Sampling Binary Contingency Tables with a Greedy Start

Ivona Bezáková* Nayantara Bhatnagar† Eric Vigoda†

October 31, 2006

Abstract

We study the problem of counting and randomly sampling binary contingency tables. For given row and column sums, we are interested in approximately counting (or sampling) $0/1$ $n \times m$ matrices with the specified row/column sums. We present a simulated annealing algorithm with running time $O((nm)^2 D^3 d_{\max} \log^5(n+m))$ for any row/column sums where D is the number of non-zero entries and $d_{\max}$ is the maximum row/column sum. In the worst case, the running time of the algorithm is $O(n^{11} \log^5 n)$ for an $n \times n$ matrix. This is the first algorithm to directly solve binary contingency tables for all row/column sums. Previous work reduced the problem to the permanent, or restricted attention to row/column sums that are close to regular. The interesting aspect of our simulated annealing algorithm is that it starts at a non-trivial instance, whose solution relies on the existence of short alternating paths in the graph constructed by a particular Greedy algorithm.

1 Introduction

Counting and randomly sampling binary contingency tables is an important problem in Statistics. Various methods have been proposed in the literature (e.g., see [6, 2, 4, 19] for recent work related to the classical problem of "Darwin's finches"), but the desired theoretical work on the efficiency of these approaches is lacking.

The problem can be formalized as follows. Given a pair of non-negative integer sequences $r(1), \ldots, r(n)$ and $c(1), \ldots, c(m)$, our goal is to generate a random $n \times m$ $0/1$ matrix where the i-th row sums to $r(i)$ and the j-th column sums to $c(j)$, for all $1 \le i \le n, 1 \le j \le m$. An efficient (approximate) sampling scheme then yields an approximation algorithm to count the number of tables with the desired row/column sums. Graph theoretically, we are generating a random bipartite graph with vertex set $u_1, \ldots, u_n$ and $v_1, \ldots, v_m$ where u_i has degree $r(i)$ and v_j has degree $c(j)$. We will primarily use the graph theoretic view in the presentation and analysis of our algorithm.

Our emphasis is on algorithms that are provably efficient for arbitrary degree sequences. More precisely, we are seeking an algorithm which approximates the number of tables within a multiplicative factor $1 \pm \varepsilon$ with probability at least $1 - \delta$ and runs in time polynomial in $n, m, 1/\varepsilon$ and $\log(1/\delta)$. The problem of counting the number of binary contingency tables is not known to be $\#P$-complete, hence it is possible that there is an algorithm which counts the number of tables exactly, not just approximately. However, until now, the only method known for approximating the number of contingency tables was by reducing the problem to approximating the permanent [12, 10]. The reduction

*Department of Computer Science, University of Chicago, Chicago, IL 60637. Email: ivona@cs.uchicago.edu.

†College of Computing, Georgia Institute of Technology, Atlanta, GA 30332. Email: {nand,vigoda}@cc.gatech.edu.
Supported by NSF grant CCR-0455666.

results in a permanent computation of an $\ell \times \ell$ matrix where $\ell = \Omega((n+m)^2)$. The quadratic increase in the size of the instance is particularly unappealing in light of the large running time for permanent algorithms. The best known algorithm for the permanent of an $n \times n$ 0/1 matrix runs in $O(n^7 \log^4 n)$ time [3] and thus results in an $O^*(n^{14})$ algorithm for generating random binary contingency tables.

We present a new algorithm for binary contingency tables with arbitrary degree sequences, by directly exploiting the combinatorial structure of the problem. The resulting algorithm is considerably faster than permanent based algorithms, although our new algorithm is still far from practical.

Before giving a high level description of our results, it is important to highlight several alternative approaches for binary contingency tables. Two popular approaches are a Markov chain, known as the Diaconis chain, which walks on the set of tables with the desired row/column sums [8], and importance sampling, e.g., see [6].

The Diaconis chain is known to converge to a random table for regular degree sequences (i.e., $r(i) = c(j)$ for all i, j) and sequences sufficiently close to regular, see Kannan, Tetali and Vempala [14], and Cooper, Dyer and Greenhill [5]. An alternative (non Markov chain) approach for regular degree sequences with degree $O(n^{1/3})$ was presented by Kim and Vu [15]. No theoretical results are known for either of these approaches for arbitrary degree sequences. The importance sampling approach appears to work well in practice, and recently Bayati and Saberi [1] proved that a variant can be used to sample efficiently when the maximum degree is at most $O(D^{1/4-\varepsilon})$, where D is the number of edges in the graph. We refer the reader to [16] for a discussion of importance sampling, and [6] for recent refinements of the approach.

Our new algorithm is inspired by the permanent algorithm of Jerrum, Sinclair, and Vigoda [12], but requires an interesting algorithmic twist. The new algorithmic idea relies on a combinatorial property of bipartite graphs satisfying a given degree sequence.

The basis of our algorithm is a Markov chain which walks on bipartite graphs with the desired degree sequence and graphs with exactly two deficiencies. We say a graph has a deficiency at vertices u_i and v_j if they have degree $r(i)-1$ and $c(j)-1$, respectively, and all other vertices have the desired degree. The number of graphs with the desired degree sequence might be exponentially fewer than the number of graphs with two deficiencies. Thus, we need to weight the random walk defined by the Markov chain so that graphs with the desired degree sequence are likely in the stationary distribution.

Let $w(i, j)$ denote the ratio of the number of graphs with the desired degree sequence versus the number of graphs with deficiencies at u_i and v_j. It turns out that given rough approximations to $w(i, j)$, for all i, j, the Markov chain weighted by these ratios quickly reaches its stationary distribution, and samples from the stationary distribution can then be used to get arbitrarily close estimates of $w(i, j)$. This type of bootstrapping procedure for recalibrating the ratios $w(i, j)$ was central to the algorithm for the permanent.

For the permanent there is an analogous Markov chain on perfect matchings and matchings with at most two unmatched vertices (or holes) where the corresponding ratios, denoted as $\widehat{w}(i, j)$, are the number of perfect matchings divided by the matchings with holes at u_i, v_j. In the case of the permanent, a bootstrapping algorithm for computing the ratios $\widehat{w}$ yields a natural simulated annealing algorithm. Consider an unweighted bipartite graph G that we wish to compute the number of perfect matchings of. In the complete bipartite graph, denoted as G_0, it is trivial to exactly compute the ratios $\widehat{w}(i, j)$ for every i, j. From G_0, we then slightly decrease the weight of edges not appearing in G, giving a new weighted graph G_1. Using $\widehat{w}$ for G_0 we use the bootstrapping to closely estimate $\widehat{w}$ for G_1. Then we, alternately, decrease (slightly) the weight of non-edges of G creating a new graph G_i, and then use the estimates of $\widehat{w}$ for G_{i-1} to bootstrap $\widehat{w}$ for G_i. A crucial element

of this algorithmic approach is that the quantities $\widehat{w}(i,j)$ are trivial to compute in the initial graph, which in this case is the complete bipartite graph.

For contingency tables, what is a starting instance where we can easily estimate the ratios $w(i,j)$'s? Recall that our final goal is to sample subgraphs of the complete bipartite graph with given degree sequence. It is not clear that there is some trivial graph which we can use to start the simulated annealing algorithm. This is the key problem we overcome.

We prove that if we construct a graph G^* with the desired degree sequence using a particular Greedy algorithm, we can estimate the ratios $w(i,j)$ in the weighted complete bipartite graph where edges of G^* have weight 1 and non-edges have sufficiently small weight. The algorithm to estimate the ratios follows immediately from the following property of G^*. For every pair of vertices u_i, v_j, there is a short alternating path between u_i and v_j, or there is no graph with the degree sequence with deficiencies at u_i, v_j. (An alternating path is a path which alternates between edges and non-edges of G^*.) Moreover, the alternating path is of length at most 5, which implies an easy algorithm to count the number of minimum length alternating paths. This combinatorial fact is the main result of this paper. It is interesting to note that this combinatorial property fails to hold for many natural variants of Greedy and max-flow algorithms for constructing a graph with a specified degree sequence. To the authors' knowledge, this is the first example of a simulated annealing algorithm which starts with a non-trivial instance.

The algorithmic consequence of our work is an $O((nm)^2 D^3 d_{\max} \log^5(n+m))$ time algorithm to approximately count the number of bipartite graphs with the desired degree sequence, where $D = \sum r(i) = \sum c(j)$ is the total degree and $d_{\max} = \max\{\max_i r(i), \max_j c(j)\}$ is the maximum degree. In the worst case this translates to an $O(n^{11} \log^5 n)$ algorithm for an $n \times n$ matrix since $D = O(n^2)$ and $d_{\max} = O(n)$. Moreover, we can count subgraphs of any input graph with the given degree sequence (see Section 2.5 for a discussion of this extension). The following is a precise statement of our main result.

Theorem 1. *For any bipartite graph $G = (U \cup V, E)$ where $U = \{u_1, \ldots, u_n\}$ and $V = \{v_1, \ldots, v_n\}$, any degree sequence $r(1), \ldots, r(n); c(1), \ldots, c(m)$, and any $0 < \varepsilon, \eta < 1$, we can approximate the number of subgraphs of G with the desired degree sequence (i.e., u_i has degree $r(i)$ and v_j has degree $c(j)$, for all i, j) in time $O((nm)^2 D^3 d_{\max} \log^5(nm/\varepsilon)\varepsilon^{-2} \log(1/\eta))$ where $D = \sum_i r(i) = \sum_j c(j)$ is the total degree and $d_{\max} = \max\{\max_i r(i), \max_j c(j)\}$ is the maximum degree. The approximation is guaranteed to be within a multiplicative factor $(1 \pm \varepsilon)$ of the correct answer with probability $\geq 1 - \eta$.*

The permanent is a special case of the problem statement in Theorem 1 when $m = n$ and for $1 \leq i \leq n$, $r_i = c_i = 1$. In fact, the problem statement in Theorem 1 can be reduced to computing the permanent in a similar manner as binary contingency tables. However, as mentioned, the reduction causes a quadratic increase in the size of the instance. The running time of our algorithm when the degrees are all constant is $O(n^7 \log^5 n)$ which matches the running time of the fastest algorithm for the permanent [3].

The paper is organized as follows. In Section 2 we give the basic definitions and present a high level description of our simulated annealing algorithm. This section aims to motivate our work on the Greedy algorithm. We prove our main result about short alternating paths in the graph constructed by a particular variant of Greedy in Section 3. In Section 4 we analyze the mixing time of the Markov chain which is used in the simulated annealing algorithm. We conclude with the details of the simulated annealing algorithm in Section 5. For completeness we sketch the standard reduction from counting to sampling in Section 6. Finally we give a breakup of the running time stated in Theorem 1 in Section 7.

2 Preliminaries

2.1 Definitions

We use U and V to denote the partitions of vertices of the bipartite graph on $n+m$ vertices. Thus, the desired degree sequence is denoted by r and c where $r : U \to \mathbf{N}_0, c : V \to \mathbf{N}_0$, and $\mathbf{N}_0$ is the set of non-negative integers.

For every vertex $v \in V(G)$, let $N(v)$ denote its neighborhood set and let $\overline{N}(v) = V \setminus N(v)$ if $v \in U$, and $\overline{N}(v) = U \setminus N(v)$ if $v \in V$. We will use a and u to denote vertices in U and b and v to denote vertices in V.

Definition 2. *We say that a bipartite graph with partitions U, V* corresponds to the degree sequences $r : U \to \mathbf{N}_0$, $c : V \to \mathbf{N}_0$ *if* $\deg(a) = r(a)$ *for every $a \in U$ and* $\deg(b) = c(b)$ *for every $b \in V$. A pair of degree sequences r, c is* feasible *if there exists a corresponding bipartite graph.*

Let $\mathcal{P}$ be the set of all graphs corresponding to r, c. Recall, our overall aim is to approximate $|\mathcal{P}|$. By a standard reduction [13] this can be done by sampling almost uniformly at random from $\mathcal{P}$.

It is easy to construct a graph with the desired degree sequence, or determine that no such graph exists, using a Greedy algorithm (there are many valid variants) or a max-flow algorithm. We study one such variant of Greedy in Section 3. Hence, we can assume that r, c defines a feasible degree sequence.

In our simulated annealing algorithm, graphs with the desired degree sequence, except at two vertices, called holes (or deficiencies), will play a central role. This is akin to the role of near-perfect matchings in algorithms for the permanent.

Definition 3. *Let $u \in U, v \in V$ and let r, c be a pair of degree sequences on U, V. We define degree sequences with holes at u, v as follows:*

$$r^{(u)}(a) := \begin{cases} r(a) & \text{if } a \neq u \\ r(a) - 1 & \text{if } a = u \end{cases} \qquad c^{(v)}(b) := \begin{cases} c(b) & \text{if } b \neq v \\ c(b) - 1 & \text{if } b = v \end{cases}$$

We say that u, v is a pair of feasible holes for the degree sequences *r, c if the pair of sequences $r^{(u)}$, $c^{(v)}$ is feasible.*

Let $\mathcal{N}(u, v)$ be the set of all graphs corresponding to $r^{(u)}, c^{(v)}$ where $u \in U$, $v \in V$, and let $\mathcal{N} = \cup_{u,v} \mathcal{N}(u, v)$. Let $\Omega = \mathcal{P} \cup \mathcal{N}$.

2.2 High-level Algorithm Description

We give a rough description of the simulated annealing algorithm for binary contingency tables. This is not the novel aspect of our paper, the algorithmic approach is very much inspired by algorithms for the permanent. Our emphasis in this section is to motivate our main result about the graph constructed by the Greedy algorithm.

The simulated annealing algorithm will consider a sequence of activities on edges of the complete bipartite graph, i.e., for all pairs (x, y) where $x \in U, y \in V$. There will be a subgraph corresponding to the Greedy algorithm which always has activity 1 on each edge, and the other edges will initially have activities $\lambda \approx 0$, and these edges will slowly increase their activities to $\lambda = 1$. More precisely, let G^* denote the graph with the desired degree sequence constructed by Greedy algorithm which

is formally defined in Section 3. (The details of this graph are not relevant at this stage.) For a positive parameter λ, we define the activity of edge $e = (x, y)$, $x \in U$, $y \in V$, as:

$$\lambda(e) = \begin{cases} 1 & \text{if } e \in E(G^*) \\ \lambda & \text{if } e \notin E(G^*) \end{cases}$$

The activity of a graph $G \in \Omega$ is then defined as:

$$\lambda(G) = \prod_{e \in E(G)} \lambda(e) = \lambda^{|E(G) \setminus E(G^*)|}$$

Finally, the activity of a set of graphs is $\lambda(S) = \sum_{G \in S} \lambda(G)$.

2.3 Bootstrapping

A key quantity are the following *ideal weights*:

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

It turns out that given close approximations of these weights, there is a Markov chain which can be used to efficiently generate samples from $\mathcal{P}$ weighted by λ. Thus, using these ideal weights for $\lambda = 1$ we can efficiently sample graphs with the desired degree sequence. Given rough approximations of the ideal weights w^* (say within a constant factor), samples from the Markov chain can be used to boost these weights into an arbitrarily close approximation of the ideal weights. This is the bootstrapping procedure. The same approach is used for the approximation of the permanent. Further details of the bootstrapping procedure can be found in Section 5.1.

Using the bootstrapping procedure to refine rough estimates of the ideal weights we can obtain a simulated annealing algorithm for sampling binary contingency tables. We start with λ_0 close to 0 (specifically with $\lambda_0 = \varepsilon(nm)^{-D}$), where, for a particular choice of G^*, it turns out to be possible to compute a $(1 \pm \varepsilon)$ approximation of the ideal weights $w^*_{\lambda_0}$ in a straightforward manner. We will then raise λ slightly to a new value λ_1. For example, suppose we set $\lambda_1 = \left(1 + 1/\ln(2^{-1/4})D\right)\lambda_0$ where D is the total number of edges in the graph. Then for any graph G, $\lambda_1(G)$ is within a factor of $2^{1/4}$ of $\lambda_0(G)$. This implies that $\lambda_1(\mathcal{P})$ and $\lambda_1(\mathcal{N}(u, v))$ will be within a factor of $2^{1/4}$ respectively of $\lambda_0(\mathcal{P})$ and $\lambda_0(\mathcal{N}(u, v))$. Then, the ideal weights $w^*_{\lambda_0}$ for λ_0 will be a $\sqrt{2}$-approximation of the ideal weights for λ_1. We use the bootstrapping procedure to boost these to get arbitrarily close estimates of $w^*_{\lambda_1}$. We can then continue to alternately raise λ by a factor $\left(1 + 1/\ln(2^{-1/4})D\right)$, and then bootstrap new estimates of the ideal weights. In $O(D^2 \log(mn))$ steps, λ becomes 1 and we will have a suitable approximation of the ideal weights for $\lambda = 1$. It turns out that we can use a more efficient algorithm for updating λ, so that the ideal weights are still constant factor approximations for the successive ideal weights, see Section 5.2.

Algorithms for the permanent use a similar simulated annealing approach, but instead start at the complete bipartite graph and slowly remove edges not appearing in the input graph. We instead start at a graph which depends on the desired degree sequence. We then slowly add in non-edges until we reach the complete bipartite graph. In some sense we are doing a reverse annealing.

2.4 Estimating Initial Weights

Now we can address how we estimate the ideal weights for λ sufficiently small. Note, $\lambda(\mathcal{P})$ and $\lambda(\mathcal{N}(u,v))$ are polynomials in λ. In particular,

$$\lambda(\mathcal{P}) = \sum_{k=0}^{D} p_k \lambda^{D-k},$$

where p_k denotes the number of graphs corresponding to r, c which contain exactly k edges of G^*. Similarly,

$$\lambda(\mathcal{N}(u,v)) = \sum_{k=0}^{D-1} p_k^{u,v} \lambda^{D-1-k},$$

where $p_k^{u,v}$ is the number of graphs corresponding to $r^{(u)}, c^{(v)}$ which contain exactly k edges of G^*.

