ to form the $(n-2) \times (n-2)$ adjacency matrix $\mathbf{A}_{ij}$ of $G_{ij}$. We have the following:

$$\begin{aligned}
\mathbb{E}[X_G] &= \sum_{j : a_{1,j} \neq 0} p_j \cdot \frac{1}{p_j} \cdot \mathbb{E}[X_{G_{ij}}] \\
&= \sum_{j : a_{1,j} \neq 0} \mathbb{E}[X_{G_{ij}}] \\
&= \sum_{j : a_{1,j} \neq 0} M(G_{ij}) \text{ by the inductive hypothesis} \\
&= M(G)
\end{aligned}$$

□

Next, we show a theorem bounding $\mathbb{E}[X_G^2]$. To achieve this we use the following Lemma.

**Lemma 3.** *Let* $\mathbf{A}$ *be a symmetric* $n \times n$ *doubly stochastically scalable matrix with* $\alpha_{1j} = \alpha_{j1} = 1$ *with all diagonal values zero. Let* $\mathbf{R}$ *be its scaling matrix. Assume we remove the rows and columns with indices* 1 *and* $j$ *from* $\mathbf{A}$*. We then discard the entries that are not in support to obtain* $\mathbf{B}$*, which is a symmetric scalable matrix of size* $n-2$*. Let* $\mathbf{D}$ *be* $\mathbf{B}$*'s scaling matrix, i.e.,* $\mathbf{DBD}$ *is doubly stochastic. Then*

$$\prod_{k=1,k\neq 1,j}^{n} r_k \leq e^{r_1 r_j - 1} \prod_{z=1}^{n-2} d_z \, .$$

*Proof.* The proof is similar to the proof of Lemma 2 and is given in Appendix. □

**Theorem 6.** *Let* $G$ *be an undirected graph,* $\mathbf{A}$ *be the* $n \times n$ *adjacency matrix of* $G$ *with all supported nonzeros, and* $\mathbf{R}$ *be the diagonal matrix which scales* $\mathbf{A}$ *into the doubly stochastic form, e.g.,* $\mathbf{RAR}$ *is doubly stochastic. Then*

$$\mathbb{E}[X_G^2] \leq M(G) \cdot \frac{1}{\prod_i r_i} \, .$$

*Proof.* We prove the theorem by induction. The base case $n = 2$ holds trivially. Let the inductive hypothesis be that the argument holds for graphs with $n-2$ vertices.

$$\mathbb{E}[X_G^2] = \sum_{a_{i,j}\neq 0} p_j \cdot \left( \frac{1}{p_j^2} \cdot \mathbb{E}[X_{G_{ij}}^2] \right)$$

$$\mathbb{E}[X_G^2] = \sum_{a_{i,j}\neq 0} \frac{1}{p_j} \cdot \mathbb{E}[X_{G_{ij}}^2]$$

$$\mathbb{E}[X_G^2] \leq \sum_{a_{i,j}\neq 0} \frac{1}{r_i \cdot r_j} \cdot \mathbb{E}[X_{G_{ij}}^2]$$

because $p_j \geq r_i \cdot r_j$ by the definition $p_j$ at Line 8 of Algorithm 2,

$$\mathbb{E}[X_G^2] \leq \sum_{a_{i,j}\neq 0} \frac{1}{r_i \cdot r_j} \cdot \frac{1}{\prod_z d_z} \cdot M(G_{ij})$$

by the inductive hypothesis, where $\mathbf{DA}_{(ij)}\mathbf{D}$ is doubly stochastic,

where $\mathbf{A}_{(ij)}$ is the filtered adjacency matrix of $G_{ij}$. Hence,

$$\mathbb{E}[X_G^2] \leq \sum_{a_{i,j}\neq 0} \frac{1}{r_i \cdot r_j} \cdot \frac{1}{\prod_{z\neq i, z\neq j}^{n} r_z} \cdot M(G_{ij}) \quad \text{by Lemma 3,}$$

$$\mathbb{E}[X_G^2] \leq \frac{1}{\prod_z r_z} \cdot \sum_{a_{i,j}\neq 0} M(G_{ij})$$

$$\mathbb{E}[X_G^2] \leq \frac{1}{\prod_z r_z} \cdot M(G) \, .$$

□

### 5.1 Filtering out redundant edges

A complication in the undirected case is that to eliminate edges that do not belong to any perfect matching it is not sufficient to only use the Dulmage-Mendelsohn decomposition of $\mathbf{A}$. We demonstrate this with the following fact.

**Fact 7.** *Let $G$ be an undirected graph, and $\mathbf{A}$ be the $n \times n$ adjacency matrix of $G$. Then $Per(\mathbf{A})$ is larger than or equal to $M(G)$.*

*Proof.* The fact holds trivially in the case that $M(G) = 0$. We hence assume in the following that $M(G) > 0$. Let $G'$ be the bipartite graph with $2n$ vertices obtained from $\mathbf{A}$. Then for each perfect matching in $G = (V, E)$, there is a corresponding perfect matching in $G' = (V', E')$. Let $\{(u_1, u_2), \ldots, (u_{n-1}, u_n)\}$ be a perfect matching in $G$ where $u_i \in V$ for $i \in \{1, \ldots, n\}$. Let the vertices in the first and the second part of the bipartite graph $G'$ be denoted as $v_i$s and $w_i$s, respectively, where $v_i \in V'$ and $w_i \in V'$ for $i \in \{1, \ldots, n\}$.

Since the edge $(u_i, u_{i+1})$ is in the matching for $i \in \{1, \ldots, n\}$, it is in $G$. Therefore $a_{i,i+1} = a_{i+1,i} = 1$ are nonzeros in $\mathbf{A}$. Hence, in the bipartite graph $G'$, we have the edges $(v_i, w_{i+1}) \in E'$ and $(v_{i+1}, w_i) \in E'$. From these edges, a perfect matching can be constructed in $G'$.

Thus for each matching in $G$, one can construct a perfect matching in $G'$ and the number of the perfect matchings in $G'$ is equal to $Per(\mathbf{A})$. Hence, $Per(\mathbf{A})$ is at least as large as $M(G)$.

