Scaling matrices and counting the perfect matchings in graphs

Fanny Dufossé, Kamer Kaya, Ioannis Panagiotas, Bora Uçar

To cite this version:

Fanny Dufossé, Kamer Kaya, Ioannis Panagiotas, Bora Uçar. Scaling matrices and counting the perfect matchings in graphs. [Research Report] RR-9161, Inria Grenoble Rhône-Alpes. 2018, pp.1-22. <hal-01743802v5>

HAL Id: hal-01743802

https://hal.inria.fr/hal-01743802v5

Submitted on 8 May 2018

Scaling matrices and counting the perfect matchings in graphs

Fanny Dufossé, Kamer Kaya,
Ioannis Panagiotas, Bora Uçar

RESEARCH REPORT
N° 9161
March 2018
Project-Team ROMA

ISSN 0249-6399 ISRN INRIA/RR--9161--FR+ENG

Scaling matrices and counting the perfect matchings in graphs

Fanny Dufossé*, Kamer Kaya†,

Ioannis Panagiotas‡, Bora Uçar§

Project-Team ROMA

Research Report n° 9161 — March 2018 — 22 pages

Abstract: We investigate efficient randomized methods for approximating the number of perfect matchings in bipartite graphs and general graphs. Our approach is based on assigning probabilities to edges.

Key-words: Permanent approximation, doubly stochastic matrices, Sinkhorn-Knopp scaling, randomized algorithms.

* Inria Grenoble, Rhône Alpes, 38330, Montbonnot-Saint-Martin, France.
† Faculty of Engineering and Natural Sciences, Sabanci University, İstanbul, Turkey.
‡ ENS Lyon, 46, allée d'Italie, ENS Lyon, Lyon F-69364, France.
§ CNRS and LIP (UMR5668 Université de Lyon - CNRS - ENS Lyon - Inria - UCBL 1), 46, allée d'Italie, ENS Lyon, Lyon F-69364, France.

Normalisation de matrice et dénombrement des couplages parfaits dans les graphes

Résumé : Nous étudions des méthodes randomisées efficaces pour approximer le nombre de couplages parfaits dans des graphes bipartis et des graphes généraux. Notre approche se base sur l'assignation de probabilités aux arêtes

Mots-clés : matrices bistochastiques, permanent, algorithmes randomisés

1 Introduction

We investigate efficient randomized methods for approximating the number of perfect matchings in bipartite graphs and general 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. For the bipartite case, a given bipartite graph with n vertices on both sides corresponds to 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. For the general (undirected) case, given a graph with n vertices, the adjacency matrix is considered to be a symmetric matrix where $a_{ij} = a_{ji} = 1$ if there is an edge between vertices i and j. Previously, we have shown that such scaling algorithms are useful in approximating the maximum cardinality of matchings in bipartite graphs [6] and general graphs [7].

Counting perfect matchings in a bipartite graph is equivalent to computing the permanent of its adjacency matrix, which is defined as $Per(\mathbf{A}) = \sum_{\sigma} \prod a_{i,\sigma(i)}$, where the summation runs over all permutations σ of $1, \ldots, n$. While we focus on the case where $\mathbf{A}$ is a 0-1 matrix, our techniques are applicable to the weighted case. Approximating the permanent is a well-studied problem. Valiant [24] showed the problem to be # P-Complete. Jerrum et al. [13] discuss an approach using Markov Chains which can provide an $(1 + \epsilon)$-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. [23] take the MCMC approach and highlight the difficulties that arise. They also propose a Markov Chain efficiently estimating the number of perfect matchings in a special graph class. Gurvits and Samorodnitsky [11] and Linial et al. [18] have used matrix scaling and proposed deterministic approximations with exponential guarantees (2^n and e^n respectively).

Our contribution is more on the practical side, as we describe an algorithm with significantly reduced complexity. We propose an alternative selection mechanism to the original algorithm due to Rasmussen [20]. The same approach has also been studied by Beichl and Sullivan [2], but we offer some new insights as well as a more detailed experimental analysis.

The rest of the report is organized as follows: Section 2 introduces the notation and the background on scaling and the permanent, and Section 3 introduces our idea and the pseudocode of the main approach for bipartite graphs. The extension to the general, undirected case is presented in Section 4. Section 5 presents the experimental results, and Section 4.2 briefly discusses related work. Section 6 concludes the report.

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 ith row and the jth 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 ith row and jth column. The column ids of the nonzeros in the ith 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 are trying to measure. In our situation, P is equivalent to the permanent. Assume that we have created 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 ith estimator, $X' = \sum_{i=1}^{N} \frac{X_i}{N}$ is the combined estimate for P.

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

2.1 Scaling to doubly stochastic form

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 all the non-zeros belong to at least one perfect matching then the matrix has *total support.*

Any nonnegative matrix $\mathbf{A}$ with total support can be scaled with two (unique) 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). The Sinkhorn-Knopp algorithm [21] 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 [16, 17].

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 $\mathbf{x} = \text{diag}(\mathbf{R})$ and $\mathbf{y} = \text{diag}(\mathbf{C})$ [14, Proposition 2]. In particular, an $n \times n$ 0-1 matrix $\mathbf{A}$ has $n \leq g_{\mathbf{A}}(\mathbf{x}, \mathbf{y}) \leq n + n \ln n$, 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].

We have the following lemma, highlighting a close connection with the scaling factors and the permanents of a matrix and its minors.

