A polynomial-time approximation algorithm for the permanent of a matrix with non-negative entries*

Mark Jerrum
Division of Informatics
University of Edinburgh
The King's Buildings
Edinburgh EH9 3JZ
United Kingdom
mrj@dcs.ed.ac.uk

Alistair Sinclair
Computer Science Division
Univ. of California, Berkeley
Soda Hall
Berkeley, CA 94720-1776
USA
sinclair@cs.berkeley.edu

Eric Vigoda
Division of Informatics
University of Edinburgh
The King's Buildings
Edinburgh EH9 3JZ
United Kingdom
vigoda@dcs.ed.ac.uk

ABSTRACT

We present a fully-polynomial randomized approximation scheme for computing the permanent of an arbitrary matrix with non-negative entries.

1. PROBLEM DESCRIPTION AND HISTORY

The permanent of an $n \times n$ matrix $A = (a(i,j))$ is defined as

$$\mathrm{per}(A) = \sum_{\pi} \prod_{i} a(i, \pi(i)),$$

where the sum is over all permutations π of $\{1, 2, \ldots, n\}$. When A is a 0,1 matrix, we can view it as the adjacency matrix of a bipartite graph $G_A = (V_1, V_2, E)$. It is clear that the permanent of A is then equal to the number of perfect matchings in G_A.

The evaluation of the permanent has attracted the attention of researchers for almost two centuries, beginning with the memoirs of Binet and Cauchy in 1812 (see [12] for a comprehensive history). Despite many attempts, an efficient algorithm for general matrices remained elusive. Indeed, Ryser's algorithm [12] remains the most efficient for computing the permanent exactly, even though it uses as many as $\Theta(n2^n)$ arithmetic operations. A notable breakthrough was achieved about 40 years ago with Kasteleyn's algorithm for counting perfect matchings in planar graphs [9], which uses just $O(n^3)$ arithmetic operations.

This lack of progress was explained in 1979 by Valiant [15], who proved that exact computation of the permanent is #P-complete, even for 0,1 matrices, and hence (under standard complexity-theoretic assumptions) not possible in polynomial time. Since then the focus has shifted to efficient approximation algorithms with precise performance guarantees. Essentially the best one can wish for is a *fully-polynomial randomized approximation scheme* (fpras), which provides an arbitrarily close approximation in time that depends only polynomially on the input size and the desired error. (For precise definitions of this and other notions, see the next section.)

Of the several approaches to designing an fpras that have been proposed, perhaps the most promising was the "Markov chain Monte Carlo" approach. This takes as its starting point the observation that the existence of an fpras for the 0,1 permanent is computationally equivalent to the existence of a polynomial time algorithm for sampling perfect matchings from a bipartite graph (almost) uniformly at random [7].

Broder [3] proposed a Markov chain Monte Carlo method for sampling perfect matchings. This was based on simulation of a Markov chain whose state space consisted of all perfect and "near-perfect" matchings (i.e., matchings with two uncovered vertices, or "holes") in the graph, and whose stationary distribution was uniform. Evidently, this approach can be effective only when the near-perfect matchings do not outnumber the perfect matchings by more than a polynomial factor. By analyzing the convergence rate of Broder's Markov chain, Jerrum and Sinclair [4] showed that the method works in polynomial time *whenever* this condition is satisfied. This led to an fpras for the permanent of several interesting classes of 0,1 matrices, including all dense matrices and a.e. random matrix.

For the past decade, an fpras for the permanent of arbitrary 0,1 matrices has resisted the efforts of researchers. There has been similarly little progress on proving the converse conjecture, that the permanent is hard to approximate in the worst case. Attention has switched to two complementary questions: how quickly can the permanent be approximated within an arbitrary close multiplicative factor; and what is the best approximation factor achievable in polynomial time? Jerrum and Vazirani [8], building upon the

*This work was partially supported by the EPSRC Research Grant "Sharper Analysis of Randomised Algorithms: a Computational Approach" and by NSF grants CCR-982095 and ECS-9873086. Part of the work was done while the first and third authors were guests of the Forschungsinstitut für Mathematik, ETH, Zürich, Switzerland. A preliminary version of this paper appears in the Electronic Colloquium on Computational Complexity, Report TR00-079, September 2000.

STOC'01, July 6-8, 2001, Hersonissos, Crete, Greece.

work of Jerrum and Sinclair, presented an approximation scheme whose running time was $\exp(O(\sqrt{n}))$ in the worst case, which though substantially better than Ryser's exact algorithm is still exponential time. In the complementary direction, there are several polynomial time algorithms that achieve an approximation factor of c^n, for various constants c (see, e.g., [10, 2]). To date the best of these is due to Barvinok [2], and gives $c \sim 1.31$.

In this paper, we resolve the question of the existence of an fpras for the permanent of a general 0,1 matrix (and indeed, of a general matrix with non-negative entries) in the affirmative. Our algorithm is based on a refinement of the Markov chain Monte Carlo method mentioned above. The key ingredient is the weighting of near-perfect matchings in the stationary distribution so as to take account of the positions of the holes. With this device, it is possible to arrange that each hole pattern has equal aggregated weight, and hence that the perfect matchings are not dominated too much. The resulting Markov chain is a variant of Broder's earlier one, with a Metropolis rule that handles the weights. The analysis of the mixing time follows the earlier argument of Jerrum and Sinclair [4], except that the presence of the weights necessitates a combinatorially more delicate application of the path-counting technology introduced in [4]. The computation of the required hole weights presents an additional challenge which is handled by starting with the complete graph (in which everything is trivial) and slowly reducing the presence of matchings containing non-edges of G, computing the required hole weights as this process evolves.

We conclude this introductory section with a statement of the main result of the paper.

Theorem 1. *There exists a fully-polynomial randomized approximation scheme for the permanent of an arbitrary $n \times n$ matrix A with non-negative entries.*

The condition in the theorem that all entries be non-negative cannot be dropped. One way to appreciate this is to consider what happens if we replace matrix entry $a(1,1)$ by $a(1,1)-\alpha$ where α is a parameter that can be varied. Call the resulting matrix A_α. Note that $\mathrm{per}(A_\alpha) = \mathrm{per}(A) - \alpha\,\mathrm{per}(A_{1,1})$, where $A_{1,1}$ denotes the submatrix of A obtained by deleting the first row and column. On input A_α, an approximation scheme would have at least to identify correctly the sign of $\mathrm{per}(A_\alpha)$; then the root of $\mathrm{per}(A) - \alpha\,\mathrm{per}(A_{1,1}) = 0$ could be located by binary search and a very close approximation to $\mathrm{per}(A)/\mathrm{per}(A_{1,1})$ found. The permanent of A itself could then be computed by recursion on the submatrix $A_{1,1}$. It is important to note that the cost of binary search scales linearly in the number of significant digits requested, while that of an fpras scales exponentially (see section 2).

The remainder of the paper is organized as follows. In section 2 we summarize the necessary background concerning the connection between random sampling and counting, and the Markov chain Monte Carlo method. In section 3 we define the Markov chain we will use and present the overall structure of the algorithm, including the computation of hole weights. In section 4 we analyze the Markov chain and show that it is rapidly mixing; this is the most technical section of the paper. Finally, in section 5 we show how to extend the algorithm to handle matrices with arbitrary non-negative entries, and in section 6 we observe some applications to constructing an fpras for various other combinatorial enumeration problems. In the interests of clarity of exposition, we make no attempt to optimize the exponents in our polynomial running times.

2. RANDOM SAMPLING AND MARKOV CHAINS

2.1 Random sampling

As stated in the Introduction, our goal is a fully-polynomial randomized approximation scheme (fpras) for the permanent. This is a randomized algorithm which, when given as input an $n \times n$ non-negative matrix A together with an accuracy parameter $\varepsilon \in (0,1]$, outputs a number Z (a random variable of the coins tossed by the algorithm) such that

$$\Pr[(1-\varepsilon)\,\mathrm{per}(A) \le Z \le (1+\varepsilon)\,\mathrm{per}(A)] \ge \tfrac{3}{4},$$

and runs in time polynomial in n and ε^{-1}. The probability $3/4$ can be increased to $1-\delta$ for any desired $\delta > 0$ by outputting the median of $O(\log \delta^{-1})$ independent trials [7].