For λ sufficiently small, to approximate $\lambda(\mathcal{N}(u,v))$ it suffices to determine the leading non-zero coefficient, i.e., $p_j^{u,v}$ such that $p_k^{u,v} = 0$ for $k > j$. Note that the sum of all the coefficients in the polynomial is at most $(nm)^D$. Then, for $\lambda \leq \varepsilon/(nm)^D$, for some $\varepsilon > 0$, we claim that $x_{u,v} = p_j^{u,v}\lambda^{D-1-j}$ is a $(1+\varepsilon)$ approximation of $\lambda(\mathcal{N}(u,v))$. Formally,

$$x_{u,v} \leq \lambda(\mathcal{N}(u,v)) = x_{u,v} + \sum_{k=0}^{j-1} p_k^{u,v}\lambda^{D-1-k} \leq x_{u,v} + \lambda^{D-j}\sum_{k=0}^{j-1} p_k^{u,v} \leq x_{u,v} + \varepsilon\lambda^{D-j-1} \leq (1+\varepsilon)x_{u,v}$$

The second last inequality follows because $\lambda(nm)^D \leq \varepsilon$. The last inequality follows since $x_{u,v} \geq \lambda^{D-1-j}$.

The graph G^* constructed by Greedy has degree sequence r, c, and hence it has exactly one subgraph (G^* itself) that has this degree sequence. Thus, the absolute coefficient of $\lambda(\mathcal{P})$ is 1 and we can approximate $\lambda(\mathcal{P})$ by 1. For $u \in U, v \in V$, if $(u,v) \in E(G^*)$, then the subgraph with edges $E(G^*) \setminus (u,v)$ has holes at u, v, and this is the only subgraph with degree sequence $r^{(u)}, c^{(v)}$. In this case we can also approximate $\lambda(\mathcal{N}(u,v))$ by 1.

If $(u,v) \notin E(G^*)$, then there is no subgraph of G^* with holes at u, v, i.e., degree sequence $r^{(u)}, c^{(v)}$ so that $p_{D-1}^{u,v} = 0$. Note, we cannot approximate $\lambda(\mathcal{N}(u,v))$ by 0, since we need an approximation that is close within a multiplicative factor. We instead need to determine a non-zero coefficient of lowest degree in the polynomial. Since $p_k^{u,v}$ is the number of graphs corresponding to $r^{(u)}, c^{(v)}$ with exactly k edges of G^*, the degree of the leading non-zero term in $\lambda(\mathcal{N}(u,v))$ is ℓ where $2\ell+1$ is the length of the shortest (de-augmenting) alternating path between u and v in G^*. We prove that for our particular choice of G^*, for every u, v there is an alternating path from u to v of length at most 5, or u, v are infeasible holes (in which case we do not need to consider their polynomial). Since these alternating paths are so short, in polynomial time we can simply enumerate all possible such paths, and exactly determine the leading non-zero coefficient, thereby obtaining a good approximation of $\lambda(\mathcal{N}(u,v))$.

This will result in the following lemma:

Lemma 4. *Let r, c be a feasible degree sequence and let $\varepsilon > 0$ and $\lambda \leq \frac{\varepsilon}{(nm)^D}$. There exists a graph G^* (independent of ε and λ) such that for any pair of feasible holes u, v we can compute a weight $w(u,v)$ satisfying*

$$(1-\varepsilon)w(u,v) \leq w_\lambda^*(u,v) \leq (1+\varepsilon)w(u,v).$$

in time $O(nmd_{\max}^2)$. Overall, the construction of G^* together with the computation of $w(u,v)$ for all feasible holes u, v takes time $O((nmd_{\max})^2)$.

2.5 Subgraphs of Arbitrary Input Graph

The above high-level algorithm description applies to the contingency tables problem, where we are generating a random subgraph of the complete bipartite graph $K_{n,m}$ with the desired degree sequence. Our approach extends to subgraphs of any bipartite graph $G = (V, E)$.

The general algorithm proceeds as in Section 2.2. Thus, regardless of G, we construct G^* using the Greedy algorithm and approximate the initial weights. For non-edges of G^*, their activity is slowly raised from $\lambda \approx 0$ to $\lambda = 1$. At this stage all edges have activity $\lambda = 1$, and thus we can generate random subgraphs of $K_{n,m}$ with the desired degree sequence. Then for non-edges of G, i.e., $(u, v) \notin E$, we slowly lower their activity from $\lambda(u, v) = 1$ to $\lambda(u, v) \approx 0$.

Lowering the activities is analogous to raising the activities, and simply requires that the weights w^* at the previous activities can be used to bootstrap the weights w^* at the new activities. Finally, the algorithm ends with close approximations to the weights w^* for the graph with activities of $\lambda(u, v) = 1$ for all $(u, v) \in E$ and $\lambda(u, v) \approx 0$ for all $(u, v) \notin E$. Therefore, we can generate random subgraphs of G with the desired degree sequence.

2.6 Analysis Details

The analysis of the Markov chain underlying the simulated annealing algorithm requires considerable technical work. It combines many of the ideas in the recent works of Cooper, Dyer and Greenhill [5], Kannan, Tetali and Vempala [14], Jerrum, Sinclair and Vigoda [12], and Bezáková et. al. [3]. This analysis is contained in Section 4. In Section 5 we give the details of the simulated annealing algorithm and analyze its running time. In the next section we prove Lemma 4.

3 Greedy graph

In this section we prove that in the graph constructed by Greedy, a standard greedy algorithm with a specific rule to break ties which will be described shortly, for all u, v there is a short alternating path from u to v or there is no graph with holes at u, v. This immediately implies Lemma 4.

Definition 5. *Let $G = (U, V, E)$ be a bipartite graph with partitions U, V and edge set E, and let $u \in U$, $v \in V$. We say that there exists an* alternating path from u to v *of length $2k + 1$, if there exists a sequence of vertices $u = w_0, w_1, \ldots, w_{2k}, w_{2k+1} = v$ such that $w_{2i} \in U, w_{2i+1} \in V$ and $(w_{2i}, w_{2i+1}) \in E$ for every $i \in \{0, \ldots, k\}$, and $(w_{2i-1}, w_{2i}) \notin E$ for every $i \in \{1, \ldots, k\}$.*

The Greedy algorithm depends on an ordering of the vertices. We need an ordering which is consistent with the degree sequence in the following sense.

Definition 6. *Fix $c : V \to \mathbf{N}_0$ and let π be a total ordering on V. We say that π is* consistent with *c, if for every b_1, b_2 with $c(b_1) > c(b_2)$, vertex b_1 precedes b_2 in π (i.e. $b_1 \prec_\pi b_2$).*

We now define the Greedy algorithm which is the focus of our analysis. It can be viewed as a recursive procedure which matches the highest degree vertex in U, say x, to $r(x)$ highest degree vertices in V. Then the procedure recurses on the residual degree sequence obtained from the original sequence by setting the degree of x to zero and decrementing the degrees of all its neighbors, until

all residual degrees equal zero. However, we need to specify how to break ties when two vertices have the same residual degree. This turns out to be the crucial aspect of our algorithm. For this purpose we introduce an additional parameter of the algorithm, a preference relation π which is initially consistent with c. For the recursive call we use a relation $\hat{\pi}$ induced by π on the residual sequence $\hat{c}$. Here is the formal description of the algorithm:

Procedure GREEDY(r, c, π),
where $r : U \to \mathbf{N}_0$, $c : V \to \mathbf{N}_0$ are degree sequences and π is a total ordering on V consistent with c

- Let $G = (U, V, \emptyset)$ be a bipartite graph with partitions U, V and no edges.
- If $\sum_{a \in U} r(a) \neq \sum_{b \in V} c(b)$, return "Sequences not feasible".
- If $\sum_{a \in U} r(a) = 0$, return G.
- Let $x \in U$ be a vertex for which $r(x)$ is maximum (if there is more than one, choose arbitrarily).
- Let $Y \subseteq V$ be the first $r(x)$ vertices in the ordering π.
- If Y contains a vertex of degree 0, return "Sequences not feasible".
- For every $y \in Y$, add the edge (x, y) to G.
- Let $\hat{G} :=$ GREEDY$(\hat{r}, \hat{c}, \hat{\pi})$, where

$$\hat{r}(a) = \begin{cases} r(a) & a \in U \setminus x \\ 0 & a = x \end{cases} \qquad \hat{c}(b) = \begin{cases} c(b) - 1 & b \in Y \\ c(b) & b \in V \setminus Y \end{cases}$$

  and $\hat{\pi}$ is a total ordering on V defined by: $b_1 \prec_{\hat{\pi}} b_2$ if and only if $\hat{c}(b_1) > \hat{c}(b_2)$ or $\hat{c}(b_1) = \hat{c}(b_2)$ and $b_1 \prec_{\pi} b_2$.
- Add the edges of $\hat{G}$ to G and return G.

For completeness we prove that the Greedy algorithm does indeed output a graph corresponding to the desired degree sequence iff it is feasible, see Lemma 17 in Section 8. The correctness of the algorithm appears to be a folklore result. A related, standard algorithmic result is a max-flow characterization for the existence of graphs satisfying a given bipartite degree sequence in [9, 18].

Now we are ready to present our main result. We claim that in the graph constructed by Greedy there is a short (constant-length) alternating path between any two *feasible* holes. One such graph is depicted in Figure 1. It shows a pair of feasible vertices a_5, b_5 and an alternating path $a_5, b_2, a_6, b_6, a_2, b_5$ between them of length 5. Notice that the holes a_7, b_7 are infeasible, thus there is no alternating path between them. In contrast with our main result, in Lemma 18 in Section 8 we construct a family of graphs which require alternating paths of linear length for certain pairs of holes. Each graph in this family is an output of a greedy algorithm which breaks ties arbitrarily.

Theorem 7. *Let r, c be a pair of feasible degree sequences and let π be a total ordering on V consistent with c. Let $G = (U, V, E)$ be the graph constructed by* GREEDY(r, c, π)*. Then for any pair of feasible holes $u \in U$, $v \in V$ in G there exists an alternating path from u to v of length ≤ 5.*

Proof. We prove the Lemma by induction on the number of non-zero entries in r. In the base case, there is a single non-zero entry in r. For any pair of feasible holes u, v, the non-zero entry is $r(u)$ and G contains the edge (u, v). Thus u, v forms an alternating path of length 1.

For the inductive hypothesis, assume that the Lemma is true for every triple (r', c', π'), where r', c' are feasible degree sequences, r' contains fewer non-zero entries than r, and π' is a total ordering consistent with c'. Let $u \in U$, $v \in V$ be a pair of feasible holes for r, c. Suppose that $U = \{a_1, \ldots, a_n\}$ and the edges adjacent to $a_i \in U$ are added in the i-th iteration (or recursive call) of GREEDY(r, c, π).

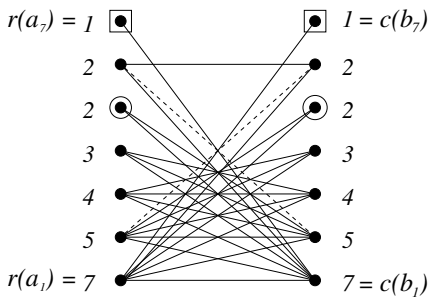

Figure 1: The Greedy graph on the sequence $(1, 2, 2, 3, 4, 5, 7)$, $(1, 2, 2, 3, 4, 5, 7)$.

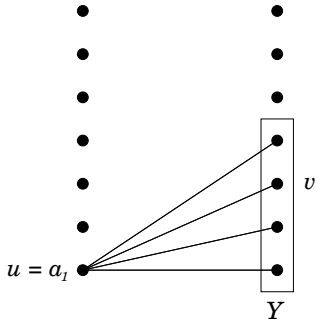

Figure 2: $u = a_1$ and $v \in Y$

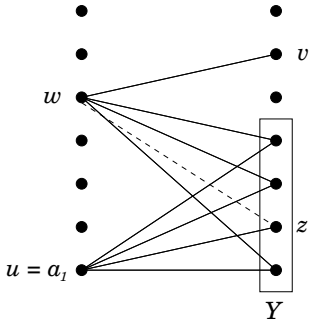

Figure 3: $u = a_1$ and $v \in V \setminus Y$

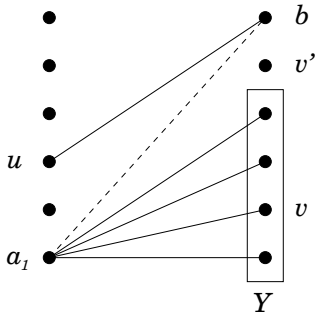

Figure 4: u has a neighbor $b \in V \setminus Y$

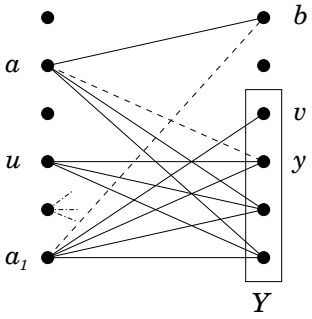

Figure 5: Vertex $b \in V \setminus Y$ of residual degree ≥ 1

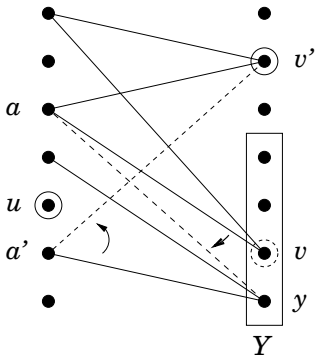

Figure 6: Neighborhoods of v, v' are identical

We say that this is the recursive call when a_i is *processed*. In the first recursive call $x = a_1$. Recall that Y denotes the set of a_1's neighbors.

- If $u = a_1$, we construct a short alternating path from u to v as follows (Figures 2,3).
  - If $v \in Y$, then (u, v) is an edge in G and thus u, v forms an alternating path of length 1.
  - If $v \notin Y$, let w be any neighbor of v. Such a neighbor exists since $\deg(v) > 0$, since u, v are feasible holes. Since u is the vertex of the highest degree, $\deg(w) \leq \deg(u)$. Hence there exists a vertex $z \in Y$ which is not a neighbor of w. (If not, then $\deg(w) \geq 1 + |Y| > \deg(u)$, a contradiction.) Then u, z, w, v forms an alternating path of length 3.

- Suppose $u \neq a_1$. Recall that $\hat{r}$, $\hat{c}$ are the reduced degree sequences corresponding to the graph $\hat{G}$ obtained from G by removing all edges adjacent to a_1.
  - If u, v are also feasible holes for $\hat{r}, \hat{c}$, then we may use the inductive hypothesis to conclude that there exists an alternating path from u to v in $\hat{G}$ of length ≤ 5. Note that the correctness of Greedy and the assumption that r, c are feasible imply that $\hat{r}, \hat{c}$ are feasible sequences, and $\hat{\pi}$ is consistent with $\hat{c}$ by definition. Hence we can indeed apply induction. Since G and $\hat{G}$ differ only in edges adjacent to a_1 and the path in $\hat{G}$ does not use a_1 (because $\hat{c}(a_1) = 0$), the path is also an alternating path of length ≤ 5 in G.

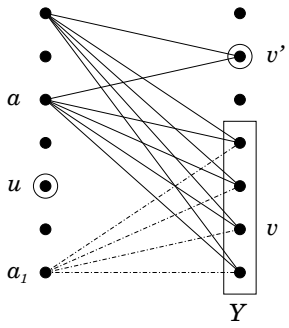

Figure 7: Every $y \in Y$ is adjacent to a

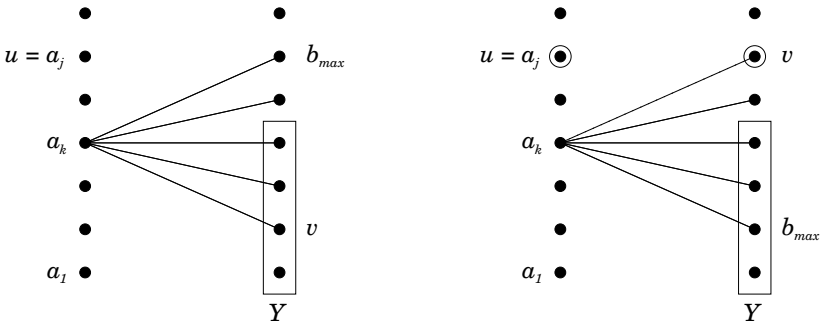

Figure 8: Constructing G and $G^{(u,v)}$ in k-th iteration

– Suppose that u, v are not feasible holes for $\hat{r}, \hat{c}$. We use the following claim:

Claim 8. *If u, v are not feasible holes for $\hat{r}, \hat{c}$, then $v \in Y$ is of degree $c(v) = 1$ and there exists $v' \in V \setminus Y$ also of degree $c(v') = 1$.*

Before we prove the claim, we check what it implies about the existence of a short alternating path between u and v.

By the claim, $v \in Y$ and there exists another $v' \in V \setminus Y$ with $c(v) = c(v') = 1$. If u has an edge to a vertex $b \in V \setminus Y$, then u, b, a_1, v forms an alternating path of length 3 (see Figure 4). Therefore, we may assume that all of u's neighbors lie in Y. Let r_j, c_j be the residual degree sequences just before the greedy algorithm for r, c starts adding edges adjacent to $u = a_j$ (i.e., r_j, c_j are GREEDY's inputs to the recursive call in which u is processed). In other words, r_j, c_j are the parameters of the j-th recursive call originated from GREEDY(r, c, π). Let b be a vertex of the highest remaining degree in $V \setminus Y$ at the start of the j-th recursive call (notice that $V \setminus Y$ is nonempty since $v' \notin Y$). The existence of a short alternating path follows from this claim:

Claim 9. *If u, v are feasible, then $c_j(b) = 1$.*

The proof of the claim is included in Section 3.1. By the claim, for feasible u, v there is a vertex $a \in U$ adjacent to b which is processed after u (see Figure 5). This follows from the fact that all of u's neighbors are in Y. Hence, $\deg(a) \leq \deg(u)$, and therefore, there exists $y \in Y$ which is a neighbor of u but it is not a neighbor of a. (If not, then $\deg(a) \geq 1 + \deg(u)$, a contradiction.) Then u, y, a, b, a_1, v is an alternating path of length 5.

□

3.1 Proofs of Claims 8 and 9 and Lemma 4

To finish the proof of the Theorem 7, it remains to prove the two claims. We re-state both claims, together with their assumptions.

