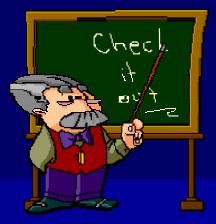
Randomized Algorithms

The Monte Carlo Method

Speaker: Chuang-Chieh Lin
Advisor: Professor Maw-Shang Chang
National Chung Cheng University

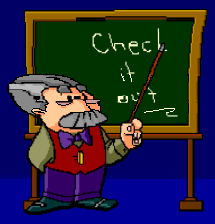
2006/7/17

References



- Professor S. C. Tsai's slides.
- [MR95] *Randomized Algorithms*, Rajeev Motwani and Prabhakar Raghavan.
- [MU05] *Probability and Computing - Randomized Algorithms and Probabilistic Analysis*, Michael Mitzenmacher and Eli Upfal.
- Wikipedia – The Free Encyclopedia

Outline



- Introduction
  - The Monte Carlo Method
  - PRAS and FPRAS
- The DNF Counting Problem
- DNF counting algorithms
  - A Naïve Approach
  - A FPRAS
  - Approximate Counting from FPAUS

Introduction

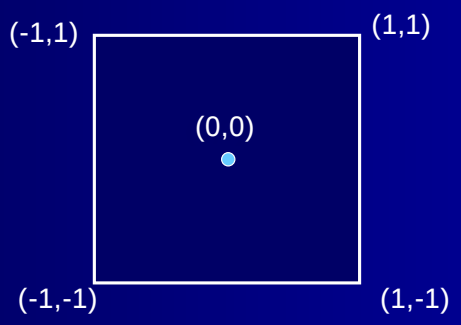
- *The Monte Carlo method* refers to a collection of tools for estimating values through sampling and simulation.
- Monte Carlo techniques are used extensively in almost all areas of physical sciences and engineering.

Introduction (cont'd)

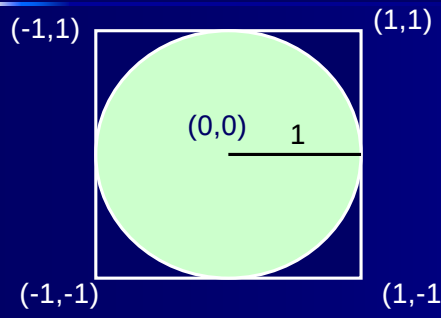
- Let us first consider the following approach for estimating the value of the constant π.

Estimating π

Let (X, Y) be a point chosen uniformly at random in a 2×2 square centered at the origin (0,0).



Estimating π (cont'd)



If we let

$$Z = \begin{cases} 1 & \text{if } \sqrt{X^2 + Y^2} \leq 1, \\ 0 & \text{otherwise.} \end{cases}$$

The probability that $Z = 1$ is exactly the ratio of the area of the circle To the area of the square. Hence,

$$\mathbf{Pr}[Z = 1] = \pi/4.$$

Estimating π (cont'd)

- Assume we run this experiment *m* times, with Z_i being the value of Z at the *i*th run.

  If $W = \sum_{i=1}^{m} Z_i$, then

$$\mathbf{E}[W] = \mathbf{E}[\sum_{i=1}^{m} Z_i] = \sum_{i=1}^{m} \mathbf{E}[Z_i] = \frac{m\pi}{4}.$$

  Hence $W' = (4/m)W$ is a natural estimate for π.

Estimating π (cont'd)

- Applying the Chernoff bound, we have

$$\begin{aligned}\mathbf{Pr}[|W' - \pi| \geq \varepsilon\pi] &= \mathbf{Pr}[|W - \frac{m\pi}{4}| \geq \frac{\varepsilon m\pi}{4}] \\ &= \mathbf{Pr}[|W - \mathbf{E}[W]| \geq \varepsilon\mathbf{E}[W]] \\ &\leq 2e^{-m\pi\varepsilon^2/12}.\end{aligned}$$

- Therefore, by using a sufficiently large number of samples we can obtain, with high probability, as tight an approximation of π as we wish.

(ε, Δ)-approximation randomized algorithm

- <u>Definition:</u>
  A randomized algorithm gives an (ε, Δ)-approximation for the value V if the output X of the algorithm satisfies

$$\mathbf{Pr}[|X - V| \leq \varepsilon V] \geq 1 - \Delta.$$

- The above method for estimating π gives an (ε, Δ)-approximation, as long as $\varepsilon < 1$ and m large enough.

$$2e^{-m\pi\varepsilon^2/12} \le \Delta \;\Rightarrow\; m \ge \frac{12\ln(2/\Delta)}{\pi\varepsilon^2}.$$

- We may generalize the idea behind our technique for estimating π to provide a relation between the number of samples and the quality of the approximation.

- We use the following simple application of the Chernoff bound throughout our discussing.

