0368-3248-01-Algorithms in Data Mining **Fall 2013**

Lecture 4: Home Assignment, Due Dec 3rd

Lecturer: Edo Liberty

Warning: *This note may contain typos and other inaccuracies which are usually discussed during class. Please do not cite this note as a reliable source. If you find mistakes, please inform me.*

---

1 Probabilistic inequalities

setup

In this question you will be asked to derive the three most used probabilistic inequalities for a specific random variable. Let $x_1, \ldots, x_n$ be independent $\{-1, 1\}$ valued random variables. Each x_i takes the value 1 with probability 1/2 and -1 else. Let $X = \sum_{i=1}^n x_i$.

questions

1. Let the random variable Y be defined as $Y = |X|$. Prove that Markov's inequality holds for Y. Hint: note that Y takes integer values. Also, there is no need to compute $\Pr[Y = i]$.

2. Prove Chebyshev's inequality for the above random variable X. You can use the fact that Markov's inequality holds for any positive variable regardless of your success (or lack of if) in the previous question. Hint: $\mathrm{Var}[X] = E[(X - E[X])^2]$.

3. Argue that
$$\Pr[X > a] = \Pr[\Pi_{i=1}^n e^{\lambda x_i} > e^{\lambda a}] \le \frac{E[\Pi_{i=1}^n e^{\lambda x_i}]}{e^{\lambda a}}$$
for any $\lambda \in [0, 1]$. Explain each transition.

4. Argue that:
$$\frac{E[\Pi_{i=1}^n e^{\lambda x_i}]}{e^{\lambda a}} = \frac{\Pi_{i=1}^n E[e^{\lambda x_i}]}{e^{\lambda a}} = \frac{(E[e^{\lambda x_1}])^n}{e^{\lambda a}}$$
What properties of the random variables x_i did you use in each transition?

5. Conclude that $\Pr[X > a] \le e^{-\frac{a^2}{2n}}$ by showing that:
$$\exists\ \lambda \in [0, 1]\ \ s.t.\ \ \frac{(E[e^{\lambda x_1}])^n}{e^{\lambda a}} \le e^{-\frac{a^2}{2n}}$$
Hint: For the hyperbolic cosine function we have $\cosh(x) = \frac{1}{2}(e^x + e^{-x}) \le e^{x^2/2}$ for $x \in [0, 1]$.

2 Approximating the size of a graph

setup

In this question we will try to approximate the size of a graph. A graph $G(V,E)$ is a set of nodes $|V| = n$ and a set of edges $|E| = m$. Each edge $e \in V \times V$ is a set of two nodes which support it. We assume the graph is simple which means there are no duplicate edges and no self loops (i.e. an edge $e = (u,u)$). The degree of a node, $\deg(u)$, is the number of edges which it supports. More formally $\deg(u) = |\{e \in E | u \in e\}|$. The degree of each node in the graph is at least 1. The question refers to the following sampling procedure:

1. $e = (u,v) \leftarrow$ an edge uniformly at random from E.
2. with probability 1/2
3. return u
4. else
5. return v

Throughout this question we assume that *i*) we can sample edges uniformly from the graph *ii*) that the number of edges m in known *iii*) that given a node u we can easily compute $\deg(u)$. The value of n, however, is unknown.

questions

1. Let $p(u)$ denote the probability that the sampling procedure returns a specific node, u. Compute $p(u)$ as a function of $\deg(u)$ and m. (Note: $\sum_{u \in V} \deg(u) = 2m$)

2. Let $f(u) = \frac{2m}{\deg(u)}$. Compute:
$$E_{x \sim smp}[f(x)]$$
where $x \sim smp$ denotes that x is chosen according to the distribution on the nodes generated by the above sampling procedure.

3. We say that a graph is d-degree-bounded if $\max_{u \in V} \deg(u) \le d$. Show that for a d-degree-bounded graph:
$$\mathrm{Var}_{x \sim smp}[f(x)] \le dn^2$$

4. Let $Y = \frac{1}{s} \sum_{i=1}^{s} f(x_i)$ where x_i are nodes chosen independently from the graph according to the above sampling procedure. Compute $E[Y]$ **and** show that $\mathrm{Var}[Y] \le dn^2/s$.

