CS648 : Randomized Algorithms

Practice sheet 2

The topics are:

- Frievald's techniques and pattern matching
- Maximum load of bin
- Chernoff Bound
- Hashing

1. **The cost of random bits**

   Recall Frievald's algorithm for checking equality $A \times B = C$. We selected a random {0,1}-vector: each entry was selected randomly uniformlyindependently from $\{0, 1\}$. The error probability was bounded by 2 essentially because there were 2 choices. What is wrong with the following arguments ?

   Select a random vector wherein each entry is a real number selected randomly uniformly and independently from $[0, 1]$. *The error probability of the algorithm will be 0. In this manner we get a deterministic algorithm for the problem !*

2. **Las Vegas algorithm for pattern matching**

   We discussed an $O(m+n)$ time Monte Carlo algorithm to detect if a pattern of length n bits appear in a text of m bits. Transform this algorithm into Las Vegas algorithm with expected $O(m + n)$ time. (Note that the algorithm has to detect just one match, if it exists. There is no need to enumerate all matches)

3. **Only for pondering**

   Can you figure out the conditions under which a Monte Carlo algorithm can be transformed into a Las Vegas algorithm ? Think over this question.

4. **random prime number**

   Design a Las Vegas algorithm that takes a number n as input and outputs a prime number from the interval $[n, 2n]$. The expected running time of the algorithm has to be polynomial of the input size. You might like to know the famous result from IITK about prime numbers to solve this problem.

5. **Maximum load of a bin again ...**

   Suppose we throw $n \log n$ balls randomly, uniformly, and independently into n bins. Prove that with high probability, the maximum load is going to be $\Theta(\log n)$. Is it not surprising ?

6. **Is Markov Inequality tight for some cases**

   This problem shows that Markov's inequality is as tight as it could possibly be. Given a positive integer k, describe a random variable X that assumes only non-negative values such that

   $$\mathbf{P}(X \geq k\mathbf{E}[X]) = \frac{1}{k}$$

7. **Proof of Chernoff Bound**

   Recall the proof of Chernoff bound that we discussed in the class. You are advised to go through the proof carefully and then try to answer the following questions. Some of these questions might not make sense if you have fully internalized the proof of Chernoff bound.

(a) In which step (or steps), the independence of random variables ?

(b) Where did we use the fact e^z is an increasing function of z?

(c) Did we ever use the fact that e^z is a 1-1 function ?

(d) Chernoff bound is based on Markov Inequality. Then how is it possible to achieve better bound using Chernoff bound ?

(e) What if we had picked some other function that e^{tX} ?

8. **Longest sequence of HEADS** We toss a fair coin n times. Show that with probability $\geq 1-1/n^2$, the length of longest contiguous sequence of HEADS will be $O(\log n)$.

9. **Familiarizing with expected value of function of random variables**

   Find $\mathbf{E}[X^2]$ for a random variable X if

   - X is a Bernoulli random variable with parameter p.
   - X is a Geometric random variable with parameter p.
   - X is a Binomial random variable with parameters n and p.
   - X is a Negative Binomial random variable with parameters n and p.

10. **Facts about Coin flipping** Supposed we flip a coin n times to obtain a sequence of flips $X_1, X_2, \ldots, X_n$. A streak of flips is a consecutive subsequence of flips that are the same. For example, if X_3,X_4,X_5 are all HEADS, there is a streak of length 3 starting at the third flip. (If X_6 is also HEADS then there is also a streak of length 4 starting at the third flip.)

    (a) Let n be power of 2. Show that the expected number of streaks of length $\log_2 n + 1$ is $1 - o(1)$.

    (b) Show that with high probability, the longest streak will be of length $O(\log_2 n)$.

    (c) Show that, the probability that there is no streak of length at least $\log_2 n - 2\log_2 \log_2 n$ is less than $1/n$.

11. p is a prime number. Let $a_1, \ldots, a_n$ be some integers in the range $[1, p-1]$. Let $X_1, ..., X_n$ be n random variables each taking integer value uniformly and independently in the range $[0, p-1]$. What is the probability that the following equation holds ?

$$(\sum_i a_i X_i) \textbf{ mod } p = 0$$

12. **Dice and Chernoff bound** We have a standard six-sided dice. Let X be the number of times that a 6 occurs over n throws of the dice. Let p be the probability of the event $X \geq n/4$. Compare the best bounds on p that you can obtain using Markov's inequality and Chernoff bounds.

13. **Universal hash functions through Boolean matrices**

    Suppose that $M = \{0,1\}^k$ and $N = \{0,1\}^\ell$. Let $\mathcal{M} = \{0,1\}^{k \times \ell}$ denote the space of Boolean matrices with k rows and ℓ columns. For any $\mathbf{x} \in M$ and $\mathbf{A} \in \mathcal{M}$, define

$$h_{\mathbf{A}}(\mathbf{x}) = \mathbf{x} \cdot \mathbf{A} \textbf{ mod } 2$$

    Note that $\mathbf{x} \cdot \mathbf{A} \in N$, and $\mathbf{x} \cdot \mathbf{A}$ **mod** 2 means we perform arithmetic modulo 2 at each of its entry.

    (a) If $\mathbf{x} \neq \mathbf{y}$, what is $\mathbf{P}_{\mathbf{A} \in_r \mathcal{M}} (h_{\mathbf{A}}(\mathbf{x}) = h_{\mathbf{A}}(\mathbf{y}))$?

    (b) Using (a), design a universal hash family.

    (c) How many bits are required to store a hash function from the hash family described in (b)? Compare it with the no. of bits required by the hash function from the universal hash family based on prime numbers discussed in the class.

14. **Hashing meets pattern matching**

The following problem is meant for those (hopefully non-zero number of) students in this class whose motivation is more than just a good grade. The solution of this problem need not to be submitted for grading.

In the class, we discussed finger-printing techniques for pattern matching. However, our solution was for any particular text and pattern. A more ambitious aim would be to preprocess a given text T so that we can find the number of occurrences of any pattern of a given size. In precise words, we wish to achieve the following.

> For any particular pattern size, after expected linear time (in the text size) preprocessing (and use of at most linear size), we can answer any query as to the number of occurrences of a given pattern in expected time proportional to the pattern size. The answer should be correct with high probability.

How will you make the query answering algorithm Las Vegas ?

The title of the problem is a hint. If you are able to solve it on your own, you may feel that you indeed have developed good skills for designing and analyzing randomized algorithms.

Note: In case you have any difficulty in any one of the problems in this practice sheet, you are welcome to have a discussion with me. Send a mail one day in advance.