CSE 6512 Randomization in Computing

Exam I; Help Sheet

Basics: We started with the basic ideas behind randomized algorithms. Two kinds of randomized algorithms were defined, namely, Monte Carlo and Las Vegas. An array example of each kind was seen. The selection problem takes as input a sequence X of n elements and an integer $i, 1 \leq i \leq n$ and the problem is to find the ith smallest element of X. We showed that this can be solved using $n + \min\{i, n-i\} + \widetilde{o}(n)$ comparisons. The randomized sorting algorithm of Frazer and McKellar was discussed. This algorithm makes $n \log n + \widetilde{O}(n \log \log n)$ comparisons.

Fingerprinting: The technique of fingerprinting can be applied to check if two given objects are the same. Instead of comparing the objects directly, we apply a function on each and compare the images. Some examples we have seen are:

1. For three given $n \times n$ matrices A, B, and C, the problem is to check if $AB = C$. We pick a random vector r from $\{0,1\}^n$ and check if $A(Br) = Cr$. (This can be done in $O(n^2)$ time). If so, we output: "$AB = C$"; else we output "$AB \neq C$". If $AB \neq C$, we showed that $Prob.[A(Br) = Cr] \leq 1/2$;

2. Given three polynomials $f(x), g(x)$, and $h(x)$, the problem is to check if $h(x) = f(x) \times g(x)$. The idea of fingerprinting can be used as follows: we pick a random element r from $\mathcal{S}$ (which is a subset of the field $\mathcal{F}$) and check if $f(r) \times g(r) = h(r)$. (This can be done in $O(n)$ time, n being the degree of $f(x)$ and $g(x)$). If so, we output: "$h(x) = f(x) \times g(x)$"; else we output: "$h(x) \neq f(x) \times g(x)$". We can show that the probability of an incorrect answer is no more than $\frac{d}{|\mathcal{S}|}$ where d is the degree of $h(x) - f(x) \times g(x)$;

3. Schwartz-Zippel theorem extends the above technique to check if a given multivariate polynomial is identically zero;

4. Edmonds' theorem in conjunction with Schwartz-Zippel theorem can be used to check for the existence of perfect matchings in graphs. This involves checking if the determinant of a matrix is zero. The time needed is $O(M(n))$, where $M(n)$ is the time needed to multiply two $n \times n$ matrices, n being the number of nodes in the input graph.

5. Karp-Rabin's algorithm applies fingerprinting to the string matching problem. To check if two given n-bit integers A and B are the same, the basic idea of this algorithm is to pick a random prime $p \leq t$ and check if $A \bmod p = B \bmod p$. Probability of an incorrect answer is $O\left(\frac{n}{t/(\log t)}\right)$.

Data structures:

1. Random skip lists have been introduced. We have shown that the dictionary operations can be performed in an expected $O(\log n)$ time each, where n is the maximum number of elements in the data structure.

2. We discussed hashing next. Let $M = \{0, 1, \ldots, m-1\}$ and $N = \{0, 1, \ldots, n-1\}$. A family H of hash functions from M into N is said to be 2-universal if for any two elements x and y from M with $x \neq y$, and for a randomly chosen h from H, $Prob.[h(x) = h(y)] \leq \frac{1}{n}$. We showed how to construct a family of 2-universal hash functions. Specifically, the following family was shown to be 2-universal: $\{h_{a,b}(x) = ((ax + b) \bmod p) \bmod n : a, b \in \mathcal{Z}_p, a \neq 0\}$. Here p is a prime larger than m. Using this family, we discussed the data structure of AKS that can be used to search in $O(1)$ time in a static table.

Chernoff bounds: If a random vraiable X is the sum of n iid Bernoulli trials with a success probability of p in each trial, the following equations give us concentration bounds of deviation of X from the expected value of np. The first equation is more useful for large deviations whereas the other two are useful for small deviations from a large expected value.

$$Prob(X \geq m) \leq \left(\frac{np}{m}\right)^m e^{m-np} \tag{1}$$

$$Prob(X \leq (1-\epsilon)pn) \leq exp(-\epsilon^2 np/2) \tag{2}$$

$$Prob(X \geq (1+\epsilon)np) \leq exp(-\epsilon^2 np/3) \tag{3}$$

for all $0 < \epsilon < 1$.

Sampling Lemma 1: Let X be a sequence of n elements. Let S be a random sample of X with $|S| = s$. Let $L = \ell_1, \ell_2, \ldots, \ell_s$ be the sorted sample. L partitions X into $s+1$ parts. The size of each part is $\widetilde{O}\left(\frac{n}{s} \log n\right)$.

Sampling Lemma 2: Let X be a sequence of n elements. Let S be a random sample of X with $|S| = s$. Let q be an element of S whose rank in S is j. If r_j is the rank of q in X, then we have:

$$Prob.\left[\left|r_j - j\frac{n}{s}\right| > \sqrt{3\alpha}\frac{n}{\sqrt{s}}\sqrt{\log n}\right] < n^{-\alpha} \tag{4}$$