Lecture Notes: A theorist's toolkit
Markov Chains and Approximate Counting

1 Overview

In this lecture, we will look at how to approximate the permanent of 0/1-matrices using Markov Chains. Calculating the permanent of such matrices is equivalent to calculating the number of perfect matches in a bipartite graph, a problem which is #**P**-complete.

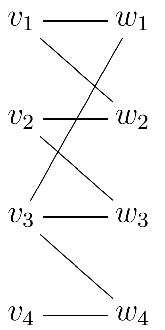
The construction of the approximation algorithm will be done in several steps. First, we will define what we actually mean by approximating, which in this case is constructing a Fully Polynomial Randomized Approximation Scheme (FPRAS). We will also define Fully Polynomial Almost Uniform Generator (FPAUG), i.e. an almost uniform random generator which can generate random samples of matchings for us. We will then show that if we can construct a family of such FPAUG:s for bipartite graphs, then we can construct a FPRAS using those FPAUG:s. This idea can be applied more generally, if it is possible to take increasingly specialized samples in some sense, that can often be used to create an approximation algorithm.

We will briefly review some theory about Markov Chains, which will be used to construct an FPAUG. This idea is also quite general, the same idea of approximation by sampling and using a Markov Chain for sampling will be demonstrated again in the next set of lecture notes, where the volume on a convex set will be approximated.

The ideas are generally applicable and not too difficult, but the devil is in the details here. We will cheat in several places. First, the sampling algorithm will only work for dense graphs. Secondly, we will only construct an FPAUG, which does not quite work with the FPRAS presented.

2 The permanent and perfect matchings

The permanent of a matrix is very similar to the determinant of a matrix.



(a) Bipartite graph

$$\begin{pmatrix} 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

(b) Corresponding matrix

Figure 1: Counting bipartite matchings with matrix permanents

Definition 1. The *permanent* of an $n \times n$-matrix A is defined as $\mathrm{per}(A) = \sum_{\pi \in \mathfrak{S}_n}^{n} \prod_{i=1}^{n} a_{i,\pi(i)}$ where $\mathfrak{S}_n$ is the symmetric group on $[n]$

Example 1.

$$\mathrm{per}\begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix} = 1 \cdot 5 \cdot 9 + 1 \cdot 6 \cdot 8 + 2 \cdot 4 \cdot 9 + 2 \cdot 6 \cdot 7 + 3 \cdot 4 \cdot 8 + 3 \cdot 5 \cdot 7$$

Calculating the permanent of an $n \times n$ $0/1$-matrix is equivalent to counting the numbers of perfect matchings in a bipartite graph with $n + n$ nodes. A bipartite graph with nodes $v_1, \ldots, v_n$ on the left side and $w_1, \ldots, w_n$ on the right side can be transformed into a "neighbor matrix" where $a_{i,j}$ is 1 if there is an edge from v_i to w_j and 0 otherwise.

The intuition for why this works is that each permutation corresponds to a potential matching. The product of the entries corresponding to this matching will be 1 if all edges exist, and 0 otherwise.

Counting the number of perfect bipartite matchings can be shown to be #**P**-complete, so chances of finding an efficient algorithm are slim.

Definition 2. (From Papadimitriou) Let Q be a polynomially balanced, polynomial-time decidable binary relation. The *counting problem* associated with Q is the following: Given x, how many y are there such that $(x, y) \in Q$? The output required is an integer in binary, say. #**P** is the class of all counting problems associated with polynomially balanced polynomial-time decidable relations.

3 Approximation

Since counting perfect bipartite matchings is #**P**-complete, we will try to create an approximation algorithm instead. The approximation algorithm will be a Fully Polynomial Randomized Approximation Scheme (FPRAS).

Definition 3. Let f be a function $f : \{0,1\}^* \to \mathbb{R}$. A Fully Polynomial Randomized Approximation Scheme (FPRAS) for f is a probabilistic algorithm $\mathcal{A}$ such that for all $x = \{0,1\}^*, \epsilon > 0$

- $\mathbf{Pr}\Big[|\mathcal{A}(x,\epsilon) - f(x)| > \epsilon|f(x)|\Big] \leq 1/4$
- $\mathcal{A}$ runs in polynomial time in $|x|$ and ϵ^{-1}

Note that, just as with e.g. **BPP**, we can shrink the error probability (which is $1/4$ for a single run) by running $\mathcal{A}$ repeatedly. To get an error probability of δ, we make $\mathcal{O}(\log \delta^{-1})$ runs of $\mathcal{A}$ and take the *median* as output.

There is an FPRAS construction for calculating the permanent of an $n \times n$-matrix with nonnegative entries (due to Jerrum, Sinclair and Vigoda, see Section 6.3), but today we will only consider 0/1-matrices. There are reasons to believe that it is not possible to handle negative entries, as can be seen from the following argument.

