1 Introduction

Last time we reduced the problem of approximating $|\mathcal{M}_n|$ to sampling almost uniformly at random from $\mathcal{M}_{k-1} \cup \mathcal{M}_k$ for k from 2 to $n-1$. Today we shall consider a sampling procedure for $k = n$ and show how sampling for any k can be reduced to this special case. The sampling procedure will be based on a random walk on a regular graph on $\mathcal{M} = \mathcal{M}_{n-1} \cup \mathcal{M}_n$ where we stop after enough steps to be near (i.e. within error δ of) the stationary distribution. From our previous work on random walks we know that this requires $6d^2/\alpha^2(\ln N + \ln 1/\delta)$ steps where $N = |\mathcal{M}|$.

2 The random walk

Starting from the current matching M

- With probability 1/2 stall. Otherwise pick $e = (u, v)$ uniformly at random from G.
- If $M \in \mathcal{M}_n$ and $e \in M$ then delete e.
- If $M \in \mathcal{M}_{n-1}$ and u, v unmatched in M then add e.
- If $M \in \mathcal{M}_{n-1}$, $e' = (u, w) \in M$ and v unmatched then add e and delete e' (rotation.)
- If none of these apply then stall.

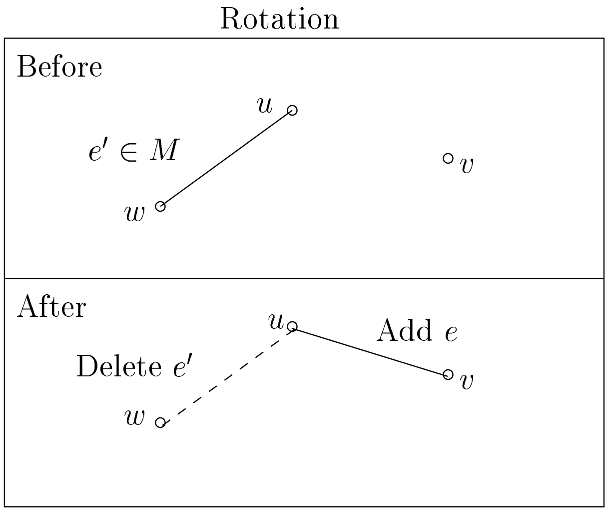

Figure 1

This is a $2m$-regular graph H on N nodes (with self-loops.) Our goal is now to lower-bound $\alpha(H)$ by $1/poly(n)$ so that the number of steps required in the random walk remains small. We start by defining an all-pairs concurrent multi-commodity flow on H. Suppose we can send $\theta = 1/bN$ units concurrently between each pair of H-nodes (where b is $poly(n)$). Then if the optimal concurrent flow value is θ^* we get,

$$\frac{1}{bN} \leq \theta \leq \theta^* \leq \eta = \min_{S \leq N/2} \frac{e(S, \overline{S})}{|S||\overline{S}|} \leq \frac{\alpha(H)}{N/2}$$

$$\Rightarrow \alpha(H) \geq \frac{1}{2b} = \frac{1}{poly(n)}$$

So our aim now is to show that we can send $1/bN$ units of concurrent flow through H. We will in fact show the stronger result that for each $M, M' \in V(H)$ we can send $1/bN$ units of flow through a single path (also called the canonical path) $P_{M,M'}$ by showing that $\leq bN$ of these paths use any single edge.

3 Construction of the paths

From last time we know that every size $n-1$ matching can be transformed into a size n matching by an augmenting path of length ≤ 3 and so from our construction of H we know that every point M in $\mathcal{M}_{n-1}$ is at most 2 steps away from a point M' in $\mathcal{M}_n$. Let $f : \mathcal{M} \to \mathcal{M}_n$ be such that,

$$f(M) = \begin{cases} M' & \text{if } M \in \mathcal{M}_{n-1} \\ M & \textit{otherwise.} \end{cases}$$

Then, we define $P_{M,M'}$ as being $P_{M,f(M)}$ followed by $P_{f(M),f(M')}$ followed by $P_{f(M'),M'}$ where $P_{M,M}$ is the empty path.