□

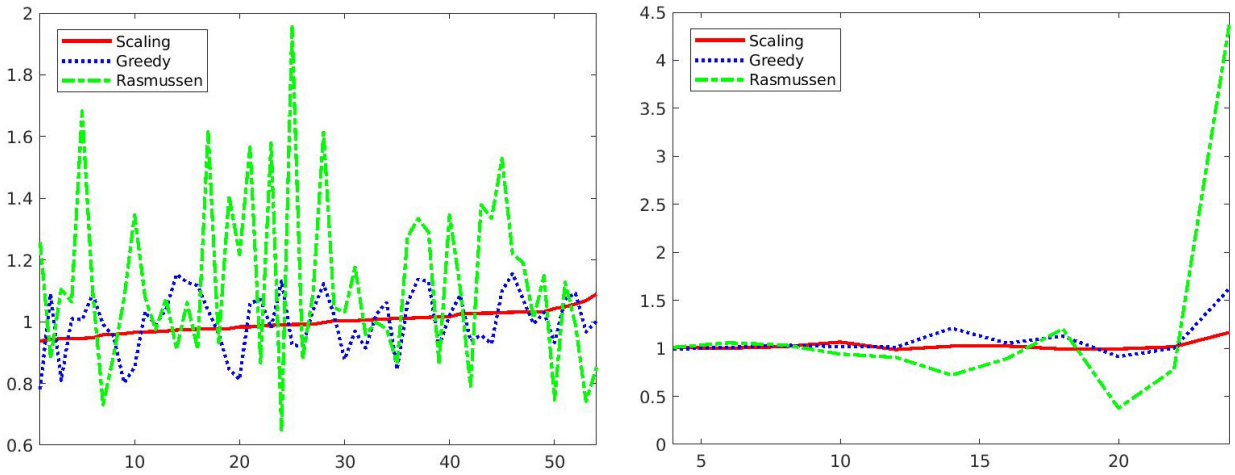
The above fact shows that any off-diagonal edge in the Dulmage-Mendelsohn decomposition of $\mathbf{A}$ cannot belong to a perfect matching in $G$, but it does not help us to eliminate possible edges that do not belong to such symmetrical matchings $(v_i, w_j)$ and $(w_j, v_i)$. These edges may be cleared by using an exact algorithm for computing maximum matchings to detect whether selecting one of them leads to a perfect matching for the remaining vertices or not. A reader may ask if one could use the Gallai-Edmonds [20, Sec. 3.2] decomposition, which partially extends the Dulmage-Mendelsohn decomposition to undirected graphs. This decomposition can be used to find perfectly matchable sub-graphs and odd components if there is no perfect matching in the graph. It does not give any useful information for our purposes in graphs with perfect matchings, while the Dulmage-Mendelsohn decomposition for bipartite graphs states which edges cannot be put into a perfect matching.

## 6 Experiments

The experiments are performed on a machine equipped with an Intel Core i7-7600 CPU and with 16 GB of available ram. For the implementation, we used MATLAB 2017. In the next two subsections, we present the experimental results on bipartite graphs and general, undirected graphs, respectively.

### 6.1 Experiments on bipartite graphs

To see how the proposed algorithm fares in practice on bipartite graphs, we compare it against the original estimator of Rasmussen as well as Greedy. We used an improved version of Rasmussen's randomized algorithm in which edges that do not participate in perfect matchings are discarded. Additionally, all three algorithms process the rows in increasing order of nonzeros, adjusted at each step, to select a random entry. For each test, we take 1000 samples and report their mean.




(a) sparse $40 \times 40$ matrices (b) grid graphs

Figure 1: Comparison of the approximation ratios of the proposed scaling-based method, Rasmussen's and Greedy (a) on 54 random sparse graphs with $n = 40$ and sparsity factor $4/n$ in increasing order of the approximation ratio of the proposed estimator; and (b) on square grids of even length in increasing order of side length.

### 6.1.1 Experiments with random and synthetic datasets

For the first set of bipartite graph experiments, we consider random matrices of size 40 and sparsity $\frac{4}{n}$, in other words there are about 160 nonzeros. We used an exact (exponential time) algorithm to compute the permanents. These matrices can have large permanents (e.g., around $10^7$) and are among the largest an exact non-parallel algorithm, which simply enumerates all the perfect matchings, can handle. The results are summarized in Figure 1a. In this figure, we display the ratio of the estimate to the exact value. As seen in the figure, our approach almost always has good performance. In contrast, Rasmussen's estimator often obtains results that are worse than the other two approaches (even if modified to avoid returning zeros). The Greedy estimator exhibits a better performance than Rasmussen's, while our approach is better than Greedy (it has a smaller and less frequent deviation from the value of the permanent).

To provide results with larger $n$, we focus on the class of matrices which correspond to grids as the second set experiments on bipartite graphs. For these matrices, an exact formula for the permanent is given independently by Kasteleyn [15] and Temperley and Fisher [25]. The number of perfect matchings in an $m \times n$ grid is given by the formula

$$\prod_{j=1}^{m} \prod_{k=1}^{n} \left( 4\cos^2\left(\frac{\pi j}{m+1}\right) + 4\cos^2\left(\frac{\pi k}{n+1}\right) \right)^{1/4} .$$

The results presented in Figure 1b concur with the previous test. We observe that Scaling seems to provide better performance than the two alternatives. This is particularly notable in the last case of the $24 \times 24$ grid where only Scaling manages to obtain an estimate in close range of the actual answer. This suggests that the assignment of probabilities via a doubly stochastic scaling method makes the overall procedure more reliable.

As the third set of experiments bipartite graphs, we give the results for 1000 simulations of the

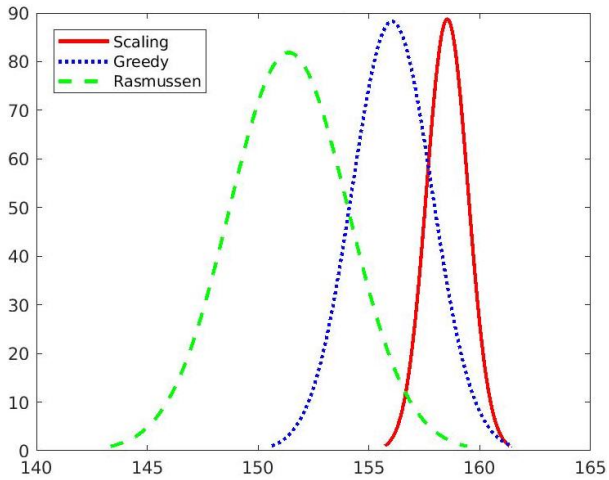



Figure 2: The performance of the estimators on the $36 \times 36$ grid. The logarithms of the 1000 estimator values are presented on the $x$-axes and the $y$-axes demonstrate the distribution of the samples for the values in each approach. We note that the logarithm of the actual permanent is 159.49.

three approaches on the matrix of the $36 \times 36$ grid to examine in detail how the algorithms behave for larger graphs. The results are presented in Figure 2. To draw this figure, we used Matlab's `histfit` command on the 1000 estimates of each algorithm, to plot a histogram and fit a bell curve so as to understand the distribution. As the values are very large, we present the results on a log-scale. In this figure, we observe less variation between the independent runs of the proposed method. Furthermore, the approximation factor of the mean estimate of our approach, 1.11, is significantly better than those of the other two approaches; Greedy's approximation ratio is 0.42 while Rasmussen's is worse than both obtaining a 0.0028 approximation (the approximation ratios are obtained using the exact formula given above).