To construct an fpras, we will follow a well-trodden path via random sampling. We focus on the 0,1 case; see section 5 for an extension to the case of matrices with general non-negative entries. Recall that when A is a 0,1 matrix, $\mathrm{per}(A)$ is equal to the number of perfect matchings in the corresponding bipartite graph G_A. Now it is well known—see for example [6]—that for this and most other natural combinatorial counting problems, an fpras can be built quite easily from an algorithm that generates the same combinatorial objects, in this case perfect matchings, (almost) uniformly at random. It will therefore be sufficient for us to construct a *fully-polynomial almost uniform sampler* for perfect matchings, namely a randomized algorithm which, given as inputs an $n \times n$ 0,1 matrix A and a bias parameter $\delta \in (0,1]$, outputs a random perfect matching in G_A from a distribution $\mathcal{U}'$ that satisfies

$$d_{\mathrm{tv}}(\mathcal{U}', \mathcal{U}) \le \delta,$$

where $\mathcal{U}$ is the uniform distribution on perfect matchings in G_A and d_{tv} denotes (total) variation distance.[1] The algorithm is required to run in time polynomial in n and $\log \delta^{-1}$.

This paper will be devoted mainly to the construction of a fully-polynomial almost uniform sampler for perfect matchings in an arbitrary bipartite graph. The sampler will be based on simulation of a suitable Markov chain, whose state space includes all perfect matchings in the graph and which converges to a stationary distribution that is uniform over these matchings.

2.2 Markov chains

Consider a Markov chain with finite state space Ω and transition probabilities P. The chain is *irreducible* if for every pair of states $x, y \in \Omega$, there exists a $t > 0$ such that $P^t(x,y) > 0$ (i.e., all states communicate); it is *aperiodic* if $\gcd\{t : P^t(x,y) > 0\} = 1$ for all x, y. It is well known from the classical theory that an irreducible, aperiodic Markov chain converges to a unique *stationary distribution* π over Ω, i.e., $P^t(x,y) \to \pi(y)$ as $t \to \infty$ for all $y \in \Omega$, regardless of the initial state x. If there exists a probability distribution

[1]The *total variation distance* between two distributions μ, η on a finite set Ω is defined as $d_{\mathrm{tv}}(\mu,\eta) = \frac{1}{2}\sum_{x\in\Omega}|\mu(x) - \eta(x)| = \max_{S\subset\Omega}|\mu(S) - \eta(S)|$.

set $\hat{\delta} \leftarrow \delta/(12n^2+3)$
repeat $T = \lceil (6n^2+2)\ln(3/\delta)\rceil$ times:
    simulate the Markov chain for $\tau_x(\hat{\delta})$ steps
    output the final state if it belongs to $\mathcal{M}$ and halt
output an arbitrary perfect matching if all trials fail

Figure 1: Obtaining an almost uniform sampler from the Markov chain.

π on Ω which satisfies the *detailed balance* conditions for all $M, M' \in \Omega$, i.e.,

$$\pi(M)P(M,M') = \pi(M')P(M',M) =: Q(M,M'),$$

then the chain is said to be (*time-*)*reversible* and π is the stationary distribution.

We are interested in the rate at which a Markov chain converges to its stationary distribution. To this end, we define the *mixing time* as

$$\tau_x(\delta) = \min\{t : d_{\text{tv}}(P^t(x,\cdot),\pi) \le \delta\}.$$

When the Markov chain is used as a random sampler, the mixing time determines the number of simulation steps needed before a sample is produced.

In this paper, the state space Ω of the Markov chain will consist of the perfect and "near-perfect" matchings (i.e., those that leave only two uncovered vertices, or "holes") in the bipartite graph G_A with $n+n$ vertices. The stationary distribution π will be uniform over the set of perfect matchings $\mathcal{M}$, and will assign probability $\pi(\mathcal{M}) \ge 1/(4n^2+1)$ to $\mathcal{M}$. Thus we get an almost uniform sampler for perfect matchings by iterating the following trial: simulate the chain for $\tau_x(\hat{\delta})$ steps (where $\hat{\delta}$ is a sufficiently small positive number), starting from state x, and output the final state if it belongs to $\mathcal{M}$. The details are given in figure 1.

LEMMA 2. *The algorithm presented in figure 1 is an almost uniform sampler for perfect matchings with bias parameter δ.*

PROOF. Let $\hat{\pi}$ be the distribution of the final state of a single simulation of the Markov chain; note that the length of simulation is chosen so that $d_{\text{tv}}(\hat{\pi},\pi) \le \hat{\delta}$. Let $\mathcal{S} \subset \mathcal{M}$ be an arbitrary set of perfect matchings, and let $M \in \mathcal{M}$ be the perfect matching that is eventually output (M is a random variable depending on the random choices made by the algorithm). Denoting by $\overline{\mathcal{M}} = \Omega \setminus \mathcal{M}$ the complement of $\mathcal{M}$, the result follows from the chain of inequalities:

$$\begin{aligned}
\Pr(M \in \mathcal{S}) &\ge \frac{\hat{\pi}(\mathcal{S})}{\hat{\pi}(\mathcal{M})} - \hat{\pi}(\overline{\mathcal{M}})^T \\
&\ge \frac{\pi(\mathcal{S}) - \hat{\delta}}{\pi(\mathcal{M}) + \hat{\delta}} - \exp(-\hat{\pi}(\mathcal{M})T) \\
&\ge \frac{\pi(\mathcal{S})}{\pi(\mathcal{M})} - \frac{2\hat{\delta}}{\pi(\mathcal{M})} - \exp(-(\pi(\mathcal{M}) - \hat{\delta})T) \\
&\ge \frac{\pi(\mathcal{S})}{\pi(\mathcal{M})} - \frac{2\delta}{3} - \frac{\delta}{3}.
\end{aligned}$$

A matching bound $\Pr(M \in \mathcal{S}) \le \pi(\mathcal{S})/\pi(\mathcal{M}) + \delta$ follows immediately by considering the complementary set $\mathcal{M} \setminus \mathcal{S}$. (Note that the total variation distance $d_{\text{tv}}(\mu,\eta)$ between distributions μ and η may be interpreted as the maximum of $|\mu(\mathcal{S}) - \eta(\mathcal{S})|$ over all events $\mathcal{S}$.) $\square$

The running time of the random sampler is determined by the mixing time of the Markov chain. We will derive an upper bound on $\tau_x(\delta)$ as a function of n and δ. To satisfy the requirements of a fully-polynomial sampler, this bound must be polynomial in n. (The logarithmic dependence on δ^{-1} is an automatic consequence of the geometric convergence of the chain.) Accordingly, we shall call the Markov chain *rapidly mixing* (from initial state x) if, for any fixed $\delta > 0$, $\tau_x(\delta)$ is bounded above by a polynomial function of n. Note that in general the size of Ω will be exponential in n, so rapid mixing requires that the chain be close to stationarity after visiting only a tiny (random) fraction of its state space.

In order to bound the mixing time we use the notion of *conductance* Φ, defined as $\Phi = \min_{\emptyset \subset \mathcal{S} \subset \Omega} \Phi(\mathcal{S})$, where

$$\Phi(\mathcal{S}) = \frac{Q(\mathcal{S},\overline{\mathcal{S}})}{\pi(\mathcal{S})\pi(\overline{\mathcal{S}})} \equiv \frac{\sum_{x\in\mathcal{S}}\sum_{y\in\overline{\mathcal{S}}} Q(x,y)}{\pi(\mathcal{S})\pi(\overline{\mathcal{S}})}.$$

The following bound relating conductance and mixing time is well known (see, e.g., [4]).

