Lecture 12

Flow Encodings:

- **Recall from last time:** If we route commodity $x \to y$ through unique path γ_{xy}, and stationary distn' is uniform, then for the flow to result in polynomial mixing time we need:

$$|\text{paths}(e)| \leq \text{poly}(n) \cdot |\Omega| \quad , \forall e \qquad (*)$$

paths through edge e — natural size of problem

Remarks: a) $(*)$ is still required in the average sense, if we use several paths for each commodity

b) if stationary distn' not uniform, $(*)$ can be suitably reweighted

- **Immediate Execution Issue:** $|\Omega|$ is unknown, and is often what we are after.

  → so need to check $(*)$ implicitly.

  → Idea: charge every element of paths(e) to some point in Ω
  ↳ each point in Ω is charged once!

Def: [flow Encoding]: An encoding for a flow f (that only uses single path γ_{xy} for each commodity $x \to y$) is a set of functions $\{\eta_e : \text{paths}(e) \to \Omega\}_e$ such that:

1. η_e is injective, for all e
2. $\underbrace{\pi(x)\cdot\pi(y) \leq b \cdot \pi(z) \cdot \pi(\eta_e(x,y))}$, $\forall (x,y)$, $\forall e \in \gamma_{xy}$, $e = (z, z')$

   - automatically true with $b = 1$, if stationary distn' is uniform
   - in general, it says that the η_e's are weight-preserving up to a factor b (want this as small as possible)

Remark: Sometimes it is ok if the η_e's are not perfect injections, but we use a little "extra information" to invert them (see proof of next claim).

Claim: If there is an encoding for f as above, then $\rho(f) \leq b \cdot \max_{P(z,z')>0} \frac{1}{P(z,z')}$.

Proof: For arbitrary $e=(z,z')$:

$$f(e) = \sum_{(x,y)\in paths(e)} \pi(x)\pi(y) \leq b \sum_{(x,y)\in paths(e)} \pi(z)\cdot\pi(\eta_e(x,y)) \leq b\cdot\pi(z) = b\,\frac{C(e)}{P(z,z')}.$$

we used the fact that η_e is an injection.
(if η_e becomes an injection w/ a little "extra information" this results in an additional factor in this inequality)

$$\Rightarrow \frac{f(e)}{C(e)} \leq \frac{b}{P(z,z')}. \quad \boxtimes$$

• Example: Lazy RW on $\{0,1\}^n$

Last time: - analyzed RW by splitting flow evenly on all shortest paths between x,y for all (x,y); then appealed to symmetries of the cube to analyze the flow, and we used that $|\Omega| = 2^n$.

This time: Pretend we don't know that $N = |\Omega| = 2^n$, and use flow encodings.

- γ_{xy} : left-to-right bit fixing path (e.g. 010 ↳ 110 ↳ 100 ↳ 101)
- clearly, $\ell(f) = n$
- analyze $\rho(f)$ using flow encodings:
  - → suppose $e = (z,z')$, where z, z' are different at i
  - → let $\gamma_{xy} \ni e$ for some pair x,y
    - clearly x agrees with z in bits $i \dots n$ and y agrees with z in bits $1 \dots i-1$
    - so can define $\eta_e(x,y) = x_1 x_2 \dots x_{i-1} y_i y_{i+1} \cdots y_n$
    - η_e is an injection because given $\eta_e(x,y)$ and z I can invert x,y.
    - also, stationary distn' is uniform, so encoding is weight-preserring with $b=1$

So n_e is a flow encoding, so claim gives:

$$\rho(f) \le \max_{P(z,z')>0} \frac{1}{P(z,z')} = 2n$$

so obtained $\rho(f) \cdot \ell(f) \le 2n^2$ (up to a constant factor this is the same we got last time; but we didn't use the cube's symmetries or size of $|\Omega|$)

- **Example 2: Matchings in Unweighted Undirected Graphs**

[Jerrum + Sinclair '89: Approximating the Permanent, SICOMP 18, 1989]

Input:
- $G = (V, E)$: unweighted, undirected
- parameter $\lambda \geq 1$ (essentially same technology is used for $\lambda < 1$)
- $\Omega = \{$ matchings of $G \}$

Goal: Sample from Gibbs distn' on Ω:

$$\pi(M) = \frac{1}{Z} \lambda^{|M|}$$

#edges in M

$$Z = Z(\lambda) = \sum_k m_k \cdot \lambda^k \quad , m_k = \#\text{matchings with } k \text{ edges}.$$

↳ matching polynomial.

#P-complete for all $\lambda > 0$

- Statistical Physics Motivation: monomer-dimer model

(edges in matching: diatomic molecules
vertices: monatomic molecules)

Markov Chain: → make it lazy

→ use 3 kinds of transitions: edge addition, edge deletion, edge exchange

→ use Metropolis Rule to make sure that stationary distn' matches Gibbs distribution.