Theorem 1

Let $X_1, \ldots, X_m$ be independent and identically distributed indicator random variables, with $\mu = \mathbf{E}[X_i]$. If $m \geq (3\ln(2/\Delta))/\varepsilon^2\mu$, then

$$\mathbf{Pr}[|\frac{1}{m}\sum_{i=1}^{m} X_i - \mu| \geq \varepsilon\mu] \leq \Delta.$$

That is, m samples provide an (ε, Δ)-approximation for μ.

Proof: Exercise!

Approximation Schemes

- There are problems for which the existence of an efficient (polynomial time) algorithm that gives an exact answer would imply that **P** = **NP**.
- Hence it is unlikely that such an algorithm will be found.
- So we eye on approximation algorithms instead.

Approximation Schemes (cont'd)

- For approximation algorithms, there are some important approximation schemes as follows.
  - Polynomial time approximation schemes (PTAS)
  - Fully polynomial time approximation schemes (FPTAS)
  - Polynomial randomized approximation schemes (PRAS)
  - Fully polynomial randomized approximation schemes (FPRAS)
  - • • •

We will focus on this scheme in this talk.

Notes...

- Here we are considering *couting* problems that map inputs x to values $V(x)$.
- For example, given an graph, we might want to know an approximation to the number of independent sets in the graph.

PRAS

A PRAS for a problem is a randomized algorithm for which, given an input x and any parameters ε and Δ with $0 < \varepsilon, \Delta < 1$, the algorithm outputs an (ε, Δ)-approximation to $V(x)$ in time $\text{poly}(|x|)$.

So, what is FPRAS?

FPRAS

A FPRAS for a problem is a randomized algorithm for which, given an input x and any parameters ε and Δ with $0 < \varepsilon, \Delta < 1$, the algorithm outputs an (ε, Δ)-approximation to $V(x)$ in time $\mathrm{poly}(|x|, 1/\varepsilon, \ln(1/\Delta))$.

The DNF Counting Problem

- Let us consider the problem of counting the number of satisfying assignments of a Boolean formula in disjunctive normal form (DNF).

The DNF Counting Problem (cont'd)

- Definition: a DNF formula is a disjunction of clauses $C_1 \vee C_2 \vee \ldots \vee C_t$, where each clause is a conjunction of literals.

- For example, the following is a DNF formula:

$$(X_1 \wedge \overline{X}_2 \wedge X_3) \vee (X_2 \wedge X_4) \vee (\overline{X}_1 \wedge X_3 \wedge X_4)$$

The DNF Counting Problem (cont'd)

- Counting the number of satisfying assignments of a DNF formula is actually #**P**-complete (pronounced "sharp-P").

  What is #**P**?

#P

- A problem is in the class #**P** if there is a polynomial time, nondeterministic Turing machine such that, for any input *I*, the number of accepting computations equals the number of different solutions associated with the input *I*.

- Clearly, a #**P** problem must be at least hard as the corresponding **NP** problem.

#P-complete

- A problem is #**P**-complete if and only if it is in #**P**, and every problem in #**P** can be reduced to it in polynomial time.

- Counting the number of Hamiltonian cycles in a graph and counting the number of perfect matching in a bipartite graph are examples of #**P**-complete problems.

How Hard Is the DNF Counting Problem?

- Given any CNF formula H, we can apply de Morgans law to obtain a DNF formula for $\overline{H}$, the negation of the formula H, with the same number of variables and clauses.

$$(\overline{X}_1 \vee X_2 \vee \overline{X}_3) \wedge (\overline{X}_2 \vee \overline{X}_4) \wedge (X_1 \vee \overline{X}_3 \vee \overline{X}_4)$$

$$\overset{\text{Negation}}{\Rightarrow} \quad (X_1 \wedge \overline{X}_2 \wedge X_3) \vee (X_2 \wedge X_4) \vee (\overline{X}_1 \wedge X_3 \wedge X_4)$$

- The formula H has a satisfying assignment $\Leftrightarrow$ the number of satisfying assignments of $\overline{H}$ is less than 2^n. (assume that H has n variables)

How Hard Is the DNF Counting Problem? (cont'd)

- So the DNF counting problem is at least hard as solving the **NP**-complete problem SAT.
- Thus the DNF counting problem is actually a #**P**-complete problem.
  - Why?

How Hard Is the DNF Counting Problem? (cont'd)

- It is unlikely that there is a polynomial time algorithm that computes the exact number of solutions of a #**P**-complete problem.
  - Such an algorithm would imply that **P** = **NP**.
- It is therefore interesting to find an approximation scheme, such as FPRAS, for the number of satisfying assignments of a DNF formula.

A Naïve Algorithm

- Let $c(F)$ be the number of satisfying assignments of a DNF formula F.
  - Here we assume that $c(F) > 0$, since it is easy to check whether $c(F) = 0$ before running our sampling algorithm.

