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

Mark Jerrum† Alistair Sinclair‡ Eric Vigoda§

June 27, 2003

Abstract

We present a polynomial-time randomized algorithm for estimating the permanent of an arbitrary $n \times n$ matrix with non-negative entries. This algorithm — technically a "fully-polynomial randomized approximation scheme" — computes an approximation that is, with high probability, within arbitrarily small specified relative error of the true value of the permanent.

*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. For most of this work, the third author was with the University of Edinburgh. 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 appeared in *Proceedings of the 33rd ACM Symposium on the Theory of Computing*, July 2001, pp. 712–721.

†School of Informatics, University of Edinburgh, The King's Buildings, Edinburgh EH9 3JZ, United Kingdom. Email: mrj@dcs.ed.ac.uk

‡Computer Science Division, University of California at Berkeley, Berkeley, CA 94720-1776, USA. Email: sinclair@cs.berkeley.edu

§Department of Computer Science, University of Chicago, Chicago, IL 60637, USA. Email: vigoda@cs.uchicago.edu

1 Problem description and history

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

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

where the sum is over all permutations σ 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 [20] for a comprehensive history). Despite many attempts, an efficient algorithm for general matrices has proved elusive. Indeed, Ryser's algorithm [22] 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 the publication of Kasteleyn's algorithm for counting perfect matchings in planar graphs [15], which uses just $O(n^3)$ arithmetic operations.

This lack of progress was explained in 1979 by Valiant [26], who proved that exact computation of the permanent is #P-complete, 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 has been 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 [13].

Broder [4] proposed a Markov chain Monte Carlo method for sampling perfect matchings. This was based on simulation of a Markov chain whose state space consists of all perfect and "near-perfect" matchings (i.e., matchings with two uncovered vertices, or "holes") in the graph, and whose stationary distribution is uniform. 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 [10] 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 arbitrarily close multiplicative factor; and what is the best approximation factor achievable in polynomial time? Jerrum and Vazirani [14], building upon the work of Jerrum and Sinclair, presented an approximation algorithm whose running time is $\exp(O(\sqrt{n}))$, 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., [18, 3]). To date the best of these is due to Barvinok [3], and gives $c \approx 1.31$ (see also [5]).

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[1]) 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 [10], except that the presence of the weights necessitates a combinatorially more delicate application of the path-counting technology introduced in [10]. 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

[1] As explained later (see §7), we cannot hope to handle matrices with negative entries as an efficient approximation algorithm for this case would allow one to compute the permanent of a 0,1 matrix exactly in polynomial time.

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 remainder of the paper is organized as follows. In §2 we summarize the necessary background concerning the connection between random sampling and counting, and the Markov chain Monte Carlo method. In §3 we define the Markov chain we will use and present the overall structure of the algorithm, including the computation of hole weights. In §4 we analyze the Markov chain and show that it is rapidly mixing; this is the most technical section of the paper. Section 5 completes the proof of Theorem 1 by detailing the procedure by which the random matchings produced by Markov chain simulation are used to estimate the number of perfect matchings in G_A, and hence the permanent of the associated 0,1-matrix A; this material is routine, but is included for completeness. At this point, the running time of our fpras as a function of n is $\widetilde{O}(n^{11})$; in §6 the dependence on n is reduced to $\widetilde{O}(n^{10})$ by using "warm starts" of the Markov chain.[2] Finally, in §7 we show how to extend the algorithm to handle matrices with arbitrary non-negative entries, and in §8 we observe some applications to constructing an fpras for various other combinatorial enumeration problems.

2 Random sampling and Markov chains

This section provides basic information on the use of rapidly mixing Markov chains to sample combinatorial structures, in this instance, perfect matchings.

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[3]

$$\Pr[e^{-\varepsilon} Z \leq \mathrm{per}(A) \leq e^{\varepsilon} Z] \geq \tfrac{3}{4},$$

[2] The notation $\widetilde{O}(\cdot)$ ignores logarithmic factors, and not merely constants.

[3] The relative error is usually specified as $1 \pm \varepsilon$. We use $e^{\pm\varepsilon}$ (which differs only slightly from $1 \pm \varepsilon$) for algebraic convenience.

and which 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 [13].

To construct an fpras, we will follow a well-trodden path via random sampling. We focus on the 0,1 case; see §7 for an extension to the case of matrices with general non-negative entries. Recall that when A is a 0,1 matrix, per(A) is equal to the number of perfect matchings in the bipartite graph G_A. Now it is well known — see for example [12] — 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

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

where $\mathcal{U}$ is the uniform distribution on perfect matchings of G_A and d_{tv} denotes (total) variation distance.[4] The algorithm is required to run in time polynomial in n and $\log \delta^{-1}$. For completeness, we will flesh out the details of the reduction to random sampling in §5.

The bulk of this paper will be devoted 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 G_A 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. In our application, states are matchings in G_A, and we use M, M' to denote generic elements of Ω. The Markov chain is *irreducible* if for every pair of states $M, M' \in \Omega$, there exists a $t > 0$ such that $P^t(M, M') > 0$ (i.e., all states communicate); it is *aperiodic* if $\gcd\{t : P^t(M, M') > 0\} = 1$ for all $M, M' \in \Omega$. 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., $P^t(M, M') \to \pi(M')$ as $t \to \infty$ for all $M' \in \Omega$, regardless

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

Set $\hat{\delta} \leftarrow \delta/(12n^2+3)$.
Repeat $T = \lceil (6n^2+2)\ln(3/\delta) \rceil$ times:
    Simulate the Markov chain for $\tau(\hat{\delta})$ 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.

of the initial state M. If there exists a probability distribution 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 a 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* (from state M) to be

$$\tau(\delta) = \tau_M(\delta) = \min\left\{t : d_{\mathrm{tv}}(P^t(M,\cdot),\pi) \le \delta\right\}.$$

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 $\tau_M(\hat{\delta})$ steps (where $\hat{\delta}$ is a sufficiently small positive number), starting from some appropriate state M,[5] 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.*

[5] As we shall, see a state is "appropriate" unless it has exceptionally low probability in the stationary distribution. Except on a few occasions when we need to call attention to the particular initial state, we may safely drop the subscript to τ.

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 $d_{\mathrm{tv}}(\hat{\pi}, \pi) \leq \hat{\delta}$. 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). The result follows from the chain of inequalities:

$$\begin{aligned}
\Pr(M \in \mathcal{S}) &\geq \frac{\hat{\pi}(\mathcal{S})}{\hat{\pi}(\mathcal{M})} - (1 - \hat{\pi}(\mathcal{M}))^T \\
&\geq \frac{\pi(\mathcal{S}) - \hat{\delta}}{\pi(\mathcal{M}) + \hat{\delta}} - \exp(-\hat{\pi}(\mathcal{M})T) \\
&\geq \frac{\pi(\mathcal{S})}{\pi(\mathcal{M})} - \frac{2\hat{\delta}}{\pi(\mathcal{M})} - \exp(-(\pi(\mathcal{M}) - \hat{\delta})T) \\
&\geq \frac{\pi(\mathcal{S})}{\pi(\mathcal{M})} - \frac{2\delta}{3} - \frac{\delta}{3}.
\end{aligned}$$

A matching bound $\Pr(M \in \mathcal{S}) \leq \pi(\mathcal{S})/\pi(\mathcal{M}) + \delta$ follows immediately by considering the complementary set $\mathcal{M} \setminus \mathcal{S}$. (Recall that the total variation distance $d_{\mathrm{tv}}(\pi, \hat{\pi})$ between distributions π and $\hat{\pi}$ may be interpreted as the maximum of $|\pi(\mathcal{S}) - \hat{\pi}(\mathcal{S})|$ over all events $\mathcal{S}$.) □

The running time of the random sampler is determined by the mixing time of the Markov chain. We will derive an upper bound on $\tau(\delta)$ 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 $\delta > 0$, $\tau(\delta)$ 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 define a multicommodity flow in the underlying graph of the Markov chain and bound its associated congestion. The graph of interest is $G_P = (\Omega, E_P)$, where $E_P := \{(M, M') : P(M, M') > 0\}$ denotes the set of possible transitions.[6] For all ordered pairs $(I, F) \in \Omega^2$ of "initial" and "final" states, let $\mathcal{P}_{I,F}$ denote a collection of simple directed paths in G_P from I to F. In this paper, we call

[6] Although G_P is formally a directed graph, its edges occur in anti-parallel pairs, by time-reversibility.

$f_{I,F} : \mathcal{P}_{I,F} \to \mathbb{R}^+$ a *flow* from I to F if the following holds:

$$\sum_{p \in \mathcal{P}_{I,F}} f_{I,F}(p) = \pi(I)\pi(F).$$

A flow for the entire Markov chain is a collection $f = \{f_{I,F} : I, F \in \Omega\}$ of individual flows, one for each pair $I, F \in \Omega$. Our aim is to design a flow f which has small *congestion* ϱ, defined as

$$\varrho = \varrho(f) = \max_{t=(M,M') \in E_P} \varrho_t, \tag{1}$$

where

$$\varrho_t = \frac{1}{Q(t)} \sum_{I,F \in \Omega} \sum_{p\,:\,t \in p \in \mathcal{P}_{I,F}} f_{I,F}(p)\,|p|, \tag{2}$$

and $|p|$ denotes the length of (i.e., number of edges contained within) the path p. Here $Q(t) = Q(M, M') = \pi(M)P(M, M')$, as defined earlier.