Assume that there is an FPRAS for the permanent of aribtrary matrices (including those with negative entries). Consider the matrix $M_\beta = (m_{i,j})$ with $m_{1,1}$ replaced with $m_{1,1} - \beta$. It will have $\mathrm{per}(M_\beta) = \mathrm{per}(M) - \beta \mathrm{per}(M_{1,1})$ where $M_{1,1}$ is M with row 1 and column 1 removed. On input M_β, the FPRAS would have to correctly identify the sign of $\mathrm{per}(M_\beta)$ (getting the wrong sign will give a distance larger than $|f(x)|$), so it will be possible to binary search and get a very close approximation of the value β. The same procedure can be applied recursively to $M_{1,1}$ until we get down to some small case (the permanent of a 1×1 matrix is quite easy to calculate). This will result in a polynomial algorithm which with high probability can calculate the exact value of $\mathrm{per}(M)$, something that is unlikely since a special case of that problem is #**P**-complete.

Definition 4. Let G be a bipartite graph and X be some set of matchings on G. A Fully Polynomial Almost Uniform Generator (FPAUG) for X is a probabilistic algorithm $\mathcal{A}$ such that $\forall x \in X, \epsilon > 0$

- $\left|\mathbf{Pr}[\mathcal{A}(G,\epsilon) = x] - \frac{1}{|X|}\right| \leq \frac{\epsilon}{|X|}$

- $\mathcal{A}$ runs in time polynomial in $|V(G)|$ and $\log \epsilon^{-1}$ and the length of the description of X

In general, if we have an algorithm which can sample in increasingly specialized ways, we can often use that sampling algorithm to construct an approximation scheme. This principle is illustrated in Proposition 1 where we will look at how to construct an FPRAS for counting perfect matchings from FPAUG:s sampling perfect matchings where a set of edges are included.

Definition 5. Let $M_k(G)$ be the set of matchings of size k in G where G is a bipartite graph.

Definition 6. $M_k^S(G) = \{m \in M_k(G) | S \subseteq m\}$, i.e. the set of all matchings of size k in the bipartite graph G where all the edges S are in the matching.

Proposition 1. *If we for all bipartite graphs G can construct an FPAUG for $M_n^S(G)$, $n = |G|$, for all $S \subseteq E(G)$, then we can construct an FPRAS for the function $f : G \to |M_n(G)|$.*

Proof. Assume $M_n \neq \emptyset$ (checking if a (bipartite) graph has a perfect matching is in $\mathbf{P}$). We will start by taking T samples $s_1, \ldots, s_T$ from M_n using the FPAUG. Let $\widetilde{e_1}$ be the edge from node v_1 occuring most frequently in these samples. Let $r_1 = \frac{|M_n^{\{\widetilde{e_1}\}}|}{|M_n|}$ and $\widetilde{r_1} = \frac{\sum_{i=1}^{T} [\widetilde{e_1} \in s_i]}{T}$ be an approximation of r_1 where $[expr]$ is 1 if the boolean expression $expr$ is true and 0 otherwise (Iverson's convention). It may be that $\widetilde{e_1}$ is not the edge maximizing r_1, but that is not important for the algorithm.

How large do we need T to be? For our analysis to work, we want to choose T large enough that $\mathbf{Pr}\big[|\widetilde{r_1} - r_1| > \frac{\epsilon}{2n} r_1\big] \leq \frac{1}{4n}$ holds. We will not cover all details of the analysis, but will mention some important points. We will use $\epsilon' = \frac{\epsilon}{30n}$ as parameter for the FPAUG.

It turns out that T will be inverse in r_1, so we need to make sure that we do not select $\widetilde{e_1}$ such that r_1 is exponentially small. There is always at least one edge occuring in at least $\frac{1}{n}$ of the samples (v_1 must be matched to some node in each sample, and there are n choices for that node). We add the requirement that $\mathbf{Pr}\big[|r_{1,i} - \widetilde{r_{1,i}}| \geq \frac{1}{2n}\big] \leq \frac{1}{8n|E_1|}$ where $r_{1,i} = \frac{|M_n^{\{e_{1,i}\}}|}{|M_n|}$, $\widetilde{r_{1,i}} = \frac{\sum_{j=1}^{T} [e_{1,i} \in S_j]}{T}$ for an edge $e_{1,i} = (v_1, w_i)$ and E_1 is the set of edges from v_1. We know that $\widetilde{e_1}$ will have $\widetilde{r_1} \geq \frac{1}{n}$ and this gives (by union bound) $\mathbf{Pr}\big[r_1 \leq \frac{1}{2n}\big] \leq \frac{1}{8n}$.

Assuming $r_1 \geq \frac{1}{2n}$, how large does T have to be for $\mathbf{Pr}\big[|\widetilde{r_1} - r_1| > \frac{\epsilon}{2n} r_1\big] \leq \frac{1}{8n}$ to hold? Using Chebychev bounds, it can be shown that $T = \frac{160n^4}{\epsilon^2}$ is sufficiently large. Using union bound again, we get that $\mathbf{Pr}\big[|\widetilde{r_1} - r_1| > \frac{\epsilon}{2n} r_1\big] \leq \frac{1}{4n}$.