THEOREM 3. *For any ergodic, reversible Markov chain with self-loop probabilities $P(y,y) \ge 1/2$ for all states y, and any initial state $x \in \Omega$,*

$$\tau_x(\delta) \le \frac{2}{\Phi^2}\left(\ln \pi(x)^{-1} + \ln \delta^{-1}\right).$$

Thus to prove rapid mixing it suffices to demonstrate a lower bound of the form $1/\operatorname{poly}(n)$ on the conductance of our Markov chain on matchings. (The term $\ln \pi(x)^{-1}$ will not cause a problem since the total number of states will be at most $(n+1)!$, and we will start in a state x that maximizes $\pi(x)$.)

3. THE SAMPLING ALGORITHM

As explained in the previous section, our goal now is to design an efficient (almost) uniform sampling algorithm for perfect matchings in a bipartite graph $G = G_A$. This will immediately yield an fpras for the permanent of an arbitrary 0,1 matrix, and hence most of the content of Theorem 1. The extension to matrices with arbitrary non-negative entries is described in section 5.

Let $G = (V_1, V_2, E)$ be a bipartite graph on $n+n$ vertices. The basis of our algorithm is a Markov chain MC defined on the collection of perfect and near-perfect matchings of G. Let $\mathcal{M}$ denote the set of perfect matchings in G, and let $\mathcal{M}(y,z)$ denote the set of near-perfect matchings with holes only at the vertices $y \in V_1$ and $z \in V_2$. The state space of MC is $\Omega := \mathcal{M} \cup \bigcup_{y,z} \mathcal{M}(y,z)$. Previous work [3, 4] considered a Markov chain with the same state space Ω and transition probabilities designed so that the stationary distribution was uniform over Ω, or assigned slightly higher weight to each perfect matching than to each near-perfect matching. Rapid mixing of this chain immediately yields an efficient sampling algorithm provided perfect matchings have sufficiently large weight; specifically, $|\mathcal{M}|/|\Omega|$ must be bounded below by a polynomial in n. In [4] it was proved that this condition — rather surprisingly — also implies that the Markov chain is rapidly mixing. This led to an fpras for the permanent of any 0,1 matrix satisfying the above condition, including all dense matrices (having at least $n/2$

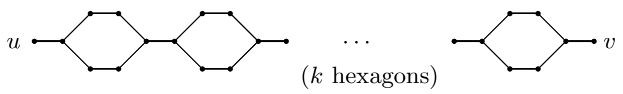

Figure 2: A graph with $|\mathcal{M}(u,v)|/|\mathcal{M}|$ exponentially large.

1's in each row and column), and a.e. (almost every) random matrix.[2]

It is not hard to construct graphs in which, for some pair of holes u, v, the ratio $|\mathcal{M}(u,v)|/|\mathcal{M}|$ is exponentially large. The graph depicted in figure 2, for example, has one perfect matching, but 2^k matchings with holes at u and v. For such graphs, the above approach breaks down because the perfect matchings have insufficient weight in the stationary distribution. To overcome this problem, we will introduce an additional weight factor that takes account of the holes in near-perfect matchings. We will define these weights in such a way that any hole pattern (including that with no holes, i.e., perfect matchings) is equally likely in the stationary distribution. Since there are only n^2+1 such patterns, π will assign $\Omega(1/n^2)$ weight to perfect matchings.

It will actually prove technically convenient to introduce edge weights also. Thus for each edge $(y,z) \in E$, we introduce a positive weight $\lambda(y,z)$, which we call its *activity*. We extend the notion of activities to matchings M (of any cardinality) by $\lambda(M) = \prod_{(i,j)\in M} \lambda(i,j)$. Similarly, for a set of matchings $\mathcal{S}$ we define $\lambda(\mathcal{S}) = \sum_{M\in\mathcal{S}} \lambda(M)$.[3] For our purposes, the advantage of edge weights is that they allow us to work with the complete graph on $n+n$ vertices, rather than with an arbitrary graph $G = (V_1, V_2, E)$: we can do this by setting $\lambda(e) = 1$ for $e \in E$, and $\lambda(e) = \xi \approx 0$ for $e \notin E$. Taking $\xi \le 1/n!$ ensures that the "bogus" matchings have little effect, as will be described shortly.

We are now ready to specify the desired stationary distribution of our Markov chain. This will be the distribution π over Ω defined by $\pi(M) \propto \Lambda(M)$, where

$$\Lambda(M) = \begin{cases} \lambda(M)w(u,v) & \text{if } M \in \mathcal{M}(u,v) \text{ for some } u,v; \\ \lambda(M) & \text{if } M \in \mathcal{M}, \end{cases}$$

and $w : V_1 \times V_2 \to \mathbb{R}^+$ is the weight function for holes to be specified shortly.

To construct a Markov chain having π as its stationary distribution, we use the original chain of [3, 4] augmented with a Metropolis acceptance rule for the transitions. Thus transitions from a matching M are defined as in figure 3. The Metropolis rule in the final step ensures that this Markov chain is reversible with $\pi(M) \propto \Lambda(M)$ as its stationary distribution. Finally, to satisfy the conditions of Theorem 3 we add a self-loop probability of $1/2$ to every state; i.e., on every step, with probability $1/2$ we make a transition as above and otherwise do nothing.

Next we need to specify the weight function w. Ideally we would like to take $w = w^*$, where

$$w^*(u,v) = \frac{\lambda(\mathcal{M})}{\lambda(\mathcal{M}(u,v))} \tag{1}$$

for each pair of holes u, v with $\mathcal{M}(u,v) \ne \emptyset$. (We leave $w(u,v)$ undefined when $\mathcal{M}(u,v) = \emptyset$.) With this choice of weights, any hole pattern (including that with no holes) is equally likely under the distribution π; since there are at most n^2+1 such patterns, when sampling from the distribution π a perfect matching is generated with probability at least $1/(n^2+1)$. In fact, we will not know w^* exactly but will content ourselves with weights w satisfying

$$w^*(y,z)/2 \le w(y,z) \le 2w^*(y,z), \tag{2}$$

with very high probability. This perturbation will reduce the relative weight of perfect matchings by at most a constant factor.

[2] I.e., the proportion of matrices that are covered by the fpras tends to 1 as $n \to \infty$.

[3] Note that if we set $\lambda(y,z) = a(y,z)$ for every edge (y,z), then $\mathrm{per}(A)$ is exactly equal to $\lambda(\mathcal{M})$. Thus our definition of the activity function λ is natural.

1. Choose an edge $e = (u,v)$ uniformly at random.
2. (i) If $M \in \mathcal{M}$ and $e \in M$, let $M' = M \setminus \{e\} \in \mathcal{M}(u,v)$;
   (ii) if $M \in \mathcal{M}(u,v)$, let $M' = M \cup \{e\} \in \mathcal{M}$;
   (iii) if $M \in \mathcal{M}(u,z)$ where $z \ne v$ and $(y,v) \in M$, let $M' = M \cup \{e\} \setminus \{(y,v)\} \in \mathcal{M}(y,z)$;
   (iv) if $M \in \mathcal{M}(y,v)$ where $y \ne u$ and $(u,z) \in M$, let $M' = M \cup \{e\} \setminus \{(u,z)\} \in \mathcal{M}(y,z)$.
3. With probability $\min\{1, \Lambda(M')/\Lambda(M)\}$ go to M'; otherwise, stay at M.

Figure 3: Transitions from a matching M.

The main technical result of this paper is the following theorem, which says that, provided the weight function w satisfies condition (2), the Markov chain is rapidly mixing. The theorem will be proved in the next section. We state the theorem as it applies to an arbitrary bipartite graph with $m = |E|$ edges; since we are working with the complete bipartite graph, for our purposes of our algorithm $m = n^2$.