The following bound relating congestion and mixing time is standard; the version presented here is due to Sinclair [24], building on work of Diaconis and Stroock [7].

Theorem 3 *For an ergodic, reversible Markov chain with self-loop probabilities $P(M, M) \geq 1/2$ for all states M, and any initial state $M_0 \in \Omega$,*

$$\tau_{M_0}(\delta) \leq \varrho \left(\ln \pi(M_0)^{-1} + \ln \delta^{-1} \right).$$

Thus to prove rapid mixing it suffices to demonstrate a flow with an upper bound of the form poly(n) on its congestion for our Markov chain on matchings. (The term $\ln \pi(M_0)^{-1}$ will not cause a problem, since the total number of states will be at most $(n+1)!$, and we will start in a state M_0 that maximizes $\pi(M_0)$.)

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, through standard considerations spelled out in §5, yield an fpras for the permanent of an arbitrary 0,1 matrix, and hence Theorem 1. The (easy) extension to matrices with arbitrary non-negative entries is described in §7.

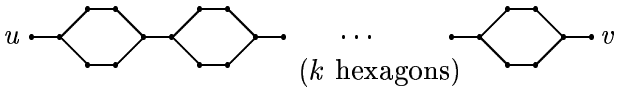

Figure 2: A graph with $|\mathcal{M}(u,v)|/|\mathcal{M}|$ exponentially large.

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 $\mathcal{M}(u,v)$ denote the set of near-perfect matchings with holes only at the vertices $u \in V_1$ and $v \in V_2$. The state space of MC is $\Omega := \mathcal{M} \cup \bigcup_{u,v} \mathcal{M}(u,v)$. Previous work [4, 10] 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 inverse polynomial in n. In [10] it was proved that this condition — rather surprisingly — is also sufficient to imply 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 $n/2$ 1's in each row and column), and a.e.[7] (almost every) random matrix [10], as well as matrices corresponding to vertex transitive graphs (including regular lattices, an important case for applications in statistical physics) [16].

On the other hand, 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 probability $\Omega(1/n^2)$ in total to perfect matchings.

It will actually prove technically convenient to introduce edge weights also. Thus for each edge $(u,v) \in E$, we introduce a positive weight $\lambda(u,v)$,

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

which we call its *activity*. We extend the notion of activities to matchings M (of any cardinality) by $\lambda(M) = \prod_{(u,v)\in M} \lambda(u,v)$. Similarly, for a set of matchings $\mathcal{S}$ we define $\lambda(\mathcal{S}) = \sum_{M\in\mathcal{S}} \lambda(M)$.[8] 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 $\xi \leq 1/n!$ 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

$$\Lambda(M) = \begin{cases} \lambda(M)w(u,v) & \text{if } M \in \mathcal{M}(u,v) \text{ for some } u,v; \\ \lambda(M), & \text{if } M \in \mathcal{M}. \end{cases} \tag{3}$$

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 a slight variant of the original chain of [4, 10] augmented with a Metropolis acceptance rule for the transitions. (The chain has been modified in order to save a factor of n from its mixing time on the complete bipartite graph.) The transitions from a matching M are defined as follows:

1. If $M \in \mathcal{M}$, choose an edge $e = (u,v)$ uniformly at random from M; set $M' = M \setminus e$.

2. If $M \in \mathcal{M}(u,v)$, choose z uniformly at random from $V_1 \cup V_2$.

   (i) if $z \in \{u,v\}$ and $(u,v) \in E$, let $M' = M \cup (u,v)$;

   (ii) if $z \in V_2$, $(u,z) \in E$ and $(x,z) \in M$, let $M' = M \cup (u,z) \setminus (x,z)$;

   (iii) if $z \in V_1$, $(z,v) \in E$ and $(z,y) \in M$, let $M' = M \cup (z,v) \setminus (z,y)$;

   (iv) otherwise, let $M' = M$.

3. With probability $\min\{1, \Lambda(M')/\Lambda(M)\}$ go to M'; otherwise, stay at M.

Thus the non-null transitions are of three types: *removing* an edge from a perfect matching (case 1); *adding* an edge to a near-perfect matching (case 2(i)); and *exchanging* an edge of a near-perfect matching with another edge adjacent to one of its holes (cases 2(ii) and 2(iii)).

[8] Note that if we set $\lambda(u,v)$ equal to the matrix entry $a(u,v)$ for every edge (u,v), then $\mathrm{per}(A)$ is exactly equal to $\lambda(\mathcal{M})$. Thus our definition is natural.

The *proposal probabilities* defined in steps 1 and 2 for selecting the candidate matching M' are symmetric, being $1/n$ in the case of moves between perfect and near-perfect matchings, and $1/2n$ between near-perfect matchings. This fact, combined with the *Metropolis rule* for accepting the move to M' applied in step 3, ensures that the 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

$$w^*(u,v) = \frac{\lambda(\mathcal{M})}{\lambda(\mathcal{M}(u,v))} \tag{4}$$

for each pair of holes u, v with $\mathcal{M}(u,v) \neq \emptyset$. (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 the event, we will not know w^* exactly but will content ourselves with weights w satisfying

$$w^*(u,v)/2 \leq w(u,v) \leq 2w^*(u,v), \tag{5}$$

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

The main technical result of this paper is the following theorem, which says that, provided the weight function w satisfies condition (5), the Markov chain is rapidly mixing. The theorem will be proved in the next section.

Theorem 4 *Assuming the weight function w satisfies inequality (5) for all $(u,v) \in V_1 \times V_2$ with $\mathcal{M}(u,v) \neq \emptyset$, then the mixing time of the Markov chain MC is bounded above by $\tau_M(\delta) = O\big(n^6\big(\ln(1/\pi(M)) + \log \delta^{-1}\big)\big)$.*

Finally we need to address the issue of computing (approximations to) the weights w^* defined in (4). 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 assign an activity of 1 to all edges $e \in E$ (i.e., all edges of our graph G), and ultimately 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) is at most $1/n!$ and there are at most $n!$ possible matchings, the combined weight of all invalid matchings is at most 1. Assuming the graph has at least one perfect matching, the invalid matchings 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 $\lambda_G(e) = 1$ for all $e \in E$, and $\lambda_G(e) = 1/n!$ 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 $\lambda_G(e) = 1/n!$ in a sequence of phases, in each of which, for some vertex v, the activities $\lambda(e)$ for all non-edges $e \notin E$ which are incident to v are updated to $\lambda'(e)$, where $\exp(-1/2)\lambda(e) \leq \lambda'(e) \leq \exp(1/2)\lambda(e)$. (In this application, we only ever need to reduce the activities, and never increase them, but the added generality costs us nothing.)

We assume at the beginning of the phase that condition (5) is satisfied; in other words, $w(u, v)$ approximates $w^*(u, v)$ within ratio 2 for all pairs (u, v).[9] Before updating an activity, we must consolidate our position by finding, for each pair (u, v), a better approximation to $w^*(u, v)$: 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

$$\frac{w(u, v)}{w^*(u, v)} = \frac{\pi(\mathcal{M}(u, v))}{\pi(\mathcal{M})}, \tag{6}$$

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 δ of π. We shall see presently that setting $\delta = O(n^{-2})$ suffices in the current situation; certainly, the exact value of δ clearly does not affect the leading term in the mixing time promised by Theorem 4. So suppose we generate S samples from $\hat{\pi}$, and for each pair $(u, v) \in V_1 \times V_2$

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

we consider the proportion $\alpha(u,v)$ of samples with hole pair u,v, together with the proportion α of samples that are perfect matchings. Clearly,

$$\mathbb{E}\,\alpha(u,v) = \hat{\pi}(\mathcal{M}(u,v)) \quad \text{and} \quad \mathbb{E}\,\alpha = \hat{\pi}(\mathcal{M}). \tag{7}$$

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 $\hat{\eta}$ 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 $1-\hat{\eta}$. Furthermore, if δ 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 (6) we have, with probability at least $1-(n^2+1)\hat{\eta}$, 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 δ that make this all work. Condition (5) entails

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

Assuming $\delta \leq 1/8(n^2+1)$, it follows from any of the standard Chernoff bounds (see, e.g., [2] or [21, Thms 4.1 & 4.2]), that $O(n^2\log(1/\hat{\eta}))$ 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 $1-\hat{\eta}$. Again using the fact that $\pi(\mathcal{M}(u,v)) \geq 1/4(n^2+1)$, we see that $\hat{\pi}(\mathcal{M}(u,v))$ will approximate $\pi(\mathcal{M}(u,v))$ within ratio $c^{1/4}$ provided $\delta \leq c_1/n^2$ where $c_1 > 0$ is a sufficiently small constant. (Note that we also satisfy the earlier constraint on δ by this setting.) Therefore, taking $c = 6/5$ and using $S = O(n^2\log(1/\hat{\eta}))$ samples, we obtain refined estimates $w(u,v)$ satisfying

$$5w^*(u,v)/6 \leq w(u,v) \leq 6w^*(u,v)/5 \tag{8}$$

with probability $1-(n^2+1)\hat{\eta}$. Plugging $\delta = c_1/n^2$ into Theorem 4, the number of steps required to generate each sample is $T = O(n^7\log n)$, provided we use a starting state that is reasonably likely in the stationary distribution; and the total time to update all the weights $w(u,v)$ is $O(n^9\log n\log(1/\hat{\eta}))$.