### 6.1.2 Experiments with real-life bipartite graphs

To further evaluate the practical performance of the proposed approximation scheme, we used a set of matrices from the SuiteSparse Matrix Collection (formerly the University of Florida Sparse Matrix Collection) [5]. The properties of the matrices can be seen in Table 1.

**On real-life matrices with known permanents** For the first set of experiments, we used six small real-life square matrices with $n < 60$. For these matrices, it is possible to calculate the permanent exactly using a recent parallel algorithm [16]. Table 1 additionally exhibits the approximation found by each of the three algorithms after performing 1000 runs and taking the mean. The results are similar to those with synthetic and random matrices. Scaling obtains the best performance, followed by Greedy, and Rasmussen, which has the worst behavior overall. In more detail, Scaling obtained a mean deviation of 2% from the permanent, whereas Greedy had an average deviation of 9%. For `impcolb`, Scaling was 16.1% closer to the permanent than Greedy, which amounted to a difference of $1.7 \times 10^7$ between the two approximations. For `dwt59`, the difference between the answers of Scaling and Greedy was $3.48 \cdot 10^{17}$ in favor of Scaling. Rasmussen's

| Name | $n$ | $nnz$ | permanent | Rasmussen | Greedy | Scaling |
|---|---|---|---|---|---|---|
| Grund/dss | 53 | 144 | $2.93 \cdot 10^5$ | 1.080 | 0.949 | 1.033 |
| DIMACS10/chesapeake | 39 | 340 | $1.32 \cdot 10^{13}$ | 0.302 | 0.918 | 0.993 |
| HB/bcsstk01 | 48 | 400 | $2.01 \cdot 10^{25}$ | 0.973 | 0.966 | 0.993 |
| HB/impcolb | 59 | 271 | $1.11 \cdot 10^{08}$ | 1.027 | 0.824 | 0.985 |
| HB/will57 | 57 | 281 | $1.07 \cdot 10^{18}$ | 1.578 | 0.944 | 1.022 |
| HB/dwt59 | 59 | 267 | $4.64 \cdot 10^{18}$ | 1.367 | 0.877 | 0.952 |
| AG-Monien/netz4504dual | 615 | 2342 | | | | |
| Bai/dw256A | 512 | 2480 | | | | |
| Bai/dw256B | 512 | 2500 | | | | |
| HB/662bus | 662 | 2474 | Permanent is too expensive to compute | | | |
| HB/685bus | 685 | 3249 | | | | |
| JGDHomology/ch5-5-b3 | 600 | 2400 | | | | |
| VDOL/dynamicSoaringProblem1 | 647 | 5367 | | | | |

Table 1: Names, dimensions, and numbers of nonzeros of the real-life square matrices corresponding to bipartite graphs used to evaluate the performance of the estimators. For the first set of six smaller matrices with $n < 60$, the exact permanent value and the approximation of the estimators with 1000 samples are also given.

average deviation from the permanent at 39% was much worse than the other methods.

To demonstrate the effectiveness of Scaling in more detail, Figure 3 presents a histogram of the estimators and fits a bell curve. We additionally show the value of the permanent with a vertical line to display how the estimators are spread around it. As it can be observed from all six subfigures, Scaling's estimations are more closely concentrated around the value of the permanent, and Scaling's samples span the smallest interval.

**On real-life matrices with unknown permanents** As the last set of bipartite graph experiments, we present the performance of estimators with 1000 estimations on seven larger matrices with $n > 500$ from Table 1. For these matrices, we do not know the exact permanent values, and we plot only the estimates in Figure 4. As before, the figures were generated with Matlab's `histfit` command, and the results are given in log-scale. In all cases, we see that the values obtained by Rasmussen's approach span a larger interval than the other two, and Greedy has a larger span than Scaling. We also note that the distributions are similar to those in Figure 3, which is another indication that Scaling most probably obtains better estimates than both Greedy and Rasmussen on these graphs as well. We also present the mean, standard deviation (std), and the std/mean ratio (i.e., coefficient of variation) of the estimators in Table 2. We calculated the standard deviation using the formula $\sigma^2 = \dfrac{\sum_{i=1}^{N}(X_i - \overline{X})^2}{N-1}$ where $N$ represents the number of samples and $X_i$ corresponds to the $i$th sample with $\overline{X}$ being their mean value.

A trend we notice in all matrices is that Rasmussen's algorithm always obtains the smallest mean value, whereas the proposed algorithm returns the largest. Furthermore, the std/mean ratio of Scaling is almost always the smallest of the three (except for the matrix `dw256B`). By combining these observations with the previous ones, we conclude that Scaling's estimations are more concentrated around the returned mean compared to the other two algorithms. In other words, the proposed approach has lesser variation than the other two.