The procedure can be repeated for $\widetilde{e_2} : r_2 = \frac{|M_n^{\{\widetilde{e_1},\widetilde{e_2}\}}|}{|M_n^{\{\widetilde{e_1}\}}|} \geq \frac{1}{n}$. This time we sample from $M_n^{\{\widetilde{e_1}\}}$ and let $\widetilde{e_2}$ be the edge from node v_2 on the left occuring most frequently.

Let $\tilde{r} = \widetilde{r_1} \cdots \widetilde{r_n} \approx r_1 \cdots r_n = \frac{|M_n^S|}{|M_n|}$ where $S = \{\widetilde{e_1}, \widetilde{e_2}, \ldots, \widetilde{e_n}\}$ is a perfect matching, so $|M_n^S| = 1$. The output of the algorithm is $\tilde{r}^{-1}$.

What is the probability that for at least one i we have $|\widetilde{r_i} - r_i| > \frac{\epsilon}{2n} r_i$? For a specific i the probability is $\frac{1}{4n}$ and so, by union bound, the probability that at least one is wrong is $\leq \frac{1}{4}$. If we can show that as long as this does not happen, we are within ϵ of r^{-1}, we have an FPRAS.

Assume that $|\widetilde{r_i} - r_i| \leq \frac{\epsilon}{2n} r_i$ for all i. This gives (for $\epsilon \leq 1$)

$$r^{-1} \frac{1}{(1+\epsilon/2n)^n} \leq \tilde{r}^{-1} \leq r^{-1} \frac{1}{(1-\epsilon/2n)^n}.$$

and in general, we have $1 - \epsilon \leq \frac{1}{1+\epsilon} \leq \frac{1}{(1+\epsilon/2n)^n}$ and $\frac{1}{(1-\epsilon/2n)^n} \leq \frac{1}{1-\epsilon/2} \leq 1 + \epsilon$. This gives us

$$r^{-1}(1-\epsilon) \leq \tilde{r}^{-1} \leq r^{-1}(1+\epsilon)$$

and thus, the propsed algorithm is an FPRAS. □

4 Markov Chains

We will now look at Markov Chains and some classifications of them. Markov Chains will later be used to construct an FPAUG for sampling matchings.

First, some notation. Let Ω be the state space for our Markov Chain. We will always have $|\Omega|$ finite. Let P be the transition matrix, where $P(x,y)$ denotes the probability to go from x to y, and $P^t(x,y)$ is the probability to go from x to y in exactly t steps.

Definition 7. A Markov Chain M is *aperiodic* if

$$\forall x, y \in \Omega : \gcd\{t : P^t(x,y) > 0\} = 1. \tag{1}$$

Definition 8. A Markov Chain M is *irreducible* if the underlying graph is strongly connected, i.e.

$$\forall x, y \in \Omega : \exists t : P^t(x,y) > 0. \tag{2}$$

Definition 9. A Markov Chain M is *ergodic* if there is a *stationary probability distribution* π over Ω such that

$$\forall x, y \in \Omega : \lim_{t \to \infty} P^t(x,y) = \pi(y). \tag{3}$$

Theorem 1. *A Markov Chain M is ergodic iff M is aperiodic and irreducible*

Ergodic Markov Chains turn out to be very useful for what we want. In essence, a Markov Chain is ergodic if there is some probability distribution over the state space for where we will be in the limit, regardless of where we started. It turns out that being ergodic is equivalent to being aperiodic and irreducible according to Theorem 1. In general, it is often easier to show that a Markov Chain is aperiodic and irreducible than it is to directly show that it is ergodic.

Definition 10. An ergodic Markov Chain M is *time-reversible* if, for the stationary distribution π we have

$$\forall x, y \in \Omega : \pi(x)P(x, y) = \pi(y)P(y, x). \tag{4}$$

Let $Q(x, y) = \pi(x)P(x, y) = \pi(y)P(y, x)$.

Lemma 1. *Let M be an ergodic Markov Chain. If there exists a probability distribution π over Ω such that $\forall x, y \in \Omega : \pi(x)P(x, y) = \pi(y)P(y, x)$ then M is time-reversible with the stationary distribution π.*

Lemma 1 will turn out to be useful for us. It says that it suffices to find a probability distribution π with the given property, and we will know that it is *the* stationary distribution.

Definition 11. The *underlying graph H_M* of a time-reversible Markov Chain M is a weighted, undirected graph $H_M = (\tilde{V}, \tilde{E})$ where $\tilde{V} = \Omega$ and $\tilde{E} = \{(x, y) \in \Omega^2 | Q(x, y) > 0\}$ where the weight for the edge $(x, y) \in \tilde{E}$ is $Q(x, y)$.

