Flow Encodings:

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

  \[ |\text{paths}(e)| \leq \text{poly}(n) \cdot |\mathcal{E}|, \forall e \]  

# paths through edge  
/ \natural \text{ size of problem}

Remarks: $\Rightarrow (\ast)$ is still required in the average sense, if we use several paths for each commodity.

$\Rightarrow$ if stationary distr' not uniform, $(\ast)$ can be suitably reweighted.

Immediate Execution Issue: $|\mathcal{E}|$ is unknown, and is often what we are after.

$\Rightarrow$ so need to check $(\ast)$ implicitly.

$\Rightarrow$ Idea: charge every element of paths(e) to some point in $\mathcal{E}$, each point in $\mathcal{E}$ is charged once!

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

1. $\eta_e$ is injective, for all $e$

2. $\pi(x) \cdot \pi(y) \leq \mathbb{E} \cdot \pi(z) \cdot \pi(\eta_e(x,y))$, $\forall (x,y), \forall e \in \mathcal{E}$, $e \in (2,2)$

- automatically true with $\mathbb{E} = 1$, if stationary distr' is uniform
- in general, it says that the $\eta_e$'s are weight-preserving up to a factor $\mathbb{E}$ (we want this as small as possible)

Remark: Sometimes it is OK if the $\eta_e$'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 \( f \) as above, then \( p(f) \leq b \cdot \max_{P(z,z')} \frac{1}{P(z,z')} \).

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

\[
f(e) = \sum_{(x,y) \in \text{paths}(z)} \pi(x) \pi(y) \leq \sum_{(x,y) \in \text{paths}(z)} \pi(x) \pi(e(x,y)) \leq \pi(z) = \frac{P(z)}{P(z, z')}.
\]

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

Example: Lazy RW on \([0,4]^n\)

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 \( |S| = 2^n \).

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

- \( g_{xy} \): left-to-right bit fixing path (e.g. 010

\[
\text{L} \rightarrow 110 \quad \text{L} \rightarrow 100 \rightarrow 101
\]

- Clearly, \( \ell(f) = n \)

- Analyze \( p(f) \) using flow encodings:
  
  - Suppose \( e = (z, z') \), where \( z, z' \) are different at \( i \)
  
  - Let \( g_{xy} \) be for some pair \( x,y \)
    
    - Clearly \( x \) agrees with \( z \) in bits \( i \ldots n \) and \( y \) agrees with \( z \) in bits \( 1 \ldots i-1 \)
    
    - So can define \( e(x,y) = x_1 x_2 \ldots x_{i-1} y_i y_{i+1} \ldots y_n \)
    
    - \( e \) is an injection because given \( e(x,y) \) and \( z \)
      
      I can invert \( x,y \).

    - Also, stationary distn is uniform, so encoding is weight-preserving with \( b = 1 \).
So $x_2$ is a flow encoding, so claim gives:

$$f(t) \leq \max_{P(2,2) \geq 0} \frac{1}{P(2,2)} = 2n$$

So obtained $g(t) \cdot \ell(t) \leq 2n^2$ (up to a constant factor.

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 $\lambda < 1$)
- $\Omega = \{m\text{ matchings of } G\}$

**Goal:** Sample from Gibbs dist'n on $\Omega$:

$$\pi(M) = \frac{1}{Z} \lambda^{\text{#edges in } M}$$

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

$\lambda$-matching polynomial.

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

- Statistical Physics Motivation: monomer-dimer model
  - (edges in matching: diatomic molecules)
  - (vertices: monatomic molecules)

Marker Chain:
- make it lazy
  - use 3 kinds of transitions: edge addition, edge deletion, edge exchange
  - use Metropolis Rule to make sure that stationary dist'n 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 $M$, goto $M+e$ (edge addition)
- if $e \in M$, go to $M-e$, w/ probability $1/2$ (edge deletion)
  and stay at $M$ w/ prob 1/2
- 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 $S_{xy}$
  - superimpose $x,y : X+Y$
    
    $L$ comprises of simple paths of alternating red, blue edges
    - simple cycles of
    - edges that are both red & blue
      
      e.g. $I \quad I \quad I$

      solid edge: red
      dashed edge: blue

  - 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 $X+Y$
- define \( f_{xy} \) as follows:

  - process paths & cycles in order specified by master index:

  - to process a path: let \( e_1, e_2, \ldots, e_k \) 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 \( e_4 \) is blue, add \( e_4 \) in the end.

    ii> if \( e_1 \) is blue, exchange \( e_2 \) for \( e_1 \), \( e_4 \) for \( e_3 \), etc, if \( e_3 \) is blue, add it in the end.

  - to process a cycle: let \( e_1, e_2, \ldots, e_k \) 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 \( e_4 \).