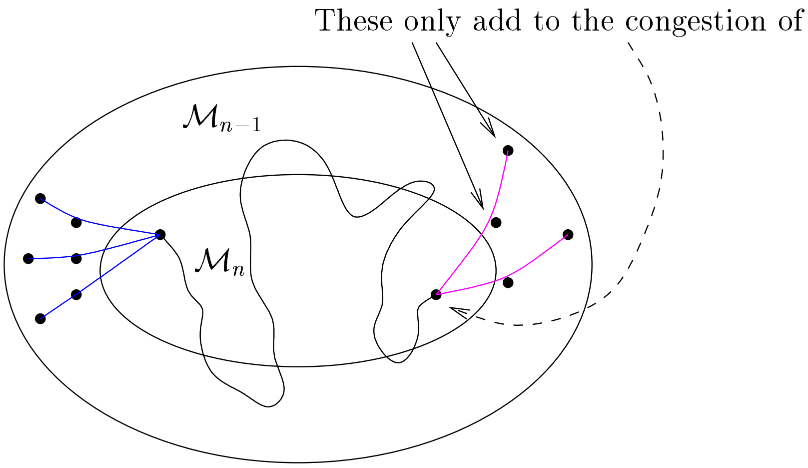

Figure 2

Now to complete the construction of the paths we only need to specify $P_{M,M'}$ where $M, M' \in \mathcal{M}_n$. Let $\Gamma = M \triangle M'$ = a union of node disjoint cycles $C_1, \ldots, C_r$. ($\triangle$ stands for "symmetric difference.")

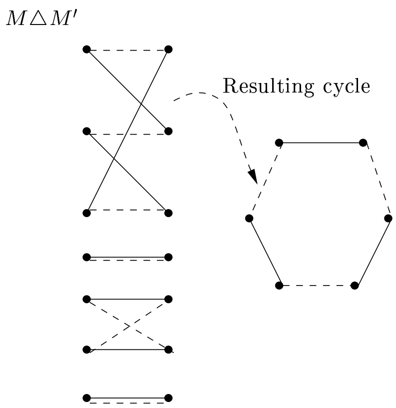

Figure 3

We label all the nodes in G and use this labelling to order the cycles lexicographically. In addition we give each cycle a "start" node i.e. the lowest labelled node in the cycle. For the path $P_{M,M'}$ we "unwind" C_i, for $i =$ 1 to r, as follows:

- delete M-edge out of start node.
- rotate edges in "clockwise" order, where we assume that the first unmatched node $\neq$ start is in clockwise order from start.
- add M'-edge to start node.

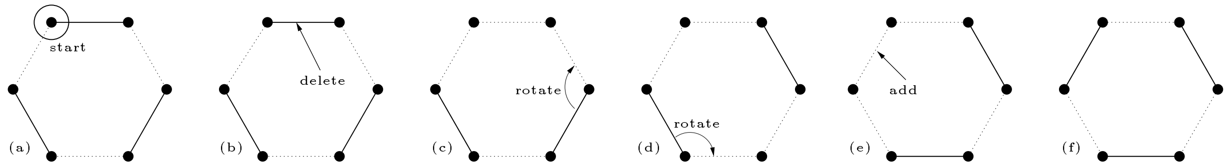

Figure 4

Claim 3.1 *The set of paths* $\{P_{M,M'} | M, M' \in \mathcal{M}_n\}$ *use each* H*-edge* $\leq N$ *times.*

To prove this, focus on an H-edge $f = (L, L')$. Let S_f = all paths $P_{M,M'}$ using f. We will be done if we can show an injective function $g_f : S_f \to \mathcal{M}$. To show that a particular g_f that we construct is injective we need to be able to retrieve M, M' from $f = (L, L')$ and $g_f(P_{M,M'})$. Consider the diagram in figure 5 illustrating this situation. Our encoding $g_f(P_{M,M'})$ should agree with M on $C_1, \ldots, C_{i-1}$ and agree with M' on $C_{i+1}, \ldots, C_r$. If we consider $g_f(P_{M,M'}) = \Gamma \triangle (L \cup L')$ then it does the right thing on $C_1, \ldots, C_{i-1}, C_{i+1}, \ldots, C_r$. On C_i however the function does not give a matching (figure 6).

