Math 8213A – Rapidly Mixing Markov Chains Lecturer: *Dana Randall*
November 27, 1996 Scribe: *Cristina G. Fernandes*

Introduction

On the work of Jerrum, Valiant and Vazirani about the relation between approximate counting and approximate uniform generation.

Approximate counting and approximate uniform generation

Definition : $R \subseteq \{0,1\}^* \times \{0,1\}^*$ is a *p-relation* if deciding whether $(x, y) \in R$ can be done in time polynomial in $|x| + |y|$.
Example : SAT

Input: $X = (x_1 \vee x_2 \vee \bar{x}_3)(\bar{x}_1 \vee x_4 \vee x_5)(\bar{x}_2 \vee x_6)$.

Let us denote a truth assignment by y. For instance, $y = 011010$ means $x_1 = F, x_2 = V, x_3 = V, x_4 = F, x_5 = T, x_6 = F$.

Let R be the set of all pairs (X, y), where y is a *satisfying* assignment for X. Then R is a p-relation, since we can verify in polynomial time if y is in fact a satisfying assignment for X.

If we construct a binary tree as in Figure 1, the leaves contain all 2^m possible y's. Only some of them are satisfying assignments.

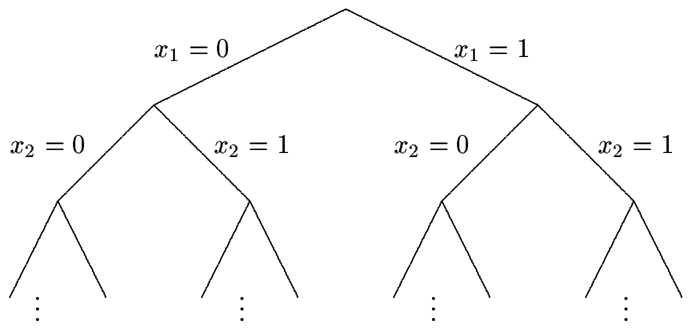

Figure 1: Binary tree representing the possibilities for y's

#SAT: How many of the leaves are satisfying assignments?

Observe that #SAT(X) = #SAT($X : x_1 = 0$) + #SAT($X : x_1 = 1$).

Definition : A p-relation is *self reducible* if we can express the set of solutions in terms of a set of smaller instances of the same problem and we can find the smaller instances in polynomial time.

In the example above, $(x_4 \vee x_5)(\bar{x}_2 \vee x_6)$ is the smaller instance for $x_1 = 1$, and $(x_2 \vee \bar{x}_3)(\bar{x}_2 \vee x_6)$ is the smaller instance for $x_1 = 0$.

When this happens, the next theorem tells us that approximate counting is the same as approximate uniform generation.

Theorem (Jerrum, Valiant and Vazirani) : If R is a self reducible p-relation, then there is a FPRAS iff there is a FPAUG.

Before sketching the proof, let us see how this applies to approximate counting perfect matchings.

Let G be a (general) graph. Pick an edge e of G.

PM(G) = PM(G^-) $\cup$ $\{M \cup \{e\} : M \in$PM$(G^+)\}$, where G^- is G after deleting edge e, and G^+ is G after deleting $e = (u, v)$ and all edges incident to u or v. This is true since, given a perfect matching M in G, either M includes e or not. If M includes e, then it corresponds to a perfect matching in G^+. If M does not include e, then it corresponds to a perfect matching in G^-.

Thus PM in general graphs define a self reducible p-relation. Observe however that this p-relation does not necessarily work for restricted classes of graphs. For example, for lattice graphs, G^+ and G^- might not be lattice graphs. But by selecting e carefully and modifying previous arguments to work for "approximate" lattice graphs (lattice graphs except maybe for the boundary), one can make PM in "approximate" lattice graphs to be self reducible.

Sketch of the proof of Jerrum, Valiant and Vazirani's theorem

Let $S^x = \{y : (x, y) \in R\}$ and $S^x_u = \{y : (x, y) \in R$ and u is a prefix of $y\}$. Assume $|y| = m$ if $(x, y) \in R$.

For instance, for SAT, we have $S^X = S^X_0 \cup S^X_1$.

FPRAS:
Given $\epsilon, \delta > 0$, and $x \in \{0,1\}^*$, output $g(x)$ such that

$$Pr\left[\frac{|S^x|}{1+\epsilon} \leq g(x) \leq |S^x|(1+\epsilon)\right] > 1 - \delta,$$

in time polynomial in $|x|, \epsilon^{-1}, \log \delta^{-1}$.