DNF counting algorithm I

Input: A DNF formula F with n variables
Output: Y = an approximation of $c(F)$

- $X \leftarrow 0$.
- **For** $k = 1$ to m, **do**:
  - Generate a random assignment for the n variables, chosen uniformly at random from all 2^n possible assignments.
  - **If** the random assignment satisfies F, **then** $X \leftarrow X + 1$.
- **Return** $Y \leftarrow (X/m)2^n$.

Analysis

- Let $X_k = \begin{cases} 1 & \text{If the } k\text{th iteration in the algorithm} \\ & \text{generated a satisfying assignment;} \\ 0 & \text{otherwise.} \end{cases}$
- Thus $\mathbf{Pr}[X_k = 1] = c(F)/2^n$.
- Let $X = \sum_{k=1}^{m} X_k$, then $\mathbf{E}[X] = m \cdot c(F)/2^n$.
- Hence,

$$\mathbf{E}[Y] = \frac{\mathbf{E}[X] \cdot 2^n}{m} = c(F).$$

Analysis (cont'd)

- By Theorem 1, X/m gives an (ε, Δ)-approximation of $c(F)/2^n$, and hence Y gives an (ε, Δ)-approximation of $c(F)$, when
$$m \geq \frac{3 \cdot 2^n \ln(2/\Delta))}{\varepsilon^2 c(F)}.$$
- If $c(F) \geq 2^n/\alpha(n)$ for some polynomial α, then we will obtain that m is polynomial in $n, 1/\varepsilon,$ and $\ln(1/\Delta)$.
- But, if $c(F) = \text{poly}(n)$, then $m = O(2^n/c(F))$, which is not (always) polynomial!

Analysis (cont'd)

- What is the problem?
- The problem with this sampling approach is that the set of satisfying assignments might not be sufficiently dense in the set of all assignments.

Revising Algorithm I

- We now revise the naïve algorithm to obtain a FPRAS.
- Let a DNF formula F = $C_1 \vee C_2 \vee \ldots \vee C_t$.
  - Assume WLOG that no clause includes a variable and its negation.
- If clause C_i has l_i literals, then there are exactly 2^{n-l_i} satisfying assignments for C_i.

Revising Algorithm I (cont'd)

- Let SC_i denote the set of assignments that satisfy clause i and let $U = \{(i, a): 1 \leq i \leq t \text{ and } a \in SC_i\}$.
- Notice that $|U| = \sum_{i=1}^{t} |SC_i|$.
- The value that we want to estimate is $c(F) = |\bigcup_{i=1}^{t} SC_i|$. Hence $c(F) \leq |U|$, since an assignment can satisfy more than one clause and thus appear in more than one pair in U.

Revising Algorithm I (cont'd)

- To estimate $c(F)$, we define a subset of U with size $c(F)$.
- We construct this set by selecting, for each satisfying assignment of F, exactly one pair of U that has this assignment.
- Specifically, we consider the following set S:

$$S = \{(i, a) | 1 \leq i \leq t \text{ and } a \in SC_i, a \notin SC_j \text{ for } j < i\}.$$

$|S| = c(F)$

- Then let us consider the second DNF counting algorithm.

DNF counting algorithm II:

Input: A DNF formula F with n variables
Output: Y = an approximation of $c(F)$

---

1. $X \leftarrow 0$.
2. **For** $k = 1$ to m, **do**:

   (a). With probability $|SC_i| / \sum_{i=1}^{t} |SC_i|$ choose, uniformly at random, an assignment $a \in SC_i$.

   (b). **If** a is not in any $SC_j, j < i$, **then** $X \leftarrow X + 1$.
3. **Return** $Y \leftarrow (X/m) \sum_{i=1}^{t} |SC_i|$.

Constructing S

Analysis

Remark: $|U| = \sum_{i=1}^{t} |SC_i|$ and $|S| = c(F)$.

- Note that $|S|/|U| \geq 1/t$.
  - Since each assignment can satisfy at most t different clauses.
- Now our S is relatively dense in U.
- Because the ith clause has $|SC_i|$ satisfying assignments, we have

$$\mathbf{Pr}[i \text{ is chosen}] = \frac{|SC_i|}{\sum_{i=1}^{t} |SC_i|} = \frac{|SC_i|}{|U|}.$$

或者用$|U| \leq t|S|$來想

Analysis (cont'd)

- Thus the probability that we choose the pair (i, a) is

$$\begin{aligned}
\mathbf{Pr}[(i,a) \text{ is chosen}] &= \mathbf{Pr}[i \text{ is chosen}] \cdot \mathbf{Pr}[a \text{ is chosen} \mid i \text{ is chosen}] \\
&= \frac{|SC_i|}{|U|} \cdot \frac{1}{|SC_i|} \\
&= \frac{1}{|U|}.
\end{aligned}$$

Theorem 2

DNF counting algorithm II is a FPRAS for the DNF counting problem when $m = \lceil (3t/\varepsilon^2) \ln(2/\Delta) \rceil$.