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. □

3 Algorithm and its Analysis

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, and produces a random variable denoted as $X_\mathbf{A}$. 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 ith 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 matching edge for the ith row. 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 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, 19] decomposition, and filtering out the entries that fall into the off-diagonal blocks in a fine decomposition [19].

Algorithm 1 PERMANENT ESTIMATION

```
Input  $n \times n$, 0-1 matrix $\mathbf{A}$
Output  Permanent estimate $X_\mathbf{A}$
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:     $[\mathbf{R}^{(\sigma,i)}, \mathbf{C}^{(\sigma,i)}] \leftarrow \textsc{Scale}(\mathbf{A}^{(i)})$
6:     Pick a random nonzero column $j \in \mathbf{A}^{(i)}(1,:)$ by using the probability density function
           $p_j = \frac{s_{1,j}}{\sum_{k\in\mathbf{A}^{(i)}(1,:)} s_{1,k}}$ for all nonzeros $a^{(i)}_{1,j}$
       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:     $\mathbf{A}^{(i+1)} \leftarrow \mathbf{A}^{(i)}_{1j}$    ▶ delete the first row and the $j$th column of $\mathbf{A}^{(i)}$
```

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 [19]. There are different algorithms for scaling; see the survey by Idel [12], and more recent papers [1, 3, 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 ε is the allowable deviation from one. Other easy to implement variants are given elsewhere [16, 17]. 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 ε. In our experience [6, 7], Sinkhorn-Knopp kind of algorithms work well for scaling 0-1 matrices for all computational (or 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. Let $\mathbf{R}^{(\sigma,i)}$ and $\mathbf{C}^{(\sigma,i)}$ denote the scaling matrices at the ith step of the algorithm, where the perfect matching σ is returned. Let also $\sigma_1^{(i)}$ denote the column chosen for the first row at the ith step. With this notation, we can write

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

Theorem 2. *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 this by induction; As the base case, for $n = 1$, the argument holds. As the inductive hypothesis, assume that the argument holds for $(n-1) \times (n-1)$ matrices. We have the following:

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

□

Lemma 3. *Let $\mathbf{A}$ be $n \times n$ matrix such that $a_{1,j} = 1$, $\mathbf{B}$ be the $(n-1) \times (n-1)$ submatrix $\mathbf{A}_{1j}$ obtained by deleting the 1st row and the jth column of $\mathbf{A}$, and by 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 scaling $\mathbf{A}$ and $\mathbf{B}$ to the doubly stochastic form ($\mathbf{RAC}$ 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

$$\mathbf{d}^T\mathbf{B}\mathbf{f} - \mathbf{r}'^T\mathbf{B}\mathbf{c}' \leq \sum_{i=1}^{n-1} \ln d_i + \sum_{i=1}^{n-1} \ln f_i - \sum_{i=1}^{n-1} \ln r'_i - \sum_{i=1}^{n-1} \ln c'_i. \tag{2}$$

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

$$\mathbf{d}^T\mathbf{B}\mathbf{f} = n - 1\ . \tag{3}$$

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

$$\mathbf{r}'^T\mathbf{B}\mathbf{c}' \leq \mathbf{r}'^T\mathbf{A}_{1j}\mathbf{c}' = n - 2 + r_1 \cdot c_j. \tag{4}$$

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

$$0 \leq 1 - r_1 \cdot c_1 \leq \mathbf{d}^T\mathbf{B}\mathbf{f} - \mathbf{r}'^T\mathbf{B}\mathbf{c}'.$$

Combining this with (2) and taking exp of all parts, we have

$$1 \leq e^{1 - r_1 \cdot c_j} \leq e^{\mathbf{d}^T\mathbf{B}\mathbf{f} - \mathbf{r}'^T\mathbf{B}\mathbf{c}'} \leq \frac{\prod_{i=1}^{n-1} d_i \cdot f_i}{\prod_{i=1}^{n-1} r'_i \cdot c'_i}, \tag{5}$$

and this concludes the proof. □

Corollary 4. *Let* $\mathbf{A}$ *be a symmetric* $n \times n$ *doubly stochastically scalable matrix with* $\alpha_{1j} = \alpha_{j1} = 1$ *with all diagonal values zero and* $\mathbf{R}$ *be its scaling matrix. Assume we remove from* $\mathbf{A}$ *the rows and columns* $1, j$*, and discard entries that are not in support to obtain* $\mathbf{B}$*, a symmetric scalable matrix of size* $n - 2$*, where* $\mathbf{D}$ *is its scaling matrix. 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 result is obtained as above by applying function g on matrix $\mathbf{B}$. Let $\mathbf{r}'$ be the vector $\mathbf{r}' = [r_2, \ldots, r_{j-1}, r_j, \ldots, r_n]^T$.

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

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

We have

$$\mathbf{d}^T\mathbf{B}\mathbf{d} = n - 2\ . \tag{6}$$

and

$$\mathbf{r}'^T\mathbf{B}\mathbf{r}' \leq n - 4 + 2 \cdot r_1 \cdot r_j. \tag{7}$$

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

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

The result then follows by taking the exponent in either side.

□

Theorem 5. *Let* $\mathbf{A}$ *be* $n \times n$ *matrix with total support, and* $\mathbf{RAC}$ *be its doubly stochastic scaling. Then* $\mathbb{E}[X_{\mathbf{A}}^2] \leq \frac{1}{\prod_i r_i \cdot c_i} \cdot Per(\mathbf{A})$.