That the graph is weighted will not matter for us, but we might as well use the standard definition.

Definition 12. The *mixing time* for an ergodic Markov Chain M is

$$\tau_M(\epsilon) = \min\left\{ t : \left|\pi(y) - P^{t'}(x, y)\right| \leq \epsilon\pi(y) \text{ for all } x, y \in \Omega \text{ and } t' \geq t \right\}. \tag{5}$$

Recall that the stationary distribution is what we acheive in the limit. Since our sampling algorithm will be based on running a Markov Chain, running it infinitely long is not really a good idea. So, we will have to be content with acheiving something close to the stationary distribution and we need to know how long we need to run it to do so.

Note that the mixing time is defined in such a way that it is ok if we run the chain for "too long" (i.e. longer than the mixing time) in the sense that it will still have a probability distribution close to the stationary distribution. This is important for us, since we will not be able to calculate the mixing time exactly, but rather show an upper bound.

Definition 13. The *conductance* of a non-empty proper subset S of Ω is

$$\Phi(S) = \frac{\sum_{x \in S, y \notin S} Q(x,y)}{\min(\pi(S), 1-\pi(S))}, \tag{6}$$

where $\pi(S) = \sum_{x \in S} \pi(x)$.

If $\pi(S) < 1 - \pi(S)$, then the conductance of S is the probability that we after the next step will be outside S, given that we were in it. If, instead $1 - \pi(S) < \pi(S)$, then it is the probability that we will be in S given that we were outside it. The conductance of a Markov Chain (Definition 14) is closely related to the mixing time of the Markov Chain (Corollary 1), which is why we are interested in it.

Definition 14. The *conductance* for a time-reversible Markov Chain M

$$\Phi(M) = \min_{S \subseteq \Omega, 0 < \pi(S) \leq 1/2} \Phi(S). \tag{7}$$

Note that $\min(\pi(S), 1 - \pi(S))$ will always be $\pi(S)$ in this case.

Theorem 2. *Let M be a time-reversible Markov Chain such that $\forall x \in \Omega$: $P(x,x) \geq \frac{1}{2}$. Let $\pi_{\min} = \min \pi(x)$. Then*

$$\forall x, y \in \Omega : \frac{|\pi(y) - P^t(x,y)|}{\pi(y)} \leq \frac{(1 - \Phi(M)^2/2)^t}{\pi_{\min}}. \tag{8}$$

Corollary 1. *For a time-reversible Markov Chain M where, for all states $x \in \Omega : P(x,x) \geq \frac{1}{2}$ we have*

$$\tau_M(\epsilon) \leq \frac{2}{\Phi(M)^2}(\ln \epsilon^{-1} + \ln \pi_{\min}^{-1}). \tag{9}$$

Proof. $\frac{|\pi(y)-P^t(x,y)|}{\pi(y)} \leq \frac{(1-\Phi(M)^2/2)^t}{\pi_{\min}}$ according to Theorem 2. Set t such that $\frac{(1-\Phi(M)^2/2)^t}{\pi_{\min}} \leq \epsilon$. This gives

$$t \ln(1 - \Phi(M)^2/2) \leq \ln \epsilon + \ln \pi_{\min}.$$

We have $0 \leq \epsilon, \pi_{\min} \leq 1$ and so $\ln \epsilon^{-1} + \ln \pi_{\min}^{-1}$ is positive. We also have $0 < 1 - \Phi(M)^2/2 < 1$ since $0 < \Phi(M)^2/2 \leq 1/2$ (the conductance cannot be 0 since the Markov Chain is ergodic and cannot be greater than 1 since it is a probability) and thus $-\ln(1 - \Phi(M)^2/2)$ is positive. Reshuffling and negating gives

$$t \geq \frac{1}{-\ln(1 - \Phi(M)^2/2)}(\ln \epsilon^{-1} + \ln \pi_{\min}^{-1}).$$

Generally, we have $-\ln(1-x) \geq x$ for $x < 1$, so $-\ln(1 - \Phi(M)^2/2) \geq \Phi(M^2)/2$. So, setting $t = \frac{1}{\Phi(M)^2/2}(\ln \epsilon^{-1} + \ln \pi_{\min}^{-1})$ will make t large enough. $\square$

5 Constructing the generator

We will now look at how to construct an FPAUG for sampling perfect matchings in a bipartite graph using Markov Chains. This FPAUG will not be the one we required for the FPRAS discussed—it will only be able to sample perfect matchings in dense graphs.

The intuition is that we will create a time-reversible Markov Chain where the states are matchings in the graph. This chain will be constructed in such a way that the stationary distribution is uniform. We will then analyze it to show that the conductance is reasonably large (inversely polynomial in n) and thus, that the mixing time is reasonably small. Sampling will be done by starting at a random position and then just simulating the Markov Chain for a number of steps, outputting the matching we end up on.