Claim 8. *Recall that $\hat{r}, \hat{c}$ denote the residual sequences after the greedy algorithm matches the first vertex $a_1 \in U$. Assume that u, v are feasible holes for r, c, where $u \neq a_1$. If u, v are not feasible for $\hat{r}, \hat{c}$, then $v \in Y = N(a_1)$ and there exists a vertex $v' \in V \setminus Y$ such that $c(v) = c(v') = 1$.*

Proof of Claim 8. Since u, v are feasible for r, c, there exists a graph with degree sequence $r^{(u)}, c^{(v)}$ (the sequence with holes at u, v). Let $G^{(u,v)}$ be the graph returned by GREEDY$(r^{(u)}, c^{(v)}, \pi^{(u,v)})$ where $\pi^{(u,v)}$ is the total order obtained from π by repositioning v right after all the vertices of degree $c(v)$ (thus v comes before all vertices of degree $c(v)-1$). We will compare $G^{(u,v)}$ with G to establish conditions under which u, v are feasible for $\hat{r}, \hat{c}$.

Notice that if $v \notin Y$ or if $v \in Y$ but $c(v) > c(b)$ for every $b \in V \setminus Y$, then the neighborhoods of a_1 in G and $G^{(u,v)}$ are identical (because the first $r(a_1) = |Y|$ elements of π and $\pi^{(u,v)}$ are the same). Thus, in this case, $\hat{c}^{(v)}$ (the sequence $\hat{c}$ with hole at v) is identical to the sequence $\widehat{c^{(v)}}$, the residual sequence used in the recursive call of GREEDY$(r^{(u)}, c^{(v)}, \pi^{(u,v)})$. Moreover, since $u \neq a_1$, we have $\hat{r}^{(u)} = \widehat{r^{(u)}}$. By the correctness of the greedy algorithm, $\widehat{r^{(u)}}, \widehat{c^{(v)}}$ are feasible. Therefore, if a_1 has the same neighbors in G and $G^{(u,v)}$, we can conclude that u, v are feasible holes for $\hat{r}, \hat{c}$.

We are left with the case when $v \in Y$ and there exists $v' \in V \setminus Y$ of the same degree $c(v') = c(v)$. We will show that if $c(v) > 1$, the holes u, v are feasible for $\hat{r}, \hat{c}$. This implies the claim.

Suppose $c(v) = c(v') > 1$ for $v \in Y$ and $v' \in V \setminus Y$. Since u, v are feasible for r, c, by symmetry u, v' are also feasible for r, c. However, $v' \notin Y$ and thus, as before, we can conclude that $\hat{r}^{(u)}, \hat{c}^{(v')}$ are feasible and there exists a corresponding graph H. Notice that $\hat{c}^{(v')} = c(v') - 1$ (because $v' \notin Y$ is a hole) and that $\hat{c}^{(v')} = c(v) - 1$ (because $v \in Y$). Since $c(v') = c(v) > 1$, vertices v and v' have the same non-zero degree in H. We will modify H to obtain H', a graph corresponding to $\hat{r}^{(u)}, \hat{c}^{(v)}$. This will prove the feasibility of u, v for $\hat{r}, \hat{c}$. To get H', we need to decrease the degree of v and increase the degree of v' by one while keeping the other degrees intact.

If there is a vertex $a \in U$ which is adjacent to v but not v' in H, we may simply set $H' = H \cup (a, v') \setminus (a, v)$. If there is no such a, then the neighborhood sets of v and v' in H are identical (see Figure 6). If there is a vertex $y \in Y$ for which there exists a neighbor a of v' (and v) in H which is not adjacent to y, then we construct H' as follows. Since $y \in Y$, and $v' \notin Y$ is of the same degree in G as $v \in Y$, by the definition of Y we have $c(y) \geq c(v) = c(v')$. Therefore the degree of y in H is not smaller than the degree of v' in H, i.e. $\hat{c}^{(v')}(y) \geq \hat{c}^{(v')}(v')$. Thus there must exist y's neighbor a' in H which does not neighbor v'. It suffices to set $H' = H \cup \{(a, y), (a', v')\} \setminus \{(a, v), (a', y)\}$ (see Figure 6).

The last case happens when v and v' share the same set of neighbors in H and every $y \in Y$ is adjacent to every neighbor of v' (see Figure 7). By contradiction we will show that this case never happens. Notice that since $r(a) \leq r(a_1)$ for every $a \in U$ and the degree of a in H remains $r(a)$ except for $u \neq a_1$ which decreases by one, the degree of every $a \in U$ in H is upper bounded by $r(a_1)$, i.e. for all $a \in U$, $\hat{r}^{(u)}(a) \leq |Y|$. Let a be any neighbor of v' (by the assumption $c(v') > 1$, the neighborhood set of v' is non-empty). Then a is adjacent to every vertex in $Y \cup \{v'\}$ and therefore $\hat{r}^{(u)}(a) > |Y|$, a contradiction. □

Claim 9. *Let $u \neq a_1$ and $v \in Y$ be such that $c(v) = c(v') = 1$ for some $v' \in V \setminus Y$. Suppose* GREEDY*(r, c, π) processes vertices from u in order $a_1, \ldots, a_n$. Let r_i, c_i, π_i be the parameters to the i-th recursive call of* GREEDY*, i.e. the call when a_i is processed. Let $u = a_j$. If $c_j(b) = 0$ for all $b \in V \setminus Y$, then u, v are not feasible for r, c.*

Proof of Claim 9. Since $c(v) = 1$, for every $b \in V \setminus Y$ we have $c(b) \leq 1$. Let $b_{\max} \in V \setminus Y$ be the vertex of degree 1 ordered last in π (there is at least one such vertex, since $c(v') = 1$). We create π' by swapping the positions of v and $b_{\max}$ in π. Notice that this ordering is consistent with $c^{(v)}$, the sequence obtained from c by decreasing the degree of v by one. We will compare the execution of GREEDY(r, c, π) and GREEDY$(r^{(u)}, c^{(v)}, \pi')$ (see Figure 8). The idea is that both executions will

behave similarly, with the roles of v and $b_{\max}$ reversed. Once $\textsc{Greedy}(r, c, \pi)$ gets to matching $b_{\max}$ to a vertex a_k from U, $\textsc{Greedy}(r^{(u)}, c^{(v)}, \pi')$ will attempt to match a_k to a vertex in V of residual degree zero and it will fail. Thus, by the correctness of $\textsc{Greedy}$, u, v cannot be feasible holes for r, c. We describe the idea in detail below.

Notice that $\textsc{Greedy}(r, c, \pi)$ and $\textsc{Greedy}(r^{(u)}, c^{(v)}, \pi')$ process all vertices a_i for $i < j$ in the same order (assume that if the second execution has multiple choices for x, it chooses the same x as the first execution, if possible). This follows from the fact that the only vertex whose degree is changed is vertex a_j and its degree only decreased. Moreover, the order of processing vertices of U is independent of c (or $c^{(v)}$), assuming the executions do not fail.

Let r_i, c_i, π_i and $r_i^{(u)}, c_i^{(v)}, \pi_i'$ be the parameters of the i-th recursive call originated from $\textsc{Greedy}(r, c, \pi)$ and $\textsc{Greedy}(r^{(u)}, c^{(v)}, \pi')$, respectively. Let $a_k \in U$ be the vertex matched to $b_{\max}$ by $\textsc{Greedy}(r, c, \pi)$. By the assumption of the claim, $c_j(b_{\max}) = 0$, and hence $k < j$, i.e. a_k is processed before $u = a_j$. By induction on i one can verify that for $i < k$ the following hold:

1. $r_i^{(u)}(a) = r_i(a)$ for $a \in U \setminus \{u\}$ and $r_i^{(u)}(u) = r_i(u) - 1$.
2. $c_i^{(v)}(b) = c_i(b)$ for $b \in V \setminus \{v, b_{\max}\}$, $c_i^{(v)}(b_{\max}) = c_i(v)$, $c_i(b_{\max}) = 1$ and $c_i^{(v)}(v) = 0$.
3. Let $supp(f) = \{x \mid f(x) \neq 0\}$.
   - If $v \in supp(c_i)$, then $supp(c_i) = supp(c_i^{(v)}) \cup \{v\}$. The total order π_i restricted to $supp(c_i)$ and the total order π_i' restricted to $supp(c_i^{(v)}) \cup \{v\}$ are identical except that the positions of $v, b_{\max}$ are reversed.
   - If $v \notin supp(c_i)$, then $supp(c_i) \setminus \{b_{\max}\} = supp(c_i^{(v)})$, and the orderings π_i restricted to $supp(c_i)$ and π_i' restricted to $supp(c_i^{(v)}) \cup \{v\}$ are identical except that b_{max} appears in π_i where v appears in π_i'.
4. For every $b \succ_{\pi_i} b_{\max}$, $c_i(b) = 0$. In words, $b_{\max}$ is the last vertex of degree 1 in π_i, if it is indeed of degree 1.

For the induction, the base case $i = 1$ is clear in each case. The case of $i = 2$ also follows in each case, and it can be checked that it holds for the second claim in 3. Now assume the claims are true up to some $2 \leq i \leq k - 2$. We show that they hold for $(i + 1)$.

1. Since only the degree of $a_i \neq u$ decreases in both sequences by $r_i(a_i) = r_i^{(u)}(a_i)$.
2. Since $b_{\max}$ is matched only in the k-th recursive call, $c_{i+1}(b_{\max}) = c_i(b_{\max}) = 1$. Also, $c_{i+1}^{(v)}(v) = c_i^{(v)}(v) = 0$. For $i = 1$, when v is matched by the outermost recursive call of $\textsc{Greedy}(r, c, \pi)$, $b_{\max}$ is matched by $\textsc{Greedy}(r^{(u)}, c^{(v)}, \pi')$, hence $c_{i+1}^{(v)}(b_{\max}) = c_{i+1}(v)$. It follows that $c_{i+1}^{(v)}(b) = c_{i+1}(b)$ for $b \in V \setminus \{v, b_{\max}\}$ since if the statement is true for i and the orderings are identical on vertices of non-zero residual degree other than $v, b_{\max}$ by 3., then exactly the same set of vertices are used by both in the i-th recursive call.
3. This part is true by the definitions $\pi_{i+1} = \hat{\pi}_i$ and $\pi_{i+1}' = \hat{\pi}_i'$ and the fact that 1, 2, and 3 hold for i.
4. This is clear by the definition of the orderings π_{i+1} and π_{i+1}' and the fact that $c_i(b_{\max}) = 1$.

Therefore a_k is joined to the first $r(a_k)$ elements in π_k by $\textsc{Greedy}(r, c, \pi)$, and by 4. the last of them is $b_{\max}$. However, by 3., the execution of $\textsc{Greedy}(r^{(u)}, c^{(v)}, \pi')$ will attempt to connect a_k to v (or some vertex with remaining degree 0), which is impossible since $c_k^{(v)}(v) = 0$. Thus, $\textsc{Greedy}(r^{(u)}, c^{(v)}, \pi')$ fails to construct a corresponding graph, and hence u, v are not feasible holes for r, c. □

This finishes the proof of the Theorem 7. As mentioned earlier, Lemma 4 is a corollary of the Theorem.

Proof of Lemma 4. We will prove that for the greedy graph G^* for any $\varepsilon > 0$ and any $\lambda \leq \frac{\varepsilon}{(nm)^D}$ we can efficiently estimate $w^*(u, v) = \lambda(\mathcal{P})/\lambda(\mathcal{N}(u, v))$ (within a $1 \pm \varepsilon$ factor) for every feasible u, v. We have already observed that $\lambda(\mathcal{P})$ and $\lambda(\mathcal{N}(u, v))$ are polynomials in λ. We will show how to approximate $\lambda(\mathcal{P})$ and $\lambda(\mathcal{N}(u, v))$. First we observe, that each of $\lambda(\mathcal{P})$ and $\lambda(\mathcal{N}(u, v))$ has a positive small-degree coefficient. In particular, the absolute coefficient of $\lambda(\mathcal{P})$ is 1, since G_0 is the only graph corresponding to r, c sharing exactly D edges with G^*. Moreover, by Lemma 7, there exists a graph $G' \in \mathcal{N}(u, v)$ which can be obtained from G^* by swapping the edges of an alternating path of length ≤ 5. Therefore G' shares at least $D - 3$ edges with G^* and thus the coefficient of x^d for some $d \leq 2$ in $\lambda(\mathcal{N}(u, v))$ is positive. Moreover,

$$|\mathcal{P}| \leq \binom{nm}{D} \leq (nm)^D,$$

where the first inequality follows from the fact that $\binom{nm}{D}$ counts the number of bipartite graphs (with partitions of sizes n, m) with exactly D edges. Thus,

$$1 \leq \lambda(\mathcal{P}) = 1 + \sum_{k=0}^{D-1} p_k \lambda^{D-k} \leq 1 + \lambda \sum_{k=0}^{D-1} p_k \leq 1 + \lambda(nm)^D \leq 1 + \varepsilon$$

To approximate $\lambda(\mathcal{N}(u, v))$, we will enumerate all graphs corresponding to $r^{(u)}, c^{(v)}$ which share at least $D - 3$ edges with G^*. This can be done by going through all possible alternating paths from u to v of length ≤ 5 and through all alternating cycles of length 4 (corresponding to the case when the symmetric difference of G^* and the graph with the degree sequence $r^{(u)}, c^{(v)}$ consists of the edge (u, v) and a 4-cycle). This way, for fixed u, v, in time $O(nmd_{\max}^2)$ we can compute

$$x_{u,v} := p_{D-1}^{u,v} + p_{D-2}^{u,v}\lambda + p_{D-3}^{u,v}\lambda^2.$$

Then, $x_{u,v}$ is a $(1 + \varepsilon)$-approximation of $\lambda(\mathcal{N}(u, v))$:

$$x_{u,v} \leq \lambda(\mathcal{N}(u, v)) = x_{u,v} + \sum_{k=0}^{D-4} p_k^{u,v} \lambda^{D-1-k} \leq x_{u,v} + \lambda^3 \sum_{k=0}^{D-4} p_k^{u,v} \leq x_{u,v} + \varepsilon\lambda^2 \leq (1 + \varepsilon)x_{u,v},$$

where the last inequality follows from $x_{u,v} \geq \lambda^2$ since there exists $j \in [3]$ for which $p_{D-j}^{u,v} \geq 1$.

Therefore in time $O((nmd_{\max})^2)$ we can compute $x_{u,v}$ for every u, v and $1/x_{u,v}$ is a $(1 + \varepsilon)$-approximation of $w_\lambda^*(u, v)$. □

4 The Markov Chain

Our Markov chain is analogous to the chain used in algorithms for the permanent [11, 12]. Recall that $\mathcal{P}$ denotes the set of graphs with the required degree sequence, and $\mathcal{N}(u,v)$ denotes the set of graphs with deficiencies at u, v and $\mathcal{N} = \cup_{u,v}\mathcal{N}(u,v)$. The state space of the chain is $\Omega = \mathcal{P} \cup \mathcal{N}$.

The Markov chain is characterized by an activity $\lambda > 0$ and a *weight function* $w : U \times V \to \mathbf{R}^+$. The weight of a graph $G \in \Omega$ is defined as

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

The transitions $G_t = (U, V, E_t) \to G_{t+1} = (U, V, E_{t+1})$ of the Markov chain MC are:

1. If $G_t \in \mathcal{P}$, choose an edge e uniformly at random from E_t. Set $G' = G_t \setminus e$.
2. If $G_t \in \mathcal{N}(u,v)$, choose an edge $e = (x,y)$ uniformly at random from the multi-set[1] $E_t \cup \{(u,v)\}$ and choose W uniformly from U, V.
   (a) If $e = (u,v)$ and $(u,v) \notin E_t$, let $G' = G_t \cup (u,v)$.
   (b) If $W = U$ and $(u,y) \notin E_t$, let $G' = G_t \setminus (x,y) \cup (u,y)$.
   (c) If $W = V$ and $(x,v) \notin E_t$, let $G' = G_t \setminus (x,y) \cup (x,v)$.
   (d) Otherwise, let $G' = G_t$.
3. With probability $\min\{1, w(G')/w(G_t)\}$, set $G_{t+1} = G'$; otherwise, set $G_{t+1} = G_t$.

It is straightforward to verify that the stationary distribution π of the chain is proportional to the weights w, i.e., for $G \in \Omega$, $\pi(G) = w(G)/Z$ where $Z = \sum_G w(G)$. The main result of this section is to show that if the weights $w(u,v)$ are within a constant factor of their ideal values $w^*(u,v)$, MC mixes in polynomial time.

We continue with some standard definitions before formally stating the main result on the convergence time of the Markov chain. The *total variation distance* between two distributions μ, ν on Ω is given by

$$d_{tv}(\mu,\nu) = \frac{1}{2}\sum_{x\in\Omega} |\mu(x) - \nu(x)|$$

Let P denote the transition matrix of the chain MC, and thus $P^t(x,\cdot)$ denotes the distribution after t steps of the chain, with starting state x. The *mixing time* $\tau_x(\delta)$ of MC starting at state $x \in \Omega$ is defined as

$$\tau_x(\delta) = \min\{t \geq 0 \mid d_{tv}(P^t(x,\cdot), \pi) \leq \delta\}$$

We can now state our main result on the mixing time of MC.

Theorem 10. *Assuming the weight function w satisfies inequality*

$$w^*(u,v)/2 \leq w(u,v) \leq 2w^*(u,v) \tag{1}$$

for every feasible hole pattern $u \in U, v \in V$, then the mixing time of the Markov chain MC started at G is $\tau_G(\delta) = O(nmD^2 d_{max}(\ln(1/\pi(G)) + \log\delta^{-1}))$, where $d_{max} = \max\{\max_i r(i), \max_j c(j)\}$.

[1] It may be that (u,v) is already in the set of edges E_t.

