Random walks on undirected graphs

$G=(V,E)$ undirected graph

consider simple random walk on this graph:

$p_i = \frac{1}{d_i}, \quad \forall (i,j) \in E$

[$MC$ is periodic iff graph is bipartite $\quad \text{but so, consider lazy r.w.}$]

$h_i = \frac{d_i}{2m}, \quad m: \# \text{ of edges}$

$\sum di \leq 1 \Rightarrow \frac{d_i}{am} = \frac{1}{ai}, \quad (i,j) \in E$

Some key quantities:

<table>
<thead>
<tr>
<th>Hitting time</th>
<th>$h_{ij} = E(T_{ij})$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Commute time</td>
<td>$c_{ij} = h_{ij} + h_{ji}$</td>
</tr>
<tr>
<td>Covertime</td>
<td>$C(t) = \exp(T_{ij})$</td>
</tr>
</tbody>
</table>

Lemma $\forall (i,j), \quad h_{ij} + h_{ji} \leq am$
Consider corresponding random walk on directed edges (states are $i, j$, $V(i) \in E$) and states

Transition probabilities:

$$ Q_{ij} = \begin{cases} \frac{1}{d_i}, & i = j \\ 0, & \text{otherwise} \end{cases} $$

$$ Q = \begin{pmatrix} q_{11} & \cdots & q_{1n} \\ \vdots & \ddots & \vdots \\ q_{n1} & \cdots & q_{nn} \end{pmatrix} $$ is doubly stochastic.

$$ \sum_{i \to j} Q_{ij} = \sum_{k \in V(i), (k, j) \in E} q_{kj} = \sum_{k \in V(i), (i, k) \in E} \frac{1}{d_i} = 1 $$

$$ \Rightarrow \pi^T \text{ is uniform, i.e. } \pi_j = \frac{1}{\lambda m} \forall e $$

$$ \Rightarrow h_{i \to j, i \to j} = 2m $$
Corollary

\( \forall G = (V,E) \)

\( C(G) \) = expected cover time \( \leq 3m(n-1) \)

**Proof:**

Let \( T \) be a spanning tree on \( G \) and let \( e \), \( e' \) be two edges in the tree.

Consider duplicated tree \( T^K \), where each edge is duplicated once in each direction.

Every vertex has indegree = outdegree = 1.

Every vertex has indegree = outdegree = 1.

\( E(\text{cover time}) \leq \sum_i E(T_{e_i,v_i} + T_{e_i,v_i}) = (n-1)2m \)

**Examples**

\[ C(G) = \Theta(n^2) \]

**Fact:** Starting from \( v_1 \), \( \Pr(\text{reach } v_0 \text{ before returning to } v_0) = \frac{1}{2} \)
Application: s-t connectivity

Given undirected graph \( G \) determine if \( s \) and \( t \) are in same CC.

DFS \( \{ O(m) \text{ time}, O(n) \text{ space} \) keep track of all vertices searched so far.

BFS \( \{ O(n) \text{ space} \) no exploration at all.

Observation: very simple randomized alg using log space
(input on separate read-only tape)

Simulate r.w. of length \( 4mn \) on \( G \) starting from \( s \)
\[ Pr( \text{r.w. doesn't reach } t \text{ when } \exists \text{ path}) \leq \frac{1}{2} \]

Both algorithms \( O(mn) \) space-time
Can we interpolate?
Monte Carlo Methods

A collection of tools for estimating values through sampling and estimation.