In this construction, the state space for the Markov Chain will not consist solely of perfect matching, but will also include *nearly-perfect* matchings, i.e. matchings with one pair of nodes unmatched. This, of course, means that after walking on the Markov Chain, we might end up on an imperfect matching. In this case, we will simply restart the procedure again. Should we end up on an imperfect matching too many times (too many being some fixed, polynomially large, value) we will simply output some fixed matching (the probability of this occuring will be very low).

Definition 15. A bipartite graph $G = (V_0 \cup V_1, E)$ is dense if every $v \in V_i$ has $\geq |V_{1-i}|/2$ neighbors (i.e. is neighbor with at least half of the nodes on the other side).

Let us now construct the Markov Chain $\mathcal{MC}_G$ from the bipartite graph $G = (V, E)$. We will consider a matching as a set of edges (the edges between nodes matched in the matching, so in a perfect matching, each node will have exactly one edge). For a matching M:

1. Select a random $e \in E$, let $e = (u, v)$
2. (a) If $M \in M_n$ and $e \in M$, $M' \leftarrow M \backslash \{e\}$
   (b) If $M \in M_{n-1}$ and u, v unmatched in M, $M' \leftarrow M \cup \{e\}$
   (c) If $M \in M_{n-1}$ and u is matched to w, v unmatched, $M' \leftarrow M \backslash \{(u, w)\} \cup \{e\}$ (analogously for v matched, u unmatched)
   (d) Otherwise, let $M' \leftarrow M$
3. With probability $1/2$, go to M'

We call the transitions of the first type *adding* transitions, those of the second type *removing* transitions and those of the third kind *flipping* transitions.

This Markov Chain $\mathcal{MC}_G$ is obviously aperiodic (since every node has an edge to itself) and irreducible, which according to Theorem 1 means that it is ergodic. Notice that for all $x, y \in \Omega$ we have $P(x, y) = P(y, x)$, i.e. the transition matrix is symmetric. This is the case since, if x and y are neighbors, then there is exactly one edge which can be chosen in step 1 which transforms x to y and the probability of following the edge, if chosen is $\frac{1}{2}$, so the probability of going from x to y is $\frac{1}{2|E(\mathcal{MC}_G)|}$. Since the transition matrix is symmetric, we have $\pi(x)P(x, y) = \pi(y)P(y, x)$ for all $x, y \in \Omega$ if we let π be the uniform distribution over Ω. By Lemma 1, this means that $\mathcal{MC}_G$ is time-reversible with a uniform stationary distribution.

Theorem 3. *If G is a dense bipartite graph, then* $\Phi(\mathcal{MC}_G) \geq \frac{1}{12n^6}$

What is important here is that the conductance will be inversely polynomial in n. This means that we can use this construction as an FPAUG, since the mixing time, per Corollary 1 is inverse to the conductance squared, and thus also polynomial in n.

The remainder of this section will consist of the proof of Theorem 3. Let us begin by looking at $\Phi(\mathcal{MC}_G)$. We have that

$$\Phi(\mathcal{MC}_G) = \min_{0<|S|\leq\frac{|\Omega|}{2}} \frac{\sum_{x\in S, y\notin S, P(x,y)>0} \frac{1}{2|E||\Omega|}}{|S|/|\Omega|} = \frac{1}{2|E|} \min_{0<|S|\leq\frac{|\Omega|}{2}} \frac{|\tilde{E}(S, \bar{S})|}{|S|},$$

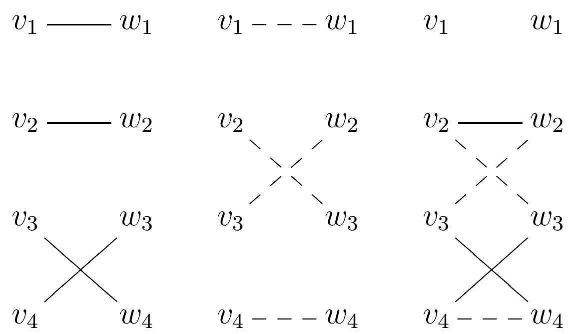
where $\tilde{E}$ are the edges of the underlying graph of the Markov Chain, per Definition 11, and $\tilde{E}(X, Y)$ are the edges in the underlying graph of the Markov Chain going from X to Y.

Now, we will construct paths $\gamma_{M,M'}$ from M to M' for every $M, M' \in \Omega$. The number of paths going from S to $\bar{S}$ is at least $|S||\bar{S}| = |S|(|\Omega| - |S|) \geq |S||\Omega|/2$. Our hope is to construct the paths in such a way that each edge in $\tilde{E}$ will occur in at most $b|\Omega|$ paths (where b is some polynomial in n), then $\tilde{E}(S, \bar{S}) \geq \frac{|S||\Omega|/2}{b|\Omega|} = \frac{|S|}{2b}$, which gives $\Phi(\mathcal{MC}_G) \geq \frac{1}{4b|E|}$. These paths are only a tool used for the analysis of the FPAUG.