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

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

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

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

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

$$\mathbb{E}[X_{\mathbf{A}}^2] \leq \sum_{a_{1,j} \neq 0} \frac{1}{r_1 \cdot c_j} \cdot \frac{1}{\prod_{z=2}^n r_z \cdot \prod_{z=1, z \neq j}^n c_z} \cdot Per(\mathbf{A}_{1j}) \quad \text{by Lemma 3,}$$

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

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

□

We can also propose a theorem which uses mean values to provide with an upper bound

Theorem 6. $\mathbb{E}[X_{\mathbf{A}}^2] \leq Per(\mathbf{A}) \cdot mean \left(\sum_\sigma e^{\sum_j {r_1}^{(\sigma,j)} c_{\sigma_1^{(j)}}^{(\sigma,j)}} \right) \cdot \frac{e^{-n}}{\prod_i r_i \cdot c_i}$.

Proof. Consider any perfect matching σ of $\mathbf{A}$. Let $X_{\mathbf{A}_\sigma}^2 = \frac{1}{\prod_{i=1}^n r_1^{(\sigma,i)} c_{\sigma_1^{(i)}}^{(\sigma,i)}}$ be its contribution to $\mathbb{E}[X_{\mathbf{A}}^2]$, which is equivalent to the value of the random variable $X_{\mathbf{A}}$ should this matching be returned.

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

$$\prod_{j=i}^{n} \frac{1}{r_1^{(\sigma,j)} c_{\sigma_1^{(j)}}^{(\sigma,j)}} \leq \frac{e^{\left(\sum_{j=i}^n {r_1}^{(\sigma,j)} c_{\sigma_1^{(j)}}^{(\sigma,j)} \right) - n - 1 + i}}{\prod_{j=i}^n r_{j-i+1}^{(\sigma,i)} c_{\sigma_1^{(j)}}^{(\sigma,i)}} . \tag{8}$$

That is we provide a relation for the scaling matrices $\mathbf{R}^{(\sigma,i)}$ and $\mathbf{C}^{(\sigma,i)}$ of the ith 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 ith step, row i occupies the position 1. The idea resembles the proof of Theorem 5 except that here a tighter upper bound is provided by having the numerator less than one.

As the base case of the induction, for $i = n$ it holds. 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 ith 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. Therefore, we can replace $\frac{1}{\prod_{j=i}^{n} r_{j-i+1}^{(\sigma,i)} c_{\sigma_1^{(j)}}^{(\sigma,i)}}$ with $\frac{e^{\left({r_1}^{(\sigma,i-1)} c_{\sigma_1^{(i-1)}}^{(\sigma,i-1)}\right)-1}}{\prod_{j=i}^{n} r_{j-(i-1)+1}^{(\sigma,i-1)} c_{\sigma_1^{(j)}}^{(\sigma,i-1)}}$ using Lemma 3 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 (8), as for $j = i-1$, we have $j-i+2 = 1$. Thus for $i = 1$ we get the following:

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

Since

$$\mathbb{E}[X_{\mathbf{A}}^2] = \sum_{\sigma} X_{\mathbf{A}_\sigma}^2, \quad \text{we get that}$$

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

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

Theorem 7. *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* $\mathbb{E}[X_{\mathbf{A}}^2] \leq \frac{e^{v-n}}{\prod_i r_i \cdot c_i} \cdot Per(A)$*, where* $v = \sum_i k_i = \|\mathbf{k}\|_1 \leq n$.

Proof. We will repeat the proof of Theorem 5 and precise how we can build $\mathbf{k}$. The base case $n = 1$ holds trivially with $k_1 = 1$. We then have

$$\begin{aligned}
\mathbb{E}[X_{\mathbf{A}}^2] &= \sum_{a_{1,j} \neq 0} r_1 \cdot c_j \cdot \left(\frac{1}{r_1^2 \cdot c_j^2} \cdot \mathbb{E}[X_{\mathbf{A}_{1j}}^2] \right) \\
\mathbb{E}[X_{\mathbf{A}}^2] &= \sum_{a_{1,j} \neq 0} \frac{1}{r_1 \cdot c_j} \cdot \mathbb{E}[X_{\mathbf{A}_{1j}}^2] \\
\mathbb{E}[X_{\mathbf{A}}^2] &\leq \sum_{a_{1,j} \neq 0} \frac{1}{r_1 \cdot c_j} \cdot \frac{e^{v^{(j)}-(n-1)}}{\prod_z d_z \cdot e_z} \cdot Per(\mathbf{A}_{1j}) \\
&\quad \text{by the inductive hypothesis, where } \mathbf{D} \text{ and } \mathbf{E} \text{ scale } \mathbf{A}_{1j}, \\
&\quad \mathbf{k}^{(j)} \text{ is the vector associated with } \mathbf{A}_{1j}, \text{ and } v^{(j)} = \|\mathbf{k}^{(j)}\| \\
\mathbb{E}[X_{\mathbf{A}}^2] &\leq \sum_{a_{1,j} \neq 0} \frac{1}{r_1 \cdot c_j} e^{r_1 c_j - 1} \cdot \frac{e^{v^{(j)}-(n-1)}}{\prod_{z=2}^n r_z \cdot \prod_{z=1, z \neq j}^n c_z} \cdot Per(\mathbf{A}_{1j}) \quad \text{by Lemma 3,} \\
&\quad \text{let } k_1 = \max_j r_1 c_j \text{ and } k_\ell = \max_j \mathbf{k}_{\ell-1}^{(j)} \text{ for } \ell = 2, \ldots, n \text{ to define } \mathbf{k} \\
\mathbb{E}[X_{\mathbf{A}}^2] &\leq \frac{e^{v-n}}{\prod_i r_i \cdot c_i} \cdot \sum_{a_{1,j} \neq 0} Per(\mathbf{A}_{1j}) \\
\mathbb{E}[X_{\mathbf{A}}^2] &\leq \frac{e^{v-n}}{\prod_i r_i \cdot c_i} \cdot Per(\mathbf{A}) \,.
\end{aligned}$$

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

Note that Theorem 7 implies Theorem 5 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 σ 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.