4.1 Analyzing the mixing time

We bound the mixing time of MC using the multicommodity flow method, which is an extension of the canonical path technique [21]. Let $\mathcal{K}$ denote the Markov kernel (i.e., the transition graph) underlying MC so that $T = (M, M')$ is an edge of $\mathcal{K}$ if $P(M, M') > 0$. Let $\mathcal{P}_{IF}$ be the set of all directed paths from I to F in $\mathcal{K}$. A *flow* in the Markov kernel is a function $g : \bigcup_{I,F \in \Omega, I \neq F} \mathcal{P}_{IF} \to \mathbb{R}_0^+$ such that $\sum_{p \in \mathcal{P}_{IF}} g(p) = \pi(I)\pi(F)$. The *congestion* of the flow g is defined as:

$$\rho(g) = \ell(g) \max_{T=(M,M')} \left\{ \frac{1}{\pi(M)P(M,M')} \sum_{p \ni T} g(p) \right\}$$

where $\ell(g)$ is the length of the longest path p such that $g(p) > 0$, Note, the summation is over all $p \in \cup_{I,F} \mathcal{P}_{IF}$, and T is restricted to be an edge of the Markov kernel so that $P(M, M') > 0$.

This implies the following bound on the mixing time, which was proved by Sinclair [21],

$$\tau_x(\delta) \leq \rho(g)(\log \pi(x)^{-1} + \log \delta^{-1})$$

To define the flow g, for each $I, F \in \Omega \times \Omega$ we must specify how to route the flow along directed paths going from I to F. As in [12] it is convenient to first define a flow f between all pairs $I \in \Omega$ and $F \in \mathcal{P}$. The flow f can be extended to a flow g between all pairs by routing the flow between a pair of near-perfect tables I, F through a random perfect table. Extending the flow from f to g causes only a modest increase in the congestion. If $\widehat{\rho}(f)$ denotes the congestion restricted to pairs $(I, F) \in \Omega \times \mathcal{P}$, then

$$\rho(g) \leq 2\frac{\pi(\mathcal{N})}{\pi(\mathcal{P})}\widehat{\rho}(f) \leq 8nm\widehat{\rho}(f) \tag{2}$$

The first inequality follows by a result of Schweinsberg, Corollary 3 in [20]. The second inequality follows assuming by (1) that the weights $w(u, v)$ approximate the ideal weights $w^*(u, v)$ up to a factor of 2. Hence it will suffice to define the flow f, which we do in the next section, and bound $\widehat{\rho}(f)$ to bound the mixing time.

4.2 Defining the Canonical Flow

We use the flows defined by Cooper, Dyer and Greenhill [5]. Therefore, we follow their notation. Let I, F be perfect or near-perfect contingency tables. We wish to define a set of canonical paths between them by decomposing $H = I \oplus F$ into a sequence of edge-disjoint alternating circuits. Different circuit decompositions will correspond to distinct paths. A circuit of H is a sequence of vertices $w_0, \cdots, w_\ell$, such that $(w_i, w_{i+1}), (w_\ell, w_0) \in E_H$, and each of these edges is distinct, though the vertices may be repeated. The set $E_H = (E_F \setminus E_I) \cup (E_I \setminus E_F)$. Let the edges in $E_F \setminus E_I$ be red and the edges in $E_I \setminus E_F$ be blue. At each vertex, we will choose a *pairing* of the red and blue edges. A pairing of $I \oplus F$ consists of a pairing at every vertex. Let $\Psi(I, F)$ be the set of all such pairings of $I \oplus F$. For each pairing in $\Psi(I, F)$, we will construct a canonical path from I to F, carrying a total flow of $\pi(I)\pi(F)/|\Psi(I, F)|$. We define the pairings and the corresponding circuit decompositions below.

$\underline{I, F \in \mathcal{P}}$: In this case, at each vertex of H, the red degree is equal to the blue degree. A pairing is constructed by pairing up the red edges at a vertex with the blue edges at each vertex. Hence, if

the red (and blue) degree in H of a vertex v is γ_v, $|\Psi(I,F)| = \Pi_v \gamma_v!$. Fix a pairing $\psi \in \Psi(I,F)$. We define an edge disjoint circuit decomposition of H, $\mathcal{C}^\psi = (C_1^\psi, \cdots, C_s^\psi)$, and then define how to "unwind" each circuit to go from I to F. To simplify notation, we omit the superscript henceforth. Let the lexicographically smallest edge in E_H be (w_0, w_1). Choose the (w_i, w_{i+1}) to be the next edge of the circuit if (w_i, w_{i+1}) is paired with (w_{i-1}, w_i) at w_i by ψ (so that we choose (w_1, w_2) if it is paired with (w_0, w_1) by ψ at w_1). This procedure terminates with the circuit $C_1 = w_0, \cdots, w_{k-1}, w_k$ when the edge (w_k, w_0) is paired with (w_0, w_1) at w_0. If $E_H = C_1$, set $\mathcal{C} = (C_1)$. Otherwise, generate C_2 by starting with the lexicographically smallest edge not in C_1. Continue until $E_H = C_1 \cup \cdots \cup C_s$. Then set $\mathcal{C} = (C_1, \cdots, C_s)$. Note that the circuits $C_1, \cdots, C_s$ are edge disjoint by construction and the edges of the circuits are alternately blue and red.

The canonical path p^ψ corresponding to the pairing ψ is defined by the concatenation of the sequence of moves which unwind $C_1, \cdots, C_s$. Let $C_r = a_0, b_0, \ldots, a_\ell, b_\ell$ be a circuit whose lexicographically smallest blue edge is (a_0, b_0). First remove the edge (a_0, b_ℓ). Then for $i = 0, \ldots, \ell - 1$, slide the edge (a_{i+1}, b_i) into (a_i, b_i). Finally, add (a_ℓ, b_ℓ). Since the set of circuits corresponding to different pairings are distinct, the corresponding canonical paths are distinct as well. Set $f(p^\psi) = \pi(I)\pi(F)/|\Psi(I,F)|$ for each path p^ψ.

<u>$I \in \mathcal{N}$ and $F \in \mathcal{P}$:</u> Suppose $I \in \mathcal{N}(u,v)$. Then, in the graph $H = I \oplus F$, every vertex except u, v is incident with an equal number of red and blue edges. The vertices u, v are each adjacent to one more red edge than the number of blue edges. Let the number of red edges adjacent to the vertex v in H be γ_v. Define the pairing ψ as follows. At each vertex other than u, v choose a pairing of red and blue edges. At each of u, v choose one red edge which remains unpaired, and pair up the remaining red and blue edges. If $\Psi(I,F)$ is the set of such pairings then, $|\Psi(I,F)| = \prod_{v \in V} \gamma_v!$. For each pairing $\psi \in \Psi(I,F)$ we decompose H into a set of circuits $\mathcal{C}$ and a walk W as follows. Let (w_0, w_1) be the red edge adjacent to $u = w_0$ which is unmatched by ψ. Choose the edge (w_i, w_{i+1}) to be the next edge of the walk if (w_i, w_{i+1}) is paired with (w_{i-1}, w_i) at w_i by ψ. The procedure terminates with the walk W, given by $u = w_0, \cdots, w_\ell = v$ when the red edge $(w_{\ell-1}, w_\ell)$ which is unpaired by ψ at v is paired with $(w_{\ell-2}, w_{\ell-1})$ at $w_{\ell-1}$. If $E_H = W$, we are done, otherwise, start with the lexicographically smallest unused edge of E_H and define the circuits $C_1, \cdots, C_s$. To define the canonical path corresponding to ψ, we unwind the walk W and then the circuits $\mathcal{C}$ in their canonical order. To augment the walk $a_0, b_0, \ldots, a_\ell, b_\ell$, we slide the edges (a_{i+1}, b_i) to (a_i, b_i) for $i = 0, \ldots, \ell - 1$. Then we add the edge (a_ℓ, b_ℓ). Set $f(p^\psi) = \pi(I)\pi(F)/|\Psi(I,F)|$ for each path p^ψ.

This completes the definition of the flow f between pairs I, F in $\Omega \times \mathcal{P}$.

4.3 Analyzing the Flow

To prove Theorem 10, we analyze the mixing time of the Markov chain which uses the ideal weights $w^*(u,v)$. We will see that the Theorem then follows immediately from the condition (1). By the construction of the flow, $\ell(f) \leq D$ and $\ell(g) \leq 2D$. Hence

$$\widehat{\rho}(f) \leq 2D \max_{T=(M,M')} \left\{ \frac{1}{\pi(M)P(M,M')} \sum_{p \ni T} f(p) \right\}$$

where the sum is over paths $p \in \Psi(I,F)$ for I, F in $\Omega \times \mathcal{P}$. Moreover, by the definition of the Markov chain MC, for any transition $T = (M, M')$ which has non-zero probability, $\pi(M)P(M,M') \geq$

$\frac{1}{2D}\min\{\pi(M), \pi(M')\}$. Hence,

$$\widehat{\rho}(f) \leq 4D^2 \max_{T=(M,M')} \left\{ \frac{1}{\min\{\pi(M), \pi(M')\}} \sum_{p \ni T} f(p) \right\} \tag{3}$$

Thus to bound $\widehat{\rho}(g)$, it is enough if we bound

$$\max_{T=(M,M')} \left\{ \frac{1}{\pi(M)} \sum_{p \ni T} f(p) \right\}$$

for every transition (M, M'), since then the bound holds for the reverse transition (M', M) as well.

Let $T = (M, M')$ be any transition of the Markov chain so that $P(M, M') > 0$. Let

$$f_T = \{(I, F) \in \Omega \times \mathcal{P} : \exists\ \psi \in \Psi(I, F) \text{ s.t. } p^\psi \ni T\}.$$

We will show that for every transition $T = (M, M')$ of the Markov kernel,

$$\sum_{(I,F) \in f_T} \frac{\pi(I)\pi(F)}{\pi(M)} \frac{|\Psi_T(I, F)|}{|\Psi(I, F)|} = O(d_{max}) \tag{4}$$

By equations (2),(3), and (4), this implies $\rho(g) = O(mnD^2 d_{max})$. This then implies the bound on the mixing time. We divide the proof of (4) into two cases according to the type of transition, in the following two subsections.

We will use the following notation. For $y, u \in U$ and $x, v \in V$ distinct, let

$$\widehat{\mathcal{N}}(y, x, (y, v), (x, u)) = \{M \in \mathcal{N}(y, x) : (y, v), (x, u) \notin M\}$$

Also, let

$$\widehat{\mathcal{P}}(u, v) = \{P \in \mathcal{P} : (u, v) \notin \mathcal{P}\}$$

We also require notation for tables with up to 4 deficiencies. For $y, u \in U$ and $v, x \in V$ (not necessarily distinct), let $\mathcal{N}(y, x, u, v)$ denote the set of tables with deficiencies at u, v, x, y. If any of the vertices y, u, v, x are the same, this means the degree at that vertex is *two* less than its required degree.

Recall, for a transition T, f_T denotes the set of $(I, F) \in \Omega \times \mathcal{P}$ which use T for some of its flow. Let

$$f_T^{u,v} = \{(I, F) \in f_T : I \in \mathcal{N}(u, v)\}$$

4.3.1 Transitions of Type 2b or 2c.

Lemma 11. *For a transition T of type 2b or 2c,*

$$\sum_{(I,F) \in f_T} \frac{\pi(I)\pi(F)}{\pi(M)} \frac{|\Psi_T(I, F)|}{|\Psi(I, F)|} = O(d_{max})$$

To prove the lemma, we use results analogous to the combinatorial lemmas proved in [3], tailored to the canonical flows in this case. We first state and prove the combinatorial results and then show how the lemma follows.

Lemma 12. *Let T be a transition between near-perfect tables, so that $M \in \mathcal{N}(u,v)$, $M' \in \mathcal{N}(u',v)$ where $u,u' \in U$, $v \in V$ and $M' = M \setminus (u',x) \cup (u,x)$ for some $x \in V$.*

i)

$$\sum_{\substack{(I,F)\in f_T \\ I\in\mathcal{P}}} \frac{|\Psi_T(I,F)|}{|\Psi(I,F)|}\lambda(I)\lambda(F) \leq \sum_{\substack{y\in U \\ (y,v)\neq(u,x)}} \lambda(u,x)\lambda(y,v)\lambda(\widehat{\mathcal{N}}(y,x,(y,v),(x,u)))\lambda(M)$$

ii) For all $s \in U$

$$\sum_{(I,F)\in f_T^{s,v}} \frac{|\Psi_T(I,F)|}{|\Psi(I,F)|}\lambda(I)\lambda(F) \leq \lambda(u,x)\lambda(\widehat{\mathcal{N}}(s,x,(x,u)))\lambda(M)$$

iii) For all $s \in U$ and $z \in V$,

$$\sum_{(I,F)\in f_T^{s,z}} \frac{|\Psi_T(I,F)|}{|\Psi(I,F)|}\lambda(I)\lambda(F) \leq \sum_{\substack{y\in U \\ (y,v)\neq(u,x)}} \lambda(u,x)\lambda(y,v)\lambda(\widehat{\mathcal{N}}(s,z,y,x,(y,v),(x,u)))\lambda(M)$$

Proof. *i*) Let $I \in \mathcal{P}$ (blue), and $F \in \mathcal{P}$ (red).

Fix a pairing $\psi \in \Psi_T(I,F)$ (at x, the edge (u,x) is always paired with (u',x)), which gives a decomposition of $I \oplus F$ into red-blue alternating circuits. Since the pairing corresponds to a path from I to F through the transition T, the vertices u,v,x lie on some circuit C. Let y be the vertex adjacent to v in C so that the edge (v,y) is blue (for the first sliding transition in the unwinding of C, y is the vertex u so that the order of vertices on the circuit is v,y,x,u'). Clearly, since (u,x) is an edge of the transition, and (y,v) is the first removed edge, $(y,v) \neq (u,x)$. In M, the circuits ordered before C agree with F, while the circuits after C agree with I.

Define the graph $E_\psi(I,F) = I \oplus F \oplus (M \cup M') \setminus \{(v,y)\}$. Then $E_\psi(I,F) \in \widehat{\mathcal{N}}(y,x,(y,v),(x,u))$. Given M and (u,x), we can recover $M \cup M' = M \cup (u,x)$, and from this, given $E_\psi(I,F)$ and (y,v), we can recover $I \oplus F = (E_\psi(I,F) \cup (y,v)) \oplus (M \cup M')$. We have that $I \cup F = M \cup E_\psi(I,F) \cup \{(u,x),(y,v)\}$, and hence

$$\frac{1}{|\Psi(I,F)|}\lambda(I)\lambda(F) = \frac{1}{|\Psi(I,F)|}\lambda(M)\lambda(E_\psi(I,F))\lambda(u,x)\lambda(y,v) \tag{5}$$

Let $\Psi'(E_\psi(I,F))$ be defined as the set of triples (I',F',ψ') with $\psi' \in \Psi(I',F')$ such that $E'_\psi(I',F') = E_\psi(I,F)$. We claim that $|\Psi'(E_\psi(I,F))| \leq |\Psi(I,F)|$. Assuming this, we claim the lemma follows. If we add up (5) for each $(I,F) \in f_T$ such that $I \in \mathcal{P}$, and each $\psi \in \Psi_T(I,F)$, then on the left hand side, each term $\lambda(I)\lambda(F)$ is counted $|\Psi_T(I,F)|$ times. On the right hand side of (5), for every graph $E \in \widehat{\mathcal{N}}(y,x,(y,v),(x,u))$ such that $E = E_\psi(I,F)$ for some I,F,ψ, the term $\lambda(E)\lambda(M)\lambda(u,x)\lambda(v,y)$ is counted $|\Psi'(E)|$ times. Formally,

$$\sum_{\substack{(I,F)\in f_T \\ I\in\mathcal{P}}} \frac{|\Psi_T(I,F)|}{|\Psi(I,F)|}\lambda(I)\lambda(F) \quad = \quad \sum_{\substack{E\in\widehat{\mathcal{N}}(y,x,(y,v),(x,u)) \\ E=E_{\psi'}(I',F')}} \frac{|\Psi'(E_{\psi'(I',F')})|}{|\Psi(I',F')|}\lambda(u,x)\lambda(y,v)\lambda(E)\lambda(M)$$

$$\leq \sum_{\substack{E\in\widehat{\mathcal{N}}(y,x,(y,v),(x,u))\\ E=E_{\psi'}(I',F')}} \lambda(u,x)\lambda(y,v)\lambda(E)\lambda(M)$$

$$\leq \sum_{\substack{y\in U\\ (y,v)\neq(u,x)}} \lambda(u,x)\lambda(y,v)\lambda(\widehat{\mathcal{N}}(y,x,(y,v),(x,u)))\lambda(M)$$

Suppose that from E_ψ and T we recover $H = I \oplus F$. Then H has even degree at every vertex. Color an edge of H *green* if it is in M and *yellow* if it is in E_ψ. To bound the number of triples $|\Psi'(E_\psi)|$, we use the fact that the pairing ψ of red and blue edges is a pairing of yellow and green edges at most vertices. A pairing of the yellow and green edges defines a decomposition of $I \oplus F = I' \oplus F'$ into alternating circuits, and further, using the transition T we can recover I' and F'. Thus the number of triples $|\Psi'(E_\psi)|$ is bounded by the number of yellow-green pairings.

In H, every vertex except possibly u, v, x, y has equal yellow and green degree. Two edges of H remain uncolored, (u, x) and (y, v).

a) Suppose $u \neq y, v \neq x$. The vertices y, x have one extra green degree and u, v have one extra yellow degree. To define the pairings at each vertex of H, define the pairings as usual for all vertices except x, y, u, v. At u, think of (u, x) as a green edge, while at x, think of it as a yellow edge. At y, think of (y, v) as a yellow edge, while at v think of it as a green edge. This ensures that there is a yellow-green pairing corresponding to the original red-blue pairing, because we know that in the red-blue pairing at v, the edge (v, y) was paired with a red edge (from F), which is now colored yellow (from E_ψ). Similar arguments can be made at the vertices y, u, x. In addition, at x, we know from T that the edge (u, x) should be paired with (u', x). The number of yellow green pairings is at most $|\Psi(I, F)|$, since if we take into account the "bicolored" edges (v, y) and (u, x), the number of yellow green pairings at each vertex is at most the number of red-blue pairings originally.

b) Suppose that $u \neq y, v = x$. Then in H at every vertex except u, y, the green degree is equal to its yellow degree. Meanwhile, y has an extra green degree and u an extra yellow degree. The pairings at each vertex are constructed as in the previous case, with the same rules for the edges (u, x) and (v, y). Again, it can be seen that the number of yellow-green pairings is the same as the number of red-blue pairings.

c) Suppose $u = y$. Note that this implies $v \neq x$. Every vertex in H except v, x has equal yellow and green degree, but again, coloring the edges (v, y) and (u, x) as before to define the pairings at v, y, x can be used to show that the number of yellow green pairings that we can construct are at most $|\Psi(I, F)|$.