5.1 Constructing the paths

We will begin by, for each $M \in \Omega$ specify a path from M to M:s "closest" perfect matching $\bar{M} \in M_n$:

1. If $M \in M_n$ then $\bar{M} = M$ and $\gamma_{M,\bar{M}} = \emptyset$



(a) I (b) F (c) $I \oplus F$

Figure 2: The xor of two perfect matchings, I and F

2. If $M \in M_{n-1}$ and $(u, v) \in E$ is unmatched, then $\bar{M} = M \cup \{(u, v)\}$ and $\gamma_{M,\bar{M}} = \langle (u, v) \rangle$
3. If $M \in M_{n-1}$ and $(u, v) \notin E$ is unmatched, then choose (in some canonical way) an edge $(u', v') \in M$ such that $(u', v), (u, v') \in E$ (such an edge always exists, as will be shown) and let $\bar{M} = M \backslash \{(u, v)\} \cup \{(u, v'), (u', v)\}$ and $\gamma_{M,\bar{M}} = \langle (u, v'), (u', v) \rangle$ (i.e. we first flip so that u, v' are matched and u', v are unmatched and then add the edge (u', v)).

We can always find (u', v') in step 3, since the graph is dense. Density by Definition 15 means that node u has at least $\frac{n}{2}$ neighbors, and none of them is v. Each of these nodes will be matched to a node, since the matching is almost perfect. The node v must be neighbor to at least one of these matched nodes, since it is neighbor to at least $\frac{n}{2}$ nodes, none of them u.

Construct the path from the closest perfect matching $\bar{M} \in M_n$ to every matching M by using the same path reversed. What remains to be done is constructing paths between pairs of perfect matchings.

Consider the path from $I \in M_n$ to $F \in M_n$. An important observation is that $I \oplus F$ (xor, i.e. $I \oplus F = \{(u, v)|(u, v) \text{ in exactly one of } I, F\}$) consists of a collection of cycles of even length and with length ≥ 4. An example of this can be seen in Figure 2.

Arrange these cycles $C_1, \ldots, C_r$ in some canonical way. A path from I to F can be constructed by *unwinding* the cycles in order. The cycles will have every other edge coming from I and every other edge coming from F.

We want to flip these so that we end up with the other edges in the cycle. This can be done by first (canonically) selecting a starting edge $e \in I \cap C_i$ and then in a canonical order selecting the edges in $F \cap C_i$, starting with one a node with e. An example can be seen in Figure 3, where we start by selecting transition (v_2, w_2), removing that edge (since it was present in

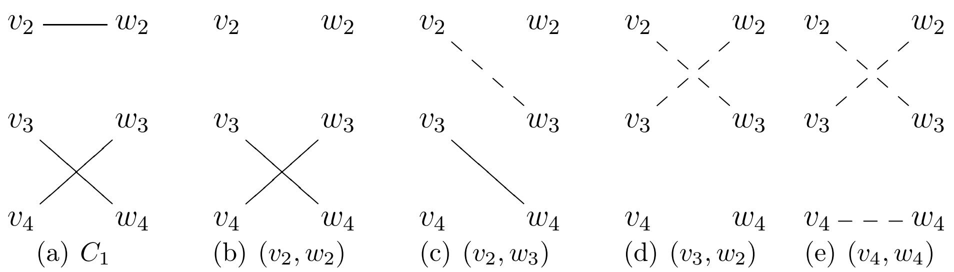


Figure 3: Unwinding a cycle

I). We then proceed by selecting the edges in $F \cap C_1$: (v_2, w_3) and (v_3, w_2) flipping edges and (v_4, w_4), adding the last edge of F.

The path $\gamma_{M,M'}$ between two arbitrary matchings M, M' will be constructed as $\gamma_{M,M'} = \langle \gamma_{M,\bar{M}}, \gamma_{\bar{M},\overline{M'}}, \gamma_{\overline{M'},M'} \rangle$. We will call $\gamma_{M,\bar{M}}$ the *start segment*, $\gamma_{\bar{M},\overline{M'}}$ the *main segment* and $\gamma_{\overline{M'},M'}$ the *end segment*.

5.2 Counting the paths

So now we have a construction of the paths in the Markov Chain. We will now attempt to analyze how many paths a single edge t in $\tilde{E}$, can occur in. We will refer to the edges in $\tilde{E}$ as transitions, since that is in some sense what they are, transitions between matchings. Recall that our goal here is to show that each transition t will occur in at most $b|\Omega|$ paths, where b is polynomial in n.

5.3 Start and end segments

Consider a transition $t = M \to M'$. How many start and end segments can t be part of? Let $K(M) = \{M' : \overline{M'} = M\}$, i.e. the set of matchings such that M is the "closest" perfect matching. Note that for $M \in M_{n-1}$ we have $K(M) = \emptyset$. An observation here is that two unmatched nodes u, v will uniquely decide an element in $K(M)$, i.e. there can be at most one element in $K(M)$ with u, v unmatched. If M is a perfect matching, it will also be in $K(M)$ and thus, $|K(M)| \leq n^2 + 1$.

