Accelerating Simulated Annealing for the Permanent and Combinatorial Counting Problems

Ivona Bezáková* Daniel Štefankovič* Vijay V. Vazirani† Eric Vigoda†

Abstract

We present an improved "cooling schedule" for simulated annealing algorithms for combinatorial counting problems. Under our new schedule the rate of cooling accelerates as the temperature decreases. Thus, fewer intermediate temperatures are needed as the simulated annealing algorithm moves from the high temperature (easy region) to the low temperature (difficult region). We present applications of our technique to colorings and the permanent (perfect matchings of bipartite graphs). Moreover, for the permanent, we improve the analysis of the Markov chain underlying the simulated annealing algorithm. This improved analysis, combined with the faster cooling schedule, results in an $O(n^7 \log^4 n)$ time algorithm for approximating the permanent of a 0/1 matrix.

1 Introduction

Simulated annealing is an important algorithmic approach for counting and sampling combinatorial structures. Two notable combinatorial applications are estimating the partition function of statistical physics models, and approximating the permanent of a non-negative matrix. For combinatorial counting problems, the general idea of simulated annealing is to write the desired quantity, say Z, (which is, for example, the number of colorings or matchings of an input graph) as a telescoping product:

$$Z = \frac{Z_\ell}{Z_{\ell-1}} \frac{Z_{\ell-1}}{Z_{\ell-2}} \cdots \frac{Z_1}{Z_0} Z_0, \tag{1.1}$$

where $Z_\ell = Z$ and Z_0 is trivial to compute. By further ensuring that each of the ratios Z_i/Z_{i-1} is bounded, a small number of samples (from the probability distribution corresponding to Z_{i-1}) suffices to estimate the ratio. These samples are typically generated from an appropriately designed Markov chain.

Each of the quantities of interest corresponds to the counting problem at a different temperature. The final quantity $Z = Z_\ell$ corresponds to zero-temperature, whereas the trivial initial quantity Z_0 is infinite temperature. The temperature slowly decreases from high temperature (easy region) to low temperature (difficult region). A notable application of simulated annealing to combinatorial counting was the algorithm of Jerrum, Sinclair and Vigoda [8] for approximating the permanent of a non-negative matrix. In their algorithm, the cooling schedule is uniform: the rate of cooling was constant.

Our first main result is an improved cooling schedule. In contrast to the previous cooling schedule for the permanent, our schedule is accelerating (the rate of cooling accelerates as the temperature decreases). Consequently, fewer intermediate temperatures are needed, and thus fewer Markov chain samples overall suffice. It is interesting to note that our schedule is similar to the original proposal of Kirkpatrick et al [13], and is related to schedules used recently in geometric settings by Lovász and Vempala [14] and Kalai and Vempala [11].

We illustrate our new cooling schedule in the context of colorings, which corresponds to the antiferromagnetic Potts model from statistical physics. We present general results defining a cooling schedule for a broad class of counting problems. These general results seem applicable to a wide range of combinatorial counting problems, such as the permanent, and binary contingency tables [1].

The permanent of an $n \times n$ matrix A is defined as

$$\mathrm{per}(A) = \sum_\sigma \prod_{i=1}^{n} a_{i,\sigma(i)},$$

where the sum goes over all permutations σ of $[n]$. The permanent of a 0/1 matrix A is the number of perfect matchings in the bipartite graph with bipartite adjacency matrix A. In addition to traditional applications in statistical physics [12], the permanent has recently been used in a variety of areas, e. g., computer

*Department of Computer Science, University of Chicago, Chicago, IL 60637. Email: {ivona,stefanko}@cs.uchicago.edu. I.B. was supported by NSF grant CCR-0455666.

†College of Computing, Georgia Institute of Technology, Atlanta, GA 30332. Email: {vazirani,vigoda}@cc.gatech.edu. V.V. is supported by NSF grants CCR-0311541 and CCR-0220343. E.V. is supported by NSF grant CCR-0455666.

vision [16], and statistics [15]. Jerrum, Sinclair, and Vigoda presented a simulated annealing algorithm [8] for the permanent of non-negative matrices with running time $O(n^{10}\log^3 n)$ for 0/1 matrices.

Our cooling schedule reduces the number of intermediate temperatures in the simulated annealing for the permanent from $O(n^2 \log n)$ to $O(n \log^2 n)$. We also improve the analysis of the Markov chain used for sampling. The improved analysis comes from several new inequalities about perfect matchings in bipartite graphs. The consequence of the new analysis and improved cooling schedule is an $O(n^7 \log^4 n)$ time algorithm for estimating the permanent of an $0/1$ $n \times n$ matrix. Here is the formal statement of our result:

THEOREM 1.1. *For all $\epsilon > 0$, there exists a randomized algorithm to approximate, within a factor $(1 \pm \epsilon)$, the permanent of a 0/1 $n \times n$ matrix A in time $O(n^7 \log^4(n) + n^6 \log^5(n)\epsilon^{-2})$. The algorithm extends to arbitrary matrices with non-negative entries.*

The remainder of the paper is organized as follows. In Section 2 we present our new cooling schedule, motivated by its application to colorings. We then focus on the permanent in Section 3.