We can then update the activity of all non-edges e incident at a common vertex v by changing $\lambda(e)$ by a multiplicative factor of $\exp(-1/2)$. Since a matching contains at most one edge incident to v, the effect of this updating 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 (8) actually satisfy the weaker

Initialize $\lambda(u,v) \leftarrow 1$ for all $(u,v) \in V_1 \times V_2$.
Initialize $w(u,v) \leftarrow n$ for all $(u,v) \in V_1 \times V_2$.
While there exists a pair y, z with $\lambda(y,z) > \lambda_G(y,z)$ do:
    Take a sample of size S from MC with parameters λ, w,
        using a simulation of T steps in each case.
    Use the sample to obtain estimates $w'(u,v)$ satisfying
        condition (8), for all u, v, with high probability.
    Set $\lambda(y,v) \leftarrow \max\{\lambda(y,v)\exp(-1/2),\ \lambda_G(y,v)\}$, for all $v \in V_2$,
        and $w(u,v) \leftarrow w'(u,v)$ for all u, v.
Output the final weights $w(u,v)$.

Figure 3: The algorithm for approximating the ideal weights.

condition (5) for the *new* activities as well, so we can proceed with the next phase. The algorithm is sketched in Figure 3.

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 entire process requires $O(n^2 \log n)$ phases, since there are n vertices in V_1, and $O(\log n!) = O(n \log n)$ phases are required to reduce the activities of edges incident at each of these vertices to their final values. We have seen that each phase takes time $O(n^9 \log n \log(1/\hat{\eta}))$. Thus the overall running time of the algorithm for computing approximate weights is $O(n^{11}(\log n)^2 \log(1/\hat{\eta}))$. It only remains to choose $\hat{\eta}$.

Recall that $\hat{\eta}$ is the failure probability on each occasion that we use a sample mean to estimate an expectation. If we are to achieve overall failure probability η then we must set $\hat{\eta} = O(\eta/(n^4 \log n))$, since there are $O(n^4 \log n)$ individual estimates to make in total. Thus

Lemma 5 *The algorithm of Figure 3 finds approximations $w(\cdot,\cdot)$ within a constant ratio of the ideal weights $w^*_G(\cdot,\cdot)$ associated with the desired activities λ_G in time $O\big(n^{11}(\log n)^2(\log n + \log \eta^{-1})\big)$, with failure probability η.*

Although it is not a primary aim of this paper to push exponents down as far as possible, we note that it is possible to reduce the running time in Lemma 5 from $\widetilde{O}(n^{11})$ to $\widetilde{O}(n^{10})$ using a standard artifice. We have seen that the number of simulation steps to generate a sample is at most $T = O(n^7 \log n)$, if we start from, say, a perfect matching M_0 of maximum activity. However, after generating an initial sample M for a phase, we are

only observing the hole pattern of M. Thus the matching M is still random with respect to its hole pattern. By starting our Markov chain from this previous sample M, we have what is known as a "warm start," in which case generating a sample requires only $O(n^6)$ simulation steps. We expand on this point in §6.

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 η so that the overall failure probability is strictly less than δ, say $\eta = \delta/2$. At the conclusion of the initialization algorithm, we have a good approximation to the ideal weights w_G^* for our desired activities λ_G. We can then simply simulate the Markov chain with these parameters to generate perfect matchings from a distribution within variation distance $\delta/2$ of uniform. By Theorem 4 the (expected) additional time required to generate such a sample is $O(n^8(n \log n + \log \delta^{-1}))$, which is negligible in comparison with the initialization procedure. (The extra factor n^2 represents the expected number of samples before a perfect matching is seen.) If we are interested in the worst-case time to generate a perfect matching, we can see from Lemma 2 that it will be $O(n^8(n \log n + \log \delta^{-1}) \log \delta^{-1})$. Again, this is dominated by the initialization procedure. Indeed the domination is so great that we could generate a sample of $\widetilde{O}(n^2)$ perfect matchings in essentially the same time bound. Again, all time bounds may be reduced by a factor $\widetilde{O}(n)$ by using warm starts.

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. Following Theorem 3, we can get an upper bound on the mixing time by defining a flow and bounding its congestion. To do this, we shall use technology introduced in [10], and since applied successfully in several other examples. The idea in its basic form is to define a *canonical path* $\gamma_{I,F}$ from each state $I \in \Omega$ to every other state $F \in \Omega$, so that no transition carries an undue burden of paths. These canonical paths then define the flow $f_{I,F}$ for all ordered pairs (I, F) by simply setting $f_{I,F}(\gamma_{I,F}) = \pi(I)\pi(F)$. By upper bounding the maximum number of such paths that pass through any particular transition, one obtains an upper bound on the congestion created by such a flow.

In the current application we can significantly reduce the technical complexity of this last step by defining canonical paths only for states $I \in \mathcal{N} := \Omega \setminus \mathcal{M}$ to states in $F \in \mathcal{M}$, i.e., from near-perfect to perfect matchings. Thus

only flows from $I \in \mathcal{N}$ to $F \in \mathcal{M}$ will be defined directly. Flows from $I \in \mathcal{M}$ to $F \in \mathcal{N}$ can safely be routed along the reversals of the canonical paths, by time-reversibility. Flows from I to F with $I, F \in \mathcal{N}$ will be routed via a random state $M \in \mathcal{M}$ using the canonical path $\gamma_{I,M}$ and the reversal of the path $\gamma_{F,M}$. Flows with $I, F \in \mathcal{M}$ will similarly be routed through a random state $M \in \mathcal{N}$. Provided — as is the case here — both $\mathcal{N}$ and $\mathcal{M}$ have non-negligible probability, the congestion of the flow thus defined will not be too much greater than that of the canonical paths. This part of the argument is given quantitative expression in Lemma 9, towards the end of the section. First, though, we proceed to define the set $\Gamma = \{\gamma_{I,F} : (I, F) \in \mathcal{N} \times \mathcal{M}\}$ of canonical paths and bound its congestion.

The canonical paths are defined by superimposing I and F. Since $I \in \mathcal{M}(y, z)$ for some $(y, z) \in V_1 \times V_2$, and $F \in \mathcal{M}$ we see that $I \oplus F$ consists of a collection of alternating cycles together with a single alternating path from y to z. 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 to F is obtained by first "unwinding" the path and then "unwinding" the cycles in the canonical order.

For convenience, denote by $\sim$ the relation between vertices of being connected by an edge in G. The alternating path $y = v_0 \sim \ldots \sim v_{2k+1} = z$ is unwound by: (i) successively, for each $0 \leq i \leq k-1$, exchanging the edge (v_{2i}, v_{2i+1}) for the edge (v_{2i+1}, v_{2i+2}); and finally (ii) adding the edge (v_{2k}, v_{2k+1}).

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 $1 \leq i \leq k-1$, exchanging the edge (v_{2i-1}, v_{2i}) for the edge (v_{2i}, v_{2i+1}); and finally (iii) adding the edge (v_{2k-1}, v_{2k}). (Refer to Figure 4.)

For each transition t denote by

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

the set of canonical paths using that transition. We define the *congestion* of Γ as

$$\varrho(\Gamma) := \max_t \left\{ \frac{L}{Q(t)} \sum_{(I,F) \in \mathrm{cp}(t)} \pi(I)\pi(F) \right\}, \tag{9}$$

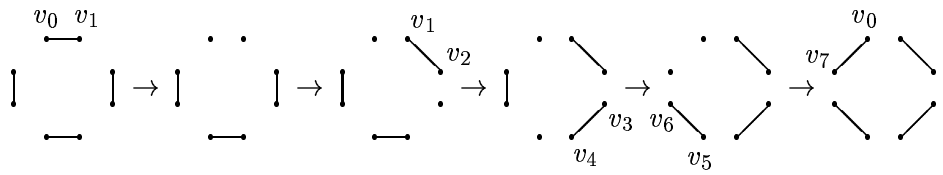

Figure 4: Unwinding a cycle with $k = 4$.

where L is an upper bound on the length $|\gamma_{I,F}|$ of any canonical path, and t ranges over all transitions. This is consistent with our earlier definition (2) when each flow $f_{I,F}$ is supported on the canonical path $\gamma_{I,F}$, and the canonical paths are restricted to pairs $(I, F) \in \mathcal{N} \times \mathcal{M}$.