If t is in a start segment on the path $\gamma_{A,B}$ then $\bar{A}$ is uniquely decided by t. There are at most $|K(\bar{A})| - 1 \leq n^2$ (the perfect matching $\bar{A}$ itself cannot be chosen now, since t can not be on the path from $\bar{A}$ to $\bar{A}$ (since that path is empty)) choices for A and $|\Omega|$ choices for B, and thus there are at most $n^2|\Omega|$ paths such that t is in the start segment. Analogously t can appear in at most $n^2|\Omega|$ end segments and thus in total t can appear in at most $2n^2|\Omega|$ start and end segments.

5.4 Main segments

The analysis for start and end segments was reasonably straightforward. For the main segment, we will have to use techniques which are more interesting.

How many paths between perfect matchings (i.e. main segments) can $t = M \to M'$ appear in? Let $P(t) = \{(I, F) \in M_n^2 \mid \text{The path } \gamma_{I,F} \text{ contains } t\}$. If we can find an injection $\sigma_t : P(t) \to \Omega$ then we know that $|P(t)| \leq |\Omega|$.

Consider $S = I \oplus F \oplus (M \cup M')$ (recall that $t = M \to M'$). For finite sets S_1, S_2, S_3 where $S_1 \cap S_2 \subseteq S_3 \subseteq S_1 \cup S_2 \Rightarrow |S_1 \oplus S_2 \oplus S_3| = |S_1| + |S_2| - |S_3|$ (this can be seen from e.g. a Wenn diagram). This gives $|S| = n$ since we have $I \cap F \subseteq (M \cup M') \subseteq I \cup F$ and $|I| = |F| = |M \cup M'| = n$.

A straightforward idea would be to try the function $\sigma_t(I, F) = I \oplus F \oplus (M \cup M')$, but that does not work since sometimes $I \oplus F \oplus (M \cup M')$ is not a matching (a node can be matched to more than one node). Luckily, this can be easily patched.

Claim 1. There is at most one node u_t such that u_t has more than one neighbor in S.

Proof. $I \oplus F$ consists of the set of cycles $C_1, \ldots, C_r$. The path between I and F unwinds these cycles in canonical order and t is on this path, so t must be modifying edges in one of the cycles, C_i. Consider the structure of $C_1, \ldots, C_{i-1}$ in $M \cup M'$. These will already be unwound, so they will be identical to F. Thus, $I \oplus F \oplus (M \cup M')$ will look like I on those cycles, and I is a matching so there is no node there with more than one neighbor. Analogously for the cycles $C_{i+1}, \ldots, C_r$.

Now consider C_i. If t was the first transition for unwinding the cycle (i.e. a removing transition), then $M \cup M' = M$ will look like I on cycle C_i and $I \oplus F \oplus (M \cup M')$ will look like F. Analogously if t was the last transition in unwinding the cycle (i.e. an adding transition).

If t was somewhere in the middle of unwinding the cycle, i.e. a flipping transition, then exactly one node (one of the nodes connected to the edge we started by removing, in fact) will be unmatched in $M \cup M'$, and thus that node will get two neighbors in $I \oplus F \oplus (M \cup M')$. □

Let $e_{I,t}$ be the edge in I which is matched to u_t. Define

$$\sigma_t(I, F) = \begin{cases} (I \oplus F \oplus (M \cup M')) - e_{I,t}) & \text{if } t \text{ is a flipping transition} \\ I \oplus F \oplus (M \cup M') & \text{otherwise} \end{cases}$$

In the first case, $\sigma_t(I, F) \in M_{n-1}$ (according to Claim 1) and otherwise $\sigma_t(I, F) \in M_n$. We will now show that $\sigma_t(I, F)$ is in fact an injection into

$\Omega = M_n \cup M_{n-1}$, by showing that given $\sigma_t(I, F)$ and t, we can reconstruct I and F. We can reconstruct $I \oplus F$ by the observation that

$$I \oplus F = \begin{cases} (\sigma_t(I, F) + e_{I,t}) \oplus (M \cup M') & \text{if } t \text{ is a flipping transition} \\ \sigma_t(I, F) \oplus (M \cup M') & \text{otherwise} \end{cases}$$

This gives the collection of cycles $C_1, \ldots, C_r$. Since we know t, we can deduce in which cycle t is, we will denote that cycle C_i. For $C_1, \ldots, C_{i-1}$ we know that F will look like M and I like $M \oplus I \oplus F$. Analogously for $C_{i+1}, \ldots, C_r$. Since we were unwinding C_i in some canonical order, we can also figure out how far we had gotten given the edge t and thus see which edges belong to I and which edges belong to F in that cycle too.