2 Improved Cooling Schedule

We begin by motivating the simulated annealing framework in the context of colorings. We then present a general method for obtaining improved cooling schedules and show how it can be applied to colorings. We conclude with the proofs of technical lemmas for improved cooling schedules.

2.1 Counting Colorings. Our focus in this section is counting all valid k-colorings of a given graph G. Let $G = (V, E)$ be the input graph and k be the number of colors. A (valid) k-coloring of G is an assignment of colors from $[k]$ to the vertices of G such that no two adjacent vertices are colored by the the same color (i. e., $\sigma(u) \neq \sigma(v)$ for every $(u, v) \in E$). Let $\Omega = \Omega(G)$ denote the set of all k-colorings of G. For input parameters ϵ, δ, our goal is to approximate $|\Omega|$ within a multiplicative factor $1 \pm \epsilon$ with probability $\geq 1 - \delta$.

Before we present our new reduction, it is worth illustrating the standard reduction (see e.g., [6]). Let $E_0 = E = \{e_1, ..., e_m\}$ (ordered arbitrarily), and, for $1 \leq i \leq m$, let $E_i = E_{i-1} \setminus e_i$ and $G_i = (V, E_i)$. Then the number of k-colorings of G can be written as a telescoping product:

$$|\Omega(G)| = k^n \prod_i \frac{|\Omega(G_{i-1})|}{|\Omega(G_i)|}$$

For $k \geq \Delta + 2$ where Δ is the maximum degree of G, it is possible to verify the following bound on the i-th ratio:

$$\frac{1}{2} \leq \frac{|\Omega(G_{i-1})|}{|\Omega(G_i)|} \leq 1,$$

Therefore we can estimate the i-th ratio by generating random k-colorings of G_i and counting the proportion that are also valid k-colorings of G_{i-1}. This reduces the approximate counting problem of estimating the cardinality of $\Omega(G)$ to m random sampling problems, see Jerrum [6] for details on the reduction as a function of the error parameter ϵ and confidence parameter δ.

We instead look at a continuous version of the problem, the anti-ferromagnetic Potts model from Statistical Physics, which allows more flexibility in how we remove edges. In addition to the underlying graph G and the number of partitions k, the Potts model is also specified by an activity[1] λ. The configuration space of the Potts model, denoted $[k]^V$, is the set of all labelings $\sigma : V \to [k]$. The partition function of the Potts model counts the number of configurations weighted by their "distance" from a valid k-coloring. The "distance" is measured in terms of the activity λ and we will specify it shortly. As the activity goes to zero, the partition function limits to $|\Omega|$.

Our reduction from approximating $|\Omega|$ to sampling from Ω, works by specifying a sequence of activities for the anti-ferromagnetic Potts model, so that the partition functions do not change by more than a constant factor between successive activities. This allows us to reduce the activity to an almost zero value while being able to estimate the ratios of two consecutive partition functions. Then, as before, we can approximate $|\Omega|$. The advantage of the new reduction lies in using fewer random sampling problems, namely instead of m problems we now need to consider only $O(n \log n)$ sampling problems to estimate $|\Omega|$.

For $\lambda > 0$, the partition function of the Potts model is

$$Z(\lambda) = \sum_{\sigma \in [k]^V} \lambda^{M(\sigma)}$$

where $M(\sigma) = M_G(\sigma) = |(u, v) \in E : \sigma(u) = \sigma(v)|$ is the number of monochromatic edges of the labeling σ.

The partition function can be viewed as a polynomial in λ. Notice that its absolute coefficient equals $|\Omega|$, the number of k-colorings of G. Moreover, $Z(1) = |\Omega(G_m)| = k^n$ is the sum of the coefficients of Z. It can be shown that for $k > \Delta$ the number of k-colorings of G is bounded from below by $(k/e)^n$ (i. e., $|\Omega| \geq (k/e)^n$).

[1] The activity corresponds to the temperature of the system. Specifically, the temperature is $1/\ln\lambda$, thus $\lambda = 1$ corresponds to the infinite temperature and $\lambda = 0$ corresponds to the zero temperature.

If we used the trivial lower bound of $|\Omega| \geq 1$, we would introduce an extra factor of $O(\log k)$ in the final running time. Observe that the value of the partition function at $\lambda = 1/e^n$ is at most $2|\Omega|$:

(2.2)

$$|\Omega| \leq Z(1/e^n) \leq |\Omega| + Z(1)(1/e^n) \leq |\Omega| + k^n/e^n \leq 2|\Omega|.$$

This will be sufficiently close to $|\Omega|$ so that we can obtain an efficient estimator for $|\Omega|$.

We will define a sequence

$$\lambda_0 = 1, \lambda_1, \ldots, \lambda_\ell \leq 1/e^n, \lambda_{\ell+1} = 0,$$

where $\ell = O(n \log n)$, and, for all $0 \leq i \leq \ell$,

$$\frac{1}{2} \leq \frac{Z(\lambda_{i+1})}{Z(\lambda_i)} \leq 1.$$

We estimate the number of k-colorings of G via the telescoping product:

$$|\Omega| = k^n \prod_{0 \leq i \leq \ell} \alpha_i,$$

where $\alpha_i = Z(\lambda_{i+1})/Z(\lambda_i)$. We will estimate α_i by sampling from the probability distribution corresponding to Z_i. Before we describe how to estimate these ratios, we first specify the cooling schedule (i.e., the sequence of activities).

