CS648 : Randomized Algorithms
Semester I, 2010-11, CSE, IIT Kanpur

Random Sampling : part III

Estimating Size of Transitive Closure of a Directed Graph

In this lecture, we shall discuss efficient randomized methods to estimate some parameters of a given input. We shall address the following interesting problem.

Problem 1 *There is a directed graph $G = (V, E)$ on $n = |V|$ vertices and $m = |E|$ edges. For each vertex v, let $\tau(v)$ be the number of vertices reachable from v in the given graph. Design an algorithm which provides a very accurate estimate on $\tau(v)$ for each $v \in V$.*

In addition to being a problem of independent theoretical interest, this problem has applications in data bases. Before answering a data base query, one would like to estimate the size of the *query-answer set* to be reported. This prior knowledge may sometimes help in optimizing query processing time.

The best deterministic algorithm to solve the above problem achieves $O(\min(mn, n^{\omega}\log n))$ running time. We shall discuss an $O(m \log n)$ time Monte Carlo randomized algorithm for this problem. The remarkable point which the reader migh appreciate is that the algorithm is capable of achieving arbitrarily high accuracy (with in $(1 \pm \epsilon)$ factor) and that too with high success probability (arbitrarily close to 1). However, to achieve this goal, one also uses his basic knowledge of elementary (deterministic) graph algorithms in a novel way.

In order to convey the essential idea underlying the estimation algorithm We shall start with a fun problem. The solution of this problem will also serve as a basis for the solution of our main problem mentioned above.

1 Estimating the number of tokens from a bag

Let there be a bag containing n tokens each assigned a unique label from the set $\{1, ..., n\}$. The value of n (number of tokens) is unknown and our aim is to estimate it. The only tool provided to us is that we can sample a few tokens from the bag. Let X be the label of a token sampled uniformly and randomly from the bag.

Question 1.1 *Can you observe some correlation between the value of X you observe and the number of tokens in the bag ?*

If there are very few tokens, you would *expect* X to be small; and if there are large number of tokens, then you would *expect* X to be large. Your underlying intuition stems from the fact that $\mathbf{E}[X] = (n+1)/2$, that is, the expected value of the label of a sampled token is half of the number of tokens present in the bag. We shall exploit this high correlation between X and n to estimate the value of n.

Based on the above discussion, we can have a simple algorithm that reports twice the value(label) of the sampled token as the number of tokens in the bag. In case X takes the value equal to its expected value, the algorithm provides a *good* estimate. But, as we all know from our experiences, nothing goes as expected in real life unless we work sincerely hard. Since X is uniformly distributed over the entire set $\{1, ..., n\}$, the deviation of the answer from the actual number of tokens may be quite huge. For example, you may observe that with probability $1/4$, the number reported by the algorithm is less than half of the actual value of n. After spending a month in the course, you must have realized by know that $1/4$ is too high an error probability. So the question is : How can we reduce this error probability ?

In order to estimate the value of n accurately, let us try repeating the sampling algorithm k times and let $\{x_1, x_2, ..., x_k\}$ be the labels of the sampled tokens for k iterations. Note that we perform sampling with replacement, that is, we put the sampled token back into the bag after each trial. There may be various strategies to use the samples $\{x_1, ..., x_k\}$ to estimate n. We follow a very simple and intuitively appealing strategy that is based on the following question.

Question 1.2 *Out of the samples $\{x_1, ..., x_k\}$, which one is likely to be closest to $\mathbf{E}[X]$, that is, closest to $(n+1)/2$?*

It can be seen that X is a discrete random variable distributed uniformly over $\{1, ..., n\}$. So it will be the median of the set $\{x_1, ..., x_k\}$ which is most likely to be closest to $(n+1)/2$. Therefore, a natural strategy will be to report twice the value of this median. Now we shall analyse the error probability of this algorithm which uses multiple sampling. In particular, let us calculate the probability that the estimated value of n happens to be smaller than half of n. How does such a bad event look like ? For this purpose, look at Figure 1.

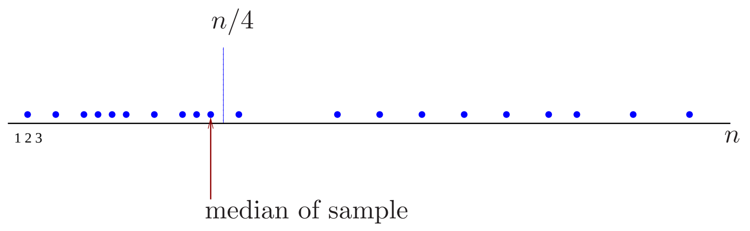


Figure 1: The event when median of the sample happens to be less than or equal to $n/4$