Note that once the bicolored edges are taken into account, in each of the above cases, every vertex of H has green degree equal to yellow degree.

ii) Let $I \in \mathcal{N}(s, v)$ (blue) and $F \in \mathcal{P}$ (red).

Fix a pairing $\psi \in \Psi_T(I, F)$ and define $E_\psi(I, F) = I \oplus F \oplus (M \cup M')$. Then $E_\psi \in \widehat{\mathcal{N}}(s, x, (x, u))$. We claim $|\Psi'(E_\psi(I, F))| \leq |\Psi(I, F)|$. Suppose that from $E_\psi(I, F)$ and T we recover $H = I \oplus F$. Then H has even degree at every vertex except s, v. Color an edge of H green if it is in M and yellow if it is in $E_\psi(I, F)$. The edge (u, x) of H remains uncolored.

We can show the bound on $|\Psi'(E_\psi(I,F))|$ by exactly the same steps as i), by substituting y in that case, with s here. The only difference is that we no longer take into consideration the edge (s,v) for constructing the pairings, as it is not in H. Notice that here, the fact that s and v have extra red degree will be compensated for by considering (u,x) to be bicolored for the purposes of constructing the yellow-green pairing. This results in s having a green edge and v having a yellow edge remaining effectively unpaired in the yellow-green pairing of H. Note that the red edges which were adjacent to s,v in the circuit being unwound will appear yellow adjacent to v and green adjacent to s in the yellow-green coloring of H. Thus the pairing red-blue ψ does indeed correspond to a yellow-green pairing in H.

iii) Let $I \in \mathcal{N}(s,z)$ (blue) and $F \in \mathcal{P}$ (red).

Fix a pairing $\psi \in \Psi_T(I,F)$. Then, ψ decomposes $I \oplus F$ into a sequence of red-blue alternating circuits and an alternating walk from s to z whose initial and final edges are red. Since T is a transition along the path corresponding to ψ from I to F, u,v,x lie on some circuit C. Let y be the vertex adjacent to v in C so that the edge (v,y) is blue.

Define the graph $E_\psi(I,F) = I \oplus F \oplus (M \cup M') \backslash \{(v,y)\}$. Then $E_\psi(I,F) \in \widehat{\mathcal{N}}(s,z,y,x,(y,v),(x,u))$. Suppose that from $E_\psi(I,F)$ and T we recover $H = I \oplus F$. Then H has even degree at every vertex except s,z. Color an edge of H green if it is in M and yellow if it is in $E_\psi(I,F)$.

First assume that the vertices u,v,s,z,x,y are distinct. Then, every vertex except u,v,s,z,x,y has equal yellow and green degree in H. Both s and z have one green degree more than their yellow degree. This is because they are the endpoints of a walk which has already been unwound, and hence the red edges adjacent to s,z which are left unpaired by ψ both appear in M and are green. Two edges of H remain uncolored, (u,x) and (y,v). The vertices y,x have one extra green degree and u,v have on extra yellow degree. To define the pairings at each vertex of H, define the pairings as usual for all vertices except x,y,u,v. At u, think of (u,x) as a green edge, while at x, think of it as a yellow edge. At y, think of (y,v) as a yellow edge, while at v think of it as a green edge. The number of yellow green pairings that we can construct are at most $|\Psi(I,F)|$.

Now, in case the vertices are not distinct, there are in all 21 possibilities, taking into account the bipartition the vertices are in and the fact that $(u,x) \neq (v,y)$. However, in each case, suppose that at u, we think of (u,x) as a green edge, while at x, we think of it as a yellow edge and at y, we think of (y,v) as a yellow edge, while at v think of it as a green edge. Then, except at s,z, the yellow degree at every vertex is equal to the green degree in H. At s,z, the green degree exceeds the yellow degree by 1. Hence, the number of yellow green pairings we can construct are at most $|\Psi(I,F)|$. □

Lemma 13. *i) Let $u,y \in U$ and $v,x \in V$ such that $(y,v) \neq (u,x)$.*

$$\lambda(u,x)\lambda(\mathcal{N}(u,v)) \sum_y \lambda(v,y)\lambda(\widehat{\mathcal{N}}(y,x,(y,v),(x,u))) \leq 6 d_{max} \lambda(\mathcal{P})^2$$

ii) Let $s,u \in U$ and $v,x \in V$.

$$\lambda(u,x)\lambda(\mathcal{N}(u,v))\lambda(\widehat{\mathcal{N}}(s,x,(s,u))) \leq 4\lambda(\mathcal{P})\lambda(\mathcal{N}(s,v))$$

iii) Fix $s \in U, z \in V$. Let $u,y \in U$ and $v,x \in V$ such that $(y,v) \neq (u,x)$.

$$\lambda(u,x)\lambda(\mathcal{N}(u,v)) \sum_y \lambda(v,y)\lambda(\widehat{\mathcal{N}}(s,z,y,x,(y,v),(x,u))) \leq 2 d_{max} \lambda(\mathcal{P})\lambda(\mathcal{N}(s,z))$$

Proof. *i*) Let $N_1 \in \mathcal{N}(u,v)$ and $N_2 \in \bigcup_y \widehat{\mathcal{N}}(y,x,(y,v),(x,u))$. We will consider the symmetric difference $N_1 \oplus N_2$ and define a modified (multi)graph $H'(N_1,N_2)$ and a set of pairings of H', $\Psi(N_1,N_2)$. From N_1, N_2 and a pairing in $\Psi(N_1,N_2)$ we construct graphs $N_3 \in \mathcal{P}$ and $N_4 \in \mathcal{P}$, and a pairing of $H'(N_3,N_4)$. The graphs will satisfy $N_1 \cup N_2 \cup \{(u,x),(v,y)\} = N_3 \cup N_4$ where the union takes into account multiplicities. Given N_3, N_4, and the pairing of $H'(N_3,N_4)$ we will be able to reconstruct N_1, N_2 and the original pairing given an additional $3 \times [d_{max}] \times \{0,1\}$ amount of information. We then show that the number of pairings of $H'(N_3,N_4)$ is at most the number of pairings of $H'(N_1,N_2)$, and this implies the claimed inequality.

First assume the vertices u,y and v,x are distinct. Consider the symmetric difference $H = N_1 \oplus N_2$ so that the edges from N_1 are blue, and those from N_2 are red. Then, x,y each have blue degree 1 more than their red degree while u,v have red degree one more than their blue degree. We will fix a pairing of the red and blue edges as follows. The graph H may or may not contain the edges $(u,x),(v,y)$ depending on whether or not they are present in N_1. If either is present, it is colored blue. To define pairings at each vertex, first add the *uncolored* edges $(u,x),(v,y)$ to H, i.e. let $H' = H \cup \{(u,x),(v,y)\}$ and retain the color of all the edges from H. Note, H' may have double edges. To define the pairings at u,v, think of both the uncolored edges $(u,x),(v,y)$ as blue and define an exact pairing of the red and blue edges. At y,x, we think of the uncolored edges as red and define an exact pairing of the red and blue edges at these vertices such that the red edge (x,u) (resp. (y,v)) is always paired with the blue (x,u) (resp. (y,v)), if it is present, for example, see Figure 9 *a*). For all other vertices, the red degree is equal to the blue degree, and we pair them up. Call this pairing in H' ψ, and let the set of pairings be $\Psi(N_1,N_2)$.

Then ψ defines a decomposition of H' into alternating circuits of even length. These are shown for the example in Figure 9 *b*). The idea of the map is to traverse the circuits and put edges alternately in $N_3 \in \mathcal{P}$ and $N_4 \in \mathcal{P}$. For each circuit not containing the uncolored edges, put edges alternately in N_3 and N_4 making the convention that the blue edges are put into N_4 and the red edges into N_3. There is only one way for a circuit to contain the uncolored edges; such a circuit must contain both. (There cannot be two distinct circuits each containing one uncolored edge, since the circuit is even, and the edges alternate red-blue, ignoring the uncolored edge). For the circuit containing the uncolored edges, put edges alternately in N_3 and N_4 starting with the uncolored (y,v) in N_3. Edges which are in both N_1, N_2 are added to both N_3, N_4. Note that this set never includes the edges $(u,x),(v,y)$, so we never attempt to add them to N_3 or N_4 twice. By the definition of ψ, if H' has any double edges, then both copies do not go into the same graph since they appear consecutively in a circuit or walk. Then, $N_3 \in \mathcal{P}$ and $N_4 \in \mathcal{P}$. The bit b of the map is set to 1 if the blue edge (v,y) was present in $N(u,v)$ and was traversed *after* the uncolored (v,y). The set $[d_{max}]$ is used to encode the vertex y.

To invert the map, consider two tables, $N_3 \in \mathcal{P}$ and $N_4 \in \mathcal{P}$, and their symmetric difference $N_3 \oplus N_4$. If the pairing ψ of H' was known, we claim we can recover N_1, N_2 uniquely. We can reconstruct H' as follows. If (u,x) (resp. (y,v)) appears in $N_3 \oplus N_4$, then it was not present in N_1, and hence appears once in H'. On the other hand, if (u,x) (resp. (y,v)) does not appear in $N_3 \oplus N_4$, then it was present in N_1, and hence appears as a double edge in H'. Thus, we can reconstruct H' from $N_3 \oplus N_4$ by adding in two copies of the edge if necessary. If ψ was known, we could partition the edges of H' into N_1, N_2 as follows. The pairing ψ determines the decomposition of H' into alternating circuits. There will be exactly one circuit which contains the edges (u,x) and (v,y). For the other circuits and the walk, we put the edges coming from N_3 into N_2, and the edges from N_4 into N_1. If there is a circuit containing $(u,x),(v,y)$, proceed as follows. If (y,v) does not appear as a double edge, start with the edge in the circuit after (y,v), and put edges alternately in N_1 and N_2,

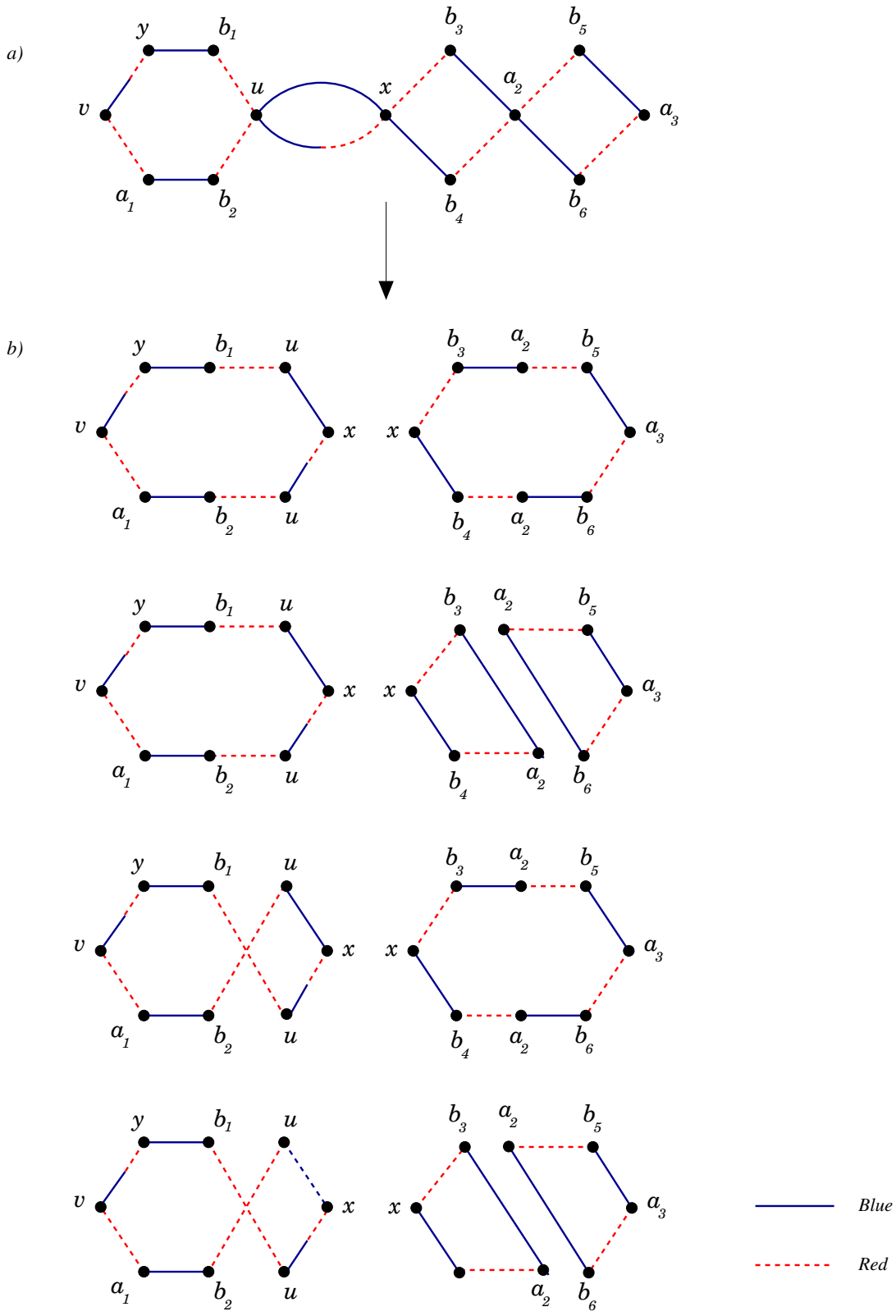

Figure 9: *a*) The graph H', *b*) Decompositions of H' into alternating circuits

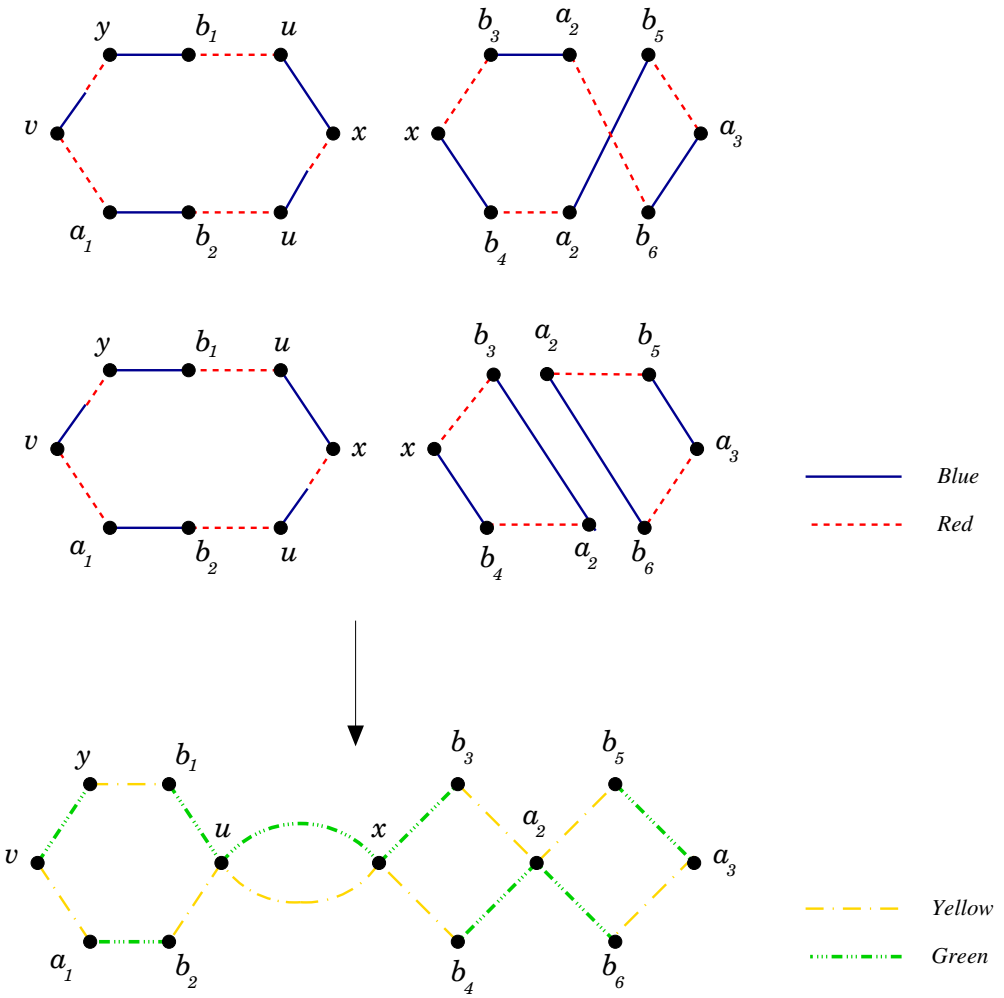

Figure 10: Graphs N_1, N_2 which map to a pair $N_3, N_4 \in \mathcal{P}$

and also skipping one copy of the edge (u, x). If (y, v) appears twice in the circuit, we can determine which copy was the uncolored one by looking at the bit b. If b is 1, it is the first one, and if b is 0, it is the second one. Proceed as before, start with the edge after the uncolored (v, y), and assign edges alternately to N_1 and N_2, and also skip one copy of (u, x). Finally, put all other common edges of N_3, N_4 into both N_1 and N_2.

Color the edges of H' green if they come from N_3, and yellow if they come from N_4. Since we do not have the pairing ψ of H', instead, we use the fact that a pairing of the original red and blue edges is a pairing of the yellow and green edges of H' at all the vertices. We know that at x, y if there is a double edge, they are colored yellow, and green, and must be matched. Also, at u, v the double edges are not paired. Hence the number of valid yellow-green pairings in H' is at most as the number of original red-blue pairings $|\Psi(N_1, N_2)|$, and so there cannot be too many initial pairs of tables mapping to N_3, N_4. This is illustrated with an example in Figure 10.