2.2 Intuition for Accelerating Cooling Schedule for Colorings. We need to ensure that for consecutive λ_i, λ_{i+1} the ratio $Z(\lambda_{i+1})/Z(\lambda_i)$ is in the interval $[\frac{1}{2}, 1]$. The polynomial Z has degree m since any labeling has at most $m = |E|$ monochromatic edges. Hence it suffices to define $\lambda_{i+1} = 2^{-1/m}\lambda_i$, then $Z(\lambda_{i+1}) \geq (2^{-1/m})^m Z(\lambda_i) \geq Z(\lambda_i)/2$. This specifies a uniform cooling schedule with a rate of decrease $2^{-1/m}$.

If we had $Z(\lambda) = k^n\lambda^m$ we could not decrease λ faster than $2^{-1/m}$. Fortunately, in our case the absolute coefficient of $Z(\lambda)$ is at least $|\Omega| \geq (k/e)^n$. To illustrate the idea of non-uniform decrease, let $f_i(\lambda) = \lambda^i$. The polynomial f_m will always decrease faster than Z. At first (for values of λ close to 1) this difference will be small, however, as λ goes to 0, the rate of decrease of Z slows down because of its absolute term. Thus, at a certain point f_{m-1} will decrease faster than Z. Once λ reaches this point, we can start decreasing λ by a factor of $2^{-1/(m-1)}$. As time progresses, the rate of Z will be bounded by the rate of polynomials f_m, then f_{m-1}, f_{m-2}, ..., all the way down to f_1 for λ close to 0. When the polynomial f_i "dominates" we can decrease λ by a factor of $2^{-1/i}$. Note that the rate of decrease increases with time, i. e., the schedule is accelerating.

2.3 General Cooling Schedule. Now we formalize the accelerated cooling approach. We state our results in a general form which proves useful in other contexts, e. g., for the permanent later in this paper, and binary contingency tables [1].

Let $Z(\lambda)$ be the partition function polynomial. Let s be the degree of $Z(\lambda)$ (note that $s = m$ for colorings). Our goal is to find $1 = \lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_\ell$ such that $Z(\lambda_i)/Z(\lambda_{i+1}) \leq c$ (e. g., for colorings we took $c = 2$). The important property of $Z(\lambda)$ for colorings is $Z(0) \geq (k/e)^n$ (i. e., $Z(\lambda)$ has large constant coefficient). For some applications it will not be possible to make the absolute coefficient large, instead we will show that a coefficient a_D of λ^D is large (for some small D). Finally, let γ be an upper bound on $Z(1)/a_D$. For colorings we can take $\gamma = e^n$. The γ measures how small λ needs to get for $Z(\lambda)$ to be within constant factor of $Z(0)$. Now we present a general algorithm in terms of parameters s, c, γ, D.

Algorithm for computing the cooling schedule λ, given parameters s, c, γ, and D:

Set $\hat{\lambda}_0 = 1$, $i = s$ and $j = 0$.
While $\hat{\lambda}_j > 1/\gamma$ do
  Set $\hat{\lambda}_{j+1} = c^{-1/i}\hat{\lambda}_j$.
  If $i > D + 1$ and $\hat{\lambda}_{j+1} < (s/\gamma)^{1/(i-D)}$,
    Set $\hat{\lambda}_{j+1} = (s/\gamma)^{1/(i-D)}$
    and decrement $i = i - 1$.
  Increment $j = j + 1$.
Set $\ell = j$.

The following lemma estimates the number of intermediate temperatures in the above cooling schedule, i.e., the length ℓ of the $\hat{\lambda}$ sequence.

LEMMA 2.1. *Let $c, \gamma > 0, D \geq 0$ and let $\hat{\lambda}_0, \ldots, \hat{\lambda}_\ell$ be the sequence computed by the above algorithm. Then $\ell = O([(D+1)\log(s-D) + s/(s-D)] \log_c \gamma)$. If c and D are constants independent of s, then $\ell = O(\log s \log \gamma)$.*

We will prove the lemma in Section 2.5. Note that for colorings $\ell = O(n \log n)$.

The following lemma shows that for the sequence of the λ_i the value of $Z(\lambda)$ changes by a factor $\leq c$ for consecutive λ_i and λ_{i+1}. We postpone the proof to Section 2.5.

LEMMA 2.2. *Let $c, \gamma, D \geq 0$ and let $Z_1, \ldots, Z_q$ be a collection of polynomials of degree s. Suppose that for every $i \in [q]$, the polynomial Z_i satisfies the following conditions:*

i) Z_i has non-negative coefficients,

ii) there exists $d \leq D$ such that the coefficient of x^d in Z_i is at least $Z_i(1)/\gamma$.