The run time of Scaling on these seven matrices are very close to those of the other two

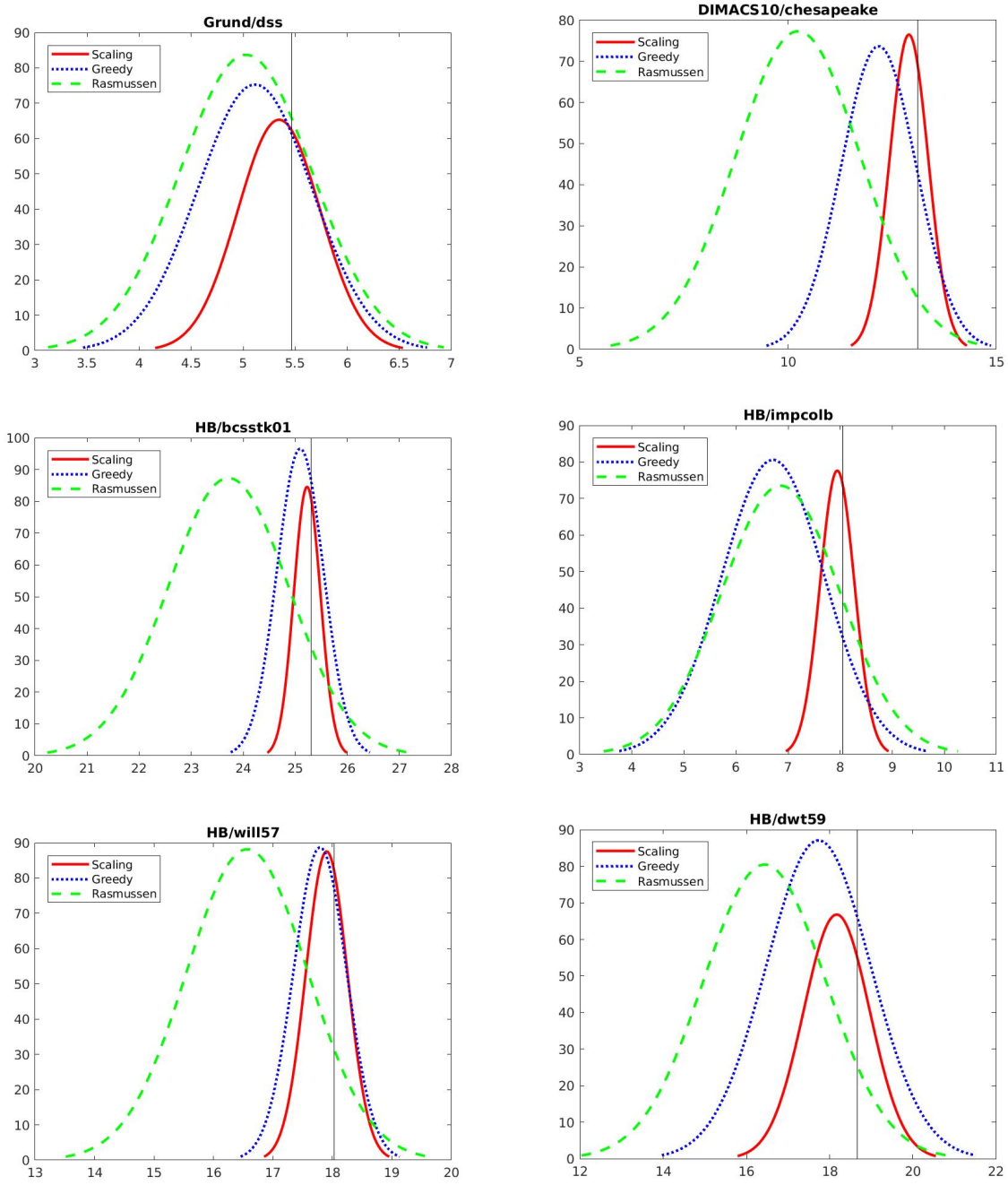



Figure 3: The performance of the estimators on the six matrices with $n < 60$ given in Table 1. The logarithms of the 1000 estimator values are presented on the $x$-axes and the $y$-axes demonstrate the distribution of the samples for the values in each approach. The vertical line corresponds to the actual value of the permanent.

| Matrix | Rasmussen | | | Greedy | | | Scaling | | |
|---|---|---|---|---|---|---|---|---|---|
| | mean | std | $\frac{\text{std}}{\text{mean}}$ | mean | std | $\frac{\text{std}}{\text{mean}}$ | mean | std | $\frac{\text{std}}{\text{mean}}$ |
| netz4504dual | $5.12 \cdot 10^{140}$ | $1.60 \cdot 10^{142}$ | 31.36 | $7.68 \cdot 10^{142}$ | $1.68 \cdot 10^{144}$ | 21.86 | $3.54 \cdot 10^{143}$ | $3.01 \cdot 10^{144}$ | 8.52 |
| dw256A | $3.59 \cdot 10^{158}$ | $1.13 \cdot 10^{160}$ | 31.54 | $2.77 \cdot 10^{162}$ | $3.75 \cdot 10^{163}$ | 13.56 | $2.11 \cdot 10^{164}$ | $2.64 \cdot 10^{165}$ | 12.53 |
| dw256B | $7.14 \cdot 10^{158}$ | $2.05 \cdot 10^{160}$ | 28.76 | $3.86 \cdot 10^{162}$ | $4.49 \cdot 10^{163}$ | 11.63 | $4.02 \cdot 10^{164}$ | $5.29 \cdot 10^{165}$ | 13.15 |
| 662bus | $1.51 \cdot 10^{142}$ | $4.69 \cdot 10^{143}$ | 31.07 | $6.48 \cdot 10^{150}$ | $1.45 \cdot 10^{152}$ | 22.32 | $1.41 \cdot 10^{151}$ | $2.83 \cdot 10^{152}$ | 20.09 |
| 685bus | $2.75 \cdot 10^{187}$ | $7.56 \cdot 10^{188}$ | 27.49 | $1.04 \cdot 10^{203}$ | $2.86 \cdot 10^{204}$ | 27.55 | $1.49 \cdot 10^{202}$ | $2.59 \cdot 10^{203}$ | 17.37 |
| ch5-5-b3 | $1.80 \cdot 10^{136}$ | $3.23 \cdot 10^{137}$ | 17.99 | $3.92 \cdot 10^{137}$ | $7.31 \cdot 10^{138}$ | 18.62 | $5.19 \cdot 10^{138}$ | $6.00 \cdot 10^{139}$ | 11.57 |
| dynamicSoar1 | $8.27 \cdot 10^{254}$ | $2.46 \cdot 10^{256}$ | 29.80 | $1.09 \cdot 10^{259}$ | $2.91 \cdot 10^{260}$ | 26.70 | $3.29 \cdot 10^{267}$ | $4.38 \cdot 10^{268}$ | 13.32 |

Table 2: Statistics for the seven larger bipartite graphs. The original name of the matrix corresponding to the last one is `dynamicSoaringProblem1` and is shortened to fit into the column. The preceding family names of the matrices are not shown.

approaches. The average run time to execute an estimator using Scaling is 0.92 seconds, whereas Rasmussen and Greedy require 0.13 and 0.14 seconds, respectively.