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 flow for all $I, F \in \Omega$ with a suitable upper bound on its congestion (see Lemma 9 below) and hence, via Theorem 3, a bound on the mixing time.

Lemma 6 *Assuming the weight function w satisfies inequality (5) for all $(u, v) \in V_1 \times V_2$, then $\varrho(\Gamma) \leq 48n^4$.*

In preparation for proving Lemma 6, we establish some combinatorial inequalities concerning weighted matchings with at most four holes that will be used in the proof. These inequalities are generalizations of those used in [16]. Before stating the inequalities, we need to extend our earlier definitions to matchings with four holes. For distinct vertices $u, y \in V_1$ and $v, z \in V_2$, let $\mathcal{M}(u, v, y, z)$ denote the set of matchings whose holes are exactly the vertices u, v, y, z. For $M \in \mathcal{M}(u, v, y, z)$, let $w(M) = w(u, v, y, z)$ where

$$w(u, v, y, z) = w^*(u, v, y, z) := \lambda(\mathcal{M})/\lambda(\mathcal{M}(u, v, y, z)).$$

Since the four-hole matchings are merely a tool in our analysis, we can set $w = w^*$ for these hole patterns. We also set $\Lambda(M) = \lambda(M)w(u, v, y, z)$.

Lemma 7 *Let G be as above, and let $u, y, y' \in V_1$ and $v, z, z' \in V_2$ be distinct vertices. Suppose that $u \sim v$. Then*

(i) $\lambda(u, v)\lambda(\mathcal{M}(u, v)) \leq \lambda(\mathcal{M})$;

(ii) $\lambda(u, v)\lambda(\mathcal{M}(u, v, y, z)) \leq \lambda(\mathcal{M}(y, z))$;

(iii) $\lambda(u, v)\lambda(\mathcal{M}(u, z))\lambda(\mathcal{M}(y, v)) \leq \lambda(\mathcal{M})\lambda(\mathcal{M}(y, z))$; *and*

(iv) $\lambda(u,v)\lambda(\mathcal{M}(u,z,y',z')\lambda(\mathcal{M}(y,v)) \leq \lambda(\mathcal{M}(y',z'))\lambda(\mathcal{M}(y,z)) + \lambda(\mathcal{M}(y',z))\lambda(\mathcal{M}(y,z')).$

Proof. The mapping from $\mathcal{M}(u,v,y,z)$ to $\mathcal{M}(y,z)$, or 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 observation dispenses with (i) and (ii).

Part (iii) is essentially a degenerate version of (iv), so we'll deal with the latter first. Our basic strategy is to define an injective map

$$\mathcal{M}(u,z,y',z') \times \mathcal{M}(y,v) \to (\mathcal{M}(y',z') \times \mathcal{M}(y,z)) \cup (\mathcal{M}(y',z) \times \mathcal{M}(y,z'))$$

that preserves activities. Suppose $M_{u,z,y',z'} \in \mathcal{M}(u,z,y',z')$ and $M_{y,v} \in \mathcal{M}(y,v)$, and consider the superposition of $M_{u,z,y',z'}$, $M_{y,v}$ and the single edge (u,v). Observe that $M_{u,z,y',z'} \oplus M_{y,v} \oplus \{(u,v)\}$ decomposes into a collection of cycles together with either: a pair of even-length paths, one joining y to y' and the other joining z to z'; or a pair of odd-length paths, one joining y to z (respectively z') and the other joining y' to z' (respectively z).[10]

First, consider the case of a pair of even-length paths. Let Π be the path that joins z to z', and let $\Pi = \{e_0, e_1, \ldots, e_{2k-1}\}$ be an enumeration of the edges of Π, starting at z. Note that Π is necessarily the path that contains the edge $\{u,v\}$. (The edges e_0 and e_{2k-1} come from the same matching, $M_{y,v}$. Parity dictates that Π cannot be a single alternating path, so it must be composed of two such, joined by the edge $\{u,v\}$.) Denote by Π_0 the k even edges of Π, and by Π_1 the k odd edges. Finally define a mapping from $\mathcal{M}(u,z,y',z') \times \mathcal{M}(y,v)$ to $\mathcal{M}(y',z') \times \mathcal{M}(y,z)$ by $(M_{u,z,y',z'}, M_{y,v}) \mapsto (M_{y',z'}, M_{y,z})$, where $M_{y',z'} := M_{u,z,y',z'} \cup \Pi_0 \setminus \Pi_1$ and $M_{y,z} := M_{y,v} \cup \Pi_1 \setminus \Pi_0$.

Now consider the case of odd-length paths. Let Π be the path with one endpoint at y. (Note that this must be the path that contains the edge $\{u,v\}$.) The other endpoint of Π may be either z or z'; we'll assume the former, as the other case is symmetrical. Let $\Pi = \{e_0, e_1, \ldots, e_{2k}\}$ be an enumeration of the edges of this path (the direction is immaterial) and denote by Π_0 the $k+1$ even edges, and by Π_1 the k odd edges. Finally define a mapping from $\mathcal{M}(u,z,y',z') \times \mathcal{M}(y,v)$ to $\mathcal{M}(y',z') \times \mathcal{M}(y,z)$ by $(M_{u,z,y',z'}, M_{y,v}) \mapsto (M_{y',z'}, M_{y,z})$, where $M_{y',z'} := M_{u,z,y',z'} \cup \Pi_0 \setminus \Pi_1$ and $M_{y,z} := M_{y,v} \cup \Pi_1 \setminus \Pi_0$. (If the path Π joins y to z' then, using the same construction, we end up with a pair of matchings from $\mathcal{M}(y',z) \times \mathcal{M}(y,z')$.)

Note that this mapping is injective, since we may uniquely recover the pair $(M_{u,z,y',z'}, M_{y,v})$ from $(M_{y',z'}, M_{y,z})$. To see this, observe that $M_{y',z'} \oplus$

[10] 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, an odd-length path and an odd-length cycle containing u and v, and the proof cannot proceed.

$M_{y,z}$ decomposes into a number of cycles, together with either a pair of odd-length paths or a pair of even-length paths. These paths are exactly those paths considered in the forward map. There is only one way to apportion edges from these paths (with edge (u, v) removed) between $M_{u,z,y',z'}$ and $M_{y,v}$. Moreover, the mapping preserves activities modulo a factor $\lambda(u, v)$.

Part (iii) is similar to (iv), but simpler. There is only one path, which is of odd length and joins y and z. The construction from part (iii) does not refer to the path ending at y', and can be applied to this situation too. The result is a pair of matchings from $\mathcal{M} \times \mathcal{M}(y, z)$, as required. □

Corollary 8 *Let G be as above, and let $u, y, y' \in V_1$ and $v, z, z' \in V_2$ be distinct vertices. Suppose $u \sim v$, and also $y' \sim z'$ whenever the latter pair appears. Then, provided in each case that the left hand side of the inequality is defined:*

(i) $w^*(u, v) \geq \lambda(u, v)$;

(ii) $w^*(u, v, y, z) \geq \lambda(u, v)w^*(y, z)$;

(iii) $w^*(u, z)w^*(y, v) \geq \lambda(u, v)w^*(y, z)$; *and*

(iv) $2w^*(u, z', y, z)w^*(y', v) \geq \lambda(u, v)\lambda(y', z')w^*(y, z)$.

Proof. Inequalities (i), (ii) and (iii) follow directly from the correspondingly labelled inequalities in Lemma 7, and the definition of w^*.

Inequality (iv) can be verified as follows. From inequality (iv) in Lemma 7, we know that either

$$2w^*(u, z', y, z)w^*(y', v) \geq \lambda(u, v)w^*(y, z)w^*(y', z') \tag{10}$$

or

$$2w^*(u, z', y, z)w^*(y', v) \geq \lambda(u, v)w^*(y, z')w^*(y', z). \tag{11}$$

(We have swapped the roles of the primed and unprimed vertices, which have the same status as far as Lemma 7 is concerned.) In the first instance, inequality (iv) of the current lemma follows from inequalities (10) and (i); in the second, from (11) and (iii). □

Armed with Corollary 8, we can now turn to the proof of our main lemma.