Let $\hat{\lambda}_0, \hat{\lambda}_1, \ldots, \hat{\lambda}_\ell$ be the sequence constructed by the above algorithm. Then

$$Z_i(\hat{\lambda}_j) \le cZ_i(\hat{\lambda}_{j+1}) \qquad \text{for every } i \in [q] \text{ and } j \in [\ell].$$

2.4 Applying the Improved Cooling Schedule to Colorings. Before applying these general results to colorings, we quickly review how to approximate the ratios α_i in the telescoping product (1.1) (see [6] for details). We can approximate α_i using the following unbiased estimator. Let $X_i \sim \pi_i$ denote a random labeling chosen from the distribution π_i defined by $Z(\lambda_i)$, (i. e., the probability of a labeling σ is $\pi_i(\sigma) = {\lambda_i}^{M(\sigma)}/Z(\lambda_i)$). Let $Y_i = (\lambda_{i+1}/\lambda_i)^{M(X_i)}$. Then Y_i is an unbiased estimator for α_i:

$$\begin{aligned}\mathbf{E}(Y_i) &= \mathbf{E}_{X_i\sim\pi_i}\left((\lambda_{i+1}/\lambda_i)^{M(X_i)}\right)\\ &= \sum_{\sigma\in[k]^V}\frac{(\lambda_{i+1})^{M(\sigma)}}{Z(\lambda_i)} = \frac{Z(\lambda_{i+1})}{Z(\lambda_i)} = \alpha_i.\end{aligned}$$

Assume that we have an algorithm for generating labelings X_i' from a distribution that is within variation distance $\le \varepsilon/\ell$ of π_i. We draw $\Theta(\ell/\varepsilon^2)$ samples of X_i' and take the mean $\overline{Y}_i$ of their corresponding estimators Y_i'. Then the expected value of $\overline{Y}_i$ is $E[\overline{Y}_i](1\pm\varepsilon/\ell)$ and the variance can be bounded as $V[\overline{Y}_i] = O(\varepsilon^2/\ell)E[\overline{Y}_i]^2$. Therefore, by the Chebyshev's inequality $k^n\prod_{i=0}^{\ell-1}\overline{Y}_i$ equals $|\Omega|(1\pm 2\varepsilon)$ with probability $\ge 3/4$.

If the algorithm generates a sample X_i' from a distribution within variation distance $\le \varepsilon/\ell$ of π_i in time $T(\varepsilon/\ell)$, then the computation of $k^n\prod_{i=0}^{\ell-1}\overline{Y}_i$ takes time $O(\ell^2/\varepsilon^2 T(\varepsilon/\ell))$.

Now we return to colorings, and conclude the final running time of the algorithm. Recall that $\ell = O(n\log n)$. For $k > 2\Delta$, it is possible to generate a labeling within variation distance $\le \varepsilon'$ of π_i in time $T(\varepsilon') = \frac{k}{k-2\Delta}n\log(n/\varepsilon')$ [3, 6]. Hence one can approximate $|\Omega|$ within a multiplicative factor $1\pm\varepsilon$ with probability $\ge 3/4$ in $O(\frac{k}{k-2\Delta}\frac{n^3\log^2 n}{\varepsilon^2}\ln(n/\varepsilon))$ time. In contrast, for $k > 2\Delta$ using the standard counting to sampling reduction, Jerrum states a running time of $O(\frac{k}{k-2\Delta}\frac{nm^2}{\varepsilon^2}\ln(n/\varepsilon))$ where m is the number of edges. For $k \le 2\Delta$ results on mixing time are known for certain classes of graphs [5]. These are proved for k-colorings, but most likely they can be extended to the non-zero temperature.

2.5 Proof of Lemmas 2.1 and 2.2. The rest of this section is devoted to the proof of Lemmas 2.1 and 2.2.

Proof of Lemma 2.1. We define intervals:

$$I_s = [(s/\gamma)^{1/(s-D)}, \infty),$$

for $i = D+2, \ldots, s-1$,

$$I_i = [(s/\gamma)^{1/(i-D)}, (s/\gamma)^{1/(i+1-D)}],$$

and finally,

$$I_{D+1} = (0, (s/\gamma)^{1/2})].$$

Let ℓ_i be the number of $\hat{\lambda}$ values lying in the interval I_i. For $i \in \{D+2, \ldots, s-1\}$ we have the estimate:

$$\ell_i \le \log_c\left(\frac{[(s/\gamma)^{1/(i+1-D)}]^i}{[(s/\gamma)^{1/(i-D)}]^i}\right) \le \frac{D+1}{i-D}\log_c\gamma.$$

Similarly,

$$\ell_s \le \log_c\left(\frac{\gamma}{[(s/\gamma)^{1/(s-D)}]^s}\right) \le \frac{2s-D}{s-D}\log_c\gamma,$$

and

$$\ell_{D+1} \le \log_c\left(\frac{[(s/\gamma)^{1/2}]^{D+1}}{[1/\gamma]^{D+1}}\right) \le \frac{D+1}{2}\log_c\gamma.$$

Putting it all together, we get the bound

$$\begin{aligned}\ell &\le \sum_{i=D+1}^{s}\ell_i\\ &\le \left((D+1)H_{s-D} + \frac{2s-D}{s-D} + \frac{D+1}{2}\right)\log_c\gamma,\end{aligned}$$

where $H_i = \sum_{j=1}^{i} 1/j = O(\log i)$ is the harmonic sum. Therefore

$$\ell = O([(D+1)\log(s-D) + s/(s-D)]\log_c\gamma).$$

□

The *log-derivative* of a function f is $(\log f)' = f'/f$. The log-derivative measures how quickly a function increases.