Proof of Theorem 2

- Step 2(a) chooses an element of U uniformly at random.
- The probability that this element belongs to S is at least $1/t$. (by the previous analysis)
- Fix any $\varepsilon, \Delta > 0$, and let $m = \lceil (3t/\varepsilon^2) \ln(2/\Delta) \rceil$.

So m is polynomial in $t, \varepsilon,$ and $\ln(1/\Delta)$.

Proof of Theorem 2

- Besides, the processing time of each sample is polynomial in t.
  - You can check this by observing 2(a) and 2(b).
- By Theorem 1, with m samples, X/m gives an (ε, Δ)-approximation of $c(F)/|U|$ and hence Y gives an (ε, Δ)-approximation of $c(F)$.

■

Approximate uniform sampling

- Now, we are going to present the outline of a general reduction
- This general reduction shows that, if we can sample almost uniformly a solution to a *self-reducible* combinatorial problem, then we can construct a randomized algorithm that approximately counts the number of solutions to the problem.

Approximate uniform sampling (cont'd)

- We will demonstrate this technique for the problem of counting the number of independent sets in a graph.
- We first need to formulate the concept of approximate uniform sampling.

Approximate uniform sampling (cont'd)

- In this setting, we are given a problem instance in the form of an input x, and there is an underlying finite sample space $\Omega(x)$ associated with the input.
- Let us see the following two definitions to make clear the concept of approximate uniform sampling.

ε - uniform Sample

Let w be the (random) output of a sampling algorithm for a finite sample space Ω. The sampling algorithm generates an ε-uniform sample of Ω if, for any subset S of Ω,

$$|\mathbf{Pr}[w \in S] - \frac{|S|}{|\Omega|}| \leq \varepsilon.$$

FPAUS

A sampling algorithm is a fully polynomial almost uniform sampler (FPAUS) for a problem if, given an input x and a parameter $\varepsilon > 0$, it generates an ε-uniform sample of $\Omega(x)$ and runs in time $\text{poly}(\ln \frac{1}{\varepsilon}, |x|)$.

FPRAS through FPAUS

- Consider an FPAUS for independent sets which would take as input a graph $G(V, E)$ and a parameter ε.
- The sample space:
  - the set of all independent sets in G.

FPRAS through FPAUS (cont'd)

- Goal:
  - Given an FPAUS for independent sets, we construct an FPRAS for counting the number of independent sets.
- Assume G has m edges, and let $e_1,\ldots,e_m$ be an arbitrary ordering of the edges.

FPRAS through FPAUS (cont'd)

- Let E_i be the set of the first i edges in E and let $G_i = (V, E_i)$.
  - Note that $G = G_m$.
  - G_{i-1} is obtained from G_i by removing a single edge e_i.
- Let $\Omega(G_i)$ denote the set of independent sets in G_i.

FPRAS through FPAUS (cont'd)

The number of independent sets in G can then be expressed as

$$|\Omega(G)| = \frac{|\Omega(G_m)|}{|\Omega(G_{m-1})|} \times \frac{|\Omega(G_{m-1})|}{|\Omega(G_{m-2})|} \times \ldots \times \frac{|\Omega(G_1)|}{|\Omega(G_0)|} \times |\Omega(G_0)|.$$

- $|\Omega(G_0)| = 2^n$. Why?
- To estimate $|\Omega(G)|$, we need good estimates for

$$r_i = |\Omega(G)| = \frac{|\Omega(G_m)|}{|\Omega(G_{m-1})|}, \text{for } i = 1, \ldots, m.$$

Since G_0 has no edges, every subset of V is an independent set and $|\Omega(G_0)| =2^n$.

r_i 就是前面所講的self-reducible的概念。

FPRAS through FPAUS (cont'd)

- Let $\widetilde{r}_i$ be an estimate for the ratio r_i, then we have $|\Omega(G)| \sim 2^n \prod_{i=1}^{m} \widetilde{r}_i$.
- To evaluate the error in our esitmate, we need to bound the ratio $R = \prod_{i=1}^{m} \frac{\widetilde{r}_i}{r_i}$
- In order to have an (ε, Δ)-approximation, we want $\mathbf{Pr}[|R-1| \leq \varepsilon] \geq 1 - \Delta$

Let us see the following lemma.

Lemma 1

Suppose that for all i, $1 \leq i \leq m$, $\widetilde{r_i}$ is an $(\varepsilon/2m, \Delta/m)$-approximation for r_i. Then

$$\mathbf{Pr}[|R-1| \leq \varepsilon] \geq 1-\Delta.$$

(By the definition of (ε, Δ)-approximation randomized algorithms.)