Proof of Lemma 6. Recall that transitions are defined by a two-step procedure: a move is first proposed, and then either accepted or rejected according

to the Metropolis rule. Each of the possible proposals is made with probability at least $1/4n$. (The proposal involves either selecting one of the n edges or $2n$ vertices u.a.r.; however, with probability $\frac{1}{2}$ we do not even get as far as making a proposal.) Thus for any pair of states M, M' such that the probability of transition from M to M' is non-zero, we have

$$P(M, M') \geq \frac{1}{4n} \min\left\{\frac{\Lambda(M')}{\Lambda(M)}, 1\right\},$$

or

$$\min\{\Lambda(M), \Lambda(M')\} \leq 4n\,\Lambda(M)P(M, M'). \tag{12}$$

Define $\Omega' := \Omega \cup \bigcup_{u,v,y,z} \mathcal{M}(u, v, y, z)$, where, as usual, u, y range over V_1 and v, z over V_2. Also define, for any collection S of matchings, $\Lambda(S) := \sum_{M \in S} \Lambda(M)$. Provided u, v, y, z is a realizable four-hole pattern, i.e., provided $\mathcal{M}(u, v, y, z)$ is non-empty, $\Lambda(\mathcal{M}(u, v, y, z)) = \Lambda(\mathcal{M})$; this is a consequence of setting $w(u, v, y, z)$ to the ideal weight $w^*(u, v, y, z)$ for all four-hole patterns. Likewise, $\frac{1}{2}\Lambda(\mathcal{M}) \leq \Lambda(\mathcal{M}(u, v)) \leq 2\Lambda(\mathcal{M})$, provided u, v is a realizable two-hole pattern; this is a consequence of inequality (5). Moreover, it is a combinatorial fact that the number of realizable four-hole patterns exceeds the number of realizable two-hole patterns by at most a factor $\frac{1}{2}(n-1)^2$. (Each realizable four-hole pattern contains at least two realizable two-hole patterns, while each realizable two-hole pattern is contained in at most $(n-1)^2$ four-hole patterns.) It follows from these considerations that $\Lambda(\Omega')/\Lambda(\Omega) \leq n^2$.

Recall $\pi(M) = \Lambda(M)/\Lambda(\Omega)$. 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 t and $\eta_t(I, F)$), and

$$\Lambda(I)\Lambda(F) \leq 8 \min\{\Lambda(M), \Lambda(M')\}\Lambda(\eta_t(I, F)). \tag{13}$$

In the light of (12), this inequality would imply

$$\Lambda(I)\Lambda(F) \leq 32n\,\Lambda(M)P(M, M')\Lambda(\eta_t(I, F)). \tag{14}$$

Summing inequality (14) over $(I, F) \in \mathrm{cp}(t)$, where $t = (M, M')$ is a most

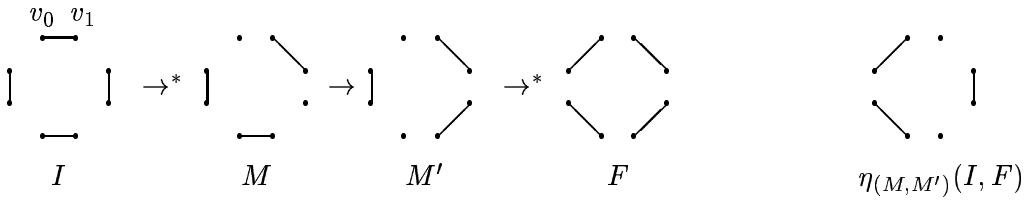

Figure 5: A canonical path through transition $M \to M'$ and its encoding.

congested transition, we get

$$\begin{aligned}
\varrho(\Gamma) &= \frac{L}{Q(t)} \sum_{(I,F)\in \mathrm{cp}(t)} \pi(I)\pi(F) \\
&= \frac{\Lambda(\Omega)\, L}{\Lambda(M)P(M,M')} \sum_{(I,F)\in \mathrm{cp}(t)} \frac{\Lambda(I)\Lambda(F)}{\Lambda(\Omega)^2} \\
&\le \frac{32nL}{\Lambda(\Omega)} \sum_{(I,F)\in \mathrm{cp}(t)} \Lambda(\eta_t(I,F)) \\
&\le \frac{48n^2\Lambda(\Omega')}{\Lambda(\Omega)} \\
&\le 48n^4, \qquad (15)
\end{aligned}$$

where we have used the following observations: canonical paths have maximum length $3n/2$ (the worst case being the unwinding of a cycle of length four), η_t is an injection, and $\Lambda(\Omega') \le n^2\Lambda(\Omega)$. Note that (15) is indeed the sought-for bound on $\varrho(\Gamma)$.

We now proceed to define the encoding η_t and show that it has the required properties, specifically that it is injective and satisfies (13). Recall that there are three stages to the unwinding of an alternating cycle: (i) the initial transition creates a pair of holes; (ii) the intermediate transitions swap edges to move one of the holes round the cycle; and (iii) the final transition adds an edge to plug the two holes. For an intermediate transition $t = (M, M')$ in the unwinding of an alternating cycle, the encoding is

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

(Refer to Figure 5, where just a single alternating cycle is viewed in isolation.) In all other cases (initial or final transitions in the unwinding of an alternating cycle, or any transition in the unwinding of the unique alternating path), 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 particular edges swapped by 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).[11] It therefore remains only to verify inequality (13) for our encoding η_t.

For the remainder of the proof, let y, z denote the holes of I, i.e., $I \in \mathcal{M}(y, z)$ where $y \in V_1$ and $z \in V_2$. Consider first the case where $t = (M, M')$ is the initial transition in the unwinding of an alternating cycle, where $M = M' \cup \{(v_0, v_1)\}$. Since $I, C \in \mathcal{M}(y, z)$, $M, F \in \mathcal{M}$ and $M' \in \mathcal{M}(v_0, v_1)$, inequality (13) 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 8, together with inequality (5). (There is a factor 4 to spare: this is not the critical case.) The situation is symmetric for the final transition in the unwinding of an alternating cycle.

Consider now an intermediate transition $t = (M, M')$ in the unwinding of an alternating cycle, say one that exchanges edge (v_{2i}, v_{2i+1}) with (v_{2i-1}, v_{2i}). In this case we have $I \in \mathcal{M}(y, z)$, $F \in \mathcal{M}$, $M \in \mathcal{M}(v_0, v_{2i-1})$, $M' \in \mathcal{M}(v_0, v_{2i+1})$ and $C \in \mathcal{M}(v_{2i}, v_1, y, z)$. 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 (13) becomes

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

[11] 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 cycles first, and then use the freedom in the choice of endpoint of the path to avoid the potential ambiguity.

This inequality can be verified by reference to Corollary 8: specifically, it follows from inequality (iv) in the general case $i \neq 1$, and by a paired application of inequalities (ii) and (i) in the special case $i = 1$, when vertices v_1 and v_{2i-1} coincide. Note that the constant $8 = 2^3$ is determined by this case (and a succeeding one), and arises from the need to apply inequality (5) twice, combined with the factor 2 in (iv).

We now turn to the unique alternating path. Consider any transition $t = (M, M')$ in the unwinding of the alternating path, except for the final one; such a transition exchanges edge (v_{2i}, v_{2i+1}) for (v_{2i+2}, v_{2i+1}). Observe that $I \in \mathcal{M}(y, z)$, $F \in \mathcal{M}$, $M \in \mathcal{M}(v_{2i}, z)$, $M' \in \mathcal{M}(v_{2i+2}, z)$ and $C \in \mathcal{M}(y, v_{2i+1})$. Moreover, $\lambda(I)\lambda(F) = \lambda(M)\lambda(C)\lambda(v_{2i}, v_{2i+1}) = \lambda(M')\lambda(C) \times \lambda(v_{2i+2}, v_{2i+1})$. In inequality (13) we are left with

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

which holds by inequality (iii) of Corollary 8 in the general case, and by inequality (i) in the special case $i = 0$ when v_{2i} and y coincide.

The final case is the last transition $t = (M, M')$ in the unwinding of an alternating path, where $M' = M \cup \{(v_{2k}, z)\}$. Note that $I, C \in \mathcal{M}(y, z)$, $F, M' \in \mathcal{M}$, $M \in \mathcal{M}(v_{2k}, z)$ and $\lambda(I)\lambda(F) = \lambda(M')\lambda(C) = \lambda(M)\lambda(C)\lambda(v_{2k}, z)$. Plugging these into inequality (13) leaves us with

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

which follows from inequality (i) of Corollary 8.

We have thus shown that the encoding η_t satisfies inequality (13) in all cases. This completes the proof of Lemma 6. □

Recall that our aim is the design of a flow $f_{I,F}$ for all $I, F \in \Omega$ with small congestion. The canonical paths Γ we have defined provide an obvious way of routing flow from a near-perfect matching I to perfect matching F. We now show how to extend this flow to all pairs of states with only a modest increase in congestion. The following lemma is similar in spirit to one used by Schweinsberg [23].

Lemma 9 *Denoting by $\mathcal{N} := \Omega \setminus \mathcal{M}$ the set of near-perfect matchings, there exists a flow f in MC with congestion*

$$\varrho(f) \leq \left[2 + 4 \left(\frac{\pi(\mathcal{N})}{\pi(\mathcal{M})} + \frac{\pi(\mathcal{M})}{\pi(\mathcal{N})} \right) \right] \varrho(\Gamma),$$

where $\varrho(f)$ is as defined in (1).

Proof. Our aim is to route flow between arbitrary pairs of states I, F along composite paths obtained by concatenating canonical paths from Γ. First some notation. For a pair of simple paths p_1 and p_2 such that the final vertex of p_1 matches the initial vertex of p_2, let $p_1 \circ p_2$ denote the simple path resulting from concatenating p_1 and p_2 and removing any cycles. Also define $\overline{p}$ to be the reversal of path p.