This bad event happens when there are at least $k/2$ sampled tokens with label $< n/4$. We now provide a formal analysis based on Chernoff bound to bound the probability of this bad event. Note that we can't consider X as a suitable candidate for applying Chernoff bound. After all, X is not sum of 0-1 random variable. But, surprising though it may appear, we shall still use Chernoff bound very effectively. **The reader should try to internalize the way Chernoff bound is used here**. Let Y be the random variable which is the total number of tokens sampled with labels in the range $[1, n/4]$. We can express Y as sum of 0-1 random variables as follows.

$$Y_i = \begin{cases} 1 & \text{if the token sampled in } i\text{th trial has label} \le n/4 \\ 0 & \text{otherwise} \end{cases}$$

So $Y = \sum_i^k Y_i$. It is easy to observe that $\mathbf{P}[Y_i = 1] = 1/4$. Furthermore, due to our sampling with replacement, each Y_i takes value independent of other random variables. $\mathbf{E}[Y] = k/4$. The bad event "$X \le n/4$" corresponds to $Y \ge k/2$, and its probability can be bounded by Chernoff bound as follows (with $\delta = 1$).

$$\mathbf{P}[Y \ge k/2] \le e^{-\frac{k}{4}/4} = e^{-k/16}$$

Compare this probability with the probability when you used single sampling. It can be seen that by sampling roughly 250 balls, the probability that your estimate of n is less than $n/2$ is less than 10^{-5}. This fun exercise is designed to make you appreciate the power of sampling to reduce the error probability in estimating some quantity of interest.

Before proceeding to the next section, the reader is encouraged to fully understand the analysis of the fun problem as given above. In particular, one should attempt the following exercise.

Exercise 1.1 *For a given $\epsilon, \delta > 0$, how many tokens one should sample to get an estimate $\hat{n}$ that satisfies $(1-\epsilon)n \le \hat{n} \le (1+\epsilon)n$ with probability at least $1-\delta$.*

This is because the analysis of the following problem will be almost along similar lines.

2 Estimating the size of transitive closure of a directed graph

As remarked earlier, there is a naive $O(mn)$ time algorithm for this problem where we perform BFS/DFS traversal from each vertex to compute the set of vertices reachable from it and then report the count of such vertices.

For many applications, it is not always required to compute the exact value of $\tau(v)$ for each $v \in V$. Instead the aim is to compute an estimate of $\tau(v)$. For example, compute a number $\hat{\tau(v)}$ such that $\tau(v)/2 < \hat{\tau(v)} < 3\tau(v)$. We shall now discuss an $O(m \log n)$ time algorithm such that for any $c_1, c_2 > 1$ the algorithm reports $\hat{\tau(v)}$ for each vertex $v \in V$ satisfying

$$\frac{\tau(v)}{c_1} < \hat{\tau(v)} < c_2\tau(v)$$

with high probability even for c_1, c_2 quite close to 1. In particular, the constant in $O(m \log n)$ running time will depend upon the values c_1, c_2 and the desired probability of success.

2.1 The underlying idea of randomization

We shall discuss a problem on continuous probability now. However, please note that you don't have to do a course on continuous probability theory to understand the underlying idea.

Firstly let us consider a random variable Y which takes value uniformly randomly from the interval $[0, 1]$. What do we mean by "uniformly randomly in the interval $[0, 1]$" ? For a finite discrete probability space, where we can define a uniform random variable as a random variable which takes a given value with probability $1/|\Omega|$. But, this definition would be meaningless in the context of a random variable which takes value from a continuous domain. This is because there are infinite number of sample points, and in fact probability it takes a given value is indeed 0. So in the given context, what "uniformly random in the interval $[0, 1]$" would mean ? You are encouraged to ponder over this question before proceeding further.

We can define a uniform random variable from interval $[0, 1]$ as follows. For any interval $\Delta \subseteq [0, 1]$, the probability that the sampled point belongs to Δ is $|\Delta|$. This basic understanding is all that we are going to use.

Let us define a random variable X over the interval $[0, 1]$ as follows. We select k numbers uniformly independently in the interval $[0, 1]$, and X is defined as the smallest of these k numbers (or equivalently the length of the first interval from the left). What can we say about the expected value of the X ? It can be shown that $\mathbf{E}[X] = 1/(k + 1)$? There are many "calculation-free" ways to show this. Two of them are sketched below.

- Let us define a random variable Y as follows. We select $k + 1$ points from a circle of circumference 1 unit. This will split the circumference into $k + 1$ intervals. Exploit uniformity in sampling the points and the symmetry to conclude that lengths of each of these $k + 1$ intervals have identical probability distribution (though they are not independent). Then conclude that expected length of an interval would be $1/(k + 1)$. Now try to relate this problem (sampling $k + 1$ points on a circle) with selecting k points from segment $[0, 1]$. All that you need is a sequence of accurately stated facts exploiting the meaning of uniformly random sampling and symmetry.