In the case that the vertices are not distinct, there 2 other possibilities :

a) $u = y, v \neq x$

b) $u \neq y, v = x$

The two cases are symmetric, except that in the second case, we have to keep track of y so we only give the argument for a). Let $N_1 \in \mathcal{N}(u, v)$ (blue) and $N_2 \in \widehat{\mathcal{N}}(u, x, (u, v), (x, u))$ (red). Then, in $H = N_1 \oplus N_2$, the vertex u has equal red and blue degree, while v has 1 extra red degree and x has 1 extra blue degree. Also, if the edges (u, v) or (u, x) are present, they are blue. Construct H' as before, and define the pairings as before. Thus at x we think of the uncolored (u, x) as red (and pair it with the blue (u, x) if it is present), while at u we think of it as blue. At v, think of the uncolored (u, v) as blue, while at u, we think of it as red, and always pair it with the blue (u, v) if it is present. The remainder of the argument is the same as when the vertices are distinct.

Now, given which case we are in (there are 3 cases in all), and $N_3, N_4 \in \mathcal{P}$, and the vertex y, the inequality follows since the number of yellow-green pairings is at most $|\Psi(N_1, N_2)|$.

ii) Let $N_1 \in \mathcal{N}(u, v)$ and $N_2 \in \bigcup_y \widehat{\mathcal{N}}(s, x, (x, u))$. As before, we will define a modified (multi)graph $H'(N_1, N_2)$ and a set of pairings of H', $\Psi(N_1, N_2)$. From N_1, N_2 and a pairing in $\Psi(N_1, N_2)$ we will construct graphs $N_3 \in \mathcal{N}(s, v)$ and $N_4 \in \mathcal{P}$, and a pairing of $H'(N_3, N_4)$. The graphs will satisfy $N_1 \cup N_2 \cup (u, x) = N_3 \cup N_4$, taking into account the multiplicity of the edges. Given N_3, N_4, and the pairing of $H'(N_3, N_4)$ we will be able to reconstruct N_1, N_2 and the original pairing given a constant amount of additional information. We then show that the number of pairings of $H'(N_3, N_4)$ is at most the number of pairings of $H'(N_1, N_2)$, and this implies the claimed inequality.

First assume the vertices s, x, u, v are distinct. Consider the symmetric difference $H = N_1 \oplus N_2$ so that the edges from N_1 are blue, and those from N_2 are red. In H, s, x each have blue degree 1 more than their red degree while u, v have red degree one more than their blue degree. Let $H' = H \cup \{(u, x)\}$ and retain the color of all the edges from H leaving the new edge (u, x) uncolored. Define a pairing ψ of H' as follows. At s, choose one blue edge which remains unpaired, and pair up the remaining red and blue edges. At v, choose one red edge to remain unpaired and pair up the others. To define the pairing at u, think of the uncolored edge (u, x) as blue and define an exact pairing of the red and blue edges. At x, we think of the uncolored edge as red and define an exact pairing of the red and blue edges at these vertices such that the red edge (x, u) is always paired with the blue (x, u), if it is present. For all other vertices, the red degree is equal to the blue degree, and we pair them up as usual. Let the set of such pairings be $\Psi(N_1, N_2)$.

Then ψ defines a decomposition of H' into circuits of even length and a walk of odd length from s to v whose initial edge is blue, final edge is red, and contains the uncolored edge (u, x), since the length of the walk is odd. The idea of the map is the same as in the previous case, to put edges from the circuits and walks alternately in N_3 and N_4 with the same color conventions as before. When we traverse the walk, starting with N_4 we put each edge alternately into N_3 and N_4. Then, $N_3 \in \mathcal{N}(s, v)$ and $N_4 \in \mathcal{P}$.

To invert the map, consider the symmetric difference $N_3 \oplus N_4$. If the pairing ψ of H' was known, we can recover N_1, N_2 uniquely. We can reconstruct H' as follows. If (u, x) appears in $N_3 \oplus N_4$, then it was not present in N_1, and hence appears once in H'. On the other hand, if (u, x) does not appear in $N_3 \oplus N_4$, then it was present in N_1, and hence appears as a double edge in H'. Thus, we can reconstruct H' from $N_3 \oplus N_4$ by adding in two copies of the edge if necessary. If ψ was known, we could partition the edges of H' into N_1, N_2. The pairing ψ determines the decomposition of H' into circuits and a walk of odd length. The walk contains (all the copies of) the edge (u, x) since the circuits are all even length. For each circuit as well as the walk, we put the edges coming from N_3 into N_2, and the edges from N_4 into N_1. Put all other common edges of N_3, N_4 into both N_1 and N_2.

Color the edges of H' green if they come from N_3, and yellow if they come from N_4. Since we do not have the pairing ψ of H', instead, we use the fact that a pairing of the original red and blue edges is a pairing of the yellow and green edges of H' at all the vertices. We know that at x if there is a double edge, they are colored yellow, and green, and must be matched. Also, at u, the double edges are not paired.

If the vertices are not distinct, there are 3 cases:

a) $u = s, v \neq x$. In this case, add the uncolored (u, x) to H. At u, think of the uncolored edge as blue, and fix a pairing by leaving out one blue edge. At x, fix the pairing by always pairing the uncolored/red (u, x) with the blue copy of (u, x) if it is present. Now in H', v has one extra red degree, while u has an extra blue degree taking into account the uncolored (u, x). Hence the pairing determines an alternating walk from s to v with initial edge blue, and final edge red, containing the uncolored edge, and alternating circuits. Put the edges along the walk alternately in N_3 and N_4. Thus we ensure N_3, N_4 each contain at most one copy of (u, x). Inverting the map is easy if the pairing of H' is known, and we can bound the number of yellow-green pairings as before.

b) $u \neq s, v = x$. In this case the argument is similar to the above, except that in H', to fix a red-blue pairing, think of the the uncolored edge (u, x) as blue at u and red at x.

c) $u = s, v = x$. This case becomes trivial. Let $N_1 \in \mathcal{N}(u, v)$ and $N_2 \in \widehat{\mathcal{N}}(u, v, (v, u))$. Set $N_3 = N_1$ and $N_4 = N_2 \cup (u, x)$. Clearly, $N_4 \in \mathcal{P}$, $N_3 \in \mathcal{N}(s, v)$, and the map is easily invertible.

Hence the number of yellow-green pairings in H' is at most the number of original red-blue pairings $|\Psi(N_1, N_2)|$. Given which case we are in (which is a factor of 4), the inequality follows.

iii) Let $N_1 \in \mathcal{N}(u, v)$ and $N_2 \in \bigcup_y \widehat{\mathcal{N}}(s, z, y, x, (y, v), (x, u))$. As before, we define a modified (multi)graph $H'(N_1, N_2)$ and a set of pairings of H', $\Psi(N_1, N_2)$. From N_1, N_2 and a pairing in $\Psi(N_1, N_2)$ we construct graphs $N_3 \in PP$ and $N_4 \in \mathcal{P}$, and pairing of $H'(N_3, N_4)$. The graphs will satisfy $N_1 \cup N_2 \cup \{(u, x), (v, y)\} = N_3 \cup N_4$, taking into account multiplicity of edges. Given N_3, N_4, and the pairing of $H'(N_3, N_4)$ we will be able to reconstruct N_1, N_2 and the original pairing given an additional $[d_{max}] \times \{0, 1\}$ amount of information. We then show that the number of pairings of $H'(N_3, N_4)$ is at most the number of pairings of $H'(N_1, N_2)$, and this implies the claimed inequality.

First assume the six vertices are distinct. Consider the symmetric difference $H = N_1 \oplus N_2$ so that the edges from N_1 are blue, and those from N_2 are red. Then, s, z, x, y each have blue degree 1 more than their red degree while u, v have red degree one more than their blue degree. Define H' as in i). We will fix a pairing ψ of the red and blue edges in H' as follows. At s, z, choose one blue edge which remains unpaired, and pair up the remaining red and blue edges. The pairing at all other vertices is defined as in i). Let the set of such pairings be $\Psi(N_1, N_2)$.

Then ψ defines a decomposition of H' into circuits of even length and a walk of odd length from s to z whose initial and final edges are blue. Traverse the walk, and starting with N_4 put each edge alternately into N_3 and N_4. The rest of the edges of H' are partitioned as in i). Then, $N_3 \in \mathcal{N}(s, z)$ and $N_4 \in \mathcal{P}$. The bit b of the map is set to 1 if the blue edge (v, y) was present in $N(u, v)$ and was traversed *after* the uncolored (v, y). The set $[d_{max}]$ is used to encode the vertex y.

We can reconstruct H' using the symmetric difference $N_3 \oplus N_4$ and the edges $(u, x), (v, y)$ exactly as in i). Since we do not have the pairing ψ of H', to recover N_1, N_2 we use the fact that a pairing of the original red and blue edges is a pairing of the yellow and green edges of H' at all the vertices. Color the edges of H' green if they come from N_3, and yellow if they come from N_4. We know that

Table 1: Enumeration of the cases

	$u \neq y, v \neq x$	$u = y, v \neq x$	$u \neq y, v = x$
$s = u$	• $z = v$ • $z = x$ • $z \neq v, x$	• $z = v$ • $z = x$ • $z \neq v, x$	• $z = v = x$ • $z \neq v, x$
$s = y$	• $z = v$ • $z = x$ • $z \neq v, x$	Counted in the case $s = u$	• $z = v$ • $z \neq v, x$
$s \neq u, y$	• $z = v$ • $z = x$ • $z \neq v, x$	• $z = v$ • $z = x$ • $z \neq v, x$	• $z = v$ • $z \neq v, x$

at x, y if there is a double edge, they are colored yellow, and green, and must be matched. Also, double edges at u or v are never paired. Hence the number of yellow-green pairings in H' is at most the number of original red-blue pairings $|\Psi(N_1, N_2)|$.

Lastly, we handle the various cases in which the vertices are not distinct. There are 21 possible distinct cases depending on which of the vertices u, y, v, x, s, z are the same. These are enumerated in Table 1 for completeness.

In each of these cases, when we add the uncolored edges $(u, x), (v, y)$, so that we think of them as red at y and x and blue at u and v, in order to define the pairing of H', we find that each of s, z have 1 extra blue degree, and at every other vertex, the blue degree equals the red degree. Then, we can restrict to the same kinds of pairings as in the case when the vertices are distinct, and the lemma follows, once we factor in which of the 22 cases we are in, and the vertex y is known in each case.

Note that in some of the cases, the map can be defined by adding the edges $(u, x), (v, y)$ to the tables, but the map can be defined in this way through the pairings as well. Since the map is defined in the same way in each case, we do not even need to retain information about which of the 22 cases we are in, and the bound now follows. □

With the above inequalities in hand, the proof of Lemma 11 is a matter of plugging them in to the expressions which bound the congestion through a transition.

Proof of Lemma 11. When T is a transition of type 2b or 2c the flow through T can come from 3 sources. First, due to being on an alternating circuit between pairs of perfect tables. Second, the congestion due to being on the augmenting walk between a near-perfect table and a perfect table. Lastly, due to being on an alternating circuit between a near-perfect table and a perfect table. The proof of the bound is similar in each of these cases, and the bottleneck is the third case. In each case, let $T = (M, M')$, where $M \in \mathcal{N}(u, v)$ and $M' \in \mathcal{N}(u', v)$, with x as the pivot vertex, so that $M' = M \cup (u, x) \setminus (u', x)$.

We can bound the congestion due to $(I, F) \in \mathcal{P} \times \mathcal{P}$ through T as follows.

$$\sum_{\substack{(I,F) \in f_T \\ I \in \mathcal{P}}} \frac{|\Psi_T(I, F)|}{|\Psi(I, F)|} \frac{\pi(I)\pi(F)}{\pi(M)}$$

$$= \frac{1}{w(\Omega)} \sum_{\substack{(I,F)\in f_T \\ I\in\mathcal{P}}} \frac{|\Psi_T(I,F)|}{|\Psi(I,F)|} \frac{\lambda(I)\lambda(F)}{\lambda(M)} \frac{\lambda(\mathcal{N}(u,v))}{\lambda(\mathcal{P})}$$

$$\text{(By Lemma 12, i)} \quad \leq \frac{1}{w(\Omega)} \sum_{\substack{y \\ (y,v)\neq(u,x)}} \lambda(u,x)\lambda(y,v) \frac{\lambda(\widehat{\mathcal{N}}(y,x,(y,v),(x,u)))\lambda(\mathcal{N}(u,v))}{\lambda(\mathcal{P})}$$

$$\begin{aligned} \text{(By Lemma 13, i)} \quad &\leq \frac{6d_{max}\lambda(\mathcal{P})}{w(\Omega)} \\ &\leq \frac{6d_{max}}{nm} \end{aligned}$$

Next, we bound the congestion due to $(I,F) \in \mathcal{N} \times \mathcal{P}$ through T when T is on the alternating walk. Note that in this case at least one of the holes of I, v is the same as a hole of M.

$$\begin{aligned} &\sum_{s\in U} \sum_{(I,F)\in f_T^{s,v}} \frac{|\Psi_T(I,F)|}{|\Psi(I,F)|} \frac{\pi(I)\pi(F)}{\pi(M)} \\ = \quad &\frac{1}{w(\Omega)} \sum_s \frac{\lambda(\mathcal{N}(u,v))}{\lambda(\mathcal{N}(s,v))} \sum_{(I,F)\in f_T^{s,v}} \frac{|\Psi_T(I,F)|}{|\Psi(I,F)|} \frac{\lambda(I)\lambda(F)}{\lambda(M)} \end{aligned}$$

$$\begin{aligned} \text{(By Lemma 12, ii)} \quad &\leq \frac{1}{w(\Omega)} \sum_s \lambda(u,x) \frac{\lambda(\mathcal{N}(u,v))}{\lambda(\mathcal{N}(s,v))} \lambda(\widehat{\mathcal{N}}(s,x,(s,u))) \\ \text{(By Lemma 13, ii)} \quad &\leq \frac{4n\lambda(\mathcal{P})}{w(\Omega)} \\ &\leq \frac{4}{m} \end{aligned}$$

Lastly, we bound the congestion due to $(I,F) \in \mathcal{N} \times \mathcal{P}$ when T is on an alternating circuit.

$$\begin{aligned} &\sum_{s\in U, z\in V} \sum_{(I,F)\in f_T^{s,z}} \frac{|\Psi_T(I,F)|}{|\Psi(I,F)|} \frac{\pi(I)\pi(F)}{\pi(M)} \\ = \quad &\frac{1}{w(\Omega)} \sum_{s,z} \frac{\lambda(\mathcal{N}(u,v))}{\lambda(\mathcal{N}(s,z))} \sum_{(I,F)\in f_T^{s,z}} \frac{|\Psi_T(I,F)|}{|\Psi(I,F)|} \frac{\lambda(I)\lambda(F)}{\lambda(M)} \\ \text{(By Lemma 12, iii)} \quad &\leq \frac{1}{w(\Omega)} \sum_{s,z} \lambda(u,x) \frac{\lambda(\mathcal{N}(u,v))}{\lambda(\mathcal{N}(s,z))} \sum_{\substack{y \\ (y,v)\neq(u,x)}} \lambda(y,v)\lambda(\widehat{\mathcal{N}}(s,z,y,x,(y,v),(x,u))) \\ \text{(By Lemma 13, iii)} \quad &\leq \frac{2d_{max}nm\lambda(\mathcal{P})}{w(\Omega)} \\ &\leq 2d_{max} \end{aligned}$$

Adding the congestion from each of these sources, the congestion through a sliding transition T is bounded by $O(d_{max})$. □

4.3.2 Transitions of Type 2a or 1.

Lemma 14. *For a transition T of type 2a or 1,*

$$\sum_{(I,F)\in f_T} \frac{\pi(I)\pi(F)}{\pi(M)} \frac{|\Psi_T(I,F)|}{|\Psi(I,F)|} = O(1) \tag{6}$$

To prove the lemma, we again tailor the corresponding combinatorial inequalities of [3] for the case of canonical flows. We first state and prove the combinatorial results and then show how the lemma follows.

Lemma 15. *Let $T = (M, M')$ be a transition between a near-perfect table in $\mathcal{N}(u,v)$ and a perfect table, so that the edge (u,v) is either deleted or added. Let N be the near-perfect table of M and M'. Then,*

i)

$$\sum_{\substack{(I,F)\in f_T \\ I\in\mathcal{P}}} \frac{|\Psi_T(I,F)|}{|\Psi(I,F)|}\lambda(I)\lambda(F) \le \lambda(u,v)\lambda(\widehat{\mathcal{P}}(u,v))\lambda(N)$$

ii) For all $s \in U$ and $z \in V$,

$$\sum_{(I,F)\in f_T^{s,z}} \frac{|\Psi_T(I,F)|}{|\Psi(I,F)|}\lambda(I)\lambda(F) \le \lambda(u,v)\lambda(\widehat{\mathcal{N}}(s,z,(u,v)))\lambda(N)$$

Proof. i) Let $I \in \mathcal{P}$ (blue) and $F \in \mathcal{P}$ (red).

Fix a pairing $\psi \in \Psi_T(I,F)$. Define the graph $E_\psi(I,F) = I \oplus F \oplus (M \cup M')$. Then, $E_\psi(I,F) \in \widehat{\mathcal{P}}(u,v)$. Given $E_\psi(I,F), T$ and ψ, we can recover I and F. Since $I \cup F = N \cup E(I,F) \cup (u,v)$,

$$\frac{1}{|\Psi(I,F)|}\lambda(I)\lambda(F) = \frac{1}{|\Psi(I,F)|}\lambda(u,v)\lambda(E_\psi(I,F))\lambda(N)$$

As before, color the edges of $I \oplus F$ yellow and green depending on whether they come from E_ψ or M. The number of yellow-green pairings of $I \oplus F$ is bounded by $\Psi(I,F)$, and the inequality follows.

ii) Let $I \in \mathcal{N}(s,z)$ (blue) and $F \in \mathcal{P}$ (red).