THEOREM 4. *Assuming the weight function w satisfies inequality (2) for all $(y,z) \in V_1 \times V_2$, then the mixing time of the Markov chain MC is bounded above by $\tau_x(\delta) = O\big(m^6 \times n^8(n\log n + \log\delta^{-1})\big)$, provided the initial state x is a perfect matching of maximum activity.*

Finally we need to address the issue of computing (approximations to) the weights w^* defined in (1). Since w^* encapsulates detailed information about the set of perfect and near-perfect matchings, we cannot expect to compute it directly for our desired edge activities $\lambda(e)$. Rather than attempt this, we instead initialize the edge activities to trivial values, for which the corresponding w^* can be computed easily, and then gradually adjust the $\lambda(e)$ towards their desired values; at each step of this process, we will be able to compute (approximations to) the weights w^* corresponding to the new activities.

Recall that we work with the *complete* graph on $n+n$ vertices, and will ultimately assign an activity of 1 to all edges $e \in E$ (i.e., all edges of our graph G), and a very small value $1/n!$ to all "non-edges" $e \notin E$. Since the weight of an invalid matching (i.e., one that includes a non-edge) will be at most $1/n!$ and there are at most $n!$ possible matchings, the combined weight of all invalid matchings will be at most 1. Assuming the graph has at least one perfect matching, the invalid matchings will merely increase by at most a small constant factor the probability that a single simulation fails to return a perfect matching. Thus our "target" activities are $\lambda^*(e) = 1$ for all $e \in E$, and $\lambda^*(e) = 1/n!$ for all other e.

As noted above, our algorithm begins with activities λ whose ideal weights w^* are easy to compute. Since we are working with the complete graph, a natural choice is to set $\lambda(e) = 1$ for all e. The activities of edges $e \in E$ will remain at 1 throughout; the activities of non-edges $e \notin E$ will converge to their target values $\lambda^*(e) = 1/n!$ in a sequence of phases, in each of which the activity $\lambda(e)$ of some chosen non-edge $e \notin E$ is updated to $\lambda'(e)$, where $\exp(-1/2)\lambda(e) \le \lambda'(e) \le \lambda(e)$. (Symmetrically, it would be possible to increase $\lambda(e)$ to any value $\lambda'(e) \le \exp(1/2)\lambda(e)$, though we shall not use that freedom.)

We assume at the beginning of the phase that condition (2) is satisfied; in other words, $w(u,v)$ approximates $w^*(u,v)$ within ratio 2 for all pairs (u,v).[4] Before updating an activity, we must consolidate our position by finding, for each pair (u,v), a better approximation to $w^*(u,v)$: specifically, one that is within ratio c for some $1 < c < 2$. (We shall see later that $c = 6/5$ suffices here.) For this purpose we may use the identity

$$\frac{w(u,v)}{w^*(u,v)} = \frac{\pi(\mathcal{M}(u,v))}{\pi(\mathcal{M})}, \tag{3}$$

since $w(u,v)$ is known to us, and the probabilities on the right hand side may be estimated to arbitrary precision by taking sample averages. (Recall that π denotes the stationary distribution of the Markov chain.)

Although we do not know how to sample from π exactly, Theorem 4 does allow us to sample, in polynomial time, from a distribution $\hat\pi$ that is within variation distance $\hat\delta$ of π. (We shall set $\hat\delta$ appropriately in a moment.) So suppose we generate S samples from $\hat\pi$, and for each pair $(u,v) \in V_1 \times V_2$ we consider the proportion $\alpha(u,v)$ of samples with holes at u,v, together with the proportion α of samples that are perfect matchings. Clearly, the expectations of these quantities satisfy

$$\mathbb{E}\,\alpha(u,v) = \hat\pi(\mathcal{M}(u,v)) \quad \text{and} \quad \mathbb{E}\,\alpha = \hat\pi(\mathcal{M}). \tag{4}$$

Naturally, it is always possible that some sample average $\alpha(u,v)$ will be far from its expectation, so we have to allow for the possibility of failure. We denote by ε the (small) failure probability we are prepared to tolerate. Provided the sample size S is large enough, $\alpha(u,v)$ (respectively α) approximates $\hat\pi(\mathcal{M}(u,v))$ (respectively $\hat\pi(\mathcal{M})$) within ratio $c^{1/4}$, with probability at least $1-\varepsilon$. Furthermore, if $\hat\delta$ is small enough, $\hat\pi(\mathcal{M}(u,v))$ (respectively $\hat\pi(\mathcal{M})$) approximates $\pi(\mathcal{M}(u,v))$ (respectively $\pi(\mathcal{M})$) within ratio $c^{1/4}$. Then via (3) we have, with probability at least $1-(n^2+1)\varepsilon$, approximations within ratio c to all of the target weights $w^*(u,v)$.

It remains to determine bounds on the sample size S and sampling tolerance $\hat\delta$ that make this all work. Condition (2) entails

$$\mathbb{E}\,\alpha(u,v) = \hat\pi(\mathcal{M}(u,v)) \ge \pi(\mathcal{M}(u,v)) - \hat\delta \ge 1/4(n^2+1) - \hat\delta.$$

[4]We say that ξ *approximates* x *within ratio* r if $r^{-1}x \le \xi \le rx$.

initialize $\lambda(e) \leftarrow 1$ for all $e \in V_1 \times V_2$
initialize $w(u,v) \leftarrow n$ for all $(u,v) \in V_1 \times V_2$
while $\exists e \notin E$ with $\lambda(e) > 1/n!$ do
    set $\lambda(e) \leftarrow \lambda(e)\exp(-1/2)$
    take S samples from MC with parameters λ, w,
        each after a simulation of T steps
    use the sample to obtain estimates $w'(u,v)$ satisfying condition (5) with high probability $\forall u,v$
    set $w(u,v) \leftarrow w'(u,v)$ $\forall u,v$
output the final weights $w(u,v)$

Figure 4: The algorithm.

Assuming $\hat\delta \le 1/8(n^2+1)$, it follows from any of the standard Chernoff bounds (see, e.g., [1] or [13, Thm. 4.1]), that $O(n^2 \log \varepsilon^{-1})$ samples from $\hat\pi$ suffice to estimate $\mathbb{E}\,\alpha(u,v) = \hat\pi(\mathcal{M}(u,v))$ within ratio $c^{1/4}$ with probability at least $1-\varepsilon$. Again using the fact that $\pi(\mathcal{M}(u,v)) \ge 1/4(n^2+1)$, we see that $\hat\pi(\mathcal{M}(u,v))$ will approximate $\pi(\mathcal{M}(u,v))$ within ratio $c^{1/4}$ provided $\hat\delta = c_1/n^2$ where $c_1 > 0$ is a sufficiently small constant. (Note that we also satisfy the earlier constraint on $\hat\delta$ by this setting.) Therefore, taking $c = 6/5$ and using $O(n^2 \log \varepsilon^{-1})$ samples, we obtain refined estimates $w'(u,v)$ satisfying

$$5w^*(u,v)/6 \le w'(u,v) \le 6w^*(u,v)/5 \tag{5}$$

with probability $1-(n^2+1)\varepsilon$. Plugging $\delta = \hat\delta$ into Theorem 4, the time to generate each sample is $T = O(n^{21}\log n)$.

We can then update the activity of an edge e by changing $\lambda(e)$ by a multiplicative factor of $\exp(-1/2)$. Note that the effect of this change on the ideal weight function w^* is at most a factor $\exp(1/2)$. Thus, since $6\exp(1/2)/5 < 2$, our estimates w' obeying (5) actually satisfy the weaker condition (2), as it applies to the *next* phase of the algorithm; that is, putting $w = w'$ in (2) and interpreting w^* as the ideal weights with respect to the *new* activities. So having assigned w' to w we can proceed with the next phase. The entire algorithm is sketched in figure 4.

Starting from the trivial values $\lambda(e) = 1$ for all edges e of the complete bipartite graph, we use the above procedure repeatedly to reduce the activity of each non-edge $e \notin E$ down to $1/n!$, leaving the activities of all edges $e \in E$ at unity. This requires $O(n^3 \log n)$ phases, and each phase requires $S = O(n^2 \log \varepsilon^{-1})$ samples. We have seen that the number of simulation steps to generate a sample is $T = O(n^{21}\log n)$. Thus the overall time required to initialize the weights to appropriate values for the target activities λ^* is $O(n^{26}(\log n)^2 \log \varepsilon^{-1})$.