Proof of Lemma 1

- By the assumption of Lemma 1, for each $1 \le i \le m$, we have

$$\mathbf{Pr}[|\widetilde{r}_i - r_i| \le \frac{\varepsilon}{2m} r_i] \ge 1 - \frac{\Delta}{m}.$$

- Equivalently, for each $1 \le i \le m$,

$$\mathbf{Pr}[|\widetilde{r}_i - r_i| > \frac{\varepsilon}{2m} r_i] < \frac{\Delta}{m}.$$

Proof of Lemma 1 (cont'd)

- By the union bound, we have

$$\mathbf{Pr}[\bigcup_{i=1}^{m}\{|\widetilde{r}_i - r_i| > \frac{\varepsilon}{2m} r_i\}] < \Delta.$$

- Then we obtain

$$\mathbf{Pr}[\bigcap_{i=1}^{m}\{|\widetilde{r}_i - r_i| \leq \frac{\varepsilon}{2m} r_i\}] \geq 1 - \Delta.$$

- Equivalently,

$$\mathbf{Pr}[\bigcap_{i=1}^{m}\{1 - \frac{\varepsilon}{2m} \leq \frac{\widetilde{r}_i}{r_i} \leq 1 + \frac{\varepsilon}{2m}\}] \geq 1 - \Delta.$$

Proof of Lemma 1 (cont'd)

- Thus we have

$$\mathbf{Pr}[(1-\frac{\varepsilon}{2m})^m \le \prod_{i=1}^{m} \frac{\widetilde{r_i}}{r_i} \le (1+\frac{\varepsilon}{2m})^m] \ge 1-\Delta.$$

- Therefore,

$$\mathbf{Pr}[1-\varepsilon \le R \le 1+\varepsilon] \ge 1-\Delta.$$

(since $1-\varepsilon \le (1-\frac{\varepsilon}{2m})^m$ and $(1-\frac{\varepsilon}{2m})^m \le 1+\varepsilon$.)

■

Lemma 1

Suppose that for all i, $1 \leq i \leq m$, $\widetilde{r_i}$ is an $(\varepsilon/2m, \Delta/m)$-approximation for r_i. Then

$$\mathbf{Pr}[|R - 1| \leq \varepsilon] \geq 1 - \Delta.$$

Estimating r_i

- Hence all we need is a method for obtaining an $(\varepsilon/2m,\ \Delta/m)$-approximation algorithm for the r_i.
- An algorithm estimating r_i is given as follows.

Estimating r_i

Input: Graphs $G_{i-1} = (V, E_{i-1})$ and $G_i = (V, E_i)$
Output: $\widetilde{r}_i =$ an approximation of r_i.

---

1. $X \leftarrow 0$.
2. **Repeat** for $M = \lceil 1296 m^2 \varepsilon^{-2} \ln(2m/\Delta) \rceil$ independent trials:
   (a) Generate an $(\varepsilon/6m)$-uniform sample from $\Omega(G_{i-1})$.
   (b) **If** the sample is an independent set in G_i, **then** $X \leftarrow X + 1$.
3. **Return** $\widetilde{r}_i \leftarrow X/M$.

Estimating r_i (cont'd)

- The constants in the procedure were chosen to facilitate the proof of the following lemma, which justifies the algorithm's approximation .

Lemma 2

When $m \geq 1$ and $0 < \varepsilon \leq 1$, the procedure for estimating r_i yields an $(\varepsilon/2m, \Delta/m)$-approximation for r_i.

Proof of Lemma 2

- First we will show that r_i is not too small,
  - avoiding the problem we have introduced previously.
- Suppose G_{i-1} and G_i differ in that edge $\{u, v\}$ is in G_i but not in G_{i-1}.
- $\Omega(G_i) \subseteq \Omega(G_{i-1})$.
  - since an independent set in G_i is also an independent set in G_{i-1}.

Proof of Lemma 2 (cont'd)

- Each independent set in $\Omega(G_{i-1})\backslash\Omega(G_i)$ contains both u and v.
  - Why?
- Associate each $I \in \Omega(G_{i-1})\backslash\Omega(G_i)$ with an independent set $I \backslash\{v\} \in \Omega(G_i)$.
- In this mapping, note that $I' \in \Omega(G_i)$ is associated with no more than one independent set $I' \cup\{v\} \in \Omega(G_{i-1})\backslash\Omega(G_i)$, thus $|\Omega(G_{i-1})\backslash\Omega(G_i)| \le |\Omega(G_i)|$.