Fix a pairing $\psi \in \Psi_T(I,F)$. Define the graph $E_\psi(I,F) = I \oplus F \oplus (M \cup M')$. Then, $E_\psi(I,F) \in \widehat{\mathcal{N}}(s,z,(u,v))$. Given $E_\psi(I,F), T$ and ψ, we can recover I and F. Since $I \cup F = N \cup E(I,F) \cup (u,v)$,

$$\frac{1}{|\Psi(I,F)|}\lambda(I)\lambda(F) = \frac{1}{|\Psi(I,F)|}\lambda(u,v)\lambda(E_\psi(I,F))\lambda(N)$$

As before, color the edges of $I \oplus F$ yellow and green depending on whether they come from E_ψ or M. The number of yellow-green pairings of $I \oplus F$ is bounded by $\Psi(I,F)$, and the inequality follows. □

Lemma 16. *i) Let $u \in U, v \in V$. Then,*

$$\lambda(u,v)\lambda(\widehat{\mathcal{P}}(u,v))\lambda(\mathcal{N}(u,v)) \le \lambda(\mathcal{P})^2$$

ii) Fix $s \in U, z \in V$. Let $u \in U, v \in V$. Then,

$$\lambda(u,v)\lambda(\widehat{\mathcal{N}}(s,z,(u,v)))\lambda(\mathcal{N}(u,v)) \leq 4\lambda(\mathcal{N}(s,z))\lambda(\mathcal{P})$$

Proof. i) Let $N_1 \in \mathcal{N}(u,v)$ (blue) and $N_2 \in \widehat{\mathcal{P}}(u,v)$ (red).

Consider the symmetric difference $H = N_1 \oplus N_2$. Both u, v have red degree one more than their blue degree. H may or may not contain the edge (u,v). If it is present, it is colored blue. We define a red-blue pairing of H to partition the edges into two perfect tables N_3, N_4. To define the pairing, we first define the multigraph $H' = H \cup (u,v)$, so that the new edge (u,v) is colored blue. Now, let Ψ_{good} be the set of possible pairings of H' so that for $\psi \in \Psi_{good}$, the corresponding decomposition of H' into alternating circuits, there is not circuit containing both copies of (u,v). In case H' contained only one copy of (u,v) all pairings are 'good'. If H' did indeed contain two copies of (u,v), we claim that $|\Psi_{good}|$ is at least $1/2$ fraction of all possible pairings. To see this, take any pairing whose circuit decomposition contains a circuit with both copies of (u,v). From this pairing, we can obtain a 'good' pairing by switching the red edges that the blue copies of (u,v) are paired with at u. Note that two such distinct pairings will always give distinct 'good' pairings.

Now, fix $\psi \in \Psi_{good}$. Let C_1, C_2 be the circuits containing the edge (u,v). For every other circuit, send all the blue edges to N_3 and the red edges to N_4. Do the same for the circuit of C_1, C_2 in which for the edge (u,v), v is adjacent to a lower numbered vertex through a red edge. For the remaining circuit, put the red edges in N_3, and the blue edges in N_4. Lastly, put all edges in $N_1 \cap N_2$ into both N_3, N_4. Then, $N_3, N_4 \in \mathcal{P}$.

As before, we can recover the uncolored H' from N_3, N_4. If the pairing of H' was known, the map can be inverted, and N_1, N_2 recovered. Since the pairing is not known, proceed as follows. Color the edges of H' green if they are from N_3 and yellow if from N_4. Now the total number of yellow-green pairings is equal to the total number of possible red-blue pairings. However, we can eliminate the ones in which, say at u the copies of (u,v) are paired, since this would give a cycle decomposition which was impossible for a pairing from Ψ_{good}. If the yellow degree of u in H is $d \geq 2$ (which is the case if there were 2 (blue) copies of (u,v) in H'), this eliminates at least $(d-1)!/(d!) \geq 1/2$ of all yellow-green pairings. Hence not too many N_1, N_2 pairs can map to N_3, N_4.

ii) In the case that $s \neq u$ and $z \neq v$, the proof is analogous to the previous case. The other cases are:

a) $s = u, z \neq v$. Let $N_1 \in \mathcal{N}(u,v)$ be blue and $N_2 \in \widehat{\mathcal{N}}(s,z,(u,v))$ be red. Then, in the symmetric difference, u has equal red and blue degree, v has 1 extra red degree, and z has one extra blue degree. If we add an extra blue edge (u,v), then s has an extra blue degree while v get equal red and blue degree. Hence in a pairing of H', there is an alternating walk from s to z whose initial and final edges are blue. As before, to take care that the two copies of (u,v) don't end up in the same table, we can exchange the pairing at one end, say u (on either the walk or any circuit), to get a pairing where the two edges are not part of the same circuit or walk.

b) $s \neq u, z = v$. The argument in this case is similar to *a*).

c) $s = u, z = v$. This case is trivial. If $N_1 \in \mathcal{N}(u,v)$ and $N_2 \in \mathcal{N}(s,z)$ such that the edge (u,v) is not present in N_2, set $N_3 = N_1 \in \mathcal{N}(s,z)$, and set $N_4 = N_2 \cup (u,v) \in \mathcal{P}$. The map is clearly invertible.

Since in all, there are 4 cases, accounting for a factor of 4, given which case we are in, we obtain the claimed bound. □

We now plug the above bounds into the expressions for congestion through a transition of the chain which either adds or deletes an edge.

Proof of Lemma 14. Let T be a transition which either adds or deletes an edge (a move in the Markov chain of type 1 or 2a). In each case, let $T = (M, M')$, where $M \in \mathcal{N}(u, v)$ and $M' \in \mathcal{P}$ (the proof in the case that the transition deletes an edge is along the same lines, with the appropriate modification to Lemmas 15 and 16). We bound the left hand side of (6) by bounding the contribution firstly, due to a pair of perfect tables, and secondly due to a near perfect and a perfect table.

We bound the congestion through T due to $(I, F) \in \mathcal{P} \times \mathcal{P}$ as follows.

$$\begin{aligned}
& \sum_{\substack{(I,F)\in f_T \\ I\in\mathcal{P}}} \frac{|\Psi_T(I,F)|}{|\Psi(I,F)|}\frac{\pi(I)\pi(F)}{\pi(M)} \\
= \quad & \frac{1}{w(\Omega)} \sum_{\substack{(I,F)\in f_T \\ I\in\mathcal{P}}} \frac{|\Psi_T(I,F)|}{|\Psi(I,F)|}\frac{\lambda(I)\lambda(F)}{\lambda(M)}\frac{\lambda(\mathcal{N}(u,v))}{\lambda(\mathcal{P})} \\
\text{(By Lemma 15, i)} \quad \leq \quad & \frac{1}{w(\Omega)}\lambda(u,v)\lambda(\widehat{\mathcal{P}}(u,v))\frac{\lambda(\mathcal{N}(u,v))}{\lambda(\mathcal{P})} \\
\text{(By Lemma 16, i)} \quad \leq \quad & \frac{\lambda(\mathcal{P})}{w(\Omega)} \\
\leq \quad & \frac{1}{nm}
\end{aligned}$$

Next, we bound the congestion through T due to $(I, F) \in \mathcal{N} \times \mathcal{P}$.

$$\sum_{s\in U, z\in V} \sum_{(I,F)\in f_T^{s,z}} \frac{|\Psi_T(I,F)|}{|\Psi(I,F)|}\frac{\pi(I)\pi(F)}{\pi(M)}$$

$$= \quad \frac{1}{w(\Omega)} \sum_{s,z} \frac{\lambda(\mathcal{N}(u,v))}{\lambda(\mathcal{N}(s,z))} \sum_{(I,F)\in f_T^{s,z}} \frac{|\Psi_T(I,F)|}{|\Psi(I,F)|}\frac{\lambda(I)\lambda(F)}{\lambda(M)}$$

$$\begin{aligned}
\text{(By Lemma 15, ii)} \quad & \leq \quad \frac{1}{w(\Omega)} \sum_{s,z} \lambda(u,v)\lambda(\widehat{\mathcal{N}}(s,z,(u,v)))\frac{\lambda(\mathcal{N}(u,v))}{\lambda(\mathcal{N}(s,z))} \\
\text{(By Lemma 16, ii)} \quad & \leq \quad \frac{4nm\lambda(\mathcal{P})}{w(\Omega)} \\
& \leq \quad 4
\end{aligned}$$

Adding the congestion from each of these sources, the congestion through a transition that adds or deletes an edge is bounded by $O(1)$. □

Lemmas 11 and 14 imply the Inequality (4). It can be seen from the proofs of the Lemmas in this section, that if the weights $w(u, v)$ satisfy (1), the bound holds for the weights w up to a small constant factor. Hence, we have that $\rho(f) = O(nmD^2 d_{max})$. This implies that the mixing time of the chain started at G is bounded by $\tau_G(\delta) = O(nmD^2 d_{max}(\ln(1/\pi(G)) + \log \delta^{-1}))$. This completes the proof of Theorem 10.

5 Approximating Ideal Weights by Simulated Annealing

Recall that our goal is to find the ideal weights (or, rather, a constant factor approximation of the ideal weights) for $\lambda = 1$.

As mentioned earlier, we will do this by progressively increasing the value of λ. We start with λ close to 0, when it is possible to compute a $(1+\varepsilon)$ approximation of the ideal weights in a straightforward manner, see Lemma 4. However, later in the algorithm we will only have a constant factor, say 2, approximation of the ideal weights. We will use samples of the corresponding Markov chain to obtain a better approximation of the ideal weights, what in turn allows us to increase the value of λ slightly so that the improved approximation of the ideal weights of the old λ sufficiently approximates the ideal weights of the new λ. Eventually, λ becomes 1 and we will have a suitable approximation of the ideal weights for $\lambda = 1$. In this section we discuss these steps in more detail.

5.1 Bootstrapping

Suppose we have weights $w_\lambda(u,v)$ which are a 2-approximation to the weights $w^*_\lambda(u,v)$. That is, suppose that $w^*_\lambda(u,v)/2 \le w_\lambda(u,v) \le 2w^*_\lambda(u,v)$. We want to use the Markov chain to tighten this approximation to a factor $c \in (1,2)$. The following computation closely mimics the computation of [12, Section 3].

Recall, that π_λ denotes the stationary distribution of the Markov chain. To simplify notation, we will omit the subscript λ. Recall, that for a given activity λ the ideal weights are defined as $w^*_\lambda(u,v) = \lambda(\mathcal{P})/\lambda(\mathcal{N}(u,v))$. Note that if $w(u,v) = w^*(u,v)$ for every $u \in U, v \in V$, then

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

Thus for the Markov chain run with weights $w = w^*$, the stationary distribution of the chain satisfies $\pi(\mathcal{N}(u,v)) = \pi(\mathcal{P})$. For arbitrary weights w, note that

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

Rearranging terms, we have

$$w^*(u,v) = w(u,v)\frac{\pi(\mathcal{P})}{\pi(\mathcal{N}(u,v))} \tag{7}$$

This implies a bootstrapping procedure to boost rough approximations to w^* into arbitrarily close approximations. By sampling from the stationary distribution of the chain with weights w, we can estimate $\pi(\mathcal{P})/\pi(\mathcal{N}(u,v))$, and thus using (7) we can estimate $w^*(u,v)$.

Here are the details. The idea is to obtain a $c^{1/2}$-approximation of both $\pi(\mathcal{P})$ and $\pi(\mathcal{N}(u,v))$. Then we will have a c-approximation, say z, of $\pi(\mathcal{P})/\pi(\mathcal{N}(u,v)) = w^*(u,v)/w(u,v)$. In other words,

$$z/c \le \frac{w^*(u,v)}{w(u,v)} \le cz$$

and thus it suffices to set the weight approximations $w_{\text{new}}(u,v) := w(u,v)z$ to get c-approximations of $w^*(u,v)$.

We can use the indicator random variables X and $X_{u,v}$ for the events "a sample from π is in $\mathcal{P}$" and "a sample from π is in $\mathcal{N}(u,v)$" as estimators of $\pi(\mathcal{P})$ and $\pi(\mathcal{N}(u,v))$. However, by running the MC we cannot obtain a sample from π, rather a sample from $\hat{\pi}$ which is δ-close to π in total

variation distance. Thus, $E[X] = \hat{\pi}(\mathcal{P})$ and $E[X_{u,v}] = \hat{\pi}(\mathcal{N}(u,v))$. It is sufficient to set δ so that $\hat{\pi}(\mathcal{P})$ and $\hat{\pi}(\mathcal{N}(u,v))$ approximate $\pi(\mathcal{P})$ and $\pi(\mathcal{N}(u,v))$, respectively, by a factor of $c^{1/4}$. Then we can use several samples of X and $X_{u,v}$ to approximate $\hat{\pi}(\mathcal{P})$ and $\hat{\pi}(\mathcal{N}(u,v))$ within a factor of $c^{1/4}$. Thus, overall we obtain a c-approximation of the ratio $\pi(\mathcal{P})/\pi(\mathcal{N}(u,v))$.

First we sketch how to set δ so that $\hat{\pi}$ is within a factor of $c^{1/4}$ of π. Recall that the distribution π is defined by a weight function w which is a 2-approximation of the ideal weights w^*. Thus, $4/(nm) \geq \pi(\mathcal{P}), \pi(\mathcal{N}(u,v)) \geq 1/(4nm)$ and we can set $\delta = \Theta(1/(nm))$ so that $\hat{\pi}(\mathcal{P}), \hat{\pi}(\mathcal{N}(u,v)) = \Theta(1/(nm))$ and $\hat{\pi}(\mathcal{P})$ and $\hat{\pi}(\mathcal{N}(u,v))$ are $c^{1/4}$-approximations of $\pi(\mathcal{P})$ and $\pi(\mathcal{N}(u,v))$.

To obtain a $c^{1/4}$ approximation of $\hat{\pi}(\mathcal{P})$ we approximate $E[X]$ within a factor of $c^{1/4}$ by averaging s random variables $X_1, \ldots, X_s$. By the Chernoff bounds, since $E[X] = \Theta(1/(nm))$, it suffices to take $s = O(nm \log \zeta^{-1})$ samples to approximate $E[X] = \hat{\pi}(\mathcal{P})$ with probability $\geq 1 - \zeta$. Analogous arguments hold for $E[X_{u,v}]$.

Putting it all together, the average of the X_i's estimates $\hat{\pi}(\mathcal{P})$ within a factor of $c^{1/4}$ with probability $\geq 1 - \zeta$ and $\hat{\pi}(\mathcal{P})$ is within a factor of $c^{1/4}$ of $\pi(\mathcal{P})$. Thus, we obtained estimates $\pi(\mathcal{P})$ within a factor of $c^{1/2}$ with probability $\geq 1 - \zeta$. Therefore, with probability at least $1 - (nm+1)\zeta$ we obtain $c^{1/2}$ approximations of all $\pi(\mathcal{P})$, $\pi(\mathcal{N}(u,v))$, resulting in factor of c approximations of $w^*(u,v)$ for every u, v. Since we do the bootstrapping for every λ_i, the overall probability of success is $\geq 1 - (nm+1)\ell\zeta$ which we want to be, say, 4/5. It suffices to set $\zeta = \Theta(1/((nm+1)\ell))$.

5.1.1 Warm Starts

For a fixed λ the improved approximation of the ideal weights includes running the Markov chain $s = O(nm \log \zeta^{-1}) = O(nm \log(nm))$ times. By Theorem 10 the mixing time of the Markov chain started at graph G is $O(nmD^2 d_{max}(\ln(1/\pi(G)) + \log \delta^{-1}))$. The term $\log \pi(G)^{-1}$ comes from the fact that the starting distribution is concentrated on the state G. The graph G^* seems to be a good starting point since $\lambda(G^*) = 1$ and thus $\log \pi(G^*)^{-1} = O(D \log(nm))$. If we start the chain at G^* we need to take $O(nmD^2 d_{max}(\ln(1/\pi(G^*)) + \log \delta^{-1}))$ steps of the chain per sample. The standard method of warm starts can be used to avoid the $\log \pi(G^*)^{-1}$ term in the running time. The idea is to obtain the first sample by taking $O(nmD^2 d_{max}(\ln(1/\pi(G^*)) + \log \delta^{-1}))$ steps, but all subsequent samples are obtained by running the Markov chain started at the previous sample. This way, the chain is effectively started from a distribution close to the stationary distribution and thus the subsequent samples each take only $O(nmD^2 d_{max}(\log \delta^{-1}))$ steps. The same idea is used in [12].

5.2 Total Number of Phases

We specify a sequence $\varepsilon(nm)^{-D} = \lambda_1 \leq \cdots \leq \lambda_\ell = 1$ such that

$$\frac{1}{\sqrt{2}} \leq \frac{w^*_{\lambda_i}(u,v)}{w^*_{\lambda_{i+1}}(u,v)} \leq \sqrt{2} \qquad \text{for every } i \in [\ell - 1] \text{ and every } u, v$$

Then, if $w_{\text{new}}(u,v)$ is a $\sqrt{2}$-approximation (remember that we are free to choose any constant $c \in (1,2)$) of $w^*_{\lambda_i}(u,v)$, by the above $w_{\text{new}}(u,v)$ is also a 2-approximation of $w^*_{\lambda_{i+1}}(u,v)$. Therefore we can increase λ from λ_i to λ_{i+1} and still be able to use Theorem 10.

We obtain the above λ sequence by reversing the output produced by the algorithm of [3] for computing the cooling schedule λ. It constructs a λ-sequence of length $\ell = O(D \log D \log(nm))$ with the additional property that $\lambda_{i+1}(\mathcal{P})$ is within a factor of $2^{1/4}$ of $\lambda_i(\mathcal{P})$ and $\lambda_{i+1}(\mathcal{N}(u,v))$ is within a factor of $2^{1/4}$ of $\lambda_i(\mathcal{N}(u,v))$ for every u, v and $i \in [\ell - 1]$ (see Lemmas 2 and 3 of [3],

notice that in our case $s = D$ and $\gamma = (nm)^D$ since we may assume that $\varepsilon \geq 1/(nm)^D$). Thus, $1/\sqrt{2} \leq w^*_{\lambda_i}(u,v)/w^*_{\lambda_{i+1}(u,v)} \leq \sqrt{2}$, as required.