DEFINITION 2.1. *We say that a polynomial f is* dominant *over a polynomial g on an interval I if $f'(x)/f(x) \ge g'(x)/g(x)$ for every $x \in I$.*

LEMMA 2.3. *Let $f, g : I \to \mathbf{R}^+$ be two non-decreasing polynomials. If f dominates over g on I, then $f(y)/f(x) \ge g(y)/g(x)$ for every $x, y \in I$, $x \le y$.*

We partition the interval $(0,\infty)$ into subintervals $I_{D+1}, \ldots, I_s$ such that x^i dominates over every Z-polynomial on the interval I_i. The $\hat{\lambda}_j$ in I_i will be such that x^i decreases by a factor c between consecutive $\hat{\lambda}$. Therefore the Z-polynomials decrease by at most a factor of c.

LEMMA 2.4. *Let $g(x) = \sum_{j=0}^{s} a_j x^j$ be a polynomial with non-negative coefficients. Then x^s dominates over g on the interval $(0,\infty)$.*

Proof. It suffices to verify that $(x^s)'/x^s \geq g'(x)/g(x)$ for every $x > 0$. □

LEMMA 2.5. *Let $g(x) = \sum_{j=0}^{s} a_j x^j$ be a polynomial with non-negative coefficients such that $g(1) \leq \gamma$ and at least one of $a_0, a_1, \ldots, a_D$ is ≥ 1. Then for any $i \geq D+1$ the polynomial x^i dominates g on the interval $(0, (s/\gamma)^{1/(i+1-D)}]$.*

Proof. The logarithmic derivative of x^i is i/x. Hence we need to prove that $ig(x) \geq xg'(x)$ for $x \leq (s/\gamma)^{1/(i+1-D)}$.

Let d be the smallest integer such that $a_d \geq 1$. From the assumptions of the lemma $d \leq D$. For $x \leq (s/\gamma)^{1/(i+1-D)}$ the following holds

$$\begin{aligned}\sum_{j=i+1}^{s} j a_j x^{j-d} &\leq \sum_{j=i+1}^{s} s a_j x^{j-D} \\ &\leq \sum_{j=i+1}^{s} s a_j \left(\frac{s}{\gamma}\right)^{(j-D)/(i+1-D)} \\ &\leq \sum_{j=i+1}^{s} s a_j \left(\frac{s}{\gamma}\right) \leq 1.\end{aligned}$$

Since $i > d$, for $x \leq (s/\gamma)^{1/(i+1-D)}$ we have

$$\begin{aligned}xg'(x) &= \sum_{j=0}^{i} j a_j x^j + \sum_{j=i+1}^{s} j a_j x^j \\ &\leq \sum_{j=d}^{i} j a_j x^j + a_d x^d \leq \sum_{j=d}^{i} i a_j x^j = ig(x).\end{aligned}$$

□

Proof of Lemma 2.2. Let $I_{D+1}, \ldots, I_s$ be as in the proof of Lemma 2.1. Let $Q_q(\lambda) = \gamma Z_q(\lambda)/Z_q(1)$. Notice that the Q_q satisfy the conditions required of g by Lemma 2.5. Therefore x^i dominates over every Q_q (and hence also Z_q) on the interval I_i for $i < s$. Moreover, Lemma 2.5 and Lemma 2.3 imply that x^s dominates over every Q_q (and hence Z_q) on the interval I_s. Notice that if $\hat{\lambda}_j, \hat{\lambda}_{j+1} \in I_i$, then $c\hat{\lambda}_{j+1}^i \geq \hat{\lambda}_j^i$ (where inequality happens only if $\hat{\lambda}_{j+1} = (s/\gamma)^{1/(i-D)}$). Therefore all of the Z_q-polynomials decrease by a factor at most c between consecutive values of $\hat{\lambda}$. □

3 Permanent Algorithm

Here we describe the simulated annealing algorithm for the permanent. We show the application of our improved cooling schedule, and our improvement in the mixing time bound for the Markov chain underlying the simulated annealing algorithm. We present the new inequalities which are key to the improved mixing time result. This analysis is more difficult than the earlier work of [8].

3.1 Preliminaries. Let $G = (V_1, V_2, E)$ be a bipartite graph with $|V_1| = |V_2| = n$. We will let $u \sim v$ denote the fact that $(u, v) \in E$. For $u \in V_1, v \in V_2$ we will have a positive real number $\lambda(u, v)$ called the *activity* of (u, v). If $u \sim v$, $\lambda(u, v) = 1$ throughout the algorithm, and otherwise, $\lambda(u, v)$ starts at 1 and drops to $1/n!$ as the algorithm evolves. The activities allow us to work on the complete graph on V_1 and V_2.