第一點：

假設 $\Omega(G_{i-1})\setminus\Omega(G_i)$ 中有一個independent set I_k 不同時包含u和v，則G_{i-1} 加入 (u,v) 變成 G_i 之後，

I_k依然屬於 $\Omega(G_i)$, 於是就矛盾了。

第三點：

因爲I' 頂多associated I'\cup {v} 或 I'\cup {u}其中之一。若兩者都有，則違反第一點。

Proof of Lemma 2 (cont'd)

- It follows that

$$r_i = \frac{|\Omega(G_i)|}{|\Omega(G_{i-1})|} = \frac{|\Omega(G_i)|}{|\Omega(G_i)| + |\Omega(G_{i-1}) \setminus \Omega(G_i)|} \geq \frac{1}{2}.$$

- Now consider our M samples.
- Let a random variable $X_k = 1$ if the kth sample is in $\Omega(G_i)$, and $X_k = 0$ otherwise

Proof of Lemma 2 (cont'd)

- Because our samples are generated by an (ε/6*m*)-uniform sampler, by definition,

$$|\ \mathbf{Pr}[X_k = 1] - \frac{|\Omega(G_i)|}{|\Omega(G_{i-1})}\ | \leq \frac{\varepsilon}{6m}.$$

$$\Rightarrow \quad |\ \mathbf{E}[X_k] - \frac{|\Omega(G_i)|}{|\Omega(G_{i-1})|}\ | \leq \frac{\varepsilon}{6m}.$$

Proof of Lemma 2 (cont'd)

- By linearity of expectations,

$$\left|\mathbf{E}\left[\frac{\sum_{k=1}^{M} X_k}{M}\right] - \frac{|\Omega(G_i)|}{|\Omega(G_{i-1})|}\right| \le \frac{\varepsilon}{6m}.$$

$\widetilde{r}_i$ r_i

- Therefore, we have

$$\begin{aligned} |\mathbf{E}[\widetilde{r}_i] - r_i| &= \left|\mathbf{E}\left[\frac{\sum_{i=k}^{M} X_k}{M}\right] - \frac{|\Omega(G_i)|}{|\Omega(G_{i-1})|}\right| \\ &\le \frac{\varepsilon}{6m}. \end{aligned}$$

因爲平均的期望值一定比最大X_k之期望值小，比最小的X_k之期望值大，

因此平均的期望值與$|\Omega(G_i)| / |\Omega(G_{i-1})|$ 的差距必定也滿足任一E[X_k]的與之的差距。

Proof of Lemma 2 (cont'd)

- Since $r_i \geq ½$, we have

$$\mathbf{E}[\widetilde{r}_i] \geq r_i - \frac{\varepsilon}{6m} \geq \frac{1}{2} - \frac{\varepsilon}{6m} \geq \frac{1}{3}.$$

Proof of Lemma 2 (cont'd)

If $M \geq \frac{3\ln(2m/\Delta)}{(\varepsilon/12m)^2(1/3)} = 1296m^2\varepsilon^{-2}\ln\frac{2m}{\Delta}$ (obtained from Theorem 1), then

$$\begin{aligned} & \mathbf{Pr}[|\frac{\widetilde{r}_i}{\mathbf{E}[\widetilde{r}_i]} - 1| \geq \frac{\varepsilon}{12m}] \\ = \; & \mathbf{Pr}[|r_i - \mathbf{E}[\widetilde{r}_i]| \geq \frac{\varepsilon}{12m}\mathbf{E}[\widetilde{r}_i] \\ \leq \; & \frac{\Delta}{m} \end{aligned}$$

Proof of Lemma 2 (cont'd)

- Equivalently, with probability $1 - \Delta/m$,

$$1 - \frac{\varepsilon}{12m} \le \frac{\widetilde{r}_i}{\mathbf{E}[\widetilde{r}_i]} \le 1 + \frac{\varepsilon}{12m}. \quad \text{------- (1)}$$

- As $|\mathbf{E}[\widetilde{r}_i] - r_i| \le \frac{\varepsilon}{6m}$, we have

$$1 - \frac{\varepsilon}{6mr_i} \le \frac{\mathbf{E}[\widetilde{r}_i]}{r_i} \le 1 + \frac{\varepsilon}{6mr_i}.$$

- Using that $r_i \ge 1/2$ then yields

$$1 - \frac{\varepsilon}{3m} \le \frac{\mathbf{E}[\widetilde{r}_i]}{r_i} \le 1 + \frac{\varepsilon}{3m}. \quad \text{---------- (2)}$$

Proof of Lemma 2 (cont'd)

- Combining (1) and (2), with probability $1 - \Delta/m$, we have

$$\boxed{(1-\frac{\varepsilon}{3m})(1-\frac{\varepsilon}{3m})} \le \frac{\widetilde{r}_i}{r_i} \le \boxed{(1+\frac{\varepsilon}{3m})(1+\frac{\varepsilon}{12m})}.$$

$$\geq 1-\frac{\varepsilon}{2m} \qquad\qquad \leq 1+\frac{\varepsilon}{2m}$$

- Thus this gives the desired ($\varepsilon/2m$, Δ/m)-approximation.