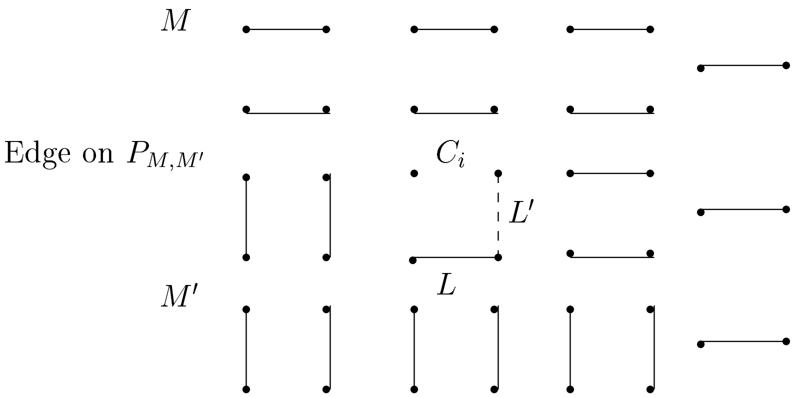

Figure 5

This problem arises only when there are 2 edges incident to the start node in C_i. So in such a case we instead define $g_f(P_{M,M'})$ to be $\Gamma \triangle (L \cup L') - \{M\text{-edge incident to start node}\}$ and this encoding has the right properties.

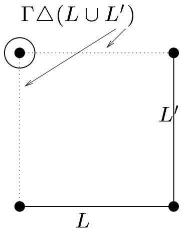

Figure 6

4 Reducing arbitrary k to $k = n$

To sample from $\mathcal{M}_{k-1} \cup \mathcal{M}_k$ in G, we construct G_k as illustrated by this diagram,

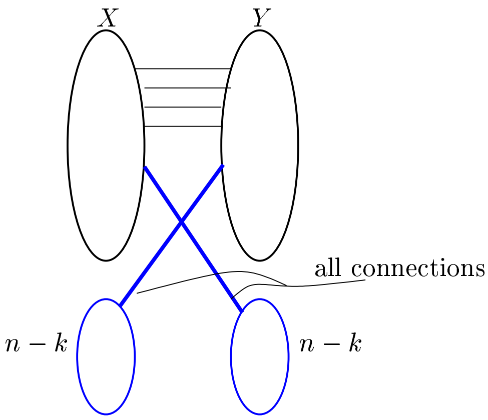

Figure 7

Since $G_2, \ldots, G_{n-1}$ are all dense if G is, we can then employ the procedure used for finding the ratio of perfect matchings to near-perfect matchings on the G_i's.

Claim 4.1 *If we know the ratio of near-perfect matchings to perfect matchings in each of $G_2, \ldots, G_{n-1}$ the we can construct $\frac{|\mathcal{M}_k|}{|\mathcal{M}_{k-1}|}$ for each k, in G.*

5 Congestion $\leq bN$ for all paths

As indicated in figure 2, each of the nodes in $\mathcal{M}_{n-1}$ can be associated with a unique node in $\mathcal{M}_n$ in terms of paths. Also, a node in $\mathcal{M}_n$ is associated with at most n nodes in $\mathcal{M}_{n-1}$ (from last time.) Therefore, when we consider the paths which have a node in $\mathcal{M}_{n-1}$ as at least one of the endpoints, the sections that are defined in terms of $P_{M,M'}$ for $M, M' \in \mathcal{M}_n$ contribute $\leq n^2 N$ to the congestion of any edge. This is because for each path with endpoints solely in $\mathcal{M}_n$ we are adding at most n^2 new paths. In addition for the rest of the edges of these paths (w.l.o.g. assuming the two-step paths have rotation edges followed by adddition edges) we note that at most n two-step nodes can have paths through a particular one-step node and therefore the additional congestion on these edges is $\leq nN$. Therefore, in general, we get $congestion \leq bN$ where b is $poly(n)$ (as required.)