3.1 Related work

Similar algorithms exist in the literature: Rasmussen [20] investigates a simpler approach in which the selection probabilities are uniform (no scaling), and no edges are discarded (so that the estimator returns zero in many cases). He shows that the estimator X_S is unbiased and also obtains the bound $\mathbb{E}[X_S]^2 \leq Per(\mathbf{A})^2 n!$, although $Per(\mathbf{A})^2$ can be reduced to $Per(\mathbf{A})$. Beichl and Sullivan [2] proposed the same algorithm and performed some analysis, where the $\mathbb{E}[X_{\mathbf{A}}^2]$ is bound using a notation to denote the average value of

$$\frac{1}{\prod_{i=1}^{n} r_1^{(\sigma,i)} \cdot c_{\sigma_1^{(i)}}^{(\sigma,i)}}$$

over all perfect matchings. This analysis is valuable in the sense to show that scaling helps, but it does not yield computable bounds. On the other hand, our analysis in Theorem 5 gives efficiently computable bounds which are loose and we believe can be extended.

4 An estimator for undirected graphs

Having applied the doubly stochastic sampling on bipartite graphs in the previous section, here we investigate how to extend our approach for the general case. Our aim is to count the number of perfect matchings in undirected graphs. This variant of the algorithm has 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

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

where $r_{i,i}$s are the diagonal entries 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.

Algorithm 2 Permanent Estimation

```
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 $\mathcal{T} = \{j : a_{1,j}^{(i)} \neq 0$ and $G^{(i)} - \{v_1, v_j\}$ has a perfect matching$\}$
 8:     Pick a random nonzero column $j$ from $\mathcal{T}$ by using the probability density function

            $p_k = \frac{s_{1,k}}{\sum_{t \in \mathcal{T}} s_{1,t}}$, for all nonzero $a_{1,k}^{(i)}$ 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 $G^{(i+1)}$
```

At the beginning of an 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 traversal; 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

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

where $\mathcal{T} = \{j : a^{(i)}_{1,j} \neq 0 \text{ and } G^{(i)} - \{v_1, v_j\} \text{ has a perfect matching}\}$, $s_{1,t} = r^{(i)}_1 \cdot r^{(i)}_t$, and $0 < \Sigma_{t\in\mathcal{T}} s_{1,t} \leq 1$. Therefore, $p_k \geq r^{(i)}_1 \cdot r^{(i)}_k$ at Line 8 of Algorithm 2.

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

Theorem 8. *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 $\mathbb{E}[X_G^2]$.

Theorem 9. *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.,* **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 $\mathbf{R}_{ij}\mathbf{A}_{ij}\mathbf{R}_{ij}$ is doubly stochastic

$$\mathbb{E}[X_G^2] \leq \sum_{a_{i,j}\neq 0} \frac{1}{r_i \cdot r_j} \cdot \frac{1}{\prod_{z\neq i, z\neq j}^{n} r_z} \cdot M(G_{ij}) \quad \text{by Corollary 4,}$$

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

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

□

Note that as in the bipartite case, we have the product of the inverse of the scaling entries for each vertex in the formula.

4.1 Filtering out redundant edges

A complication for the undirected case is that eliminating the edges that are not in any perfect matching no longer works. We demonstrate this by using the fact below:

Fact 10. *Let G be an undirected graph, $\mathbf{A}$ be the $n \times n$ adjacency matrix of G with all supported nonzeros. Let G' be the bipartite graph with $2n$ vertices obtained from $\mathbf{A}$. Then $Per(\mathbf{A})$ is larger than or equal to the number of perfect matchings in G.*

Proof. 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 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' are denoted as v_is and w_is, 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 $a_{i,i+1} = a_{i+1,i+1} = 1$ are nonzeros in $\mathbf{A}$. Therefore 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'.

Therefore 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 larger than or equal to the number of perfect matchings in G.

□

Although the above fact ensures that any off-diagonal edge in the Dulmage-Mendelsohn decomposition of $\mathbf{A}$ cannot belong to a perfect matching in G, it does not help us eliminate edges that do not belong to such symmetrical matchings (v_i, w_j) and (w_j, v_i).

4.2 Related work

Fürer and Kasiviswanathan [10] discuss three randomized algorithms (Simple, REP, and Greedy) similar to ours. 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 k 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 have also proposed 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. They conclude that Simple is easy to analyze but has with high worst case bound; Greedy looks good on many graphs but difficult to analyze; and REP is the fastest for random (Erdös-Renyi) graphs with again a high worst case bound. If Greedy were to chose 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. Our approach can be seen as more complicated variant of this approach, yet its analysis seems simpler thanks to the minimization properties of the scaling factors of the function $g_{\mathbf{A}}(x, y)$ shown in (1).