### 6.2 Experiments on general, undirected graphs

We examine the performance of the estimators on five undirected graphs obtained from matrices available in the SuiteSparse Matrix Collection. After downloading the matrices, we made them pattern-wise symmetric by adding all missing symmetric entries. Furthermore, we discarded the values in the diagonal and set all remaining nonzero values to one so that the resulting matrix is the adjacency matrix of an undirected graph. The properties of the resulting graphs are presented in Table 3.

As discussed in Section 5.1, getting rid of the entries that do not belong to any perfect matching is a more time consuming procedure than its equivalent for bipartite graphs. As Fürer and Kasiviswanathan [10] discard all edges that cannot be put into a perfect matching from the graph in the original description of Greedy, we implemented this cleaning for Greedy. For the extension of Rasmussen's approach for undirected graphs, first, we select a row and then we remove any of its entries that do not participate in any perfect matching, so that we always end up with a valid perfect matching. For the proposed scaling-based algorithm we opted to test the following two alternatives:

1. The results labeled *Scaling* in the figures correspond to Algorithm 2. In this version, the Dulmage-Mendelsohn decomposition is used to ensure that the matrix has total support. Then we cleaned only the entries from the selected row.

2. The results labeled with *Clear-Scaling* are obtained by discarding all edges that are not part of some perfect matching, using first the Dulmage-Mendelsohn decomposition, and then by exhaustively testing the remaining edges (due to Fact 7, the resulting matrix has total support). Then we proceed to scale the matrix to obtain the scaling matrix $\mathbf{R}$ and do the appropriate selections. This corresponds to the cleaning performed in Greedy.

We do not know the actual number of perfect matchings on these graphs. As in the previous subsection, we plot the distribution (in logarithmic scale) of the results for 1000 trials. The results are presented in Figure 5. We observe that the two scaling-based alternatives have similar curves, and there does not seem to be an advantage in applying the more expensive method of extensive cleaning. In addition, we observe that our approach seems to minimize the variance in respect to the other alternatives by providing a closer range of reported values.

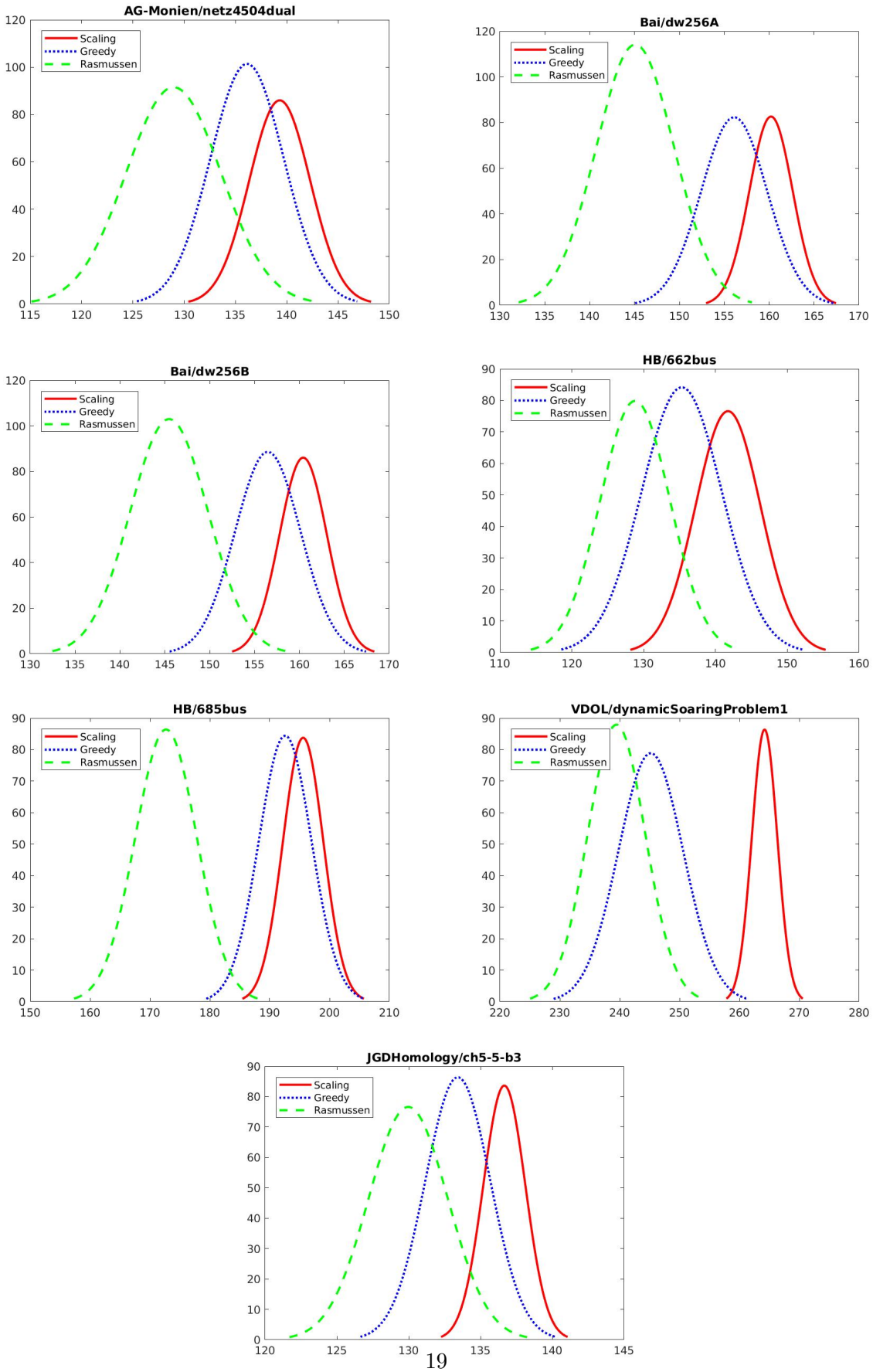





Figure 4: The performance of the estimators on the seven matrices with $n > 500$ given in Table 1. The logarithms of the 1000 estimator values are presented on the $x$-axes and the $y$-axes demonstrate the distribution of the samples for the values in each approach. For these matrices, the permanent values are unknown.