The flow $f_{I,F}$ from I to F is determined by the location of the initial and final states I and F. There are four cases:

- If $I \in \mathcal{N}$ and $F \in \mathcal{M}$ then use the direct path from Γ. That is $\mathcal{P}_{I,F} = \{\gamma_{I,F}\}$, and $f_{I,F}(p) = \pi(I)\pi(F)$ for the unique path $p \in \mathcal{P}_{I,F}$.
- If $I \in \mathcal{M}$ and $F \in \mathcal{N}$ then use the reversal of the path $\gamma_{F,I}$ from Γ. That is $\mathcal{P}_{I,F} = \{\overline{\gamma_{I,F}}\}$, and $f_{I,F}(p) = \pi(I)\pi(F)$ for the unique path $p \in \mathcal{P}_{I,F}$.
- If $I \in \mathcal{N}$ and $F \in \mathcal{N}$ then route flow through a random state $X \in \mathcal{M}$. So $\mathcal{P}_{I,F} = \{p_X : X \in \mathcal{M}\}$, where $p_X = \gamma_{I,X} \circ \overline{\gamma_{F,X}}$, and $f_{I,F}(p_X) = \pi(I)\pi(F)\pi(X)/\pi(\mathcal{M})$. (We regard the paths in $\mathcal{P}_{I,F}$ as being labelled by the intermediate state X, so that two elements of $\mathcal{P}_{I,F}$ are distinguishable even if they happen to be equal as paths.)
- If $I \in \mathcal{M}$ and $F \in \mathcal{M}$ then route flow through a random state $X \in \mathcal{N}$. So $\mathcal{P}_{I,F} = \{p_X : X \in \mathcal{N}\}$, where $p_X = \overline{\gamma_{X,I}} \circ \gamma_{X,F}$, and $f_{I,F}(p_X) = \pi(I)\pi(F)\pi(X)/\pi(\mathcal{N})$.

It is immediate in all cases that $\sum_p f_{I,F}(p) = \pi(I)\pi(F)$, where the sum is over all $p \in \mathcal{P}_{I,F}$.

Let $t = (M, M')$ be a most congested transition under the flow f just defined, and recall that $Q(t) = \pi(M)P(M, M')$. Then

$$\varrho(f) = \frac{1}{Q(t)} \sum_{I,F \in \Omega} \sum_{p\,:\,t \in p \in \mathcal{P}_{I,F}} f_{I,F}(p)\,|p|.$$

Decompose $\varrho(f)$ into contributions from each of the above four types of paths, by writing

$$\varrho(f) = \varrho(f_{\mathcal{N},\mathcal{M}}) + \varrho(f_{\mathcal{M},\mathcal{N}}) + \varrho(f_{\mathcal{N},\mathcal{N}}) + \varrho(f_{\mathcal{M},\mathcal{M}}),$$

where

$$\varrho(f_{\mathcal{N},\mathcal{M}}) = \frac{1}{Q(t)} \sum_{I \in \mathcal{N}, F \in \mathcal{M}} \sum_{p\,:\,t \in p \in \mathcal{P}_{I,F}} f_{I,F}(p)\,|p|,$$

etc.

Recall that $\mathrm{cp}(t)$ denotes the set of pairs $(I, F) \in \mathcal{N} \times \mathcal{M}$ such that the canonical path from I to F passes along t. Letting L be the maximum length of any canonical path in Γ,

$$\begin{aligned}\varrho(f_{\mathcal{N},\mathcal{M}}) &\le \frac{L}{Q(t)} \sum_{I\in\mathcal{N}, F\in\mathcal{M}} \sum_{p\,:\,t\in p\in\mathcal{P}_{I,F}} f_{I,F}(p) \\ &= \frac{L}{Q(t)} \sum_{(I,F)\in \mathrm{cp}(t)} \pi(I)\pi(F) \\ &= \varrho(\Gamma).\end{aligned}$$

Likewise, by time reversibility, $\varrho(f_{\mathcal{M},\mathcal{N}}) \le \varrho(\Gamma)$. Furthermore,

$$\begin{aligned}\varrho(f_{\mathcal{N},\mathcal{N}}) &\le \frac{2L}{Q(t)} \sum_{I\in\mathcal{N}, F\in\mathcal{N}} \sum_{p\,:\,t\in p\in\mathcal{P}_{I,F}} f_{I,F}(p) \\ &\le \frac{2L}{Q(t)} \sum_{I\in\mathcal{N}, F\in\mathcal{N}} \sum_{X\in\mathcal{M}} \sum_{p\,:\,t\in p=\gamma_{I,X}\circ\overline{\gamma_{F,X}}} f_{I,F}(p) \\ &\le \frac{2L}{Q(t)} \Bigg[\sum_{(I,X)\in\mathrm{cp}(t)} \sum_{F\in\mathcal{N}} \frac{\pi(I)\pi(F)\pi(X)}{\pi(\mathcal{M})} \\ &\qquad + \sum_{(F,X)\in\mathrm{cp}(t)} \sum_{I\in\mathcal{N}} \frac{\pi(I)\pi(F)\pi(X)}{\pi(\mathcal{M})} \Bigg] \\ &= \frac{4\,\pi(\mathcal{N})}{\pi(\mathcal{M})}\varrho(\Gamma).\end{aligned}$$

Likewise,

$$\varrho(f_{\mathcal{M},\mathcal{M}}) \le \frac{4\,\pi(\mathcal{M})}{\pi(\mathcal{N})}\varrho(\Gamma).$$

Putting the four inequalities together, the claimed bound on congestion follows. □

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

Proof of Theorem 4. The theorem follows from Lemma 6, Lemma 9, Theorem 3, and the fact that $\pi(\mathcal{N})/\pi(\mathcal{M}) = \Theta(n^2)$. □

It is perhaps worth remarking, for the benefit of those familiar with Diaconis and Saloff-Coste's comparison argument [6], that the proof of Lemma 9 could be viewed as comparing the Markov chain MC against the random walk in a complete bipartite graph.

5 Using samples to estimate the permanent

For convenience, we adopt the graph-theoretic view of the permanent of a $0,1$-matrix as the number of perfect matchings in an associated bipartite graph G. From Lemma 2 and Theorem 4 we know how to sample perfect matchings from an almost uniform distribution. Now, Broder [4] has demonstrated how an almost uniform sampler for perfect matchings in a bipartite graph may be converted into an fpras. Indeed, our Theorem 1 (the existence of an fpras for the permanent of a 0,1-matrix) follows from Lemma 2 and Theorem 4 via Broder's Corollary 5. Nevertheless, with a view to making the article self contained, and at the same time deriving an explicit upper bound on running time, we present in this section an explicit proof of Theorem 1. Our proposed method for estimating the number of perfect matchings in G given an efficient sampling procedure is entirely standard (see, e.g., [9, §3.2]), but we are able to curb the running time by tailoring the method to our particular situation.

So suppose G is a bipartite graph on $n+n$ vertices and that we want to estimate the number of perfect matchings in G within ratio $e^{\pm\varepsilon}$, for some specified $\varepsilon > 0$. Recall that the initialization procedure of §3 converges to suitable hole-weights $w(\cdot,\cdot)$ through a sequence of phases. In phase i, a number of samples are obtained using Markov chain simulation with edge-activities $\lambda_{i-1}(\cdot,\cdot)$ (say) and corresponding hole-weights $w_{i-1}(\cdot,\cdot)$. At the beginning, before phase 1, λ_0 is the constant function 1, and $w(u,v) = n$ for every hole-pair u, v. Between one phase and the next, the weights and activities change by small factors; ultimately, after the final phase r, the activity $\lambda(u,v)$ is 1 if (u,v) is an edge of G, and a very small value otherwise. The number of phases is $r = O(n^2 \log n)$.

Let Λ_i be the weight function associated with the pair (λ_i, w_i) through definition (3). The quantity $\Lambda_i(\Omega) = \sum_{M\in\Omega} \Lambda_i(M)$ is a "partition function" for weighted matchings after the ith phase. Initially, $\Lambda_0(\Omega) = (n^2+1)n!$; while, at termination, $\Lambda_r(\Omega)$ is roughly n^2+1 times the number of perfect matchings in G. Considering the "telescoping product"

$$\Lambda_r(\Omega) = \Lambda_0(\Omega) \times \frac{\Lambda_1(\Omega)}{\Lambda_0(\Omega)} \times \frac{\Lambda_2(\Omega)}{\Lambda_1(\Omega)} \cdots \times \frac{\Lambda_r(\Omega)}{\Lambda_{r-1}(\Omega)}, \tag{16}$$

we see that we may obtain a rough estimate for the number of perfect matchings in G by estimating in turn each of the ratios $\Lambda_{i+1}(\Omega)/\Lambda_i(\Omega)$. We now explain how this is done.