Let $\mathcal{P}$ denote the set of perfect matchings (recall that we are working on the complete graph now), and let $\mathcal{N}(u, v)$ denote the set of near-perfect matchings with holes (or unmatched vertices) at u and v. Similarly, let $\mathcal{N}(x, y, w, z)$ denote the set of matchings that have holes only at the vertices x, y, w, z. For any matching M, denote its activity as

$$\lambda(M) := \prod_{(u,v)\in M} \lambda(u, v).$$

For a set S of matchings, let $\lambda(S) := \sum_{M\in S} \lambda(M)$. For $u \in V_1, v \in V_2$ we will have a positive real number $w(u, v)$ called the *weight* of the hole pattern u, v. Given weights w, the weight of a matching $M \in \Omega$ is

$$w(M) := \begin{cases} \lambda(M)w(u, v) & \text{if } M \in \mathcal{N}(u, v), \text{and} \\ \lambda(M) & \text{if } M \in \mathcal{P}. \end{cases}$$

The weight of a set S of matchings is

$$w(S) := \sum_{M\in S} w(M).$$

For given activities, the *ideal weights* on hole patterns are the following:

$$w^*(u, v) = \frac{\lambda(\mathcal{P})}{\lambda(\mathcal{N}(u, v))} \tag{3.3}$$

Note, for the ideal weights all hole patterns have the same weight, in particular, $w^*(\mathcal{N}(u, v)) = w^*(u, v)\lambda(\mathcal{N}(u, v)) = \lambda(\mathcal{P}) = w^*(\mathcal{P})$.

3.2 Markov chain definition. At the heart of the algorithm lies a Markov chain MC, which was used in [8], and a slight variant was used in [2, 7]. Let $\lambda : V_1 \times V_2 \to \mathbb{R}_+$ be the activities and $w : V_1 \times V_2 \to \mathbb{R}_+$ be the weights. The set of states of the Markov chain is $\Omega = \mathcal{P} \cup \mathcal{N}_2$. The stationary distribution π is proportional to w, i. e., $\pi(M) = w(M)/Z$ where $Z = \sum_{M\in\Omega} w(M)$.

The transitions $M_t \to M_{t+1}$ of the Markov chain MC are defined as follows:

1. If $M_t \in \mathcal{P}$, choose an edge e uniformly at random from M_t. Set $M' = M_t \setminus e$.
2. If $M_t \in \mathcal{N}(u,v)$, choose vertex x uniformly at random from $V_1 \cup V_2$.
   (a) If $x \in \{u,v\}$, let $M' = M \cup (u,v)$.
   (b) If $x \in V_2$ and $(w,x) \in M_t$, let
   $$M' = M \cup (u,x) \setminus (w,x).$$
   (c) If $x \in V_1$ and $(x,z) \in M_t$, let
   $$M' = M \cup (x,v) \setminus (x,z).$$
   (d) Otherwise, let $M' = M_t$.
3. With probability $\min\{1, w(M')/w(M_t)\}$, set $M_{t+1} = M'$; otherwise, set $M_{t+1} = M_t$.

Note, cases 1 and 2a move between perfect and near-perfect matchings, whereas cases 2b and 2c move between near-perfect matchings with different hole patterns.

The key technical theorem is that the Markov chain quickly converges to the stationary distribution π if the weights w are close to the ideal weights w^*. The mixing time $\tau(\delta)$ is the time needed for the chain to be within variation distance δ from the stationary distribution.

THEOREM 3.1. *Assuming the weight function w is within a multiplicative factor 2 of the ideal weights for all $(u,v) \in V_1 \times V_2$ with $\mathcal{M}(u,v) \neq 0$, then the mixing time of the Markov chain MC is bounded above by $\tau(\delta) = O(n^4(\ln(1/\pi(M_0)) + \log \delta^{-1}))$.*

This theorem improves the mixing time bound by $O(n^2)$ over the corresponding result in [8].

3.3 Simulated Annealing with New Cooling Schedule. We will run the chain with weights w close to w^*, and then we can use samples from the stationary distribution to redefine w so that they are arbitrarily close to w^*. For the Markov chain run with weights w, note that

$$\begin{aligned}\pi(\mathcal{N}(u,v)) &= \frac{w(u,v)\lambda(\mathcal{N}(u,v))}{Z} = \frac{w(u,v)\lambda(\mathcal{P})}{Zw^*(u,v)} \\ &= \pi(\mathcal{P})\frac{w(u,v)}{w^*(u,v)}\end{aligned}$$

Rearranging, we have

$$w^*(u,v) = \frac{\pi(\mathcal{P})w(u,v)}{\pi(\mathcal{N}(u,v))} \tag{3.4}$$

Given weights w which are a rough approximation to w^*, identity (3.4) implies an easy method to recalibrate weights w to an arbitrarily close approximation to w^*. We generate many samples from the stationary distribution, and observe the number of perfect matchings in our samples versus the number of near-perfect matchings with holes u, v. By generating sufficiently many samples, we can estimate $\pi(\mathcal{P})/\pi(\mathcal{N}(u,v))$ within an arbitrarily close factor, and hence we can estimate $w^*(u,v)$ (via (3.4)) within an arbitrarily close factor.

For weights that are within a factor of 2 of the ideal weights, it follows that $\pi(\mathcal{N}(u,v)) \geq 1/4(n^2+1)$, which implies that $O(n^2 \log(1/\hat{\eta}))$ samples of the stationary distribution of the chain are enough to obtain a $\sqrt{2}$-approximation w' of w^* with probability $\geq 1-\hat{\eta}$. Theorem 3.1 (with $\delta = O(1/n^2)$) implies that $O(n^4 \log n)$ time is needed to generate each sample.