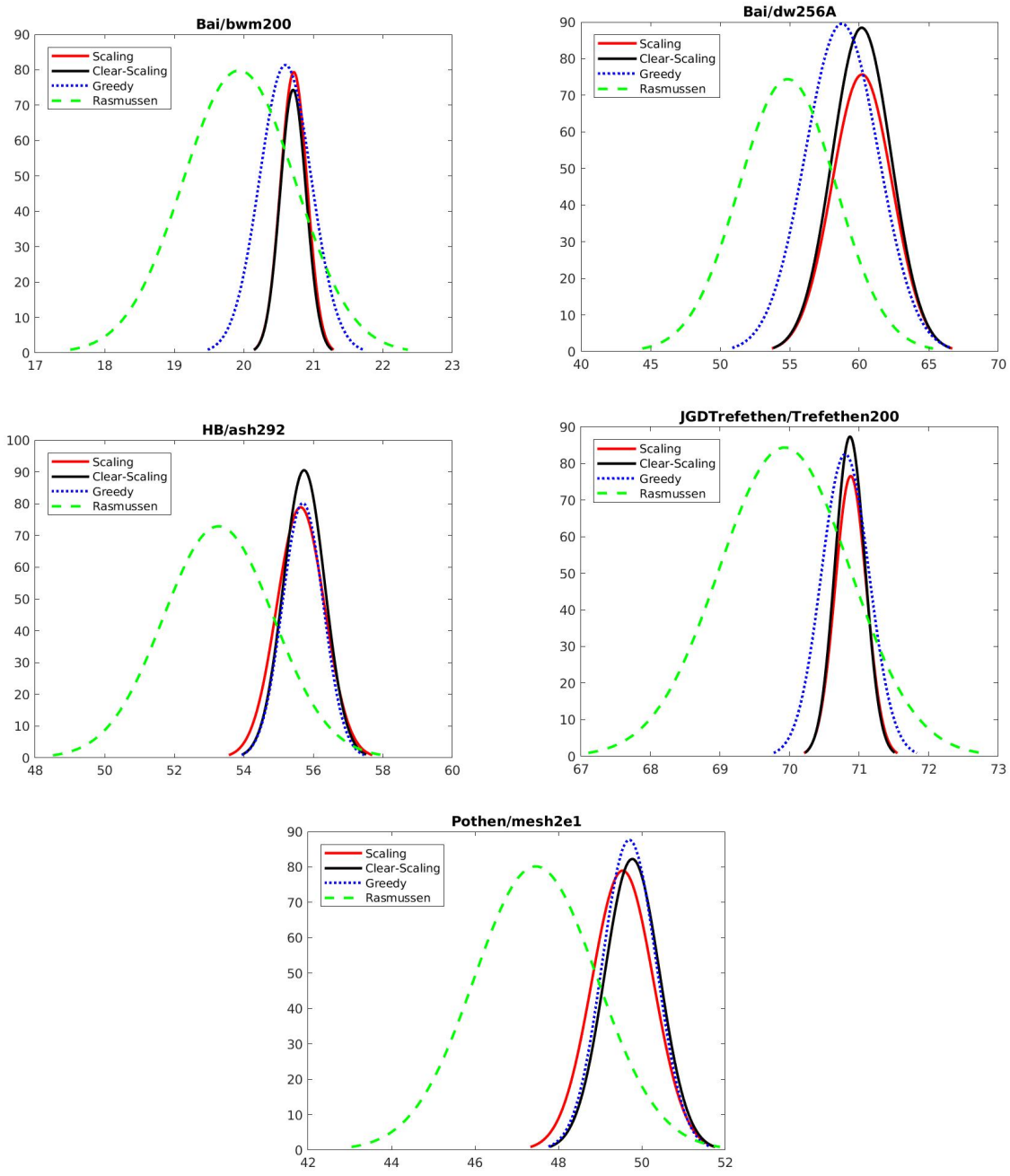



Figure 5: The performance of the estimators on the five undirected graphs with $n < 260$ in Table 3. The logarithms of the 1000 estimator values are presented on the $x$-axes and the $y$-axes demonstrate the distribution of the samples for the values in each approach. For these graphs, the number of perfect matchings are unknown.

| Name | $\|V\|$ | $\|E\|$ |
|---|---|---|
| Bai/bwm200 | 100 | 298 |
| Bai/dw256A | 256 | 1004 |
| HB/ash292 | 146 | 958 |
| JGDTrefethen/Trefethen200 | 100 | 1345 |
| Pothen/mesh2em1 | 153 | 856 |

Table 3: The properties of undirected graphs corresponding to real-life matrices and the names of those matrices.

| | Rasmussen | | | Greedy | | |
|---|---|---|---|---|---|---|
| Matrix | mean | std | $\frac{std}{mean}$ | mean | std | $\frac{std}{mean}$ |
| Bai/bwm200 | $5.44 \cdot 10^{20}$ | $2.71 \cdot 10^{21}$ | 4.98 | $5.74 \cdot 10^{20}$ | $5.84 \cdot 10^{20}$ | 1.02 |
| Bai/dw256A | $1.71 \cdot 10^{61}$ | $3.26 \cdot 10^{62}$ | 19.10 | $2.58 \cdot 10^{62}$ | $2.47 \cdot 10^{63}$ | 9.59 |
| HB/ash292 | $7.23 \cdot 10^{55}$ | $8.73 \cdot 10^{56}$ | 12.07 | $1.22 \cdot 10^{56}$ | $2.26 \cdot 10^{56}$ | 1.86 |
| JGDTrefethen/Trefethen200 | $8.33 \cdot 10^{70}$ | $4.14 \cdot 10^{71}$ | 4.98 | $8.63 \cdot 10^{70}$ | $7.81 \cdot 10^{70}$ | 0.90 |
| Pothen/mesh2e1 | $7.21 \cdot 10^{49}$ | $1.03 \cdot 10^{51}$ | 14.35 | $1.45 \cdot 10^{50}$ | $3.80 \cdot 10^{50}$ | 2.63 |
| | Scaling | | | Clear-Scaling | | |
| | mean | std | $\frac{std}{mean}$ | mean | std | $\frac{std}{mean}$ |
| Bai/bwm200 | $5.81 \cdot 10^{20}$ | $2.81 \cdot 10^{20}$ | 0.48 | $5.65 \cdot 10^{20}$ | $2.62 \cdot 10^{20}$ | 0.46 |
| Bai/dw256A | $6.88 \cdot 10^{62}$ | $4.87 \cdot 10^{63}$ | 7.08 | $3.16 \cdot 10^{62}$ | $2.05 \cdot 10^{63}$ | 6.49 |
| HB/ash292 | $1.43 \cdot 10^{56}$ | $3.66 \cdot 10^{56}$ | 2.56 | $1.32 \cdot 10^{56}$ | $2.77 \cdot 10^{56}$ | 2.10 |
| JGDTrefethen//Trefethen200 | $8.63 \cdot 10^{70}$ | $4.44 \cdot 10^{70}$ | 0.51 | $8.32 \cdot 10^{70}$ | $4.11 \cdot 10^{70}$ | 0.49 |
| Pothen/mesh2e1 | $1.38 \cdot 10^{50}$ | $3.14 \cdot 10^{50}$ | 2.27 | $1.85 \cdot 10^{50}$ | $5.20 \cdot 10^{50}$ | 2.81 |

Table 4: Statistics for the five undirected graphs obtained from the matrices in Table 3.