- Another method is to use the red-blue balls problem : there are r red balls and b blue balls in a bin and we take out the balls uniformly randomly. We showed that the expected number of blue balls preceding all the red ball is $\frac{b}{r+1}$. Use it for the discrete version of the above problem : If we select k numbers from $[1, n]$ where $n >> k$, then what can we say about the expected value of the smallest number selected ?

The fact that the expected value of the smallest number among the k numbers selected uniformly independently from $[0, 1]$ is $1/(k+1)$ tells you something very important. This tells you that the number of random variables k is related to $\mathbf{E}[X]$ very closely. We can use X to infer the number of random variables which define it. In particular, if X takes value a, we may return $1/a - 1$ as the estimate of the number of random variables. This randomization idea is the underlying idea of the algorithm for estimating the value $\tau(v)$ for each $v \in V$. (pause for a few minites to see how it will be used in the algorithm). If everything goes as expected, this will be a good estimate. However, to improve the accuracy of our estimate and the associated probability, we would like to use the idea of multiple sampling as used in the fun problem solved above.

2.2 Algorithm

Though there is no deterministic algorithm to compute $\tau(v)$ for all $v \in V$ in $O(m)$ time, there are many problems on directed graphs which can be solved in $O(m)$ time (for example, computing the strongly connected components of the graph). One such problem is the following : Let each vertex stores some key which is a real number. Our aim is to compute, for each each $v \in V$, the key of the smallest key vertex reachable from v. There is a deterministic $O(m)$ time algorithm for this problem based on DFS (depth first search) of the graph. We discussed it in the class as well as doubt clearing session. We shall use this algorithm and the randomization idea used above to solve the problem of estimating $\tau(v)$ for each $v \in V$.

The algorithm will perform ℓ iterations. In ith iteration, we assign a key to each vertex in the graph by selecting a number uniformly randomly from the interval $[0, 1]$. After this we execute $O(m)$ time algorithm which computes, for each $v \in V$, $k_i[v]$: the key of the smallest key vertex reachable from v. At the end of k iterations, we shall have a set $\{k_1[v], \ldots, k_\ell[v]\}$ of ℓ such labels for each $v \in V$. We know that expected value of any element of the set is $\frac{1}{\tau(v)}$. (Actually, it is $\frac{1}{\tau(v)+1}$, but for simplicity and clarity of exposition we ignore the additive term of 1 from the denominator). Out of these labels, we would like to select one label and report its inverse as the estimate of $\tau(v)$. But how do we select that particular label ? For this objective, we ask a question similar to that which was asked in the case of the problem of estimating number of tokens.

Question 2.1 *Out of the samples $\{k_1[v], ..., k_\ell[v]\}$, which one is likely to be closest to $\frac{1}{\tau(v)}$?*

To answer the above question, we would ask another question (The reader is encouraged to enquire about the relation between these two questions). What would be the probability that $k_i[v]$ takes value more than $1/\tau(v)$? We need to know the probability distribution of $k_i[v]$ for answering this question. For this purpise, we state the following Lemma whose proof is elementary.

Lemma 2.1 *If we select j numbers uniformly independently from interval $[0, 1]$, the probability that the smallest number is greater than c is $(1 - c)^j$ for any $0 < c < 1$.*

We now use Lemma 2.1, the probability that $k_i[v]$ takes value greater than $\frac{1}{\tau(v)}$ is

$$\mathbf{P}[k_i[v] > \frac{1}{\tau(v)}] = \left(1 - \frac{1}{\tau(v)}\right)^{\tau(v)} \approx 1/e \qquad (1)$$

What does Equation 1 mean ? It means that out of ℓ values from $\{k_1[v], \ldots, k_\ell[v]\}$, on an average, $1/e$ fraction of the values will be greater than $1/\tau(v)$. How can we use this observation to select suitable key from $\{k_1[v], \ldots, k_\ell[v]\}$? It is evident from the above argument that the (ℓ/e)th largest element from this set is most likely to be closest to $1/\tau(v)$. This label is denoted as k^*_v. So the algorithm finally reports $1/k^*_v$ as estimate $\hat{\tau(v)}$ of $\tau(v)$.

We provide the algorithm in complete details now.