5 Experiments

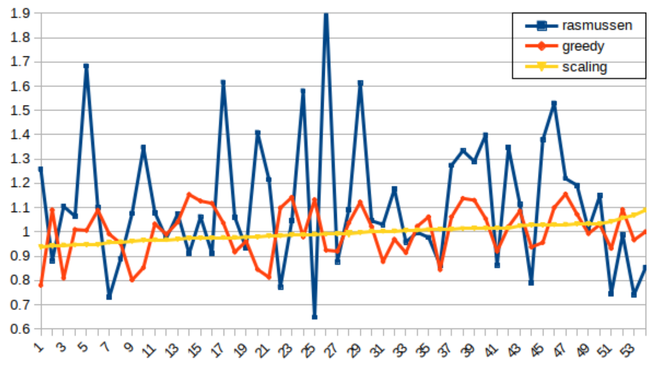
The experiments are performed on a machine equipped with an Intel Core i7-7600 CPU and with 16 GB of ram. We used MATLAB 2017.

5.1 Experiments on bipartite graphs

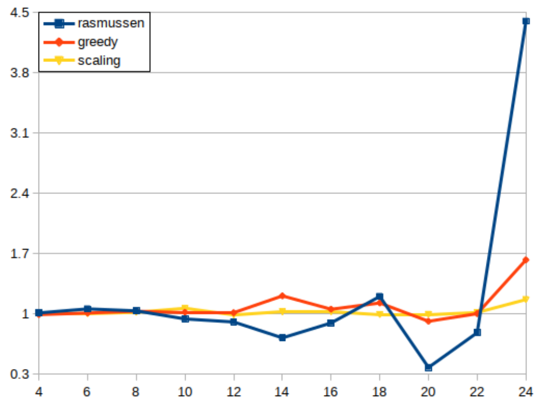
To see how the proposed heuristic fares in practice on bipartite graphs, we compare it against the original estimator of Rasmussen as well as Greedy. We used an improved version of Rasmussen's randomized algorithm in which edges that do not participate in perfect matchings are discarded. Additionally, all three variants in this paper use a small heuristic and select the row with the least nonzeros remaining at each step. For each test, we take 1000 samples and report their mean.

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. Such matrices can have large permanents (e.g., around 10^7) and are among the largest an exact algorithm can handle. The results which are summarized in Figure 1(a) showcase that our approach almost always has good performance. In contrast, Rasmussen's estimator often reports results that are a lot worse than the other two approaches (even if modified to avoid returning zeros). The Greedy heuristic exhibits a better performance than Rasmussen's, while our approach is better than Greedy (it has smaller and less frequent deviation from the value of the permanent).

To provide results with larger n, for the second set of bipartite graph experiments, we focus on the class of matrices which correspond to grids. For these matrices, an exact formula for the



(a) sparse 40×40 matrices



(b) grid graphs

Figure 1: The approximation ratios of the three approaches in 54 random graphs with $n = 40$ and sparsity factor $4/n$ in increasing order of the approximation ratio of our estimator (top). The approximation ratios on square grids of even length in increasing order of the length (bottom).

permanent is given by independently by Kasteleyn [15] and Temperley and Fisher [22]; 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 1(b) concur with the previous test, suggesting that the assignment of probabilities via scaling makes the overall procedure more reliable.

As the third set of bipartite graph experiments, we provide the results for 1000 simulations of the three estimators on a 36×36 grid to examine in detail how the three algorithms behave for larger graphs. The results are presented in Figure 2. 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 just a 0.0028 approximation.

As the last set of bipartite graph experiments, we present results on seven matrices downloaded

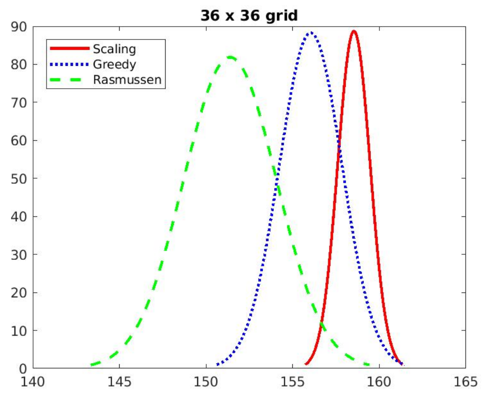


Figure 2: Result of all estimators on the matrix corresponding to a large 36×36 grid. The values of the x axis correspond to the logarithm of returned estimation values while those in the y axis give the population of samples corresponding to each method. We note that the logarithm of the actual permanent is 159.49.

Name	n	nnz
AG-Monien/netz4504dual	615	2342
Bai/dw256A	512	2480
Bai/dw256B	512	2500
HB/662bus	662	2474
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.