Finally, for this set of experiments, we provide the means as well as the standard deviations in Table 4 using the same definitions as in the previous subsection. The table shows that once again the proposed algorithm obtains usually the smallest standard deviation to mean ratio. In addition, we can observe that the extensive cleaning of entries can help in reducing this ratio, though only slightly.

Focusing on the run time of the estimators, we have that again Scaling is fast. In these graphs, Rasmussen's algorithm requires 0.05 seconds to calculate an estimate for $M(G)$, whereas an estimator based on Scaling requires 0.32 seconds. The other two methods Greedy and Clear-Scaling are both significantly slower. The increase in the run time is due to their excessive cleaning procedure which at east step eliminates all edges which do not participate in a perfect matching. Greedy requires on average 8.68 seconds while Clear-Scaling requires 8.92 seconds. Since Scaling and Clear-Scaling obtain similar estimators while Scaling being the faster of the two, we suggest avoiding costly cleaning step of Clear-Scaling.

# 7 Conclusion

We have proposed a technique for approximating the permanent and counting perfect matchings in undirected graphs. Our approach uses matrix scaling in order to sample randomly a perfect matching of the graph. At each step of the algorithm, we select a vertex and match it randomly with one of its neighbors. The probabilities upon which we base the selection are obtained by scaling the adjacency matrix of the graph into a doubly stochastic form and examining the entries that correspond to the neighbors of the given vertex. At the end of the algorithm an estimate $X$ is

returned based on the values of the chosen probabilities. This process is repeated a sufficient amount of times and a mean estimate $\overline{X}$ is returned by taking the mean of all returned $X$ values. We proved loose yet computable upper bounds for the expected value of the square of $X$ which affects the number of samples required. The experimental analysis demonstrated improvements over previous similar methodologies. Future work involves bounding the estimates for special graph classes where one can potentially analyse the upper bounds more tightly.

## References


[1] Z. Allen-Zhu, Y. Li, R. Oliveira, A. Wigderson, Much faster algorithms for matrix scaling, arXiv preprint arXiv:1704.02315.

[2] I. Beichl, F. Sullivan, Approximating the permanent via importance sampling with application to the dimer covering problem, J. Comput. Phys. 149 (1) (1999) 128–147.

[3] D. Chakrabarty, S. Khanna, Better and simpler error analysis of the Sinkhorn-Knopp algorithm for matrix scaling, arXiv preprint arXiv:1801.02790.

[4] M. B. Cohen, A. Madry, D. Tsipras, A. Vladu, Matrix scaling and balancing via box constrained Newton's method and interior point methods, arXiv preprint arXiv:1704.02310.

[5] T. A. Davis, Y. Hu, The University of Florida sparse matrix collection, ACM Trans. Math. Softw. 38 (1) (2011) 1:1–1:25.

[6] F. Dufossé, K. Kaya, B. Uçar, Two approximation algorithms for bipartite matching on multicore architectures, J. Parallel Distr. Com. 85 (2015) 62–78.

[7] F. Dufossé, K. Kaya, I. Panagiotas, B. Uçar, Approximation algorithms for maximum matchings in undirected graphs, Proceedings of the 8th SIAM Workshop on Combinatorial Scientific Computing (May 2018).

[8] A. L. Dulmage, N. S. Mendelsohn, Coverings of bipartite graphs, Canad. J. Math. 10 (1958) 517–534.

[9] M. Fürer, S. P. Kasiviswanathan, An almost linear time approximation algorithm for the permanent of a random (0-1) matrix, in: International Conference on Foundations of Software Technology and Theoretical Computer Science, Springer, 2004, pp. 263–274.

[10] M. Fürer, S. P. Kasiviswanathan, Approximately counting perfect matchings in general graphs., in: ALENEX/ANALCO, 2005, pp. 263–272.

[11] L. Gurvits, A. Samorodnitsky, Bounds on the permanent and some applications, arXiv preprint arXiv:1408.0976.

[12] M. Idel, A review of matrix scaling and Sinkhorn's normal form for matrices and positive maps, arXiv preprint arXiv:1609.06349.

[13] M. Jerrum, A. Sinclair, E. Vigoda, A polynomial-time approximation algorithm for the permanent of a matrix with nonnegative entries, J. ACM 51 (4) (2004) 671–697.

[14] B. Kalantari, L. Khachiyan, On the complexity of nonnegative-matrix scaling, Linear Algebra Appl. 240 (1996) 87–103.

[15] P. W. Kasteleyn, The statistics of dimers on a lattice, Physica 27 (1961) 1209–1225.

[16] K. Kaya, Parallel algorithms for computing sparse matrix permanents, Turkish Journal of Electrical Engineering & Computer Sciences 27 (6) (2019) 4284–4297.

[17] P. A. Knight, D. Ruiz, A fast algorithm for matrix balancing, IMA J. Numer. Anal. 33 (3) (2013) 1029–1047.

[18] P. A. Knight, D. Ruiz, B. Uçar, A symmetry preserving algorithm for matrix scaling, SIAM J. Matrix Anal. A. 35 (3) (2014) 931–955.

[19] N. Linial, A. Samorodnitsky, A. Wigderson, A deterministic strongly polynomial algorithm for matrix scaling and approximate permanents, Combinatorica 20 (2000) 545–568.

[20] L. Lovasz, M. D. Plummer, Matching theory, Elsevier Science Publishers, Netherlands, 1986.

[21] A. Pothen, C.-J. Fan, Computing the block triangular form of a sparse matrix, ACM T. Math. Software 16 (1990) 303–324.

[22] L. E. Rasmussen, Approximating the permanent: A simple approach, Random Struct. Algor. 5 (2) (1994) 349–361.

[23] R. Sinkhorn, P. Knopp, Concerning nonnegative matrices and doubly stochastic matrices, Pacific J. Math. 21 (1967) 343–348.