MC step if at state M:

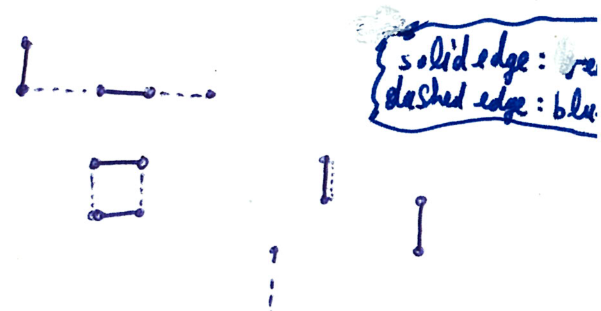
- with prob. $1/2$, stay at M (laziness)
- o.w. choose an edge $e=(u,v) \in E$ u.a.r.
- if u,v unmatched in M, go to $M+e$ (edge addition)
- if $e \in M$, go to $M-e$, w/ probability $1/\lambda$ (edge deletion) and stay at M w/ prob $1-1/\lambda$
- if exactly one of u or v is matched in M, go to $M+e-e'$ (edge exchange) where e' is edge adjacent to u or v
- if both u,v matched, do nothing

Chain follows metropolis rule, hence stationary distn' is Gibbs distn'.

Flow:

- Let x,y be matchings; color x red, y: blue.
- want to specify γ_{xy}
- superimpose x,y: $X+y$
  - ↳ comprises of
    - simple paths of alternating red, blue edges
    - simple cycles of –″–
    - edges that are both red & blue

e.g.



- fix (for our analysis) a total ordering of all simple paths & cycles that are subgraphs of G and designate a "start vertex" in all of them (the start vertex of a path should be an endpoint)
  - ↳ this ordering induces an ordering on the paths and cycles that appear on $x+y$

master index

- define γ_{xy} as follows:

  - process paths & cycles in order specified by master index:
  - to process a path: let $e_1, e_2, \dots, e_f$ be its edges where e_1 is adjacent to start vertex; process it via the following transitions

    i> if e_1 is red, remove it, then exchange e_3 for e_2, exchange e_5 for e_4 etc; if e_f is blue, add e_f in the end.

    ii> if e_1 is blue, exchange e_2 for e_1, e_4 for e_3, etc; if e_f is blue, add it in the end
  - to process a cycle: let $e_1, e_2, \dots, e_f$ be its edges, where e_1 is red edge adjacent to start vertex; process cycle as follows: remove e_1, exchange e_3 for e_2, e_5 for e_4, etc; finally add e_f

- flow encoding: - suppose transition $t = (z, z')$ corresponds to edge exchange; we proceed to define η_t (for other types of transitio[n] the encoding is similar)

  - suppose t involves exchanging e' for e, where $e', e \in E$
  - let $x, y \in \Omega$ be matchings for which the path γ_{xy} uses transition $t = (z, z')$
  - define

$$\eta_t(x,y) = \begin{cases} x \oplus y \oplus (z \cup z') & \text{, if } \gamma_{xy} \text{ uses } t \text{ when processing a path (case 1)} \\ x \oplus y \oplus (z \cup z') \setminus \{e_1\} & \text{, if } \gamma_{xy} \text{ uses } t \text{ when processing a cycle whose first (red) edge is } e_1 \text{ (case 2)} \end{cases}$$

- **Demystifying $\eta_t(x,y)$:**

 - Suppose transition $t = (Z, Z')$ is used in γ_{xy} when processing a path or cycle S. in $x+y$

 - easy to check: ∘ Z, Z' differ only in e, e' (z has e & not e', z' has e' & not e)

 ∘ Z, Z': - all edges that are colored both red and blue in $x+y$ appear in Z, Z'

 - for all connected components of $x+y$ that precede S in the total ordering Z, Z' have only kept the blue edges of these components

 - for all connected components of $x+y$ following S in the total ordering Z, Z' have the red edges of these components

 - with regards to S:

 ∘ Z has kept all blue edges before e, except all red edges after (& including) e

 ∘ Z' has kept all blue edges before (& including) e' and all red edges after e', except

 - So easy to check that $\eta_t(x,y)$ is a matching (removing e_1 in the case that S is a cycle is crucial for this to hold)

- **Now is it an injection?**

 - If we knew whether "$\setminus \{e_1\}$" exists in definition of $\eta_t(x,y)$, we can recover $x \oplus y$ as follows:

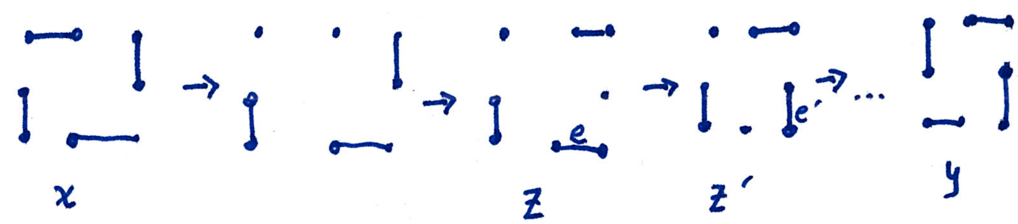
$$x \oplus y = \eta_t(x,y) \oplus (Z \cup Z'), \text{ for case 1}$$

$$x \oplus y = (\eta_t(x,y) \cup \{e_1\}) \oplus (Z \cup Z'), \text{ for case 2}$$