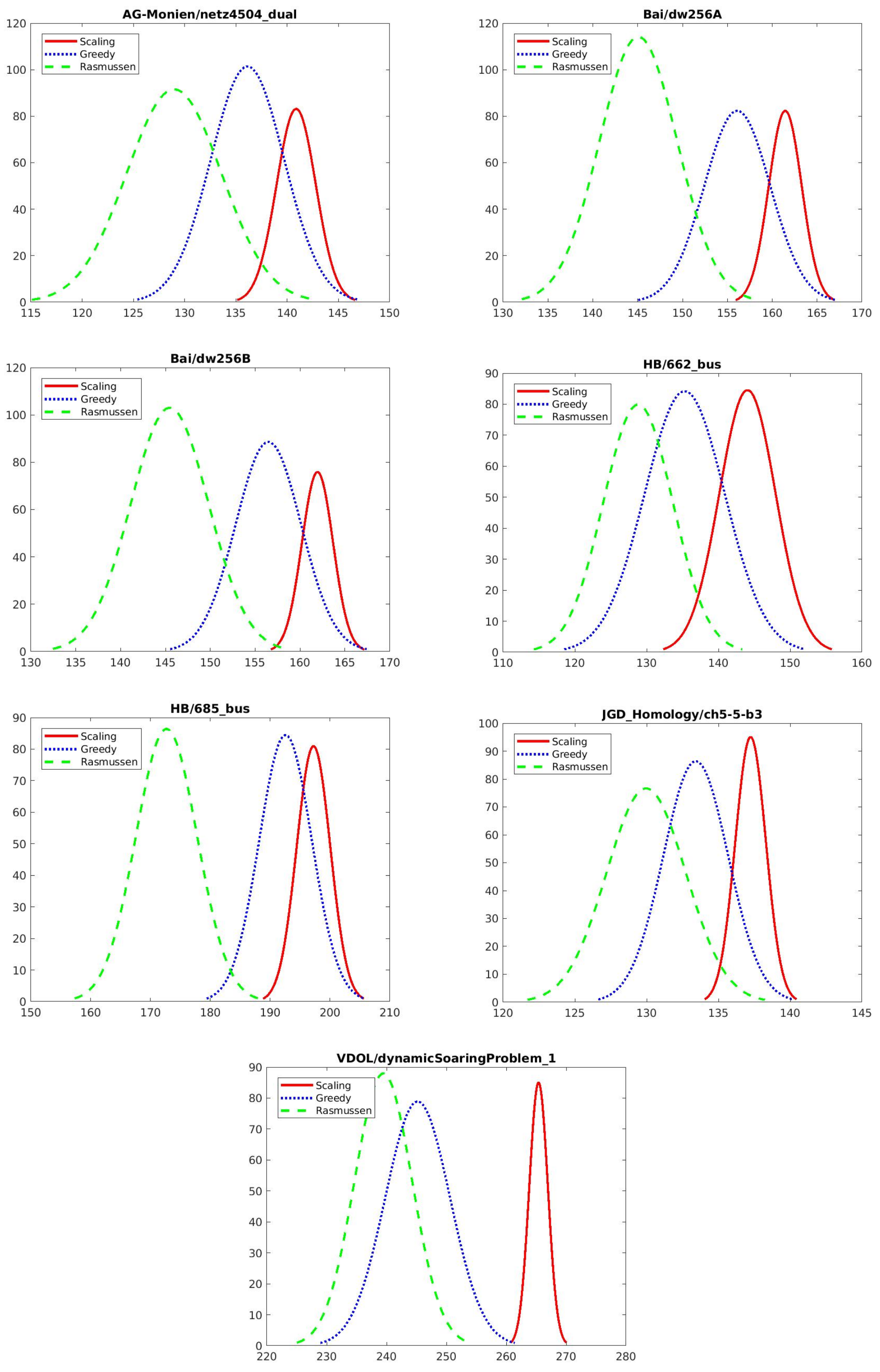


Figure 3: The performance of the estimators on seven matrices corresponding to bipartite graphs in Table 1. In each figure, x axis shows the logarithm of the estimations and the y axis shows the populations of samples corresponding to the curves.

	Rasmussen			Greedy			Scaling		
Matrix	mean	std	std/mean	mean	std	std/mean	mean	std	std/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	$5.27 \cdot 10^{143}$	$3.83 \cdot 10^{144}$	7.26
`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	$5.34 \cdot 10^{164}$	$6.01 \cdot 10^{165}$	11.27
`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	$4.61 \cdot 10^{164}$	$5.01 \cdot 10^{165}$	10.85
`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	$6.16 \cdot 10^{151}$	$1.09 \cdot 10^{153}$	17.68
`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	$1.17 \cdot 10^{203}$	$3.33 \cdot 10^{204}$	28.56
`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	$2.95 \cdot 10^{138}$	$2.19 \cdot 10^{139}$	7.42
`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	$6.05 \cdot 10^{267}$	$6.09 \cdot 10^{268}$	10.07

Table 2: Statistics for the seven matrices. The last matrix's original name is `dynamicSoaring-Problem_1`.

from the SuiteSparse Matrix Collection (formerly the University of Florida Sparse Matrix Collection) [5]. The properties of these matrices are given in Table 1. We have run the three algorithms 1000 times on these seven matrices. Since we do not know the permanents of these matrices, we only plot the estimates in Figure 3. For each algorithm, we create a histogram and then try to fit a bell curve around it to understand its distribution. As the values are very large, we present the results on a log-scale. In all the seven bipartite graphs, 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 the proposed scaling-based algorithm. For this experiment, we also present the mean, standard deviation (std), and the std/mean ratio of the three algorithms in Table 2. For the standard deviation we opt to use the formula $\sigma^2 = \frac{\sum_{i=1}^{N}(X_i - \overline{X})^2}{N-1}$ where N represents the number of samples and X_i corresponds to the ith 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 our Scaling Algorithm returns the largest. Furthermore, our std/mean ratio is almost always the smallest of the three (except for the matrix `685bus`). This clearly demonstrates that with Scaling we have less variation and the proposed algorithm's results are more concentrated around the returned mean compared to the other two.