Assume that the initialization procedure runs successfully, so that (5) holds at every phase. (We shall absorb the small failure probability of the initialization phase into the overall failure probability of the fpras.) Observe that the rule for updating the activities from λ_i to λ_{i+1}, together with the constraints on the weights w_i and w_{i+1} specified in (5), ensure

$$\frac{1}{4e} \leq \frac{\Lambda_{i+1}(M)}{\Lambda_i(M)} \leq 4e, \quad \text{for all } M \in \Omega. \tag{17}$$

Thus we are in good shape to estimate the various ratios in (16) by Monte Carlo sampling. The final task is to improve this rough estimate to a more accurate one.

Let π_i denote the stationary distribution of the Markov chain used in phase $i+1$, so that $\pi_i(M) = \Lambda_i(M)/\Lambda_i(\Omega)$. Let Z_i denote the r.v. which is the outcome of the following experiment:

> By running the Markov chain MC of §3 with parameters $\Lambda = \Lambda_i$ and $\delta = \varepsilon/80e^2r$, obtain a sample matching M from a distribution that is within variation distance $\varepsilon/80e^2r$ of π_i.
> Return $\Lambda_{i+1}(M)/\Lambda_i(M)$.

If we had sampled M from the exact stationary distribution π_i instead of an approximation, then the resulting modified r.v. Z_i' would have satisfied

$$\mathbb{E}\, Z_i' = \sum_{M \in \Omega} \frac{\Lambda_i(M)}{\Lambda_i(\Omega)} \frac{\Lambda_{i+1}(M)}{\Lambda_i(M)} = \frac{\Lambda_{i+1}(\Omega)}{\Lambda_i(\Omega)}.$$

As it is, noting the particular choice for δ and bounds (17), and using the fact that $e^{-x/4} \leq 1 - \frac{1}{5}x \leq 1 + \frac{1}{5}x \leq e^{x/4}$ for $0 \leq x \leq 1$, we must settle for

$$\exp\left(-\frac{\varepsilon}{4r}\right) \frac{\Lambda_{i+i}(\Omega)}{\Lambda_i(\Omega)} \leq \mathbb{E}\, Z_i \leq \exp\left(\frac{\varepsilon}{4r}\right) \frac{\Lambda_{i+i}(\Omega)}{\Lambda_i(\Omega)}.$$

Now suppose s independent trials are conducted for each i using the above experiment, and denote by $\overline{Z}_i$ the sample mean of the results. Then $\mathbb{E}\,\overline{Z}_i = \mathbb{E}\, Z_i$ (obviously), and

$$\exp\left(-\frac{\varepsilon}{4}\right) \frac{\Lambda_r(\Omega)}{\Lambda_0(\Omega)} \leq \mathbb{E}(\overline{Z}_0 \overline{Z}_1 \ldots \overline{Z}_{r-1}) \leq \exp\left(\frac{\varepsilon}{4}\right) \frac{\Lambda_r(\Omega)}{\Lambda_0(\Omega)}. \tag{18}$$

For s sufficiently large, $\prod_i \overline{Z}_i$ will be close to $\prod_i \mathbb{E}\,\overline{Z}_i$ with high probability. With a view to quantifying "sufficiently large", observe that in the light of (17),

$$\frac{\mathrm{Var}[\overline{Z}_i]}{(\mathbb{E}\,\overline{Z}_i)^2} \leq \frac{16}{s}.$$

Thus taking $s = \Theta(r\varepsilon^{-2})$ we get

$$\begin{aligned}
\frac{\mathrm{Var}[\overline{Z}_0 \ldots \overline{Z}_{r-1}]}{(\mathbb{E}[\overline{Z}_0 \ldots \overline{Z}_{r-1}])^2} &= \prod_{i=0}^{r-1} \frac{\mathbb{E}\,\overline{Z}_i^2}{(\mathbb{E}\,\overline{Z}_i)^2} - 1 \\
&= \prod_{i=0}^{r-1} \left(1 + \frac{\mathrm{Var}\,\overline{Z}_i}{(\mathbb{E}\,\overline{Z}_i)^2}\right) - 1 \\
&\leq \left(1 + \frac{16}{s}\right)^r - 1 \\
&= O(\varepsilon^2).
\end{aligned}$$

So, by Chebyshev's inequality,

$$\Pr\left[e^{-\varepsilon/4}\,\mathbb{E}(\overline{Z}_0 \ldots \overline{Z}_{r-1}) \leq \overline{Z}_0 \ldots \overline{Z}_{r-1} \leq e^{\varepsilon/4}\,\mathbb{E}(\overline{Z}_0 \ldots \overline{Z}_{r-1})\right] \geq \frac{11}{12}, \quad (19)$$

assuming the constant implicit in the setting $s = \Theta(r\varepsilon^{-2})$ is chosen appropriately. Combining inequalities (18) and (19) with the fact that $\Lambda_0(\Omega) = (n^2+1)n!$ we obtain

$$\Pr\left[e^{-\varepsilon/2}\Lambda_r(\Omega) \leq (n^2+1)n!\,\overline{Z}_0 \ldots \overline{Z}_{r-1} \leq e^{\varepsilon/2}\Lambda_r(\Omega)\right] \geq \frac{11}{12}. \quad (20)$$

Denote by $\mathcal{M}_G \subset \mathcal{M}$ the set of perfect matchings in the graph G. Inequality (20) provides an effective estimator for $\Lambda_r(\Omega)$, already yielding a rough estimate for $|\mathcal{M}_G|$. The final step is to improve the accuracy of this estimate to within ratio $e^{\pm\varepsilon}$, as required. Observe that $\Lambda_r(M) = 1$ for any matching $M \in \mathcal{M}_G$, so that $\Lambda_r(\mathcal{M}_G)$ is equal to the number of perfect matchings in G. Consider the following experiment:

By running the Markov chain MC of §3 with parameters $\Lambda = \Lambda_r$ and $\delta = \varepsilon/80e^2$, obtain a sample matching M from a distribution that is within variation distance $\varepsilon/80e^2$ of π_r.
Return 1 if $M \in \mathcal{M}_G$, and 0 otherwise.

The outcome of this experiment is a random variable that we denote by Y. If M had been sampled from the exact stationary distribution π_r then its expectation would have been $\Lambda_r(\mathcal{M}_G)/\Lambda(\Omega)$; as it is, we have

$$\exp\left(-\frac{\varepsilon}{4}\right)\frac{\Lambda_r(\mathcal{M}_G)}{\Lambda_r(\Omega)} \leq \mathbb{E}\,Y \leq \exp\left(\frac{\varepsilon}{4}\right)\frac{\Lambda_r(\mathcal{M}_G)}{\Lambda_r(\Omega)}.$$

Let $\overline{Y}$ denote the sample mean of $s' = \Theta(n^2\varepsilon^{-2})$ independent trials of the above experiment. Since $\mathbb{E}\,\overline{Y} = \mathbb{E}\,Y = \Omega(n^{-2})$, Chebyshev's inequality gives

$$\Pr\left[e^{-\varepsilon/4}\,\mathbb{E}\,\overline{Y} \leq \overline{Y} \leq e^{\varepsilon/4}\,\mathbb{E}\,\overline{Y}\right] \geq \frac{11}{12},$$

as before. Combining this with (20), we get

$$\Pr\left[e^{-\varepsilon}|\mathcal{M}_G| \leq (n^2+1)n!\,\overline{Y}\,\overline{Z}_0\overline{Z}_1\ldots\overline{Z}_{r-1} \leq e^{\varepsilon}|\mathcal{M}_G|\right] \geq \frac{5}{6}.$$

All this was under the assumption that the initialization procedure succeeded. But provided we arrange for the failure probability of initialization to be at most $1/12$, it will be seen that $(n^2+1)n!\,\overline{Y}\,\overline{Z}_0\overline{Z}_1\ldots\overline{Z}_{r-1}$ is an estimator for the permanent that meets the specification of an fpras.

In total the above procedure requires $rs+s' = O(\varepsilon^{-2}n^4(\log n)^2)$ samples; by Theorem 4, $O(n^7\log n)$ time is sufficient to generate each sample. (Since there is no point in setting $\varepsilon = o(1/n!)$, the $\log\delta^{-1}$ term in Theorem 4 can never dominate.) The running time is thus $O(\varepsilon^{-2}n^{11}(\log n)^3)$. Note that this is sufficient to absorb the cost of the initialization procedure as well, which by Lemma 5 is $O(n^{11}(\log n)^3)$.

6 Reducing the running time by using "warm starts"

In this article, we have concentrated on simplicity of presentation, rather than squeezing the degree of the polynomial bounding the running time. However, a fairly simple (and standard) observation allows us to reduce the dependence on n from $\widetilde{O}(n^{11})$ — which was the situation at the end of the previous section — to $\widetilde{O}(n^{10})$.

The observation is this. We use Markov chain simulation to generate samples from a distribution close to the stationary distribution. These samples are used to estimate the expectation $\mathbb{E}\,f$ of some function $f : \Omega \to \mathbb{R}^+$. The estimator for $\mathbb{E}\,f$ is naturally enough the mean of f over the sample.

By restarting the Markov chain MC before generating each sample, we ensure that the samples are independent. This allows the performance of the estimator to be analysed using classical Chebyshev and Chernoff bounds. The down-side is that we must wait the full mixing time of MC between samples.