6 Counting by Sampling

We will sketch a standard reduction from counting to sampling, due to Jerrum, Valiant and Vazirani [13]. Our goal is to estimate $|\mathcal{P}|$. For any sequence $\lambda_1, \ldots, \lambda_\ell$,

$$|\mathcal{P}| = \frac{|\mathcal{P}|}{w_{\lambda_\ell}(\Omega)} \frac{w_{\lambda_\ell}(\Omega)}{w_{\lambda_{\ell-1}}(\Omega)} \frac{w_{\lambda_{\ell-1}}(\Omega)}{w_{\lambda_{\ell-2}}(\Omega)} \cdots \frac{w_{\lambda_2}(\Omega)}{w_{\lambda_1}(\Omega)} w_{\lambda_1}(\Omega)$$

Let us fix the λ-sequence from the previous section. We first estimate

$$w_{\lambda_1}(\Omega) = \lambda_1(\mathcal{P}) + \sum_{u,v} w_{\lambda_1}(u,v)\lambda_1(\mathcal{N}(u,v))$$

where $1 \leq \lambda_1(\mathcal{P}) \leq 1+\varepsilon$ and $x_{u,v} \leq \lambda_1(\mathcal{N}(u,v)) \leq (1+\varepsilon)x_{u,v}$ for every u, v, see Lemma 4. Since $w_{\lambda_1}(u,v) = 1/x_{u,v}$, we get that $nm+1$ is a $(1+\varepsilon)$ approximation of $w_{\lambda_1}(\Omega)$. We define $s_* := |\mathcal{P}|/w_{\lambda_\ell}(\Omega)$ and $s_i := w_{\lambda_i}(\Omega)/w_{\lambda_{i-1}}(\Omega)$ and we will use samples of the Markov chain to estimate each s_i within a factor of $e^{\varepsilon/(2\ell)}$ and s_* within a $e^{\varepsilon/2}$ factor. Then, if s'_* and s'_i denote the estimates for s_* and the s_i, the quantity $(nm+1)s'_*s'_2 \ldots s'_\ell$ estimates $|\mathcal{P}|$ within a factor $(1+\varepsilon)e^\varepsilon = 1+\varepsilon'$, as required.

Recall that with probability $\geq 4/5$ the weights w_{λ_i} correctly approximate the ideal weights $w^*_{\lambda_i}$ for every i, see Section 5.1. In what follows we will assume that the weights w are correct estimates of w^* for every λ_i.

Notice that since $\lambda_\ell = 1$ and each $w(u,v)$ is within a factor of 2 of $w^*(u,v)$, we have that $|\mathcal{P}| = w_{\lambda_\ell}(\mathcal{P})$ is within a constant factor of $w_{\lambda_\ell}(\Omega)/(nm+1)$. Thus, $s_* = \Theta(1/(nm))$. By a similar argument, $s_i = \Theta(1)$ for each i. Therefore we can estimate the s_i as follows. We take a random sample X of the Markov chain for λ_i and consider the value of $\mathrm{est}(X) := w_i(M)/w_{i-1}(M)$. The expectation of this value is exactly s_i. Then we take $O(\ell\varepsilon^{-2})$ samples of the Markov chain for λ_i with the variation distance $\delta = O(\varepsilon/\ell)$ and average their $\mathrm{est}_i(X)$ values, obtaining est_i. By the Chebyshev's inequality, $\prod_{i=2}^{\ell} \mathrm{est}_i$ estimate $\prod_{i=2}^{\ell} s_i$ within an $e^{\varepsilon/2}$ factor with a probability $\geq 11/12$ (for suitable constants within the O notation).

Similarly, we sample X by the Markov chain for λ_ℓ and $\delta = O(\varepsilon)$ and define $\mathrm{est}_*(X)$ to be indicator variable for the event $X \in \mathcal{P}$. Then the expectation of $\mathrm{est}_*(X)$ is s_* and we average the values of $O(nm\varepsilon^{-2})$ samples to get within a factor $e^{\varepsilon/2}$ of s_* with probability $\geq 11/12$. Then, $(nm+1)\mathrm{est}_* \prod_{i=2}^{\ell} \mathrm{est}_i$ approximates $|\mathcal{P}|$ within a factor of $(1+\varepsilon)e^\varepsilon$ with probability $\geq 5/6$.

Thus, with probability $\geq 4/5$ we have correct estimates w of the ideal weights w^* and conditioned on the correct weight estimates the algorithm outputs a $(1+\varepsilon')$-approximation of $|\Omega|$ with probability $\geq 5/6$. Unconditionally, with probability $\geq 2/3$ the algorithm produces an answer within $(1+\varepsilon')$ factor of $|\Omega|$.

See [12] and [3] for details of the computation.

7 Proof of Theorem 1

We now recall the statement of our main theorem, and conclude its proof.

Theorem 1. *For any bipartite graph $G = (U \cup V, E)$ where $U = \{u_1, \ldots, u_n\}$ and $V = \{v_1, \ldots, v_n\}$, any degree sequence $r(1), \ldots, r(n); c(1), \ldots, c(m)$, any $0 < \varepsilon, \eta < 1$, we can approximate the number of subgraphs of G with the desired degree sequence (i.e., u_i has degree $r(i)$ and v_j has degree $c(j)$, for all i, j) in time $O((nm)^2 D^3 d_{\max} \log^5(nm/\varepsilon)\varepsilon^{-2} \log(1/\eta))$ where $D = \sum_i r(i) = \sum_j c(j)$ is the total degree and $d_{\max} = \max\{\max_i r(i), \max_j c(j)\}$ is the maximum degree. And, the approximation is guaranteed to be within a multiplicative factor $(1 \pm \varepsilon)$ of the correct answer with probability $\geq 1 - \eta$.*

The Theorem states that we can approximately count the number of bipartite graphs with a given degree sequence which are subgraphs of any given bipartite graph G. In the previous sections we dealt with the case when $G = K_{n,m}$, the complete bipartite graph on $n + m$ vertices. If G is not complete, we can perform the annealing algorithm in two stages. In the first stage, we run the simulated annealing algorithm described previously. Thus, we estimate the ideal weights for $\lambda = 1$ for the complete graph at the end of the first stage. In the second stage, we do the simulated annealing starting with the weights at the end of the first stage (notice that now all edge activities are 1). However, the annealing will *decrease* the activities of edges *not present in* G from 1 to $\lambda \approx 0$ (hence, we may be decreasing the activities of different edges than the ones whose activities were previously increased). The analysis of the annealing algorithm and the mixing time of the Markov chain remain the same. Thus, the two stage process only doubles the running time.

Now we break up the running time in the first stage. Initially, we spend $O((nmd_{\max})^2)$ time to construct the Greedy graph G^* and to approximate the initial weights, see Lemma 4. We need $\ell = O(D \log^2(nm))$ intermediate temperatures for the simulated annealing (Section 5.2). As discussed in Section 5.1, at each temperature we need to generate $O(nm \log(nm))$ samples from the stationary distribution of the Markov chain in order to do the bootstrapping. By Theorem 10, see also Section 5.1.1, each sample takes $O(D^2 nmd_{\max} \log(nm)$ steps of the Markov chain (recall that we set $\delta = \Theta(1/(nm))$, Section 5.1). Thus, as discussed in Section 5.1, with probability $\geq 4/5$ in time $O((nm)^2 D^3 d_{\max} \log^4(nm))$ we compute correct approximations of the ideal weights w^* for $\lambda = 1$. Therefore, we can generate a random bipartite graph with the desired degree sequence, from a distribution within variation distance $\leq \delta$ of uniform, in time $O((nm)^2 D^3 d_{\max} \log^4(nm/\delta))$. The computation of the initial weights is absorbed by this quantity.

For the counting, see Section 6, we use $O(nm\varepsilon^{-2})$ samples of the Markov chain to approximate s^* and for every intermediate temperature we need $O(\ell\varepsilon^{-2}) = O(D \log^2(nm)\varepsilon^{-2})$ samples to approximate the corresponding s_i. Taking into account the mixing time of the Markov chain, the counting phase takes time $O(D^4 nmd_{\max} \log^5(nm/\varepsilon)\varepsilon^{-2})$. Thus, the final running time of the algorithm including the weight estimation phase is $O(D^3 (nm)^2 d_{\max} \log^5(nm/\varepsilon)\varepsilon^{-2})$. With probability $\geq 2/3$ the algorithm outputs a $(1+\varepsilon)$ approximation of the number of bipartite graphs with the desired degree sequence. This can be boosted to probability $\geq 1 - \eta$ by running the algorithm $O(\log \eta^{-1})$ times and outputting the median of the resulting values. □

8 Remaining proofs for Greedy

For completeness we include a proof of the correctness of the Greedy algorithm.

Lemma 17 (Correctness of GREEDY). *Let r, c be a pair of degree sequences and let π be a total ordering consistent with c. If the pair of sequences is feasible, GREEDY(r, c, π) outputs a corresponding bipartite graph G. If the sequences are infeasible, the algorithm returns "Sequences not feasible".*

Proof. The proof is by induction on the number of non-zero entries in r. In the base case, there is a single non-zero entry in r, say $r(u)$. In this case, the sequences are feasible iff $c(v) \leq 1$ for

all $v \in V$, and $\sum_{v \in V} c(v) = r(u)$. Thus, if the sequences are feasible, GREEDY(r, c, π) outputs the bipartite graph. If it is the case that $\sum_{v \in V} c(v) \neq r(u)$, GREEDY returns "Sequences not feasible" in step 2. Else, it must be that the set $Y \subseteq V$ contains a vertex of degree 0, in which case GREEDY returns "Sequences not feasible" in step 6. Assume inductively that the Lemma holds for any degree sequences r', c' such that r' has fewer non-zero entries than r, and any total ordering π' consistent with c'. We claim that GREEDY works correctly in both of the possible cases:

- The sequences r, c are a feasible pair.
  Fix any total ordering π. We show that if r, c is a feasible sequence, then there is a corresponding bipartite graph G such that the vertex of highest degree $x \in U$ is joined to the set Y, the $r(x)$ vertices of highest degree in V with respect to the ordering π. Let G' be a bipartite graph with degree sequence r, c not satisfying this property. We obtain the graph G'', with degree sequence r, c, with strictly fewer vertices of Y not joined to x. Let $y \in Y$, such that $(x, y) \notin E(G')$. Then, there is a vertex $u \in V \setminus Y$ such that $(x, u) \in E(G')$. Note that $c(u) \leq c(y)$. Hence there is a vertex $w \in N(y)$ such that $w \notin N(u)$. Let $E(G'') = (E(G') \cup \{(x, y), (w, u)\}) \setminus \{(y, w), (x, u)\}$. Hence, there exists a graph such that x is matched to Y. Hence, $\hat{r}, \hat{c}$ is a feasible pair. By induction, GREEDY$(\hat{r}, \hat{c}, \hat{\pi})$ outputs a graph $\hat{G}$. Then $G = \hat{G} \cup \bigcup_{y \in Y}(x, y)$ is the required graph corresponding to the sequence r, c.

- The sequences r, c are an infeasible pair.
  - If GREEDY fails while processing x, we are done.
  - Assume GREEDY successfully matches x to Y, then, $\hat{r}, \hat{c}$ must be infeasible, otherwise, by induction, the algorithm produces $\hat{G}$, and $G = \hat{G} \cup \bigcup_{y \in Y}(x, y)$ would be a graph corresponding to r, c.

□

Finally, we present a family of graphs, each resulting from a greedy algorithm breaking ties arbitrarily, which for some feasible holes u, v require an alternating path from u to v of linear length.

Lemma 18. *For every $n \geq 0$, there exist degree sequences r_n, c_n and corresponding graphs G_n such for some feasible pair of holes u, v, there is no alternating path from u to v of length $\leq 2n$ in G_n.*

Proof. Denote the vertices in the two bipartitions by $U = \{u_1, \cdots, u_{n+1}\}$ and $V = \{v_1, \cdots, v_{n+1}\}$. For $n = 0$, let $r_0 = c_0 = (1)$. For $n \geq 1$ let $r_n = c_n = (1, 1, 2, 3, \ldots, n)$. Construct G_n inductively as follows.

---

1. If $n = 0$, set $E(G_n) = \{(u_1, v_1)\}$.
2. If $n = 1$, set $E(G_n) = \{(u_1, v_2), (u_2, v_1)\}$.
3. For $n \geq 2$,
   i) Set $E(G_n) := \bigcup_{v \in U \setminus \{v_1\}} (u_{n+1}, v) \cup \bigcup_{u \in V \setminus \{u_1\}} (u, v_{n+1})$.
   ii) The degree requirements of $u_2, u_{n+1}, v_2, v_{n+1}$ are now satisfied. The residual degree sequence is of the form r_{n-2}, c_{n-2} on the vertices $u_1, u_3, \cdots, u_n$ and $v_1, v_3, \cdots, v_n$ if $n \geq 3$, and on u_1, v_1 if $n = 2$.

- If $n \geq 3$, construct the graph G'_{n-2} on $U' = \{u_3, u_1, \cdots, u_n\} = \{u'_1, \cdots, u'_{n-1}\}$ and $V' = \{v_3, v_1, \cdots, v_n\} = \{v'_1, \cdots, v'_{n-1}\}$. (Note that the order of u_1, u_3 and v_1, v_3 are reversed, so that u_n, v_n will be joined to all the vertices of V', U' except v_3, u_3 respectively.)
- If $n = 2$, construct the graph G'_{n-2} on $U' = \{u_1\}$ and $V' = \{v_1\}$.

iii) Set $E(G_n) := E(G_n) \cup E(G'_{n-2})$.

For every $n \geq 1$, u_2, v_2 is a pair of feasible holes. In the base cases, we can check that , the shortest alternating path in G_0 from u_2 to v_2 is of length 1, u_2, v_2, and the shortest alternating path in G_1 from u_2 to v_2 is of length 3, u_2, v_1, u_1, v_2. In G_2, the shortest alternating path from u_2 to v_2 is of length 5, $u_2, v_3, u_1, v_1, u_3, v_2$. Assume the statement is true for all $k < n$ for $n \geq 3$. We claim that the shortest alternating path from u_2 to v_2 in G_n is of length $\geq 2n + 1$. Any alternating path from u_2 to v_2 must begin with the sequence of vertices u_2, v_{n+1}, u_1, and end with v_1, u_{n+1}, v_2, and consist of an alternating path from u_1 to v_1, not using the vertices $u_2, u_{n+1}, v_2, v_{n+1}$. I.e., an alternating path in G'_{n-2} from u'_2 to v'_2. By induction, the path in G'_{n-2} has length $\geq 2n - 3$, and hence any alternating path in G_n from u_2 to v_2 has length $2n + 1$. □

9 Conclusions

We have presented an algorithm for directly solving binary contingency tables for arbitrary degree sequences. While our algorithm has many similarities to the permanent algorithm of [12], the new algorithm relies on a surprising combinatorial property of the greedy graph.

An interesting open problem is the efficiency of the Diaconis chain on arbitrary degree sequences. Does there exist a degree sequence for which the chain converges slowly to the stationary distribution, or is the mixing time polynomial for all degree sequences?

References

[1] M. Bayati and A. Saberi. Fast Generation of Random Graphs via Sequential Importance Sampling. Preprint. Available as Technical Report 06-06-4123-22, Stanford University, MS&E Department, 2006.

[2] J. Besag and P. Clifford. Generalized Monte-Carlo Significance Tests. *Biometrika*, 76:633-642, 1989.

[3] I. Bezáková, D. Štefankovič, V. Vazirani, and E. Vigoda. Accelerating Simulated Annealing for Combinatorial Counting. In *Proceedings of the 17th Annual ACM-SIAM Symposium on Discrete Algorithms* (SODA), 900-907, 2006.

[4] G. Cobb and Y. Chen. An Application of Markov Chain Monte-Carlo to Community Ecology. *American Mathematical Monthly*, 110:265-288, 2003.

[5] C. Cooper, M. Dyer, C. Greenhill. On Markov Chains for Random Regular Graphs. In *Proceedings of the 16th Annual ACM-SIAM Symposium on Discrete Algorithms* (SODA), 980-988, 2005.

[6] Y. Chen, P. Diaconis, S. P. Holmes, and J. S. Liu. Sequential Monte Carlo Methods for Statistical Analysis of Tables. *J. Am. Stat. Assoc.*, 100:109-120, 2005.

[7] P. Diaconis and B. Efron. Testing for Independence in a Two-Way Table: New Interpretations of the Chi-Square Statistic. *Ann. of Stat.*, 13:845-874, 1985.

[8] P. Diaconis and A. Gangolli. Rectangular Arrays with Fixed Margins. In *Discrete Probability and Algorithms*, eds. D. Aldous et al, New York: Springer-Verlag, 15-41, 1995.

[9] D. Gale. A theorem on flows in networks, *Pacific J. Math.*, 7:1073-1082, 1957.

[10] M. Jerrum and A. Sinclair. Fast uniform generation of regular graphs. *Theor. Comp. Sci.*, 73:91-100, 1990.

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

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

[13] M. Jerrum, L. Valiant, and V. Vazirani. Random generation of combinatorial structures from a uniform distribution. *Theor. Comp. Sci.*, 43:169-188, 1986.

[14] R. Kannan, P. Tetali, and S. Vempala. Simple Markov-chain algorithms for generating bipartite graphs and tournaments. In *Proceedings of the 8th Annual ACM-SIAM Symposium on Discrete Algorithms* (SODA), 193-200, 1997.

[15] J. H. Kim and V. H. Vu. Generating random regular graphs. In *Proceedings of the 35th Annual ACM Symposium on Theory of Computing* (STOC), 213-222, 2003.

[16] J. S. Liu. *Monte Carlo Strategies In Scientific Computing*, New York: Springer-Verlag, 2001.

[17] B. McKay and N. Wormald. Uniform generation of random regular graphs of moderate degrees, *Journal of Algorithms*, 11:52-67, 1990.

[18] H. J. Ryser. *Combinatorial Mathematics*, Carus Math. Monograph, No. 14, New York, Wiley, 1963.

[19] J.G. Sanderson. Testing Ecological Patterns. *American Scientist*, 88:332-339, 2000.

[20] J. Schweinsberg. An $O(n^2)$ Bound for the Relaxation Time of a Markov Chain on Cladograms. *Random Structures and Algorithms*, 20:59-70, 2002.

[21] A. Sinclair. Improved Bounds for Mixing Rates of Markov Chains and Multicommodity Flow. *Combinatorics, Probability and Computing*, 1:351-370, 1992.