\((\varepsilon, \delta)\) Approximation

A randomized alg gives an \((\varepsilon, \delta)\) approx for value \(V\) if the output \(X\) of the alg satisfies

\[ \Pr(|X - V| > \varepsilon V) \leq \delta \]

Example

Sample independent random vars whose mean is quantity we want to estimate

Let \(X_1, X_2, \ldots, X_m\) iid. Bernoulli with \(E(X_i) = p\)

If \(m > \frac{3 \ln(\frac{1}{\delta})}{\varepsilon^2 p}\) then

\[ \Pr\left( \frac{1}{m} \sum_{i=1}^{m} X_i - p \geq \varepsilon p \right) \leq \delta \]

DNF Counting

Suppose want to know \# satisfying assignments

\((x_1 \land \neg x_2) \lor (x_3 \land x_4) \lor (x_1 \land \neg x_2 \land \neg x_3 \land x_4) \lor (x_3 \land \neg x_4)\)

Obviously satisfying such a formula is easy

\(\#\) counting \# satisfying assignments is hard

If could do this, could solve 3-SAT

Problem actually \#P-complete \(\rightarrow\) strong intractability

\#P is counting analogue of NP

problems of form:

compute \(f(x)\) where \(f(x)\) is \# solutions to problem \(x\) in NP
Approximate DNF Counting

Obvious approach: sample random assignments, indep m times
\[ X_i = \begin{cases} 1 & \text{if random assignment satisfies } \varphi \\ 0 & \text{otherwise} \end{cases} \]

return \[ \frac{\sum X_i}{m} \cdot 2^n \] as estimate

\[ \exists \text{ FPRAS for DNF counting} \]

Fully polynomial randomized approx scheme

aka: a randomized alg for which, given an input \( x \) and any parameters \( \epsilon, \delta \) with \( 0 < \epsilon, \delta < 1 \), the alg outputs an \( (\epsilon, \delta) \) approx to \( V(x) \) in time poly in \( \frac{1}{\epsilon}, \frac{1}{\delta} \) & size of input

\[ \overline{V} \text{ is } (\epsilon, \delta) \text{ approx to } V \equiv \Pr((\overline{V} - V) > \epsilon V) \leq \delta \]
DNF counting illustrates fundamental connection
sampling \leftrightarrow \text{counting}

From approx sampling \rightarrow approx counting

**Counting Independent Sets**

**Input:** $X, \varepsilon$ 

**FPAUS**
FPTAS
Fully polynomial time approximation scheme

\[ \varepsilon \text{ uniform sample } w \text{ from } \mathcal{L}(x) \]
\[ \Pr(w) = \frac{|I|}{|X|} \leq \varepsilon \]

\[ \text{runs in time polynomial in size of } x \]
\[ \& \ln(\varepsilon) \]

Want to show:

**Given** FPAUS for independent sets $\implies$ **construct** FPRAS for ISS

**FPRAS**

\[ \text{polynomial } \]
\[ \text{algorithm to } \]
\[ \text{FPAUS}(x, \varepsilon) \]
\[ \text{runs in time } \]
\[ \text{poly in } \varepsilon, \text{ size of input} \]

$G = (V, E)$ arbitrary ordering of edges

$E_i = \{e_i, e_2, \ldots, e_m \}$

$G_i = (V, E_i)$

$G_m = G$

$G_0 = (V, E)$

$\mathcal{L}(G_i)$: set of ISSs in $G_i$

\[ |\mathcal{L}(G)| = \frac{|\mathcal{L}(G_1)|}{|\mathcal{L}(G_m)|} \times \frac{|\mathcal{L}(G_2)|}{|\mathcal{L}(G_m)|} \times \ldots \times \frac{|\mathcal{L}(G_i)|}{|\mathcal{L}(G_m)|} \times \frac{|\mathcal{L}(G)|}{|\mathcal{L}(G)|} \]
So need to bound \( R = \frac{m}{\sum_{i} \frac{c_i}{n_i}} \).