However, it is known that once a Markov chain has reached near-stationarity it is possible to draw samples at a faster rate than that indicated by the mixing time; this "resampling time" is proportional to the inverse spectral gap of the Markov chain. Although the samples are no longer independent, they are as good as independent for many purposes. In particular, there exist versions of the Chernoff and Chebyshev bounds that are adapted to exactly this setting. Versions of the Chernoff bound that fit our application (specifically estimating the expectations in identities (7) have been presented by Gillman [8, Thm 2.1] and Lezaud [17, Thm 1.1, Remark 3]; a version of the Chebyshev bound (that we used twice in §5) by Aldous [1].

The appropriate versions of Chernoff and Chebyshev bounds have slight differences in their hypotheses. For the estimates requiring Chernoff bounds we use every matching visited on the sample path, whereas for those estimates requiring Chebyshev bounds we only use samples spaced by the resampling time. Doing both simultaneously presents no contradiction.

Now the inverse spectral gap is bounded by the congestion ϱ (see [24, Thm 5]), which in the case of MC is $O(n^6)$, by Lemmas 6 and 9. In contrast, the mixing time of MC is $O(n^7 \log n)$. Thus, provided we consume at least $O(n \log n)$ samples (which is always the case for us) we can use the higher resampling rate and save a factor $O(n \log n)$ in the running time. This observation reduces all running times quoted in earlier sections by a similar factor; in particular, the running time of the approximation algorithm for the permanent in §5 comes down to $\widetilde{O}(n^{10})$.

7 Arbitrary weights

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 $a_{\min} = \min_{i,j} a(i,j)$. 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 6.

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 $a_{\min}$,

Initialize $\lambda(u,v) \leftarrow a_{\max}$ for all $(u,v) \in V_1 \times V_2$.
Initialize $w(u,v) \leftarrow n a_{\max}$ for all $(u,v) \in V_1 \times V_2$.
While there exists a pair y,z with $\lambda(y,z) > a(y,z)$ do:
    Take a sample of size S from MC with parameters λ, w,
        using a simulation of T steps in each case.
    Use the sample to obtain estimates $w'(u,v)$ satisfying
        condition (8), for all u,v, with high probability.
    Set $\lambda(y,v) \leftarrow \max\{\lambda(y,v)\exp(-1/2),\ a(y,v)\}$, for all $v \in V_2$,
        and $w(u,v) \leftarrow w'(u,v)$ for all u,v.
Output the final weights $w(u,v)$.

Figure 6: The algorithm for non-negative entries.

assuming arithmetic operations are treated as unit cost. The algorithm of Linial, Samorodnitsky and Wigderson [18] 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 scaling factor. 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.

Finally, note that we cannot realistically hope to handle matrices which contain negative entries. One way to appreciate this is to consider what happens if we replace matrix entry $a(1,1)$ by $a(1,1) - \beta$ where β is a parameter that can be varied. Call the resulting matrix A_β. Note that $\mathrm{per}(A_\beta) = \mathrm{per}(A) - \beta\,\mathrm{per}(A_{1,1})$, where $A_{1,1}$ denotes the submatrix of A obtained by deleting the first row and column. On input A_β, an approximation scheme would have at least to identify correctly the sign of $\mathrm{per}(A_\beta)$; then the root of $\mathrm{per}(A) - \beta\,\mathrm{per}(A_{1,1}) = 0$ could be located by binary search and a *very* close approximation (accurate to within a polynomial number of digits) to $\mathrm{per}(A)/\mathrm{per}(A_{1,1})$ found. The permanent of A itself could then be computed to similar accuracy (and therefore exactly!) by recursion on the submatrix $A_{1,1}$, giving us a polynomial time randomized algorithm that with high probability computes $\mathrm{per}(A)$ exactly. It is important to note here that the cost of binary search scales linearly with the number of significant digits requested, while that of an fpras scales exponentially.

8 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 [25]. A perfect matching in a graph G may be viewed as a spanning[12] 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 [11] 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 certain restricted setting. Combining the same reduction with Theorem 1 yields the following unconditional result.

Corollary 10 *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.[13]

The second example concerns the notion of a 0,1-flow.[14] Consider a directed graph $\overrightarrow{G} = (\overrightarrow{V}, \overrightarrow{E})$, where the in-degree (respectively, out-degree) of a vertex $v \in \overrightarrow{V}$ is denoted by $d_-(v)$ (respectively, $d_+(v)$). A *0,1-flow* is defined as a subset of edges $\overrightarrow{E'} \subset \overrightarrow{E}$ such that in the resulting subgraph $(\overrightarrow{V}, \overrightarrow{E'})$, $d_-(v) = d_+(v)$ for all $v \in \overrightarrow{V}$. Counting the number of $0,1$-flows in $\overrightarrow{G}$ is reducible to counting perfect matchings in an undirected bipartite graph. Specifically, let $G = (V, E)$ be the graph with the following vertex

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

[13] 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.

[14] This notion should not be confused with the notion of flow we used earlier in the analysis of the Markov chain.

and edge sets:

$$\begin{aligned} V &= \left\{h_{i,j}, m_{i,j}, t_{i,j} : \forall i,j \text{ with } \overrightarrow{v_i v_j} \in \overrightarrow{E}\right\} \\ &\quad \cup \left\{u_i^1, \ldots, u_i^{d_-(v_i)} : \forall i \text{ with } v_i \in \overrightarrow{V}\right\}, \\ E &= \left\{(h_{i,j}, m_{i,j}), (m_{i,j}, t_{i,j}) : \forall i,j \text{ with } \overrightarrow{v_i v_j} \in \overrightarrow{E}\right\} \\ &\quad \cup \left\{(u_i^k, h_{i,j}) : \forall i,j,k \text{ satisfying } u_i^k, h_{i,j} \in \overrightarrow{V}\right\} \\ &\quad \cup \left\{(u_i^k, t_{j,i}) : \forall i,j,k \text{ satisfying } u_i^k, t_{j,i} \in \overrightarrow{V}\right\}. \end{aligned}$$

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

Corollary 11 *For an arbitrary directed graph $\overrightarrow{G}$, there exists an fpras for counting the number of $0,1$-flows.*

Suppose the directed graph $\overrightarrow{G}$ 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, we note 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 estimating ice configurations considered by Mihail and Winkler [19].

References

[1] David Aldous, On the Markov chain simulation method for uniform combinatorial distributions and simulated annealing, *Probability in the Engineering and Informational Sciences* **1** (1987), 33–46.

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

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

[4] 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.

[5] Steve Chien, Lars Rasmussen and Alistair Sinclair, Clifford algebras and approximating the permanent, *Proceedings of the 34th Annual ACM Symposium on Theory of Computing*, ACM Press, 2002, 222–231.

[6] Persi Diaconis and Laurent Saloff-Coste, Comparison theorems for reversible Markov chains, *The Annals of Applied Probability* **3** (1993), 696–730.

[7] Persi Diaconis and Daniel Stroock, Geometric bounds for eigenvalues of Markov chains, *The Annals of Applied Probability* **1** (1991), 36–61.

[8] David Gillman, A Chernoff bound for random walks on expander graphs, *SIAM Journal on Computing* **27** (1998), 1203–1220.

[9] Mark Jerrum, Counting, Sampling and Integrating: algorithms and complexity, *Lectures in Mathematics – ETH Zürich*, Birkhäuser, Basel, 2003.

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

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

[12] 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 Publishing, 1996, 482–520.

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

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

[15] 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.

[16] Claire Kenyon, Dana Randall and Alistair Sinclair, Approximating the number of dimer coverings of a lattice, *Journal of Statistical Physics* **83** (1996), 637–659.

[17] Pascal Lezaud, Chernoff-type bounds for finite Markov chains, *Annals of Applied Probability* **8** (1998), 849–867.

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

[19] Milena Mihail and Peter Winkler, On the number of Eulerian orientations of a graph, *Proceedings of the 3rd Annual ACM-SIAM Symposium on Discrete Algorithms*, ACM Press, 1992, 138–145.

[20] Henryk Minc, *Permanents*, Encyclopedia of Mathematics and its Applications Vol. 6, Addison-Wesley, 1982.

[21] Rajeev Motwani and Prabhakar Raghavan, *Randomized Algorithms*, Cambridge University Press, 1995.

[22] Hebert J. Ryser, *Combinatorial Mathematics*, The Carus Mathematical Monographs No. 14, Mathematical Association of America, 1963.

[23] Jason Schweinsberg, An $O(n^2)$ bound for the relaxation time of a Markov chain on cladograms, *Random Structures and Algorithms* **20** (2002), 59–70.

[24] Alistair Sinclair, Improved bounds for mixing rates of Markov chains and multicommodity flow, *Combinatorics, Probability and Computing* **1** (1992), 351–370.

[25] W. T. Tutte, A short proof of the factor theorem for finite graphs, *Canadian Journal of Mathematics* **6** (1954), 347–352.

[26] L. G. Valiant, The complexity of computing the permanent, *Theoretical Computer Science* **8** (1979), 189–201.