FPAUG:
Given $x \in \{0,1\}^*$, and $\alpha > 0$, output y, or "try again", so that the whole process runs in time polynomial in $|x|$ and $\log \alpha^{-1}$, and
1. if $(x, y) \in R$ then

$$\frac{1}{|S^x|(1+\alpha)} \leq Pr[\text{output } y] \leq \frac{1+\alpha}{|S^x|}.$$

2. if $(x, y) \notin R$ then $Pr[\text{output } y] = 0$.
3. if $|S^x| > 0$, then $Pr[\text{"try again"}] < 1/2$. (Expected number of "try again"s is 2. Any constant would also work.)

The no error case: exact counters and exact generators

Let us assume we have a perfect counter, and we want to generate y uniformly.

Let $N = |\{y = y_1, \ldots, y_m : (x, y) \in R\}|$ and $N_0 = |\{y = y_2, \ldots, y_m : (x, 0y) \in R\}|$ (use the perfect counter to compute N and N_0).

To generate y uniformly, branch left in the tree with probability N_0/N and branch right with probability N_1/N, and recurse: if you first branched left, compute N_{00}, and branch left again with probability N_{00}/N_0. Otherwise, branch right (with probability N_{01}/N_0).

Then the probability of being at a (satisfying) leaf $y = 010\ldots$ is

$$\frac{N_0}{N} \cdot \frac{N_{01}}{N_0} \cdot \frac{N_{010}}{N_{01}} \cdots = \frac{1}{N}.$$

Thus in fact we end up with a perfect uniform generator.

Now, let us assume we have a perfect uniform generator. We will describe how to get an approximate counter.

Generate enough samples to estimate N_0/N and N_1/N (the proportions of satisfying y's starting with a 0 or 1, respectively) with error within $\epsilon/2m$. Let β_1 be the larger between N_0/N and N_1/N (the smaller one can have a large deviation). So we have an estimate for β_1. Let us say N_1/N is larger (i.e., $\beta_1 = N_1/N$). Estimate β_2: the larger between N_{10}/N_1 and N_{11}/N_1, and so on, until a leaf (denoted β_m) has being estimated.

The β_i's are a telescoping series whose product is $1/N$. The product of the inverse of their estimates is within ϵ from N:

$$\frac{N}{(1+\epsilon)} \leq \prod_{i=1}^{m} \frac{1}{\beta_i} \frac{1}{(1+\frac{\epsilon}{2m})^m} \leq \prod_{i=1}^{m} \frac{1}{(\text{estimate for } \beta_i)} \leq \prod_{i=1}^{m} \frac{1}{\beta_i}(1+\frac{\epsilon}{2m})^m \leq (1+\epsilon)N.$$

So we obtain an approximate counter. In fact, even if we have started with an approximate generator, we would have to generate more samples (to compensate for the errors) to get our estimates with error within $\epsilon/2m$, but still we would end up with an approximate counter.

From an approximate counter to an approximate uniform generator

What if we have errors in our counter? How can we get an approximate uniform generator?

The method above applied to an approximate counter results in a generator which, let us say, outputs y with probability $p_y \approx 1/|S^x|$.

If we have b such that $b \leq \min\{p_y : (x,y) \in R\}$, then we could output y with probability b/p_y (if $p_y \neq 0$), otherwise try again. This would have the effect of "uniformizing" the distribution: any y such that $(x,y) \in R$ would be output with the same probability $p_y \cdot b/p_y = b$.

To apply this idea, however, we would have to know the p_y's and b. We can use our approximate counter to obtain estimates for the p_y's and for b. Observe that $1-|S^x|b$ is the probability of rejection (i.e., Pr["try again"]) which we want to be bounded by some constant smaller than one (so that the expected number of "try again"s is constant). So choose some constant $\epsilon > 0$, and use the approximate counter to get estimates $\hat{p}_y$'s for the p_y's such that $\hat{p}_y$ is $1/|S^x|$ within at most ϵ, with probability at least $1-\alpha$.

Generate a number t of estimates $\hat{p}_y$'s and let $\hat{b}$ be the minimum of all these estimates. Choose t large enough so that a subsequent estimate $\hat{p}_y$ is smaller than $\hat{b}$ with probability smaller than α.

Now, here is the algorithm to approximate uniform generation:

```
Generate a sample $y$ and get an estimate $\hat{p}_y$ for $p_y$.
If $\hat{p}_y \geq \hat{b}$, then output $y$ with probability $\hat{b}/\hat{p}_y$.
If $\hat{p}_y < \hat{b}$, then output $y$ with probability $\hat{p}_y$.
Otherwise, try again.
```

There are a few important things to be noticed. First, most of the y's are output with the same probability $\hat{b}$. Second, y's which are not output with probability $\hat{b}$ happen with probability smaller than α. Third, the rejection probability $1-\hat{b}|S^x|$ is small ($\leq \epsilon$).

About the running time, since we are using the approximate counter only with a constant ϵ and $\delta = O(\alpha)$, the algorithm is polynomial in $|x|$ and $\log \alpha^{-1}$, as required.