3.4 Algorithm for estimating ideal weights. Now we can present the simulated annealing algorithm for 0/1 matrices. The algorithm runs in phases, each characterized by a parameter $\hat{\lambda}$. In every phase,

$$\lambda(e) = \begin{cases} 1 & \text{for } e \in E \\ \hat{\lambda} & \text{for } e \notin E \end{cases} \tag{3.5}$$

We start with $\hat{\lambda} = 1$ and slowly decrease $\hat{\lambda}$ until it reaches its target value $1/n!$.

At the start of each phase we have a set of weights within a factor 2 of the ideal weights, for all u, v, with high probability. Applying Theorem 3.1 we generate many samples from the stationary distribution. Using these samples and (3.4), we refine the weights to within a factor $\sqrt{2}$ of the ideal weights. This allows us to decrease $\hat{\lambda}$ so that the current estimates of the ideal weights for $\hat{\lambda}_i$ are within a factor 2 of the ideal weights for $\hat{\lambda}_{i+1}$.

In [8], $O(n^2 \log n)$ phases are required. A straightforward way to achieve this is to decrease $\hat{\lambda}$ by a factor $(1 - 1/3n)$ between phases. This ensures that the weight of any matching changes by at most a factor $(1-1/3n)^n \leq \exp(1/3) < \sqrt{2}$.

We use only $O(n \log^2 n)$ phases by using the cooling schedule presented in Section 2. In the notation of Section 2, the permanent algorithm requires $s = n$, $c = \sqrt{2}$, $\gamma = n!$, and $D = 1$. Thus, our total running time is $O(n \log^2 n)$ phases, $O(n^2 \log n)$ samples per phase, and $O(n^4 \log n)$ time to generate each sample. Thus, the total running time is $O(n^7 \log^4 n)$.

3.5 Analyzing Mixing Time: Key Technical Inequalities. The following lemma contains the new combinatorial inequalities which are the key to our improvement of $O(n^2)$ in Theorem 3.1. In [8] simpler inequalities were proved without the sum in the left-

hand side, and were a factor of 2 smaller in the right-hand side. Using these new inequalities to bound the mixing time requires more work than the analysis of [8].

LEMMA 3.1. *Let $u, w \in V_1$, $v, z \in V_2$ be distinct vertices. Then,*

1.
$$\sum_{x,y:(u,y),(x,v)\in E} |\mathcal{N}(u,v)||\mathcal{N}(x,y)| \leq 2|\mathcal{P}|^2.$$

2.
$$\sum_{x:(x,v)\in E} |\mathcal{N}(u,v)||\mathcal{N}(x,z)| \leq 2|\mathcal{N}(u,z)||\mathcal{P}|.$$

3.
$$\sum_{x,y:(u,y),(v,x)\in E} |\mathcal{N}(u,v)||\mathcal{N}(x,y,w,z)| \leq 2|\mathcal{N}(w,z)||\mathcal{P}|.$$

Proof.

1. We will construct a one-to-one map:

$$f_1 : \mathcal{N}(u,v) \times \bigcup_{x,y:(u,y),(v,x)\in E} \mathcal{N}(x,y) \to \mathcal{P} \times \mathcal{P} \times b,$$

where b is a bit, i.e., b is 0/1.

Let $L_0 \in \mathcal{N}(u,v)$ and $L_1 \in \cup_{x,y:(u,y),(v,x)\in E}\mathcal{N}(x,y)$. In $L_0 \oplus L_1$ the four vertices u, v, x, y each have degree one, and the remaining vertices have degree zero or two. Hence these four vertices are connected by two disjoint paths. Now there are three possibilities:

- If the paths are u to x and v to y, they must both be even.
- If the paths are u to v and x to y, they must both be odd.
- The third possibility, u to y and v to x is ruled out since such paths start with an L_0 edge and end with an L_1 edge and hence must be even length; on the other hand, they connect vertices across the bipartition and hence must be of odd length.

Now, the edges (u,y) and (v,x) are in neither matching, and so $(L_0 \oplus L_1) \cup \{(u,y),(v,x)\}$ contains an even cycle, say C, containing (u,y) and (v,x). We will partition the edges of $L_0 \cup L_1 \cup \{(u,y),(v,x)\}$ into two perfect matchings as follows. Let M_0 contain the edges of L_0 outside of C and alternate edges of C starting with edge (u,y). M_1 will contain the remaining edges. Bit b is set to 0 if $(x,v) \in M_0$ and to 1 otherwise. This defines the map f_1.

Next, we show that f_1 is one-to-one. Let M_0 and M_1 be two perfect matchings and b be a bit. If u and v are not in one cycle in $M_0 \oplus M_1$ then (M_0, M_1, b) is not mapped onto by f_1. Otherwise, let C be the common cycle containing u and v. Let y be the vertex matched to u in M_0. If $b = 0$, denote by x the vertex that is matched to v in M_0; else denote by x the vertex that is matched to v in M_1. Let L_0 contain the edges of M_0 outside C and let it contain the near-perfect matching in C that leaves u and v unmatched. Let L_1 contain the edges of M_1 outside C and let it contain the near-perfect matching in C that leaves x and y unmatched. It is easy to see that $f_1(L_0, L_1) = (M_0, M_1, b)$.