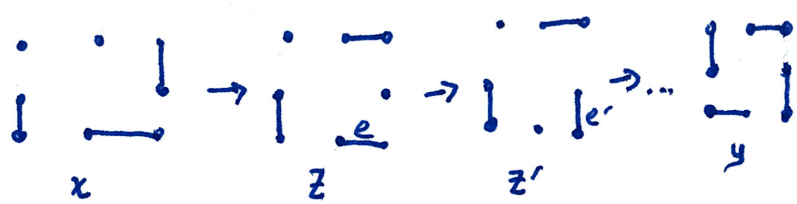
- and if we have $x \oplus y$ we can obtain x and y by looking at $x \oplus y, z, z'$
  - → edges in $z \setminus x \oplus y$ are in both x & y
  - → can identify the order in which the components of $x+y$ where processed by looking at master ordering and $x \oplus y$ (note: the c.c. that need processing in $x+y$ are the c.c.'s of $x \oplus y$)
  - → by looking at z, z' and the connected components of $x \oplus y$ can identify which c.c. was being processed during the transition $t = (z, z')$, which c.c. were processed before, and which after
  - → then $z, z', x \oplus y$ will identify what edges are blue and what are red in these components (see discussion in page 6).
- However, cannot quite get $x \oplus y$ by looking at $\eta_t(x,y), z, z'$

e.g.

scenario 1:



vs scenario 2:



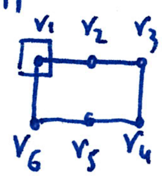
in both cases: $\eta_t(x,y) =$ 

and z, z' are obviously the same.

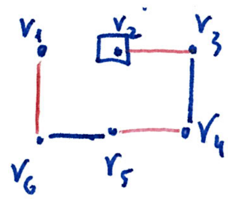
- **Remedy**: → Problem arises because cycle and path that was missing an edge of the cycle (that was adjacent to start vertex) were processed in the same order

  → solution: master index specifies different start vertex for the same path but w/ opposite red/blue edge alternations, if the endpoints of the path are adjacent and the corresponding cycle's start vertex is adjacent to missing edge.

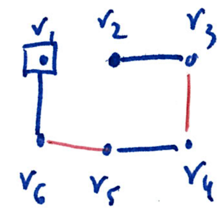
  e.g. suppose v_1 is the start vertex of the cycle $v_1, v_2, v_3, v_4, v_5, v_6$



consider the paths:



and



↑ choose v_2 as start vertex

↑ choose v_1 as start vertex

- **Claim**: If we use the above convention for processing paths & cycles, given $\eta_t(x,y)$, z and z' we can always recover $x \oplus y$ and therefore x and y.

Proof: Can figure out $x \oplus y$ except maybe one edge will be missing, if the current c.c. being processed during transition $t = (z, z')$ is a cycle; but looking at z, z' we can figure out the direction in which the component was being processed and using the convention above we can tell if it was a path or a cycle.

- **weight-preservation?**
  - compare $|x|+|y|$ to $|z| + |\eta_t(x,y)|$
  - not hard to see, that the latter has at most 2 fewer edges (possible edge deletion at beginning of cycle, & edge missing from current transition).

$$\Rightarrow \quad \pi(x)\cdot\pi(y) \leq \lambda^2 \pi(z)\cdot \pi(\eta_t(x,y)) \qquad (*)$$

- analysis of τ_{mix}: for any transitition $t=(z,z')$, $P(z,z') \geq \frac{1}{2\lambda|E|}$ $(**)$

$$\overset{(*)}{\underset{(**)}{\Rightarrow}} \quad \rho(f) \leq \lambda^2 \cdot \left(\frac{1}{2\lambda|E|}\right)^{-1} = O(\lambda^3 \cdot |E|).$$

also clearly $\ell(f) \leq |V|$ (since $|x| \leq \frac{|V|}{2}$, $|y| \leq \frac{|V|}{2}$ and every vertex in $x \oplus y$ is processed once)

$$\Rightarrow \quad \alpha \geq \frac{1}{\rho(f)\cdot \ell(f)} \geq \Omega\left(\frac{1}{\lambda^3 \cdot |E|\cdot|V|}\right).$$

$$\Rightarrow \quad \tau_x(\varepsilon) \leq O\left(\lambda^3 \cdot |E| \cdot |V| \cdot \left(\log \pi(x)^{-1} + 2\log(1/\varepsilon)\right)\right)$$

- starting x?

  start from a max-weight matching (can be found efficiently)

$$\pi(x) \geq \frac{1}{|\Omega|} \geq \frac{1}{2^{|E|}}$$

$$\Rightarrow \quad \tau_x(\varepsilon) = O(\lambda^3 |E|^2 \cdot |V|).$$

exercise (1pt): Improve bound to $O(\lambda^2 \cdot |E|^2 \cdot |V|)$.