■

Lemma 2

When $m \geq 1$ and $0 < \varepsilon \leq 1$, the procedure for estimating r_i yields an $(\varepsilon/2m, \Delta/m)$-approximation for r_i.

Remark

- The number of samples M is polynomial in m, ε, and $\ln \Delta^{-1}$, and the time for each sample is polynomial in the size of the graph and $\ln 1/\varepsilon$, we therefore have the following theorem.

Theorem 3

Given an FPAUS for independent sets in any graph, we can construct an FPRAS for the number of independent sets in a graph G.

However,...

- How to obtain an FPAUS for independent sets for graphs?
  - See Chapter 11, *Coupling of Markov Chains*, page 286-289 in [MU05].
  - Or consider the Markov chain Monte Carlo (MCMC) method.

The Markov Chain Monte Carlo Method

- The Markov Chain Monte Carlo method provides a very general approach to sampling from a desired probability distribution.
- Basic idea:
  - Define an ergodic Markov chain whose set of states is the sample space and whose stationary distribution is the required sampling distribution.

The Markov Chain Monte Carlo Method (cont'd)

- Let $X_0, X_1, \ldots, X_n$ be a run of the chain.
- The Markov chain converges to the stationary distribution from any starting state X_0.
- After a sufficiently large number of steps r, the distribution of the state X_r will be close to the stationary distribution, so it can be used as a sample.

The Markov Chain Monte Carlo Method (cont'd)

- Similarly, repeating this argument with X_r as the starting point, we can use X_{2r} as another sample, and so on.
- We can therefore use the sequence of states X_r, X_{2r}, … as almost independent samples from the stationary distribution of the Markov chain.

The Markov Chain Monte Carlo Method (cont'd)

- The efficiency of MCMC depends on
  - how large r must be to ensure a suitable good sample
  - how much computation is required for each step of the Markov chain

"Coupling", see Ch. 11 of [MU05]

- Here we focus on finding efficient Markov chains with the appropriate stationary distribution.
  - For simplicity, we consider constructing a Markov chain with a **uniform** stationary distribution over the state space Ω.

Revisiting the Independent Sets in a Graph

- Given a graph $G(V, E)$.
- Let the state space be all of the independent sets of G.
- Two independent states x and y are neighbors if they differ in just one vertex.

Revisiting the Independent Sets in a Graph (cont'd)

- The neighbor relationship guarantees that the state space is *irreducible*.
  - Since all independent sets can reach (resp., can be reached from) the empty independent set by a sequence of vertex deletions (resp., vertex additions).

Revisiting the Independent Sets in a Graph (cont'd)

- Next we need to establish the transition probabilities.
- A naïve approach:
  - Random walk on the graph of the state space.
  - Yet the probability of a vertex is proportional to its degree, so this may not lead to a uniform distribution.
- Consider the following lemma.

Lemma 3

For a finite state space Ω and neighborhood structure $\{N(x) \mid x \in \Omega\}$, let $N = \max_{x \in \Omega} |N(x)|$. Let M be any number such that $M \geq N$. For all $x \in \Omega$, let $\pi_x = 1/\Omega$ be the desired probability of a state x in the stationary distribution. Consider a Markov chain where

$$P_{x,y} = \begin{cases} 1/M & \text{if } x \neq y \text{ and } y \in N(x), \\ 0 & \text{if } x \neq y \text{ and } y \notin N(x), \\ 1 - N(x)/M & \text{if } x = y. \end{cases}$$

If this chain is irreducile and aperiodic, then the stationary distribution is the uniform distribution.

That is, if we modify the random walk by giving each vertex an appropriate self-loop probability, then we can obtain a uniform stationary distribution.

Let us see the proof as follows.

Proof of Lemma 3

- For any $x \neq y$, since $\pi_x = \pi_y$ and $P_{x,y} = P_{y,x}$ (= $1 / M$), we have

$$\pi_x P_{x,y} = \pi_y P_{y,x}.$$

- Then apply the following theorem (Theorem 7.10 at [MU05]), it follows that the uniform distribution is the stationary distribution. ■

Theorem 4 (for the proof)

Consider a finite, irreducible, and ergodic Markov chain with transition matrix $\mathbf{P}$. If there are nonnegative numbers $\bar{\pi} = (\pi_0, \ldots, \pi_n)$ such that $\sum_{i=0}^{n} \pi_i = 1$ and if, for any pair of states i, j,

$$\pi_i P_{i,j} = \pi_j P_{j,i},$$

then $\bar{\pi}$ is the stationary distribution corresponding to $\mathbf{P}$.

Proof: Pleae refer to page 172 in [MU05].

Example: Independent Sets in a Graph

- Consider the following simple Markov chain whose states are independent sets in $G(V, E)$.