- flow encodings: suppose transition \( t = (z, z') \) corresponds to edge exchange; we proceed to define \( n_t \) (for other types of transitions the encoding is similar)

  - suppose \( t \) involves exchanging \( e' \) for \( e \), where \( e, e' \in E \)

  - let \( x, y \in \mathcal{X} \) be matchings for which the path \( f_{xy} \) uses transition \( t = (z, z') \)

  - define

    \[
    n_t(x, y) = \begin{cases} 
    x \oplus y \oplus (z \cup z') & \text{if } f_{xy} \text{ uses } t \text{ when processing a path} \\
    x \oplus y \oplus (z \cup z') \setminus \{e_1\} & \text{if } f_{xy} \text{ uses } t \text{ when processing a cycle whose first (red)edge is } e_1 
    \end{cases}
    \]

    (case 1)
- Demystifying $\eta_t(x,y)$:

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

- Easy to check: $z,z'$ differ only in $e,e'$ ($z$ has $e$ and not $e'$)
  $z',z$ have 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 $e$.
  $z'$ has kept all blue edges before $e'$, and all red edges after $e'$, except $e'$. (removing $e'$ in the case that $S'$ is a cycle is crucial for this to hold.)

- So easy to check that $\eta_t(x,y)$ is a matching.

- Now is it an injection? If we knew whether "$\setminus e,3"$ exists in definition of $\eta_t(x,y)$, we can recover $x@y$ as follows:
  $x@y = \eta_t(x,y) \oplus (z \cup z')$, for case 1.
  $x@y = (\eta_t(x,y) \cup \{e\}) \oplus (z \cup z')$, for case 2.
- and if we have $x \bowtie y$ we can obtain $x$ and $y$ by looking at $x \bowtie y, z, z'$

- edges in $z \setminus x \bowtie y$ are in both $x \bowtie y$
- can identify the order in which the components of $x \bowtie y$ were processed by looking at master ordering and $x \bowtie y$ (note: the c.c. that needs processing in $x \bowtie y$ are in $z \setminus x \bowtie y$)
- by looking at $z, z'$ and the connected components of $x \bowtie y$ can identify which c.c. was being processed during the transition $t = (z, z')$, which c.c. were processed before, and which after
- then $z, z'$, $x \bowtie y$ will identify what edges are blue and what are red in those components (see discussion in page 6).

However, cannot quite get $x \bowtie y$ by looking at $\eta_e(x, y), z, z'$

E.g.

**Scenario 1:**

<p>| | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>$x$</td>
<td>---</td>
<td>$z$</td>
<td>$z'$</td>
<td>$y$</td>
<td></td>
</tr>
</tbody>
</table>

**vs Scenario 2:**

<p>| | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>$x$</td>
<td>---</td>
<td>$z$</td>
<td>$z'$</td>
<td>$y$</td>
<td></td>
</tr>
</tbody>
</table>

in both cases: $\eta_e(x, y) = \begin{array}{c} 1 \\ \hline 1 \end{array}$

and $z, z'$ are obviously the same.
- **Remedy**: Problem arises because cycle and path that was missing an edge of the cycle were processed in the same order (that was adjacent to start vertex).

- **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 $V_2$ is the start vertex of the cycle $V_2, V_3, V_4, V_5,

  \[ V_6 \]

  Consider the paths:

  \[ V_6 \rightarrow V_3 \rightarrow V_4 \text{ and } V_6 \rightarrow V_5 \rightarrow V_4 \]

  \[ \uparrow \text{ choose } V_2 \text{ as start vertex} \quad \uparrow \text{ choose } V_1 \text{ as start vertex} \]

- **Claim**: If we use the above convention for processing paths & cycles, given $N_x(x,y)$, $z$ 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, 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.
- \textbf{weight-preservation?}

- compare $|x| + |y|$ to $|z| + |n_e(x,y)|$

- not hard to see, that the latter has at most 2 fewer edges (possible edge deletion at beginning of cycle, a edge missing from current transition).

\[ \Rightarrow \pi(x) \cdot \pi(y) \leq \lambda^2 \pi(z) \cdot \pi(n_e(x,y)) \quad (\star) \]

- analysis of $\tau_{\text{mix}}$: for any transition $t = (z, z')$, $P(z, z') \geq \frac{1}{2 |E|}$ \quad (\star\star)

\[ \Rightarrow p(t) \leq \lambda^2 \cdot \left( \frac{1}{2 |E|} \right)^{-1} = O(\lambda^2 \cdot |E|). \]

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

\[ \Rightarrow \ell \geq \frac{1}{p(t) \cdot \ell(f)} \geq \Omega \left( \frac{1}{\lambda^3 |E| \cdot |v|} \right). \]

\[ \Rightarrow \tau_{\text{x}}(x) \leq O \left( \lambda^3 |E| \cdot |v| \cdot \left( \log \pi(x)^{-1} + 2 \log |v| \right) \right). \]

- \textbf{starting $x$?}

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

\[ \pi(x) \geq \frac{1}{|E|} \geq \frac{1}{2 |E|} \]

\[ \Rightarrow \tau_{\text{x}}^k(x) = O \left( \lambda^3 |E|^2 \cdot |v| \right). \]

\textit{exercise (1pt): Impose bound to $O(\lambda^2 |E|^2 \cdot |v|)$.}