5.2 Experiments on general, undirected graphs

For the last test case, we examine the performance of the estimators on five undirected graphs obtained from the SuiteSparse Matrix Collection. After downloading these matrices, we make them pattern-wise symmetric by adding all missing symmetric entries. Furthermore, we eliminate the values in the diagonal and set all remaining nonzero values to one so that the resulting matrix resembles the adjacency matrix of an undirected graph. The properties of the selected matrices are presented in Table 3.

As discussed in Section 4.1, 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. Adhering to the original description of Greedy, we have chosen to implement it as is. For the extension of Rasmussen for undirected graphs, first we select a row and then we remove any of its entries that do not participate in any matchings, so that we always end up with a valid perfect matching.

For the proposed scaling-based algorithm we opted to test both alternatives:

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

Name	n	nnz
Bai/bwm200	200	796
Bai/dw256A	512	2480
HB/ash292	292	2208
JGDTrefethen/Trefethen200	200	2890
Pothen/mesh2em1	306	2018

Table 3: Names, dimensions and numbers of nonzeros of the real-life square matrices corresponding to undirected graphs used to evaluate the performance of the estimators.

	Rasmussen			Greedy			Scaling			Clear-Scaling		
Matrix	mean	std	std/mean	mean	std	std/mean	mean	std	std/mean	mean	std	std/mean
`bwm200`	$6.89 \cdot 10^{20}$	$5.51 \cdot 10^{21}$	8.00	$5.85 \cdot 10^{20}$	$5.80 \cdot 10^{20}$	0.99	$5.66 \cdot 10^{20}$	$2.66 \cdot 10^{20}$	0.47	$5.58 \cdot 10^{20}$	$2.61 \cdot 10^{20}$	0.47
`dw256A`	$5.54 \cdot 10^{61}$	$1.11 \cdot 10^{63}$	19.94	$2.82 \cdot 10^{62}$	$3.01 \cdot 10^{63}$	10.68	$5.52 \cdot 10^{62}$	$2.67 \cdot 10^{63}$	4.84	$4.93 \cdot 10^{62}$	$2.09 \cdot 10^{63}$	4.24
`ash292`	$9.13 \cdot 10^{55}$	$1.15 \cdot 10^{57}$	12.57	$1.20 \cdot 10^{56}$	$2.61 \cdot 10^{56}$	2.16	$1.32 \cdot 10^{56}$	$2.68 \cdot 10^{56}$	2.04	$1.20 \cdot 10^{56}$	$2.15 \cdot 10^{56}$	1.80
`Trefethen200`	$7.37 \cdot 10^{70}$	$2.48 \cdot 10^{71}$	3.37	$8.44 \cdot 10^{70}$	$7.23 \cdot 10^{70}$	0.86	$8.36 \cdot 10^{70}$	$4.23 \cdot 10^{70}$	0.51	$8.54 \cdot 10^{70}$	$4.33 \cdot 10^{70}$	0.51
`mesh2e1`	$2.06 \cdot 10^{50}$	$4.64 \cdot 10^{51}$	22.55	$1.35 \cdot 10^{50}$	$2.53 \cdot 10^{50}$	1.87	$1.62 \cdot 10^{50}$	$5.02 \cdot 10^{50}$	3.11	$1.480 \cdot 10^{50}$	$3.57 \cdot 10^{50}$	2.41

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

only on cleaning entries from the selected row.

2. The results labeled with *Clear-Scaling* are obtained by first removing all those entries who do not participate in any matching of the undirected graph (due to Fact 10 the resulting matrix has total support) and then we proceed to scale the matrix to acquire the scaling matrix $\mathbf{R}$ and do the appropriate selections.

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 estimators. The results are presented in Figure 4. We observe that the two scaling alternatives have similar curves and there is almost no advantage in applying the more expensive method of extensive cleaning. In addition, we observe that our approach seems to reduce the variance much better compared to the other algorithms.

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 section. The table shows that once again Scaling obtains usually the smallest standard deviation to mean ratio. In addition, we can observe that Clear Scaling can help in reducing this ratio although slightly so.

6 Conclusion

We have proposed a technique for approximating the permanent based on matrix scaling. We manage to prove loose yet computable upper bounds for $\mathbb{E}[X_A^2]$. The experimental analysis suggests that an improvement over previous similar methodologies is possible. Future work involves bounding the estimated factors for special graph classes.