5. Use Chebyshev's inequality to find a value for s such that for any d-degree-bounded graph and any two constants $\varepsilon \in [0,1]$ and $\delta \in [0,1]$:
$$\Pr[|Y - n| > \varepsilon n] < \delta.$$
s should be a function of d, ε and δ.

3 Approximate median

setup

Given a list A of n numbers $a_1, \ldots, a_n$, we define the rank of an element $r(a_i)$ as the number of elements which are smaller than it. For example, the smallest number has rank zero and the largest has rank $n-1$. Equal elements are ordered arbitrarily. The median of A is an element a such that $r(a) = n/2$ (rounded either up or down). An α-approximate-median is a number a such that:

$$n(1/2 - \alpha) \le r(a) \le n(1/2 + \alpha)$$

In this question we sample k elements uniformly at random *with replacement* from the list A. Let the samples be $\{x_1, \ldots, x_k\} = X$. You will be asked to show that the median of X is an α-approximate-median of A.

questions

1. What is the probability the a randomly chosen element x is such that:

$$r(x) > n(1/2 + \alpha)$$

2. Let us define $X_{>\alpha}$ as the set of samples whose rank is greater than $n(1/2+\alpha)$. More precisely, $X_{>\alpha} = \{x_i \in X | r(x_i) > n(1/2+\alpha)\}$. Similarly we define $X_{<\alpha} = \{x_i \in X | r(x_i) < n(1/2-\alpha)\}$. Prove that if $|X_{>\alpha}| < k/2$ and $|X_{<\alpha}| < k/2$ then the median of X is an α-approximate-median of A.

3. Let $Z = |X_{>\alpha}|$. Find t for which:

$$\Pr[Z \ge k/2] = \Pr[Z \ge (1+t)E[Z]]$$

4. Bound from above the probability that $Z \ge k/2$ as tightly as possible. If you do so using a probabilistic inequality, justify your choice.

5. Compute the minimal value for k which will guarantee that $|X_{>\alpha}| < k/2$ **and** $|X_{<\alpha}| < k/2$ with probability at least $1 - \delta$.

4 Simple high capacity hashing

setup

In this question we try to evaluate the capacity of a special hash table. For simplicity, we assume that the hashed elements are a subset of $[N]$ ($[N]$ denots the set $\{1, \ldots, N\}$). The hash table consists of an array A of length n and L perfect hash functions $h_\ell : [N] \to [n]$. Throughout the exercise we assume the existence of perfect hash functions. That is, $\Pr[h(x) = i] = 1/n$ for all $x \in [N]$ and $i \in [n]$ independently of the values $h(x')$. For convenience we also assume that the entries in A are initialized to the value 0.

Algorithm 1 *Add*(x)

```
for ℓ ∈ [L] do
  if A[h_ℓ(x)] == 0 or A[h_ℓ(x)] == x then
    A[h_ℓ(x)] = x
    Return Success
  end if
end for
Return Fail
```

Algorithm 2 *Query*(x)

```
for ℓ ∈ [L] do
  if A[h_ℓ(x)] == x then
    Return True
  else if A[h_ℓ(x)] == 0 then
    Return False
  end if
end for
Return False
```

questions

1. Argue the correctness of the hashing scheme. a) If an element was **successfully** added to the table by $Add(x)$ it will be found by $Query(x)$. b) If an element was not added to the table by $Add(x)$ it will not be found by $Query(x)$.

2. Assume that exactly m cells in the array are occupied. That is, m cells contain values $A[j] > 0$ and for the rest $A[j] = 0$. Given a new element x which is in not stored in the hash table. What is the probability that location $h_1(x)$ in A is occupied.

3. What is the probability that procedure $Add(x)$ fails for an element x not in the hash table? (here we still assume there are exactly m elements already in the table)

4. Assume we start with an empty hash table and insert m elements one after the other. Use the union bound to get a value for L for which $Add(x)$ succeeds in **all** m element insertions with probability at least $1 - \delta$

5. Argue that the **expected** running time of both $Add(x)$ and $Query(x)$ is $O(1)$. That is, it does not depend on L.