

We refer the reader to Jerrum's book [1] for the analysis of a Markov chain for generating a random matching of an arbitrary graph. Here we'll look at how to extend the argument to sample *perfect* matchings in dense graphs and arbitrary bipartite graphs. Both of these results are due to Jerrum and Sinclair [2].

Dense graphs

We'll need some definitions and notations first:

Definition 7.1 A graph $G = (V, E)$ is said to be dense if for every $v \in V$, $\text{degree}(v) > n/2$, where $n = |V|$.

Definition 7.2 Let G be a graph and let \mathcal{M}_G be the set of all perfect matchings of G . Similarly, for two distinct vertices u, v , let $\mathcal{N}_G(u, v)$ be the set of all perfect matchings of the graph $G \setminus \{u, v\}$. We refer to matchings in $\mathcal{N}_G(u, v)$ as near-perfect matchings, or matchings with holes u and v . Whenever G is clearly understood from the context, we shall use simply \mathcal{M} and $\mathcal{N}(u, v)$.

Jerrum's book presents the analysis of a Markov chain for sampling matchings (not necessarily perfect). The Markov chain for sampling perfect matchings in dense graphs is only a slight modification of the original chain. The state space Ω is the set of perfect and near-perfect matchings, i.e., $\Omega = \mathcal{M} \cup \cup_{u,v \in V} \mathcal{N}(u, v)$. The near-perfect matchings are needed to navigate between perfect matchings.

- For $X_t \in \Omega$,
 1. Choose $e = (u, v)$ uniformly at random from E .
 2. If u, v are both unmatched, let $X' = X_t \cup e$.
If $e \in X_t$, then let $X' = X_t \setminus e$.
If u is unmatched and $(v, w) \in X_t$, let $X' = X_t \cup e \setminus (v, w)$.

3. If $X' \in \Omega$, with probability $1/2$ set $X_{t+1} = X'$, else $X_{t+1} = X_t$.

The chain is symmetric, thus its stationary distribution π is uniform over Ω . For this Markov chain to efficiently (in polynomial time) sample perfect matchings, we have to have a polynomial bound on the ratio of perfect matchings to near-perfect matchings.

Lemma 7.3 For a dense graph, $\frac{|\Omega|}{|\mathcal{M}|} \leq n^2$. Thus, $\pi(\mathcal{M}) \geq 1/n^2$.

Proof: We shall prove that $|\mathcal{M}| \geq |\mathcal{N}(u, v)|$ for every u, v . We do this by defining an injective map $f_{u,v} : \mathcal{N}(u, v) \rightarrow \mathcal{M}$.

Case 1: If u and v are adjacent, then $f_{u,v}(M) = M \cup \{(u, v)\}$ for $M \in \mathcal{N}(u, v)$. Clearly, this map is injective.

Case 2: If u and v are not adjacent, then for $M \in \mathcal{N}(u, v)$ we claim there exists $e = \{w, x\} \in M$ such that $w \in N(u)$ and $x \in N(v)$. (The set $N(u)$ denotes the neighbors of vertex u .) This will then give an augmenting path of length three to transform M into a perfect matching. To prove this claim, let $S(v) = \{w \mid \{w, x\} \in M, x \in N(v)\}$. Since v is unmatched, every vertex in $S(v)$ is matched and therefore $|S(v)| = |N(v)| > n/2$. Since we also know $|N(u)| > n/2$, we have $S(v) \cap N(u) \neq \emptyset$. Let $z \in S(v) \cap N(u)$ and $\{w, z\} \in M$. Finally, let $f_{u,v}(M) = M \cup \{v, w\} \cup \{z, u\} \setminus \{w, z\}$. Again, $f_{u,v}$ is injective. ■

Theorem 7.4 For dense graphs, $\tau(\epsilon) \leq \text{poly}(n, \log 1/\epsilon)$.

Proof:(sketch): Our aim is to use the canonical paths and associated encoding η_t which were we used for the chain on all matchings. We will take it for granted that the reader remembers the paths and encoding that were used earlier. However, there are two problems we must address. By taking a brief look at the various cases, we will see that there are two scenarios where the proof from last class uses matchings with 4 holes. This is the issue we must resolve since 4-hole matchings are not contained in Ω .

Consider a pair of perfect matchings I, F . The canonical path used previously as well as the associated encoding are legitimate in the sense that both always contain at most two holes.

Suppose $I \in \mathcal{N}(u, v)$ and $F \in \mathcal{M}$. Looking at $I \oplus F$, we have an augmenting path from u to v and a set of alternating cycles. We define the canonical path γ_{IF} to unwind, i.e., augment, the path, and then unwind the cycles in some canonical order. Notice that after we unwind the path, we are at a perfect matching. It is easy to verify that this path never has more than two holes (the holes arise in the midst of unwinding an alternating cycle).