Suppose our aim is to generate one perfect matching from a distribution that is within variation distance δ of uniform. Then we need to set ε so that the overall failure probability is strictly less than δ, say $\delta/2$. The probability of violating condition (5) in *any* phase is at most $O(\varepsilon n^5 \log n)$, since there are $O(n^5 \log n)$ values to be estimated, and we can fail in any individual case with probability ε. So for adequate reliability we must take $\varepsilon = c_2\delta/n^5 \log n$. The running time of the entire algorithm of figure 4 is thus $O(n^{26}(\log n)^2(\log n + \log \delta^{-1}))$. By Theorem 4, the (expected) additional time required to generate the sample is $O(n^{22}(n\log n + \log \delta^{-1}))$, which is negligible in comparison with the initialisation procedure. (The extra factor n^2 represents the expected number of samples before a perfect matching is seen.)

At the conclusion of this algorithm we have a good approximation to the ideal weights w^* for our desired activities λ^*. We can then simply simulate the Markov chain with these parameters to generate perfect matchings uniformly at random at an (expected) cost of $O(n^{22}(n \log n + \log \delta^{-1}))$ per sample, where δ is the permitted deviation from uniformity. As remarked earlier, we have not attempted to minimize the exponents, and the analysis could certainly be tightened to reduce the exponent 22 somewhat.

4. ANALYSIS OF THE MARKOV CHAIN

Our goal in this section is to prove our main technical result on the mixing time of the Markov chain MC, Theorem 4, stated in the previous section. Following Theorem 3, we can get an upper bound on the mixing time by bounding the conductance from below. To do this, we shall use technology introduced in [4], and since applied successfully in several other examples. The idea is to define a *canonical path* $\gamma_{I,F}$ from each state $I \in \Omega$ to each state $F \in \Omega$. By upper bounding the maximum number of such paths that pass through any particular transition (the "congestion"), one obtains a lower bound on the conductance.

Using the fact that perfect matchings have non-negligible weight under the stationary distribution, it will be sufficient to consider only paths between pairs (I, F) where $F \in \mathcal{M}$ (and $I \in \Omega$ is arbitrary), and moreover to count only a portion of some of these paths. Denote the set of all canonical paths by $\Gamma = \{\gamma_{I,F} : (I, F) \in \Omega \times \mathcal{M}\}$. Certain transitions on a canonical path will be deemed *chargeable*. For each transition t denote by

$$\mathrm{cp}(t) = \{(I, F) : \gamma_{I,F} \text{ contains } t \text{ as a chargeable transition}\}.$$

The canonical paths are defined by superimposing I and F. If $I \in \mathcal{M}$, then $I \oplus F$ consists of a collection of alternating cycles. We assume that the cycles are ordered in some canonical fashion; for example, having ordered the vertices, we may take as the first cycle the one that contains the least vertex in the order, as the second cycle the one that contains the least vertex amongst those remaining, and so on. Furthermore we assume that each cycle has a distinguished start vertex (e.g., the least in the order).

The canonical path $\gamma_{I,F}$ from $I \in \mathcal{M}$ to F is obtained by unwinding these cycles in the canonical order. A cycle $v_0 \sim v_1 \sim \ldots \sim v_{2k} = v_0$, where we assume w.l.o.g. that the edge (v_0, v_1) belongs to I, is unwound by: (i) removing the edge (v_0, v_1), (ii) successively, for each $1 \le i \le k-1$, exchanging the edge (v_{2i}, v_{2i+1}) with (v_{2i-1}, v_{2i}), and (iii) adding the edge (v_{2k-1}, v_{2k}). (Refer to figure 5.) All transitions on the path $\gamma_{I,F}$ are deemed chargeable. A canonical path joining two perfect matchings, as just described, will be termed "type A".

If $I \in \mathcal{M}(y, z)$ for some $(y, z) \in V_1 \times V_2$, then $I \oplus F$ consists of a collection of alternating cycles together with a single alternating path from y to z. The canonical path $\gamma_{I,F}$ from I to F is obtained by unwinding the path and then unwinding the cycles, as above, in some canonical order. In this case, only the transitions involved in the unwinding of the path are deemed chargeable. The alternating path $y = v_0 \sim \ldots \sim v_{2k+1} = z$ is unwound by: (i) successively, for each $1 \le i \le k$, exchanging the edge (v_{2i-1}, v_{2i}) with (v_{2i-2}, v_{2i-1}), and (ii) adding the edge (v_{2k}, v_{2k+1}). A canonical path joining a near-perfect to a perfect matching will be termed "type B".

We define a notion of congestion of Γ that accounts only for the chargeable transitions:

$$\varrho(\Gamma) := \max_t \left\{ \frac{1}{Q(t)} \sum_{(I,F)\in \mathrm{cp}(t)} \pi(I)\pi(F) \right\}, \qquad (6)$$

where the maximum is over all transitions t.

Our main task will be to derive an upper bound on $\varrho(\Gamma)$, which we state in the next lemma. From this, it will be a straightforward matter to obtain a lower bound on the conductance Φ (see Corollary 8 below) and hence, via Theorem 3, a bound on the mixing time. In the following lemma recall that $m = |E|$, where for our purposes $m = n^2$.

LEMMA 5. *Assuming the weight function w satisfies inequality (2) for all $(y, z) \in V_1 \times V_2$, then $\varrho(\Gamma) \le 16m$.*

In preparation for proving Lemma 5, we establish some combinatorial inequalities concerning weighted near-perfect matchings that will be used in the proof.

LEMMA 6. *Let G be as above, and let $u, y \in V_1$, $v, z \in V_2$.*

(i) $\lambda(u, v)\lambda(\mathcal{M}(u, v)) \le \lambda(\mathcal{M})$, for all vertices u, v with $u \sim v$;

(ii) $\lambda(u, v)\lambda(\mathcal{M}(u, z))\lambda(\mathcal{M}(y, v)) \le \lambda(\mathcal{M})\lambda(\mathcal{M}(y, z))$, for all distinct vertices u, v, y, z with $u \sim v$.

PROOF. The mapping from $\mathcal{M}(u, v)$ to $\mathcal{M}$ defined by $M \mapsto M \cup \{(u, v)\}$ is injective, and preserves activities modulo a factor $\lambda(u, v)$; this dispenses with (i). For (ii), suppose $M_{u,z} \in \mathcal{M}(u, z)$ and $M_{y,v} \in \mathcal{M}(y, v)$, and consider the superposition of $M_{u,z}$, $M_{y,v}$ and the single edge (u, v). Observe that $M_{u,z} \oplus M_{y,v} \oplus \{(u, v)\}$ decomposes into a collection of cycles together with an odd-length path O joining y and z.[5] Let $O = \{e_0, e_1, \ldots, e_{2k}\}$ be an enumeration of the edges of this path, starting at y and working towards z. Denote by O_0 the $k+1$ even edges, and by O_1 the k odd edges. Finally define a mapping from $\mathcal{M}(u, z) \times \mathcal{M}(y, v)$ to $\mathcal{M} \times \mathcal{M}(y, z)$ by $(M_{u,z}, M_{y,v}) \mapsto (M, M_{y,z})$, where $M := M_{u,z} \cup O_0 \setminus O_1$ and $M_{y,z} := M_{y,v} \cup O_1 \setminus O_0$. Note that this mapping is injective, since we may uniquely recover $(M_{u,z}, M_{y,v})$ from $(M, M_{y,z})$. (To see this, observe that $M \oplus M_{y,z}$ decomposes into a number of cycles, together with a single odd-length path joining y and z. This path is exactly the path O considered in the forward map. There is only one way to apportion edges from $O \setminus \{(u, v)\}$ between $M_{u,z}$ and $M_{y,v}$.) Moreover, the mapping preserves activities modulo a factor $\lambda(u, v)$. □

COROLLARY 7. *Let G be as above, and let $u, y \in V_1$, $v, z \in V_2$. Then, provided in each case that the left hand side of the inequality is defined,*

(i) $w^*(u, v) \ge \lambda(u, v)$, for all vertices u, v with $u \sim v$;

(ii) $w^*(u, z)w^*(y, v) \ge \lambda(u, v)w^*(y, z)$, for all distinct vertices u, v, y, z with $u \sim v$;

[5]It is at this point that we rely crucially on the bipartiteness of G. If G is non-bipartite, we may end up with an even-length path and an odd-length cycle, and the proof cannot proceed.