The proofs for cases (2) and (3) of the Lemma follow a similar approach. □

In fact, the following extension of the previous lemma, is used in the proof of Theorem 3.1. This is a weighted version of the previous lemma.

LEMMA 3.2. *Let $u, w \in V_1$, $v, z \in V_2$ be distinct vertices. Then,*

1.
$$\sum_{x\in V_1, y\in V_2} \lambda(u,y)\lambda(x,v)\lambda(\mathcal{N}(u,v))\lambda(\mathcal{N}(x,y)) \leq 2\lambda(\mathcal{P})^2.$$

2.
$$\sum_{x\in V_1} \lambda(x,v)\lambda(\mathcal{N}(u,v))\lambda(\mathcal{N}(x,z)) \leq 2\lambda(\mathcal{N}(u,z))\lambda(\mathcal{P}).$$

3.
$$\sum_{x\in V_1, y\in V_2} \lambda(u,y)\lambda(x,v)\lambda(\mathcal{N}(u,v))\lambda(\mathcal{N}(x,y,w,z)) \leq 2\lambda(\mathcal{N}(w,z))\lambda(\mathcal{P}).$$

To bound the mixing time we use the canonical paths technique, and then bound the congestion, which is the total weight of paths through any transition. That follows the same basic approach used in previous MCMC algorithms for the permanent [7, 8]. However, to use the new inequalities in our analysis of the congestion, we need to partition the analysis into several cases based on the type of transition and canonical path.

4 Conclusion

With the improvement in running time of the approximation algorithm for the permanent, computing permanent of $n \times n$ matrices with $n \approx 100$ now appears feasible. Further improvement in running time is an important open problem.

Some avenues for improvement are the following. We expect that the mixing time of the underlying chain

is better than $O(n^4)$. Some slack in the analysis is in the application of the new inequalities to bound the congestion. In their application we simply use a sum over y, whereas the inequalities hold for a sum over x and y as stated in Lemma 3.2. It is possible that fewer samples are needed at each intermediate activity for estimating the ideal weights w^*. Perhaps the w^* satisfy relations which allow for fewer samples to infer them.

References

[1] I. Bezáková, N. Bhatnagar, and E. Vigoda, Sampling Binary Contingency Tables with a Greedy Start, to appear in *Proceedings of the 17th Annual ACM-SIAM Symposium on Discrete Algorithms* (SODA), 2006.

[2] A. 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), 1986, 50–58. Erratum in *Proceedings of the 20th Annual ACM Symposium on Theory of Computing*, 1988, p. 551.

[3] R. Bubley and M. Dyer, Path coupling: A technique for. proving rapid mixing in Markov chains. In *Proceedings of the 38th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, 223–231, 1997.

[4] M.E. Dyer, A.M. Frieze and R. Kannan, A random polynomial time algorithm for approximating the volume of convex bodies, *Journal of the Association for Computing Machinery*, 38(1):1–17, 1991.

[5] T. Hayes, E. Vigoda, A Non-Markovian Coupling for Randomly Sampling Colorings. In *Proceedings of the 44th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, 618–627, 2003.

[6] M. Jerrum, A Very Simple Algorithm for Estimating the Number of k-Colorings of a Low-Degree Graph. *Random Struct. Algorithms*, 7(2):157–166, 1995.

[7] M. Jerrum and A. Sinclair, Approximating the permanent, *SIAM Journal on Computing*, 18:1149–1178, 1989.

[8] M.R. Jerrum, A. Sinclair and E. Vigoda, A polynomial-time approximation algorithm for the permanent of a matrix with non-negative entries. *Journal of the Association for Computing Machinery*, 51(4):671-697, 2004.

[9] M. Jerrum, L. Valiant and V. Vazirani, Random generation of combinatorial structures from a uniform distribution, *Theoretical Computer Science*, 43:169–188, 1986.

[10] M. Jerrum and U. Vazirani, A mildly exponential approximation algorithm for the permanent, *Algorithmica*, 16:392–401, 1996.

[11] A. Kalai and S. Vempala, Simulated Annealing for Convex Optimization, Preprint.

[12] P. W. Kasteleyn, The statistics of dimers on a lattice, I., The number of dimer arrangements on a quadratic lattice, *Physica*, 27:1664-1672, 1961.

[13] S. Kirkpatrick, C. D. Gelatt Jr., and M. P. Vecchi, Optimization by Simulated Annealing, *Science*, 220:671-680, 1983.

[14] L. Lovász and S. Vempala, Simulated Annealing in Convex Bodies and an $O^*(n^4)$ Volume Algorithm. In *Proceedings of the 44th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, 650–659, 2003.

[15] P. McCullagh, On prediction and density estimation, Preprint, 2004.

[16] H. Pasula, S.J. Russell, M. Ostland, Y. Ritov, Tracking Many Objects with Many Sensors, *Sixteenth International Joint Conference on Artificial Intelligence (IJCAI)*, 1160–1171, 1999.

[17] A. Sinclair, Improved bounds for mixing rates of Markov chains and multicommodity flow, *Combinatorics, Probability and Computing*, 1:351–370, 1992.

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