However, the associated encoding will have 4 holes. After we unwind the path, the encoding will have holes at u and v . During the unwinding of any cycle, we will get a further two holes in the encoding. Despite this fact, we keep the same encoding as before, but we now

have that $\eta_t : cp(t) \rightarrow \Omega'$, where Ω' is the set of matchings with at most 4 holes and $cp(t)$ is the set of canonical paths that traverse t . The encoding is simply an ingredient in the proof, used to bound $|cp(t)|$. Therefore, there is no inherent reason why the encoding must be contained in Ω . It will suffice to simply prove that $|\Omega'| \leq n^2|\Omega|$.

Finally, consider a pair of near-perfect matchings I, F . The canonical path will visit 4-hole matchings. To avoid this, we construct different canonical paths. We first choose a *random* $M \in \mathcal{M}$ and the new canonical path is the concatenation of γ_{IM} and γ_{MF} .

Now we want to bound the congestion. We first analyze the congestion created by paths between $I \in \Omega$ and $F \in \mathcal{M}$. For every $t = M \rightarrow M'$, we can define an injective map $\eta_t : cp(t) \rightarrow \Omega'$. Thus, we can bound $|cp(t)| \leq |\Omega'| \leq n^2|\Omega|$. (The second inequality follows from Lemma 7.3.) In order to account for the congestion added by flows between pairs of near-perfect matchings we use the fact that $|\mathcal{N} = \cup_{u,v} \mathcal{N}(u,v)| \leq n^2|\mathcal{M}|$. Fix a near-perfect matching I and a perfect matching M . For each near-perfect matching F , we use γ_{IM} with probability $1/|\mathcal{M}|$. Summing over F , the expected extra load on γ_{IM} is $|\mathcal{N}|/|\mathcal{M}| \leq n^2$. Therefore, paths between pairs of near-perfect matchings increase the congestion by a factor of n^2 .

If π is uniform on Ω , we can bound the congestion through $t = M \rightarrow M'$:

$$\begin{aligned} \rho &= \frac{1}{\pi(M)P(M, M')} \sum_{(I,F) \in cp(t)} \pi(I)\pi(F)|\gamma_{IF}| \\ &\leq |\Omega|m \sum_{(I,F) \in cp(t)} \frac{n}{|\Omega|^2} \\ &= mn \frac{|cp(t)|}{|\Omega|} \leq mn \frac{|\Omega'|}{|\Omega|} \leq mn^3 \end{aligned}$$

■

For dense graphs we can generate a random perfect matching in polynomial time, by running the chain for the mixing time. If the final state is a perfect matching (with probability at least $1/n^2$), then it's a random perfect matching. Otherwise, we run the chain again.

General Bipartite Graphs

Consider a bipartite graph $G = (V_1, V_2, E)$. We'll present the algorithm of Jerrum, Sinclair and Vigoda [3] for generating a random perfect of an arbitrary bipartite graph (and thus approximate the permanent of any non-negative matrix).

Once we remove the min-degree condition, the ratio of perfect matchings to near-perfect matchings can be exponentially small. To ensure perfect matchings are likely in the stationary distribution we introduce suitable weights on matchings and modify the Markov chain.

The weight of a matching is a function of the location of its holes (if any). Let

$$w^*(M) = \begin{cases} 1 & \text{for } M \in \mathcal{M} \\ w^*(u, v) = \frac{|\mathcal{M}|}{|\mathcal{N}(u, v)|} & \text{for } M \in \mathcal{N}(u, v) \end{cases}$$

We then modify the Markov chain introduced for dense graphs so that the stationary distribution is proportional to the weights. To do this we simply add what is known as a Metropolis filter. More precisely we simply modify the last step of the Markov chain to only move to the proposed new matching with an appropriate probability:

3. If $X' \in \Omega$, with probability $1/2 \min\{1, \frac{w^*(X')}{w^*(X_t)}\}$ set $X_{t+1} = X'$, else $X_{t+1} = X_t$.

It is straightforward to verify that, for $M \in \Omega$, $\pi(M) = w(M)/Z$ where $Z = \sum_{M' \in \Omega} w(M')$.

There is an obvious concern with our proposed weights. In order to compute the weights, we need to be able to compute the number of perfect matchings and near-perfect matchings – this was our original task. We will tackle this problem later. Let us first show that if we can obtain the weights, then we can generate a random perfect matching efficiently.

To motivate the above weights, let's look at the effect on the stationary distribution. We have

$$\pi(\mathcal{N}(u, v)) = \frac{1}{Z} \sum_{M \in \mathcal{N}(u, v)} w^*(M) = \frac{1}{Z} \sum_{M \in \mathcal{N}(u, v)} \frac{|\mathcal{M}|}{|\mathcal{N}(u, v)|} = |\mathcal{M}|/Z.$$

Similarly,

$$\pi(\mathcal{M}) = |\mathcal{M}|/Z.$$

Therefore, each hole pattern is equally likely, i.e., for all $u \in V_1, v \in V_2$,

$$\pi(\mathcal{M}) = \pi(\mathcal{N}(u, v)).$$

Hence,

$$\pi(\mathcal{M}) = \frac{1}{n^2 + 1}.$$

Thus, in the stationary distribution we output a random perfect matching with probability $1/(n^2 + 1)$. Equally important, with these ideal weights, or even a reasonable approximation, our Markov chain is rapidly mixing.