1. X_0 is an arbitrary independent set in G.
2. To compute X_{i+1}:
   (a) choose a vertex v uniformly at random from V;
   (b) if $v \in X_i$ then $X_{i+1} = X_i \setminus \{v\}$;
   (c) if $v \notin X_i$ and if adding v to X_i still gives an independent set, then set $X_{i+1} = X_i \cup \{v\}$;
   (d) otherwise, $X_{i+1} = X_i$.

Example: Independent Sets in a Graph (cont'd)

- The neighbors of a state X_i are independent sets that differ from X_i in just one vertex.
- Since every state is reachable from the empty set, the chain is irreducible.
- Assume G has at least one edge (u,v), then the state $\{v\}$ has a self-loop ($P_{\{v\},\{v\}} > 0$), thus aperiodic.
- When $X_i \neq X_j$, it follows that $P_{X_i, X_j} = 1/|V|$ or 0, by the previous lemma, the stationary distribution is the uniform distribution.

How about the non-uniform cases?

- However, in some other cases, we may want to sample from a chain with non-uniform stationary distribution.
- What should we do?
- Solution: *the Metropolis Algorithm*.

The Metropolis Algorithm

- Let us again assume that we have designed an irreducible state space for our Markov chain.
- Now we want to construct a Markov chain on this state space with a stationary distribution $\pi_x = b(x) / B$, where for all $x \in \Omega$ we have $b(x) > 0$ and such that $B = \sum_{x \in \Omega} b(x)$ is finite.

Lemma 4

For a finite state space Ω and neighborhood structure $\{N(x) \mid x \in \Omega\}$, let $N = \max_{x \in \Omega} |N(x)|$. Let M be any number such that $M \geq N$. For all $x \in \Omega$, let $\pi_x > 0$ be the desired probability of a state x in the stationary distribution. Consider a Markov chain where

$$P_{x,y} = \begin{cases} (1/M)\min(1, \pi_y/\pi_x) & \text{if } x \neq y \text{ and } y \in N(x), \\ 0 & \text{if } x \neq y \text{ and } y \notin N(x), \\ 1 - \sum_{y \neq x} P_{x,y} & \text{if } x = y. \end{cases}$$

If this chain is irreducile and aperiodic, then the stationary distribution is the uniform distribution π_x.

Proof of Lemma 4

- The proof is similar to the one of Lemma 3 as follows.
- For any $x \neq y$, if $\pi_x \leq \pi_y$, then $P_{x,y} = 1/M$ and $P_{y,x} = (1/M)(\pi_x/\pi_y)$.
- It follows that $P_{x,y} = 1/M = (\pi_y/\pi_x)\, P_{y,x}$.

  $\Rightarrow \pi_x P_{x,y} = \pi_y P_{y,x}$.
- The case for $\pi_x > \pi_y$ is similar.
- Again, by the previous theorem, π_x's form the stationary distribution. ■

Example: Independent Sets in a Graph

- Create a Markov chain, in the stationary distribution, each independent set I has probability proportional to $\lambda^{|I|}$, for some $\lambda > 0$.
- That is, $\pi_x = \lambda^{|I_x|}/B$, where I_x is the independent set corresponding to state x and $B = \sum_x \lambda^{|I_x|}$.
- Note that, when $\lambda=1$, this is the uniform distribution.

Example: Independent Sets in a Graph (cont'd)

- Consider the following variation on the previous Markov chain for independent sets in a graph $G(V, E)$.

1. X_0 is an arbitrary independent set in G.
2. To compute X_{i+1}:
   (a) choose a vertex v uniformly at random from V;
   (b) if $v \in X_i$ then $X_{i+1} = X_i \setminus \{v\}$ with probability $\min(1, 1/\lambda)$;
   (c) if $v \notin X_i$ and if adding v to X_i still gives an independent set, then set $X_{i+1} = X_i \cup \{v\}$ with probability $\min(1, \lambda)$;
   (d) otherwise, $X_{i+1} = X_i$.

Example: Independent Sets in a Graph (cont'd)

- First, we propose a move by choosing a vertex v to add or delete.
  - Each vertex is chosen with probability $1/M$, here $M = |V|$.
- Second, this proposal is then accepted with probability $\min(1, \pi_y / \pi_x)$, where x is the current state and y is the proposed state where the chain will move.

Example: Independent Sets in a Graph (cont'd)

- π_y / π_x
  - is "λ" if the chain attempts to add a vertex, and
  - is "1/λ" if the chain attempts to delete a vertex.
- Then we obtain the transition probability $P_{x,y}$ is

$$P_{x,y} = \frac{1}{M} \min(1, \frac{\pi_y}{\pi_x}).$$

- Thus Lemma 4 applies.

■

Example: Independent Sets in a Graph (cont'd)

- Comments:
  - We never need to know $B = \sum_x \lambda^{|I_x|}$
    - Calculating this sum would cost much time.
  - Our Markov chains gives the correct stationary distribution by using the ratios π_y / π_x, which are much easier to deal with.

Thank you.