CS5120: Probability & Computing	**Jan-May 2020**
Markov Chains 4	

1 Approximate Counting

We have a universe U of elements and a subset $S \subseteq U$. Our goal is to find $|S|$. It may not be possible to find this value exactly with an efficient algorithm, thus our goal will be to find an approximate value, with a trade-off between running time and approximation (and error probability for a randomized algrorithm). Formally, we define the following.

Definition An algorithm to estimate z is a **Fully Polynomial-time Randomized Approximation Scheme** or **FPRAS**, if the output $\bar{z}$ of the algorithm satisfies:

$$Pr\left(|z - \bar{z}| > \varepsilon z\right) < \delta$$

and the algorithm runs in time:

$$poly(|\text{input}|)\ \frac{1}{\varepsilon^2}\log(\frac{1}{\delta}).$$

An output $\bar{z}$ such as above is called an (ε, δ)-approximation (to the value z).

Basic Monte Carlo Algorithm:

- Pick samples $X = \{X_1, \ldots, X_N\}$ u.a.r. from U.
- Let $s = |\{x \in X \cap S\}|$.
- Output $\frac{s}{N}|U|$.

For an $\varepsilon, \delta)$-approximation, we have the following which follows from Chernoff bounds.

Proposition 1 *The number N of samples needed for the above algorithm is:* $N \sim \Theta\left(\frac{|U|}{|S|}\frac{1}{\varepsilon^2}\log(\frac{1}{\delta})\right)$.

The above algorithm would be a FPRAS only if $|U|/|S|$ is bounded polynomially by the input size, which need not be the case.

In the second step, to find $|X \cap S|$, we assume that we have an algorithm that can test whether a given element belongs to S; and the running time will be that of this algorithm multiplied by N, the number of samples.

2 Counting DNF solutions

A DNF (Disjunctive Normal Form) formula consists of a disjunction of clauses, each of which is an AND of literals, that is: $\varphi = C_1 \vee C_2 \vee \ldots \vee C_m$; where each C_i is a conjunction of literals. For example, let $\varphi = C_1 \vee C_2 \vee C_3$, where $C_1 = (x \wedge \neg y \wedge z \wedge w)$, $C_2 = (y \wedge z)$, $C_3 = (x \wedge \neg w)$. Note that a DNF formula has satisfying assignments that are easy to find; just set all the literals of one clause to True, and arbitrarily assign other literals. In the above example, two satisfying assignments are $x = y = z = w = T$ and $x = y = T, z = w = F$. The interesting question here is to find the number of distinct satisfying assignments.

Let S_i denote the set of satisfying assignments to C_i, the ith clause. Then we want to find $|S|$, where $S = S_1 \cup S_2 \cup \ldots \cup S_m$. One approach is to use inclusion-exclusion. Write $S = \sum_i |S_i| - \sum_{i,j} |S_i \cap S_j| + \ldots$ and note that intersections can be computed easily; for example, if a conjunctive clause has k distinct literals (and no two negations of each other), then the number of solutions to that clause is 2^{n-k}. However, there are $\Theta(2^m)$ terms in the sum, which makes the running time prohibitively large.

We will instead use an approach that still makes use of the fact that we can write $S = S_1 \cup \ldots \cup S_m$, with each $|S_i|$ known. We will also use the fact that we can sample from each S_i.

Assumptions: We know $|S_i|$, and we can sample from each S_i. Note that these assumptions are satisfied for the DNF counting problem; to sample from S_i, set the literals of that clause to True, and set other variables u.a.r to True or False.

Let $T_i = \{(i, x) | x \in S_i\}$ and $T = T_1 \cup \ldots \cup T_m$. $|T| \leq m|S|$.

Algorithm to sample u.a.r. from T

- Pick $i \in \{1, 2, \ldots, m\}$ with prob $\dfrac{|S_i|}{|T|}$
- Pick x u.a.r from S_i.

We note that $Pr[(i,x) \text{ is output}] = \frac{|S_i|}{|T|}\frac{1}{|S_i|} = \frac{1}{|T|}$ as desired.

Algorithm to find $|S|$ where $S = S_1 \cup \ldots \cup S_m$.

- Pick $(i_1, x_1), \ldots, (i_N, x_N)$ u.a.r. from T.
- For each k, set $C_k = 1$ for (i,x) if S_i is the first set $\ni x$. Else set $C_k = 0$.
- Let $C = C_1 + \ldots + C_N$, $C_i \in \{0, 1\}$.
- Output $\frac{C}{N}|T|$ as the estimate of $|S|$.

Note that $Pr[C_k = 1] = \frac{|S|}{|T|}$; thus from Proposition 1, it follows that $N = \Theta\left(m\frac{1}{\varepsilon^2}\log(\frac{1}{\delta})\right)$ is sufficient to get an (ε, δ)- approximation.