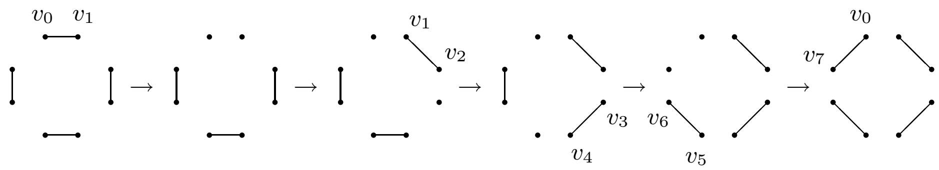

Figure 5: Unwinding a cycle with $k = 4$.

(iii) $w^*(u,z)w^*(y,v) \geq \lambda(u,v)\lambda(y,z)$*, for all distinct vertices* u, v, y, z *with* $u \sim v$ *and* $y \sim z$*.*

Proof. Inequalities (i) and (ii) follow from the correspondingly labelled inequalities in Lemma 6, and the definition of w^* in equation (1). Inequality (iii) is implied by (i) and (ii). □

Armed with Corollary 7, we can now turn to the proof of our main lemma.

Proof of Lemma 5. Note from the Metropolis rule that for any pair of states M, M' such that the probability of transition from M to M' is non-zero, we have $Q(M, M') = \min\{\pi(M), \pi(M')\}/2m$. We will show that for any transition $t = (M, M')$ and any pair of states $I, F \in \mathrm{cp}(t)$, we can define an encoding $\eta_t(I, F) \in \Omega$ such that $\eta_t : \mathrm{cp}(t) \to \Omega$ is an injection (i.e., (I, F) can be recovered uniquely from $\eta_t(I, F)$), and

$$\begin{aligned}\pi(I)\pi(F) &\leq 8\min\{\pi(M), \pi(M')\}\pi(\eta_t(I,F)) \\ &= 16m\, Q(t)\pi(\eta_t(I,F)). \qquad (7)\end{aligned}$$

(The factor 8 here comes from approximating w^* by w.) Note that inequality (7) requires that the encoding $\eta_t(I, F)$ be "weight-preserving". in some sense. Summing inequality (7) over $(I, F) \in \mathrm{cp}(t)$, we get

$$\frac{1}{Q(t)} \sum_{(I,F)\in\mathrm{cp}(t)} \pi(I)\pi(F) \leq 16m \sum_{(I,F)\in\mathrm{cp}(t)} \pi(\eta_t(I,F)) \leq 16m,$$

where we have used the fact that η_t is an injection. This immediately yields the claimed bound on $\varrho(\Gamma)$.

We now proceed to define the encoding η_t and show that it has the above properties. For a transition $t = (M, M')$ which is involved in stage (ii) of unwinding a cycle, the encoding is

$$\eta_t(I,F) = I \oplus F \oplus (M \cup M') \setminus \{(v_0, v_1)\},$$

where (v_0, v_1) is the first edge in the unwinding of the cycle. (Refer to figure 6, where just a single alternating cycle is viewed in isolation.) Otherwise, the encoding is

$$\eta_t(I,F) = I \oplus F \oplus (M \cup M').$$

It is not hard to check that $C = \eta_t(I, F)$ is always a matching in Ω (this is the reason that the edge (v_0, v_1) is removed in the first case above), and that η_t is an injection. To see this for the first case, note that $I \oplus F$ may be recovered from the identity $I \oplus F = (C \cup \{(v_0, v_1)\}) \oplus (M \cup M')$; the apportioning of edges between I and F can then be deduced from the canonical ordering of the cycles and the position of the transition t. The remaining edges, namely those in the intersection $I \cap F$, are determined by $I \cap F = M \cap M' \cap C$. The second case is similar, but without the need to reinstate the edge (v_0, v_1).[6] It therefore remains only to verify inequality (7) for our encoding η_t.

Consider first the case where $I \in \mathcal{M}$ and $t = (M, M')$ is the first transition in the unwinding of an alternating cycle in a type A canonical path, where $M = M' \cup \{(v_0, v_1)\}$. Since $I, F, C, M \in \mathcal{M}$ and $M' \in \mathcal{M}(v_0, v_1)$, inequality (7) simplifies to

$$\lambda(I)\lambda(F) \leq 8\min\{\lambda(M), \lambda(M')w(v_0,v_1)\}\lambda(C).$$

The inequality in this form can be seen to follow from the identity

$$\lambda(I)\lambda(F) = \lambda(M)\lambda(C) = \lambda(M')\lambda(v_0,v_1)\lambda(C),$$

using inequality (i) of Corollary 7 and inequality (2). The situation is symmetric for the final transition in unwinding an alternating cycle.

Staying with the type A path, i.e., with the case $I \in \mathcal{M}$, suppose the transition $t = (M, M')$ is traversed in stage (ii) of unwinding an alternating cycle, i.e., exchanging edge (v_{2i}, v_{2i+1}) with (v_{2i-1}, v_{2i}). In this case we have $I, F \in \mathcal{M}$ while $M \in \mathcal{M}(v_0, v_{2i-1}), M' \in \mathcal{M}(v_0, v_{2i+1})$ and $C \in \mathcal{M}(v_{2i}, v_1)$. Since

$$\begin{aligned}\lambda(I)\lambda(F) &= \lambda(M)\lambda(C)\lambda(v_{2i},v_{2i-1})\lambda(v_0,v_1) \\ &= \lambda(M')\lambda(C)\lambda(v_{2i},v_{2i+1})\lambda(v_0,v_1),\end{aligned}$$

inequality (7) simplifies to

$$1 \leq 8\min\left\{\frac{w(v_0,v_{2i-1})}{\lambda(v_{2i},v_{2i-1})}, \frac{w(v_0,v_{2i+1})}{\lambda(v_{2i},v_{2i+1})}\right\}\frac{w(v_{2i},v_1)}{\lambda(v_0,v_1)}.$$

This inequality can again be verified by reference to Corollary 7: specifically, it follows from part (iii) in the general case $i \neq 1$, and by two applications of part (i) in the special case $i = 1$. Again we use inequality (2) to relate w and w^*.

We now turn to the type B canonical paths. Suppose $I \in \mathcal{M}(y, z)$, and consider a transition $t = (M, M')$ from stage (i) of the unwinding of an alternating path, i.e., exchanging edge (v_{2i}, v_{2i-1}) with (v_{2i-2}, v_{2i-1}). Observe that $F \in \mathcal{M}, M \in \mathcal{M}(v_{2i-2}, z), M' \in \mathcal{M}(v_{2i}, z)$ and $C \in \mathcal{M}(y, v_{2i-1})$. Moreover, $\lambda(I)\lambda(F) = \lambda(M)\lambda(C)\lambda(v_{2i-2}, v_{2i-1}) = \lambda(M') \times \lambda(C)\lambda(v_{2i}, v_{2i-1})$. In inequality (7) we are left with

$$w(y,z) \leq 8\min\left\{\frac{w(v_{2i-2},z)}{\lambda(v_{2i-2},v_{2i-1})}, \frac{w(v_{2i},z)}{\lambda(v_{2i},v_{2i-1})}\right\} w(y,v_{2i-1}),$$

[6] We have implicitly assumed here that we know whether it is a path or a cycle that is currently being processed. In fact, it is not automatic that we can distinguish these two possibilities just by looking at M, M' and C. However, by choosing the start points for cycles and paths carefully, the two cases can be disambiguated: just choose the start point of the cycle first, and then use the freedom in the choice of endpoint of the path to avoid the potential ambiguity.