[24] D. Štefankovič, E. Vigoda, J. Wilmes, On counting perfect matchings in general graphs, arXiv preprint arXiv:1712.07504.

[25] H. N. V. Temperley, M. E. Fisher, Dimer problem in statistical mechanics-an exact result, The Philosophical Magazine: A Journal of Theoretical Experimental and Applied Physics 6 (68) (1961) 1061–1063.

[26] L. G. Valiant, The complexity of computing the permanent, Theor. Comput. Sci. 8 (2) (1979) 189–201.

# Appendix

This appendix contains the omitted proofs. For convenience, we also restate the results that are proved here.

**Restatement of Lemma 3.** *Let* $\mathbf{A}$ *be a symmetric* $n \times n$ *doubly stochastically scalable matrix with* $\alpha_{1j} = \alpha_{j1} = 1$ *with all diagonal values zero. Let* $\mathbf{R}$ *be its scaling matrix. Assume we remove the rows and columns with indices* 1 *and* $j$ *from* $\mathbf{A}$*. We then discard the entries that are not in support to obtain* $\mathbf{B}$*, which is a symmetric scalable matrix of size* $n-2$*. Let* $\mathbf{D}$ *be* $\mathbf{B}$*'s scaling matrix, i.e.,* $\mathbf{DBD}$ *is doubly stochastic. Then*

$$\prod_{k=1,k\neq 1,j}^{n} r_k \leq e^{r_1 r_j - 1} \prod_{z=1}^{n-2} d_z \ .$$

*Proof of Lemma 3.* The result is obtained by applying the function $g_{\mathbf{B}}(\mathbf{x},\mathbf{y}) = \mathbf{x}^T\mathbf{B}\mathbf{y} - \sum_{i=1}^{n} \ln x_i - \sum_{j=1}^{n} \ln y_j$ shown in (1) as in the proof of Lemma 2. Let $\mathbf{r}'$ be $[r_2, \ldots, r_{j-1}, r_{j+1}, \ldots, r_n]^T$.

$$g_{\mathbf{B}}(\mathbf{d},\mathbf{d}) \leq g_{\mathbf{B}}(\mathbf{r}',\mathbf{r}')$$

$$\mathbf{d}^T\mathbf{B}\mathbf{d} - 2 \cdot \sum_{z=1}^{n-2} \ln d_z \leq \mathbf{r}'^T\mathbf{B}\mathbf{r}' - 2 \cdot \sum_{z=1}^{n-2} \ln r'_z$$

We have

$$\mathbf{d}^T\mathbf{B}\mathbf{d} = n - 2 \ .$$

and

$$\mathbf{r}'^T\mathbf{B}\mathbf{r}' \leq n - 4 + 2 \cdot r_1 \cdot r_j.$$

$$n - 2 - 2 \cdot \sum_{z=1}^{n-2} \ln d_z \leq n - 2 - 2 + 2 \cdot (r_1 \cdot r_j) - 2 \cdot \sum_{z=1}^{n-2} \ln r'_z$$

$$-\sum_{z=1}^{n-2} \ln d_z \leq -1 + r_1 \cdot r_j - \sum_{z=1}^{n-2} \ln r'_z$$

The result then follows by taking the exponent in both sides. □

**Restatement of Theorem 4.** Let $\mathbf{A}$ be $n \times n$ matrix with total support, and $\mathbf{RAC}$ be its doubly stochastic scaling. There exists a positive vector $\mathbf{k}$ such that $\mathbb{E}[X_{\mathbf{A}}^2] \leq \dfrac{e^{v-n}}{\prod_i r_i \cdot c_i} \cdot Per(\mathbf{A})$, where $v = \sum_i k_i = \|\mathbf{k}\|_1 \leq n$.

*Proof.* We will repeat the proof of Theorem 2 and precise how we can build $\mathbf{k}$. The base case $n = 1$ holds trivially with $k_1 = 1$. We then have

$$\mathbb{E}[X_{\mathbf{A}}^2] = \sum_{a_{1,j}\neq 0} r_1 \cdot c_j \cdot \left( \frac{1}{r_1^2 \cdot c_j^2} \cdot \mathbb{E}[X_{\mathbf{A}_{1j}}^2] \right)$$

$$\mathbb{E}[X_{\mathbf{A}}^2] = \sum_{a_{1,j}\neq 0} \frac{1}{r_1 \cdot c_j} \cdot \mathbb{E}[X_{\mathbf{A}_{1j}}^2]$$

$$\mathbb{E}[X_{\mathbf{A}}^2] \leq \sum_{a_{1,j}\neq 0} \frac{1}{r_1 \cdot c_j} \cdot \frac{e^{v^{(j)}-(n-1)}}{\prod_z d_z \cdot e_z} \cdot Per(\mathbf{A}_{1j})$$

by the inductive hypothesis, where $\mathbf{D}$ and $\mathbf{E}$ scale $\mathbf{A}_{1j}$,
$\mathbf{k}^{(j)}$ is the vector associated with $\mathbf{A}_{1j}$, and $v^{(j)} = \|\mathbf{k}^{(j)}\|$. Hence,

$$\mathbb{E}[X_{\mathbf{A}}^2] \leq \sum_{a_{1,j}\neq 0} \frac{1}{r_1 \cdot c_j} e^{r_1 c_j - 1} \cdot \frac{e^{v^{(j)}-(n-1)}}{\prod_{z=2}^{n} r_z \cdot \prod_{z=1, z\neq j}^{n} c_z} \cdot Per(\mathbf{A}_{1j}) \quad \text{by Lemma 2.}$$

To define $\mathbf{k}$, let $k_1 = \max_j r_1 c_j$ and $k_\ell = \max_j \mathbf{k}_{\ell-1}^{(j)}$ for $\ell = 2, \ldots, n$,
and let $v = \sum_i k_i$ . Then,

$$\mathbb{E}[X_{\mathbf{A}}^2] \leq \frac{e^{v-n}}{\prod_i r_i \cdot c_i} \cdot \sum_{a_{1,j}\neq 0} Per(\mathbf{A}_{1j})$$

$$\mathbb{E}[X_{\mathbf{A}}^2] \leq \frac{e^{v-n}}{\prod_i r_i \cdot c_i} \cdot Per(\mathbf{A}) \; .$$

Since each entry used in defining $\mathbf{k}$ is no larger than one, $v = \sum_i k_i = \|\mathbf{k}\|_1 \leq n$ holds, hence the proof. □