**Claim:** For \( R \), is \( \left( \frac{\sum_{i} c_i / n_i}{m} \right) \) approx to \( R \), \( \forall i \\
then \( X \) is \( (\varepsilon, \delta) \) approx to \( \mathcal{L}(G) \).

\begin{align*}
\Pr & \left( \left( \frac{n_{i+1}}{n_{i+1}} \right) > \frac{1}{m} \right) \rightarrow
\Pr & \left( \left( \frac{r_{i+1}}{r_{i+1} + r_{i+2}} \right) > \frac{1}{m} \right) \\
& \Rightarrow \left( \frac{r_{i+1}}{r_{i+1} + r_{i+2}} \right) \approx \frac{1}{m} \\
& \Rightarrow \left( \frac{r_{i+1}}{r_{i+1} + r_{i+2}} \right) \approx \frac{1}{m} \\
& \Rightarrow \left( \frac{r_{i+1}}{r_{i+1} + r_{i+2}} \right) \approx \frac{1}{m} \\
& \Rightarrow \left( \frac{r_{i+1}}{r_{i+1} + r_{i+2}} \right) \approx \frac{1}{m} \\
& \Rightarrow \Pr \left( \frac{r_{i+1}}{r_{i+1} + r_{i+2}} > \frac{1}{m} \right) \leq \delta
\end{align*}

To get \( R \), \( \left( \frac{\sum_{i} c_i / n_i}{m} \right) \)-approx for \( r \), use \( \text{FPANS} \) for \( D\delta \).

**Idea:** (approx) sample indep sets in \( G_{i+1} \) & compute fraction \( j \) those that are indep in \( G_i \):

\[
\text{FPANS} \quad \text{FPANS algorithm with uniform sampler} \\
\text{Input:} \quad G_{i+1} \quad \text{Output:} \quad \frac{1}{m} \text{ uniform sample w/ from } \mathcal{L}(G_{i+1})
\]

For this to work, need \( r_i = \frac{\lambda(G_i)}{|\mathcal{L}(G_i)|} \) not too small (no results on average)

**Claim:** \( r_i \geq \frac{1}{2} \)

\( \Rightarrow \) with polynomially many calls to \( \text{FPANS} \), we can get good approx \( R \) to \( r \) using Monte Carlo. Thm
Two errors we need to bound:

1. \( \text{FPAUS} \neq \text{exact sampler} \) so exp value \( \neq r \):
   \[
   \text{But using, say, } \frac{e}{6m} \text{ sampler } |E(F_i) - r_i| \leq \frac{e}{6m}
   \]

2. With samples we get approx to \( E(F_i) \):
   since \( r_i \) big \( (\geq \frac{5}{6}) \), \( E(F_i) \) big \( \Rightarrow \)
   Monte Carlo Thm \( \Rightarrow \) \( O \left( \frac{m^2}{e^2} \ln \left( \frac{1}{\delta} \right) \right) \) samples suffice

**Thm:** Given a FPAUS for sampling ISSs, one can construct a FPRAS for ISSs

Approach works for many "self-reducible" problems.

Another example: counting # matchings in a graph

Again: \( E = \{e_1, \ldots, e_m\} \)

\( G_i = (V, E_i) \) where \( E_i = \{e_1, \ldots, e_i\} \)

\[
|M(G)| = \frac{|M(G)_{e_1}| |M(G)_{e_2}| \ldots |M(G)_{e_i}|}{|M(G)_{e_1}| |M(G)_{e_2}| \ldots |M(G)|}
\]

Like before: \( \frac{|M(G)|}{|M(G)_{e_i}|} \geq \frac{1}{2} \)

Big question remains: how to construct approx samplers?
How to sample sets from a universe $U$ according to some dist $\Pi$?

Cool idea! Design MC whose state space is $U$ that has stationary dist $\Pi$.
- Simulate MC until it "mixes".
- Use state at that time as sample.

2 key questions:
1. how to design chain with right $\Pi$?
2. how to bound mixing time?

Example: Sampling indep sets uniformly from $G=(V,E)$

States: indep sets
$X_k$: some indep set

MC: in each step choose vertex $v$ u.a.r. from $V$
  - if $v \notin X_k$ then $X_{t+1} = X_t \setminus \{v\}$
  - if $v \notin X_k$ & can be added w/o violating indep then $X_{t+1} = X_t \cup \{v\}$
  - otherwise $X_{t+1} = X_t$