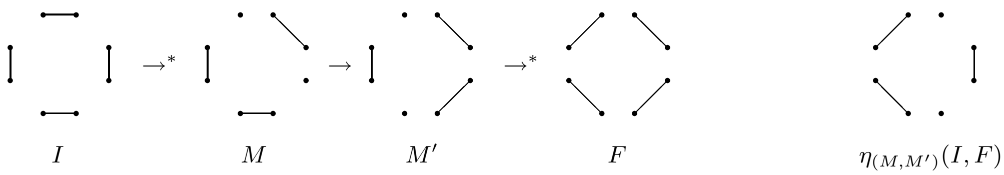

Figure 6: A canonical path through transition $M \to M'$ and its encoding.

which holds by inequality (ii) of Corollary 7. Note that the factor $8 = 2^3$ is determined by this case, since we need to apply inequality (2) three times.

The final case is the last transition $t = (M, M')$ in unwinding an alternating path, where $M' = M \cup (z', z)$. Note that $I, C \in \mathcal{M}(y, z)$, $F, M' \in \mathcal{M}$, $M \in \mathcal{M}(z', z)$ and $\lambda(I)\lambda(F) = \lambda(M')\lambda(C) = \lambda(M)\lambda(z', z)\lambda(C)$. (Here we have written z' for v_{2k}.) Plugging these into inequality (7) leaves us with

$$1 \leq 8 \min\left\{\frac{w(z', z)}{\lambda(z', z)}, 1\right\},$$

which follows from inequality (i) of Corollary 7 and inequality (2).

We have thus shown that the encoding η_t satisfies inequality (7) in all cases. This completes the proof of the lemma. $\square$

The upper bound on path congestion given in Lemma 5 readily yields a lower bound on the conductance:

Corollary 8. *Assuming the weight function w satisfies inequality (2) for all $(y, z) \in V_1 \times V_2$, then $\Phi \geq 1/100\varrho^3 n^4 \geq 1/10^6 m^3 n^4$.*

Proof. Set $\alpha = 1/10\varrho n^2$. Let $\mathcal{S}, \overline{\mathcal{S}}$ be a partition of the state-space. (Note that we do not assume that $\pi(\mathcal{S}) \leq \pi(\overline{\mathcal{S}})$.) We distinguish two cases, depending on whether or not the perfect matchings $\mathcal{M}$ are fairly evenly distributed between $\mathcal{S}$ and $\overline{\mathcal{S}}$. If the distribution is fairly even, then we can show $\Phi(\mathcal{S})$ is large by considering type A canonical paths, and otherwise by using the type B paths.

Case I. $\pi(\mathcal{S} \cap \mathcal{M})/\pi(\mathcal{S}) \geq \alpha$ and $\pi(\overline{\mathcal{S}} \cap \mathcal{M})/\pi(\overline{\mathcal{S}}) \geq \alpha$. Just looking at canonical paths of type A we have a total flow of $\pi(\mathcal{S} \cap \mathcal{M})\pi(\overline{\mathcal{S}} \cap \mathcal{M}) \geq \alpha^2 \pi(\mathcal{S})\pi(\overline{\mathcal{S}})$ across the cut. Thus $\varrho\, Q(\mathcal{S}, \overline{\mathcal{S}}) \geq \alpha^2 \pi(\mathcal{S})\pi(\overline{\mathcal{S}})$, and $\Phi(\mathcal{S}) \geq \alpha^2/\varrho = 1/100\varrho^3 n^4$.

Case II. Otherwise (say) $\pi(\mathcal{M} \cap \mathcal{S})/\pi(\mathcal{S}) < \alpha$. Note the following estimates:

$$\begin{aligned}
\pi(\mathcal{M}) &\geq \frac{1}{4n^2+1} \geq \frac{1}{5n^2};\\
\pi(\mathcal{S} \cap \mathcal{M}) &< \alpha\pi(\mathcal{S}) < \alpha;\\
\pi(\mathcal{S} \setminus \mathcal{M}) &= \pi(\mathcal{S}) - \pi(\mathcal{S} \cap \mathcal{M}) > (1-\alpha)\pi(\mathcal{S}).
\end{aligned}$$

Consider the cut $\mathcal{S} \setminus \mathcal{M} : \overline{\mathcal{S}} \cup \mathcal{M}$. The weight of canonical paths (all chargeable as they cross the cut) is $\pi(\mathcal{S} \setminus \mathcal{M})\pi(\mathcal{M}) \geq (1-\alpha)\pi(\mathcal{S})/5n^2 \geq \pi(\mathcal{S})/6n^2$. Hence $\varrho\, Q(\mathcal{S} \setminus \mathcal{M}, \overline{\mathcal{S}} \cup M) \geq \pi(\mathcal{S})/6n^2$. Noting $Q(\mathcal{S} \setminus \mathcal{M}, \mathcal{S} \cap M) \leq \pi(\mathcal{S} \cap M) \leq \alpha\pi(\mathcal{S})$ we have

$$\begin{aligned}
Q(\mathcal{S}, \overline{\mathcal{S}}) &\geq Q(\mathcal{S} \setminus \mathcal{M}, \overline{\mathcal{S}})\\
&= Q(\mathcal{S} \setminus \mathcal{M}, \overline{\mathcal{S}} \cup M) - Q(\mathcal{S} \setminus \mathcal{M}, \mathcal{S} \cap M)\\
&\geq (1/6\varrho n^2 - \alpha)\pi(\mathcal{S})\\
&\geq \pi(\mathcal{S})/15\varrho n^2\\
&\geq \pi(\mathcal{S})\pi(\overline{\mathcal{S}})/15\varrho n^2.
\end{aligned}$$

Rearranging, $\Phi(\mathcal{S}) = Q(\mathcal{S}, \overline{\mathcal{S}})/\pi(\mathcal{S})\pi(\overline{\mathcal{S}}) \geq 1/15\varrho n^2$.

Clearly, it is Case I that dominates, giving the claimed bound on Φ. $\square$

initialize $\lambda(e) \leftarrow a_{\max}$ for all $e \in V_1 \times V_2$
initialize $w(u, v) \leftarrow n a_{\max}$ for all $(u, v) \in V_1 \times V_2$
while $\exists e$ with $\lambda(e) > a(e)$ do
  set $\lambda(e) \leftarrow \max\{\lambda(e)\exp(-1/2), a(e)\}$
  take S samples from MC with parameters λ, w, each after a simulation of T steps
  use the sample to obtain estimates $w'(u, v)$ satisfying condition (5) with high probability $\forall u, v$
  set $w(u, v) \leftarrow w'(u, v)\ \forall u, v$
output final weights $w(u, v)$

Figure 7: The algorithm for non-negative entries.

Our main result, Theorem 4 of the previous section, now follows immediately.

Proof of Theorem 4. The condition that the starting state is one of maximum activity ensures $\log(\pi(X_0)^{-1}) = O(n \log n)$, where X_0 in the initial state. The lemma now follows from Corollary 8 and Theorem 3. $\square$

5. ARBITRARY NON-NEGATIVE ENTRIES

Our algorithm easily extends to compute the permanent of an arbitrary matrix A with non-negative entries. Let $a_{\max} = \max_{i,j} a(i, j)$ and $a_{\min} = \min_{i,j:a(i,j)>0} a(i, j)$. Assuming $\mathrm{per}(A) > 0$, then it is clear that $\mathrm{per}(A) \geq (a_{\min})^n$. Rounding zero entries $a(i, j)$ to $(a_{\min})^n/n!$, the algorithm follows as described in figure 7.