So, we now have that $|P(t)| \leq |\Omega|$. The number of paths where t is in the main segment is $\sum_{\gamma_{I,F} \in P(t)} |K(I)||K(F)| \leq (n^2+1)^2 |P(t)| \leq (n^2+1)^2 |\Omega|$

The total number of paths in which t appear is $\leq (2n^2 + (n^2+1)^2)|\Omega| \leq 3n^4|\Omega| \Rightarrow b \leq 3n^4$. Recall that $\Phi(\mathcal{MC}_G) \geq \frac{1}{4b|E|} \geq \frac{1}{12n^4n^2} = \frac{1}{12n^6}$ which is inversely polynomial in n, which concludes the proof of Theorem 3.

6 Loose ends

In this section, we will look quickly at some loose ends, such as why the graph must be dense and if there is some easy way to fix the FPAUG so that we can use it with the FPRAS construction we have. References for further reading will also be given.

6.1 Why the graph must be dense

The sampler we have constructed samples perfect matchings in a dense bipartite graph. Why do we require the graph to be dense? We used it at one point to show that the distance in the Markov Chain from an arbitrary matching to a perfect matching is at most 2. However, the fact that the bipartite graph is dense will also limit $\frac{|M_{n-1}|}{|M_n|}$ to be polynomial in n. For general graphs, that fraction can be exponential. This would be problematic for the sampler, since, if it stops on a matching in M_{n-1} it will be re-run, and in the highly improbable case that it has been rerun too many times (some large polynomial in n), it will output some fixed matching. If M_{n-1} is exponentially larger than M_n, this trick would not work anymore.

6.2 Why we can't use the FPAUG with the FPRAS

If we have a fully polynomial almost uniform generator for perfect matchings, can't we still use that to construct the FPRAS discussed? Recall that the

FPRAS needed to be able to sample perfect matchings which were guaranteed to include some set of edges. Can't we simply remove the nodes those edges were matching and then sample perfect matchings in the resulting graph? That would work out great if it was not for the fact that the FPAUG we have constructed requires the graph to be dense, and even if we start out with a dense graph, we cannot be certain that it will still be dense after removing nodes.

6.3 Further reading

For the interested reader, the idea for the algorithm was first presented by Broder (1986) in a paper titled "How hard is it to marry at random? (On the approximation of the permanent)" published in the Proceedings of the 18th Annual ACM Symposium on Theory of Computing (STOC), pages 50–58. However, the proof for that construction was not correct.

The algorithm, as presented here, is due to Jerrum and Sinclair (1989) and was published in a paper called "Approximating the permanent" published in SIAM Journal on Computing 18, pages 1149–1178. They fixed the proof and also made extensions to work for almost all graphs (not only dense graphs). More specifically, it works for all graphs such that $\frac{|M_{n-1}|}{|M_n|}$ is at most polynomial in n.

More recently, Jerrum, Sinclair and Vigoda (2000) showed how to approximate the permanent of a matrix with nonnegative entries. The algorithm can be found in e.g. "A Polynomial-Time Approximation Algorithm for the Permanent of a Matrix with Nonnegative Entries" published in the Journal of the ACM, Vol. 51, No. 4, July 2004, pages 671–697.

7 Wrapping it up

To sum it up, we started out with a problem, calculating the permanent of a matrix. It turned out that counting the number of perfect matchings in a bipartite graph was a special case of this problem. Since that problem is #**P**-complete, we quickly lost any hope for finding an algorithm for solving the problem, and instead turned to approximation.

We defined approximation in the form of a Fully Polynomial Randomized Approximation Scheme (FPRAS) and showed that we could construct such an approximation scheme given that we had something which could (almost uniformly) sample matchings from an increasingly specialized subset.

To do this sampling, we used the idea of creating a Markov Chain over the sampling space with a suitable (i.e. uniform) stationary distribution and then simulating the Markov Chain for a large number of steps and taking the position where we end up as the sample. The number of steps we needed to simulate to (almost) reach the stationary distribution was called the mixing time.

To illustrate the principle, we constructed a sampler which could only sample perfect matchings in dense bipartite graphs, which is not enough for the FPRAS construction. To analyze the time complexity of the sampler, we wanted to bound the mixing time which we did (using Corollary 1) by showing that the conductance of the graph was not too small.

To analyze the conductance it turned out that constructing a set of canonical paths between matchings in the underlying graph was useful, and in particular, what we needed to show was that each edge would not be in too many such paths. The construction of the paths here is only a tool for the analysis.

The analysis of how many start or end segments a certain edge could be in was quite simple, but the main segment was a bit more involved. It also demonstrated a technique which can often be useful—bounding the size of a set by creating an injective function from that set to some other set.

In the end, the analysis worked out and the sampler was shown to only need to go a polynomial number of steps. It is however unlikely to be very interesting in practice, given that the degree of the polynomial is prohibitively large. However, the construction and analysis demonstrated several interesting tools.