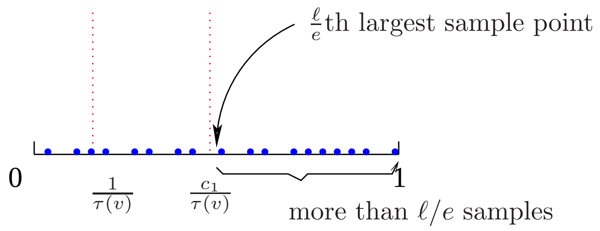
ALGORITHM

```
1. For $i = 1$ to $\ell$ do
    (a) Assign all the vertices numbers uniformly independently in interval (0,1).
    (b) For each vertex $v \in V$ compute the label of the smallest label vertex reachable from $v$.
        Let $k_i[v]$ be this label.
2. let $k^*_v$ be $(\ell/e)$th largest element from from the set $\{k_1[v], k_2[v], ..., k_\ell[v]\}$.
3. For each vertex $v \in V$, $\hat{\tau}(v) \leftarrow \frac{1}{k^*_v}$.
```

3 Analysis of the algorithm

What can you say about the closeness of $\hat{\tau(v)}$ with respect to $\tau(v)$? We shall try to get a bound on the probability for the event "$\tau(v)/c_1 < \hat{\tau}(v) < c_2\tau(v)$" for any given constants $c_1, c_2 > 1$. For this purpose we need to calculate a bound on the probability for the event "$\hat{\tau}(v) < \tau(v)/c_1$" and a bound on the probability for the event "$\hat{\tau}(v) > c_2\tau(v)$" separately.

We shall make use the following inequality for proving these bounds.



$\frac{\ell}{e}$th largest sample point
0
1
$\frac{1}{\tau(v)}$
$\frac{c_1}{\tau(v)}$
more than ℓ/e samples

Figure 2: The event when (ℓ/e)th largest element happens to be greater than $c_1/\tau(v)$

For all $a, n \in \mathbf{R}$ with $n \geq 1, |a| < n$,

$$e^a\left(1 - \frac{a^2}{n}\right) \leq \left(1 + \frac{a}{n}\right)^n \leq e^a$$

3.1 Bounding the probability of "$\hat{\tau}(v) < \tau(v)/c_1$"

$\hat{\tau}(v) < \tau(v)/c_1$ means that the (k/e)th largest element from $\{k_1[v], k_2[v], ..., k_\ell[v]\}$ happened to be greater than $c_1/\tau(v)$. This means that at least ℓ/e of the elements from $\{k_1[v], k_2[v], ..., k_\ell[v]\}$ turned out to be greater than even $c_1/\tau(v)$. See Figure 3.1.

Let us introduce random variables $X_i, i \leq t$ at this moment.

$$X_i = \begin{cases} 1 & \text{if } k_i[v] \text{ is greater than } c_1/\tau(v) \\ 0 & \text{otherwise} \end{cases}$$

Note that each of X_i, $i \leq \ell$ chooses its value independent of other X_j's $j \neq i$ since every iteration of the algorithm assigns labels to vertices independent of other iterations. Let $X = \sum_i X_i$. So we can see that the event "$\hat{\tau(v)} < \tau(v)/c_1$" is associated with "$X > \ell/e$". Using Lemma 2.1 and linearity of expectation, it is easy to observe that $\mathbf{E}[X]$ is $(1 - c_1/\tau(v))^{\tau(v)}\ell$, which is at most $e^{-c_1}k$ using the inequality mentioned in the box above. So the situation is the following. Compared to its expected value of ℓ/e^{c_1}, the random variable X took value greater than ℓ/e which is quite large (depending upon c_1).

Let us compute the probability for this event using Chernoff bound. Convince yourself that X fulfills all requirements for applying Chernoff bound. Let us for sake of clarity of exposition, assume the value of c_1 such that $\frac{1}{e^{c_1}} = \frac{1}{2e}$ (the value of c_1 is close to 1.7). So the expected value of X is $\ell/(2e)$. Since the value taken by X is at least ℓ/e, it implies that $\delta > 1$. Hence applying the Chernoff bound

$$\mathbf{P}[X > \ell/e] < e^{-\frac{\ell}{2e}/4} = e^{-\ell/8e}$$

If we choose $\ell = 24e \ln n$, that is, if we repeat the main iteration of the algorithm more than $24e \ln n$ times, the probability that $\tau(v)$ takes value less than $\tau(v)/(1.7)$ is less than $1/n^3$. Using Boole's inequality (union theorem) we can thus state the following Lemma.

Lemma 3.1 *If we repeat the main iteration of the algorithm at least $24e \ln n$ times, the probability that $\tau(v)$ is less than $\tau(v)/1.7$ for any v is less than $1/n^2$.*

3.2 Bounding the probability of "$\hat{\tau}(v) > c_2\tau(v)$"

This part is more nontrivial. This is bacause, we wish to achieve extremely small bound on the probability of this event and that too for any arbitrarily small $c_2 > 1$. The problem comes when you use inequality mentioned in the box above. This part is left for those students whose expectation from this course is more than just a good grade. From perspective of exams, you may ignore this portion if you wish.