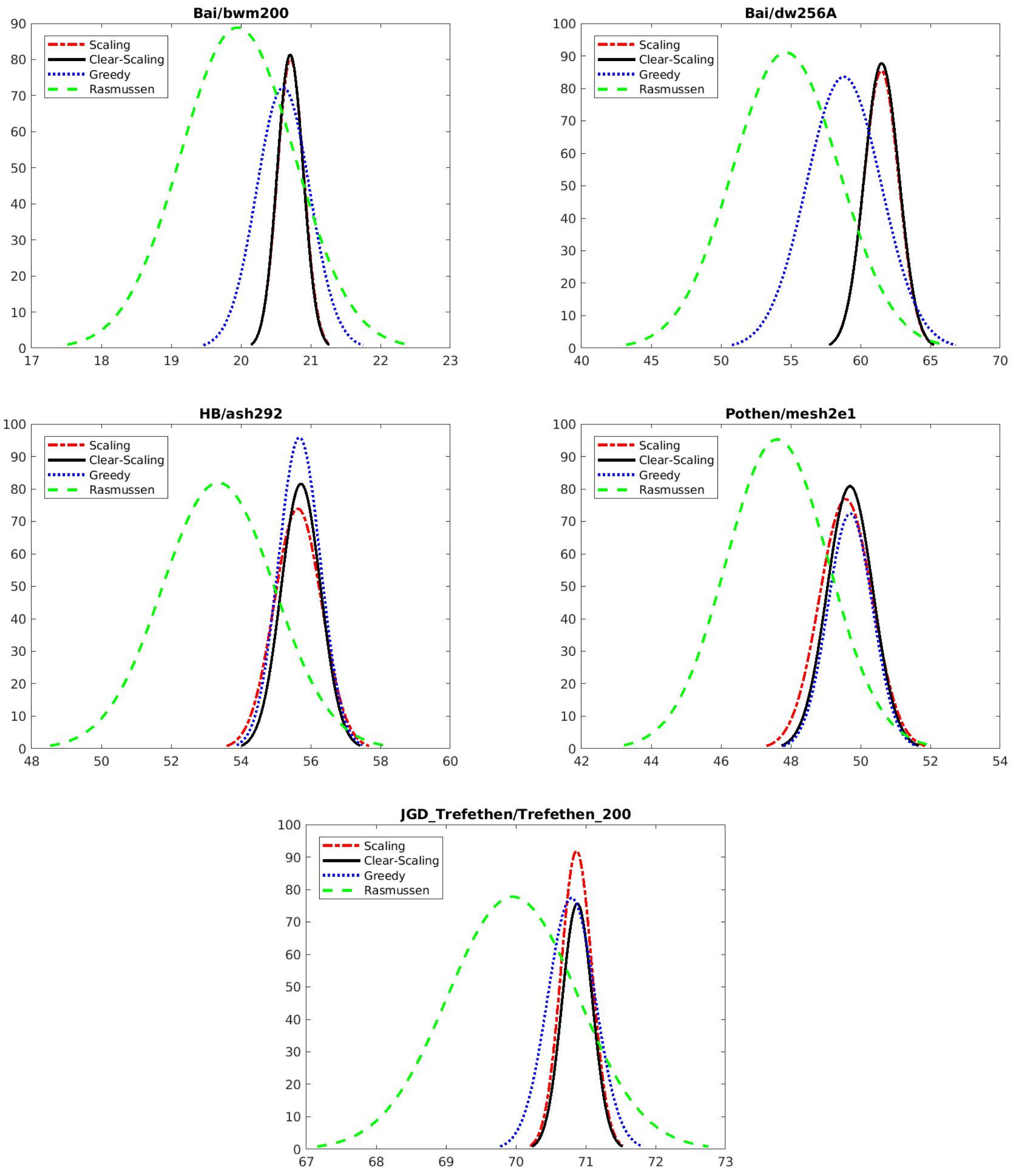


Figure 4: The performance of the estimators on the five matrices in Table 3 after symmetrization and removing diagonal entries. The logarithms of the 1000 estimator values are presented on the x-axes and the y axes demonstrate the distribution of the samples for the values in each approach

References

[1] Z. Allen-Zhu, Y. Li, R. Oliveira, and A. Wigderson. Much faster algorithms for matrix scaling. *arXiv preprint arXiv:1704.02315*, 2017.

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

[3] D. Chakrabarty and S. Khanna. Better and simpler error analysis of the Sinkhorn-Knopp algorithm for matrix scaling. *arXiv preprint arXiv:1801.02790*, 2018.

[4] M. B. Cohen, A. Madry, D. Tsipras, and A. Vladu. Matrix scaling and balancing via box constrained Newton's method and interior point methods. *arXiv preprint arXiv:1704.02310*, 2017.

[5] T. A. Davis and Y. Hu. The University of Florida sparse matrix collection. *ACM Trans. Math. Softw.*, 38(1):1:1–1:25, 2011. ISSN 0098-3500.

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

[7] F. Dufossé, K. Kaya, I. Panagiotas, and B. Uçar. Approximation algorithms for maximum matchings in undirected graphs. accepted to be published in the proceedings of the 8th SIAM Workshop on Combinatorial Scientific Computing, Dec 2018.

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

[9] M. Fürer and 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*, pages 263–274. Springer, 2004.

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

[11] L. Gurvits and A. Samorodnitsky. Bounds on the permanent and some applications. *arXiv preprint arXiv:1408.0976*, 2014.

[12] M. Idel. A review of matrix scaling and Sinkhorn's normal form for matrices and positive maps. *arXiv preprint arXiv:1609.06349*, 2016.

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

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

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

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

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

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

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

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

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

[22] H. N. V. Temperley and M. E. Fisher. Dimer problem in statistical mechanics-an exact result. *The Philosophical Magazine: A Journal of Theoretical Experimental and Applied Physics*, 6 (68):1061–1063, 1961. doi: 10.1080/14786436108243366.

[23] D. Štefankovič, E. Vigoda, and J. Wilmes. On counting perfect matchings in general graphs. *arXiv preprint arXiv:1712.07504*, 2017.

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

RESEARCH CENTRE
GRENOBLE – RHÔNE-ALPES

Inovallée

655 avenue de l'Europe Montbonnot

38334 Saint Ismier Cedex

Publisher
Inria
Domaine de Voluceau - Rocquencourt
BP 105 - 78153 Le Chesnay Cedex
inria.fr

ISSN 0249-6399