Lemma 7.5 *For the Markov chain with weights w satisfying, for all u, v where the associated quantities are defined,*

$$w^*(u, v)/2 \leq w(u, v) \leq 2w^*(u, v),$$

we have

$$\tau(\epsilon) \leq \text{poly}(n, \log 1/\epsilon).$$

The proof idea is the same as for dense graphs. We use the same canonical paths as before. The critical difference in the analysis is that we now have to guarantee that for a transition $t = M \rightarrow M'$,

$$w(I)w(F) \leq w(M)P(M, M')w(\eta_t(I, F)),$$

where $\eta_t(I, F) : cp(t) \rightarrow \Omega'$ is the same injective map we used for dense graphs. This inequality is equivalent to

$$\pi(I)\pi(F) \leq \pi(M)P(M, M')\pi(\eta_t(I, F)).$$

This ensures that the map η has sufficient weight to encode the initial, final pair. Assuming this inequality we can still bound the congestion thru a transition $t = M \rightarrow M'$ as follows:

$$\begin{aligned} \rho_t &= \frac{1}{\pi(M)P(M, M')} \sum_{(I, F) \in cp(t)} \pi(I)\pi(F)|\gamma_{IF}| \\ &\leq \frac{1}{\pi(M)P(M, M')} \sum_{(I, F) \in cp(t)} \pi(M)P(M, M')\pi(\eta_t(I, F))n \\ &= n \sum_{(I, F) \in cp(t)} \pi(\eta_t(I, F)) \\ &\leq n. \end{aligned}$$

For bipartite graphs, it turns out that the same canonical paths and encodings η_t as used for dense graphs, also satisfy the desired inequality on weights. This follows by checking several cases. (The inequality does not hold for non-bipartite graphs.)

Therefore, if we have these ideal weights w^* we can generate a random perfect matching in a polynomial number of steps (with high probability).

What happens if we run the chain with an approximation w to the ideal weights? The chain still converges rapidly to its stationary distribution, but the stationary distribution is no longer uniform over hole patterns. The likelihood, in the stationary distribution, of each hole pattern will be skewed by how far off the weights are from ideal. In particular, we will have

$$\pi(\mathcal{M}) \propto |\mathcal{M}|, \quad \pi(\mathcal{N}(u, v)) \propto w(u, v)|\mathcal{N}(u, v)|.$$

Therefore, we have

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

The left-hand side we can estimate by simply generating many samples from the chain and count the number of occurrences of each hole pattern. Since we know $w(u, v)$, we can then

estimate $w^*(u, v)$. In other words, given weights which are reasonably close to ideal, we can revise these weights arbitrarily close to ideal, i.e., we can compute an arbitrarily close approximation to w^* .

Before presenting the algorithm for generating a random perfect matching, we need to convert from unweighted graphs to weighted graphs. To avoid confusion with the weights w^* we call the weights on edges as *fugacities*. Given a bipartite graph $G = (V_1, V_2, E)$ we view it as a graph with fugacities where for $u \in V_1, v \in V_2$, the edge (u, v) has fugacity

$$\lambda^*(u, v) = \begin{cases} 1 & \text{if } (u, v) \in E \\ 0 & \text{otherwise.} \end{cases}$$

Then a matching M has a fugacity $\lambda(M) = \prod_{e \in M} \lambda(e)$. Similarly, we redefine $w^*(u, v)$ to be over matchings weighted by their fugacity.

Our algorithm starts with a simple graph where we can compute the weights w^* . We then learn the graph along with the weights w^* in a slow manner. Our initial graph is the complete bipartite graph, here we can easily compute w^* . Iteratively, we reduce the fugacities of some non-edge by a factor of $1/2$. We then run the chain with the ideal weights from the previous graph on this new graph. The old ideal weights are sufficiently close to the new ideal weights, so that we can estimate the new ideal weights as described above. We repeat this process until all non-edges have negligible fugacities (fugacities $1/n!$ suffice since there are at most $n!$ perfect matchings in any graph).

References

- [1] M. Jerrum. *Counting, sampling and integrating: algorithms and complexity*. Lectures in Mathematics ETH Zürich. Birkhäuser Verlag, Basel, 2003.
- [2] M. Jerrum and A. Sinclair. Approximating the permanent. *SIAM J. Comput.*, 18(6):1149–1178, 1989.
- [3] M. Jerrum, A. Sinclair, and E. Vigoda. A polynomial-time approximation algorithm for the permanent of a matrix with non-negative entries. In *ACM Symposium on Theory of Computing*, pages 712–721, 2001.