The running time of this algorithm is polynomial in n and $\log(a_{\max}/a_{\min})$. For completeness, we provide a strongly polynomial time algorithm, i.e., one whose running time is polynomial in n and independent of $a_{\max}$ and $a_{\min}$, assuming arithmetic operations are treated as unit cost. The algorithm of Linial, Samorodnitsky and Wigderson [10] converts, in strongly polynomial time, the original matrix A into a nearly doubly stochastic matrix B such that $1 \geq \mathrm{per}(B) \geq \exp(-n - o(n))$ and $\mathrm{per}(B) = \alpha\,\mathrm{per}(A)$ where α is an easily computable function. Thus it suffices to consider the computation of $\mathrm{per}(B)$, in which case we can afford to round up any entries smaller than, say, n^{-2n} to n^{-2n}. The analysis for the 0,1 case now applies with the same running time.

6. OTHER APPLICATIONS

Several other interesting counting problems are reducible (via approximation-preserving reductions) to the 0,1 permanent. These were not accessible by the earlier approximation algorithms for restricted cases of the permanent because the reductions yield a matrix A whose corresponding graph G_A may have a disproportionate number of near-perfect matchings. We close the paper with two such examples.

The first example makes use of a reduction due to Tutte [14]. A perfect matching in a graph G may be viewed as a spanning[7] subgraph of G, all of whose vertices have degree 1. More generally, we may consider spanning subgraphs whose vertices all have specified degrees, not necessarily 1. The construction of Tutte reduces an instance of this more general problem to the special case of perfect matchings. Jerrum and Sinclair [5] exploited the fact that this reduction preserves the *number* of solutions (modulo a constant factor) to approximate the number of degree-constrained subgraphs of a graph in a restricted setting.[8] Combining the same reduction with Theorem 1 yields the following unconditional result.

COROLLARY 9. *For an arbitrary bipartite graph G, there exists an fpras for computing the number of labelled subgraphs of G with a specified degree sequence.*

As a special case, of course, we obtain an fpras for the number of labelled bipartite graphs with specified degree sequence.[9]

The second example concerns the notion of a "0,1 flow". Consider a directed graph $G = (V, E)$, where the in-degree (respectively, out-degree) of a vertex $v \in V$ is denoted by $d_G^-(v)$ (respectively, $d_G^+(v)$). A *0,1 flow* is defined as a subset of edges $E' \subseteq E$ such that, in the resulting subgraph $H = (V, E')$, $d_H^-(v) = d_H^+(v)$ for all $v \in V$. Counting the 0,1 flows in G is reducible to counting the matchings in an undirected bipartite graph. Specifically, let $\widehat{G} = (\widehat{V}, \widehat{E})$ be the graph with the following vertex and edge sets:

$$\begin{aligned}\widehat{V} &= \{h_{i,j}, m_{i,j}, t_{i,j} : \forall i,j \text{ with } (v_i, v_j) \in E\} \\ &\qquad \cup \{u_i^1, \ldots, u_i^{d_G^+(v_i)} : \forall i \text{ with } v_i \in V\}, \\ \widehat{E} &= \{\{h_{i,j}, m_{i,j}\}, \{m_{i,j}, t_{i,j}\} : \forall i,j \text{ with } (v_i, v_j) \in E\} \\ &\qquad \cup \{\{u_i^k, h_{i,j}\}, \{t_{i,j}, u_j^{k'}\} : \forall i,j \text{ with } (v_i, v_j) \in E, \\ &\qquad\qquad \forall k, k' \text{ with } 1 \le k \le d_G^+(v_i), 1 \le k' \le d_G^+(v_j)\}.\end{aligned}$$

A 0,1 flow E' in G corresponds to a set of perfect matchings M in $\widehat{G}$ in the following manner. For each $(v_i, v_j) \in E'$ add the edge $\{h_{i,j}, m_{i,j}\}$ to M, while for each $(v_i, v_j) \in E \setminus E'$ add the edge $\{m_{i,j}, t_{i,j}\}$ to M. Now for $v_i \in V$, observe that the set of vertices $\{h_{i,j}\}_j \cup \{t_{j',i}\}_{j'}$, consists of exactly $d_H^-(v_i) + d_G^+(v_i) - d_H^+(v_i) = d_G^+(v_i)$ unmatched vertices. There are $d_G^+(v_i)!$ ways of pairing these unmatched vertices with the set of vertices $\{u_i^k\}_k$. Thus the flow E' corresponds to $\prod_{v \in V} d_G^+(v)!$ perfect matchings in $\widehat{G}$, and it is clear that every perfect matching in $\widehat{G}$ corresponds to precisely one 0,1 flow. This implies the following corollary.

COROLLARY 10. *For an arbitrary directed graph G, there exists an fpras for counting the number of 0,1 flows.*

Suppose the directed graph G has a fixed source s and sink t. After adding a simple gadget from t to s we can estimate the number of maximum 0,1 flows from s to t of given value by estimating the number of 0,1 flows in the resulting graph.

Finally, it is perhaps worth observing that the "six-point ice model" on an undirected graph G may be viewed as a 0,1 flow on an appropriate orientation of G, giving us an alternative approach to the problem of counting ice configurations considered by Mihail and Winkler [11].

[7] A subgraph of G is *spanning* if it includes all the vertices of G; note that a spanning subgraph is not necessarily connected.

[8] More specifically, both the actual vertex degrees of the graph and the desired vertex degrees of the subgraph had to satisfy a certain algebraic relationship.

[9] Note that this special case is not known to be #P-complete, and hence may conceivably be solvable *exactly* in polynomial time. It seems likely, however, that an fpras is the best that can be achieved.

7. REFERENCES

[1] Noga Alon and Joel Spencer, *The Probabilistic Method*, John Wiley, 1992.

[2] Alexander Barvinok, Polynomial time algorithms to approximate permanents and mixed discriminants within a simply exponential factor, *Random Structures and Algorithms* **14** (1999), 29–61.

[3] Andrei Z. Broder, How hard is it to marry at random? (On the approximation of the permanent), *Proceedings of the 18th Annual ACM Symposium on Theory of Computing* (STOC), ACM Press, 1986, 50–58. Erratum in *Proceedings of the 20th Annual ACM Symposium on Theory of Computing*, 1988, p. 551.

[4] Mark Jerrum and Alistair Sinclair, Approximating the permanent, *SIAM Journal on Computing* **18** (1989), 1149–1178.

[5] Mark Jerrum and Alistair Sinclair, Fast uniform generation of regular graphs, *Theoretical Computer Science* **73** (1990), 91–100.

[6] Mark Jerrum and Alistair Sinclair, The Markov chain Monte Carlo method: an approach to approximate counting and integration. In *Approximation Algorithms for NP-hard Problems* (Dorit Hochbaum, ed.), PWS, 1996, 482–520.

[7] Mark Jerrum, Leslie Valiant and Vijay Vazirani, Random generation of combinatorial structures from a uniform distribution, *Theoretical Computer Science* **43** (1986), 169–188.

[8] Mark Jerrum and Umesh Vazirani, A mildly exponential approximation algorithm for the permanent, *Algorithmica* **16** (1996), 392–401.

[9] P. W. Kasteleyn, The statistics of dimers on a lattice, I., The number of dimer arrangements on a quadratic lattice, *Physica* **27** (1961) 1664–1672.

[10] Nathan Linial, Alex Samorodnitsky and Avi Wigderson, A deterministic strongly polynomial algorithm for matrix scaling and approximate permanents, *Combinatorica* **20** (2000), 545–568.

[11] Milena Mihail and Peter Winkler, On the number of Eulerian orientations of a graph, *Algorithmica* **16**

(1996), 402–414.
[12] Henryk Minc, Permanents, *Encyclopedia of Mathematics and its Applications* **6** (1982), Addison-Wesley Publishing Company.
[13] Rajeev Motwani and Prabhakar Raghavan, *Randomized Algorithms*, Cambridge University Press, 1995.
[14] W. T. Tutte, A short proof of the factor theorem for finite graphs, *Canadian Journal of Mathematics* **6** (1954) 347–352.
[15] L. G. Valiant, The complexity of computing the permanent, *Theoretical Computer Science* **8** (1979), 189–201.