Assignment 1

Hao Lee 141070027

Nanjing University

1 Problem 1

Recall that in class we show by the probabilistic method how to deduce a $\frac{n(n-1)}{2}$ upper bound on the number of distinct min-cuts in any multigraph with vertices from the $\frac{2}{n(n-1)}$ lower bound for success probability of Karger's min-cut algorithm.

Also recall that the algorithm taught in class guarantees to return a min-cut with probability at least . Does this imply a much tighter upper bound on the number of distinct min-cuts in any multigraph with vertices? Prove your improved upper bound if your answer is "yes", and give a satisfactory explanation if your answer is "no".

1.1 solution:

the problem is: how many different min-cuts in a graph G, given Pr[c is returned]$\geq \frac{2}{n(n-1)}$, c is a min-cut in G. figure out the upper bound. assume G has r min-cuts, c_1,c_2...,c_r. thus Pr[c_i is returned]$\geq$ $\frac{2}{n(n-1)}$, given i $\in 1, 2, ...r.andtheseeventthat$c$_i$ is returned is mutually exclusive. Pr[$\bigcup_{i=1}^{r}$c$_i$ is returned] $\geq \sum_{i=1}^{r} \frac{2}{n(n-1)} = \frac{2\times r}{n(n-1)}$. and Pr[$\bigcup_{i=1}^{r}c_i$ is returned]$\leq$ $1.thus, r\leq$ $\frac{n(n-1)}{2}$.
I can give a counter-example. for a complete graph with n vertices, due to its symmetry, the number of distinct min-cuts in this kind of graph must be upper than n. and $O(\log n)$ is lower than n. so $O(\log n)$ cannot be the upper bound. so my answer is 'no'.

2 Problem 2

Give an efficient randomized algorithm with bounded one-sided error (false positive), for testing isomorphism between rooted trees with n vertices. Analyze your algorithm.

2.1 solution:

the thought we want to solve the problem is to find an expression of the trees using invariant 0 and 1. then using the fingerprint algorithm to check the identity.

we can use the AHU algorithm to figure out the problem. the idea of the algorithm is to find a sole number only have 0 and 1 to represent a rooted tree. the figure shows that how the algorithm works. and then the problem becomes the communication complexity. and one-sided error $\leq \frac{1}{2}$

Algorithm 1 Tree isomorphism (AHU) AHUSORT(v)

Input:
two rooted trees with n vertices.

Output:
whether the two trees are isomorphed ;

```
if v is a leaf then
    Name v '10'
else
    for all child w of v do
        AHUSORT(w);
sort the names of the children of v;
concatenate the names of all children of v to temp;
Give v the name 1temp0;
```

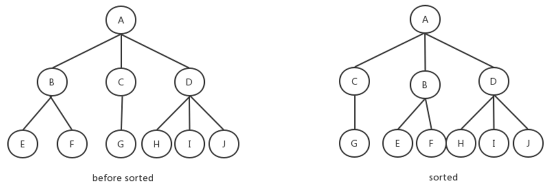

Fig. 1. the tree before and after sorted

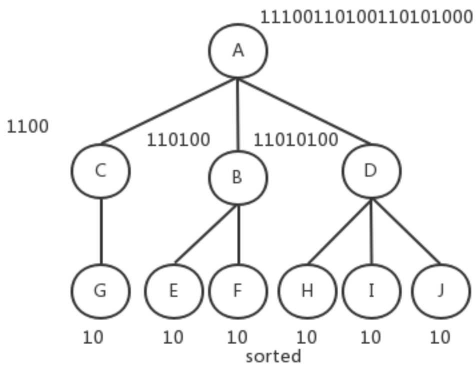

Fig. 2. the tree named

3 Problem 3

describe a strategy of choosing an x from the sampled set $\{Y_1, Y_2, ... Y_t\}$ such such that rank(x) is approximately k. here rank(x) denotes the rank of x in the original list $\{x_1, x_2, ...x_n\}$: The rank of the largest number among $x_1, x_2, , x_n$ is 1; the rank of the second largest number among $x_1, x_2, , x_n$ is 2, Choose your t as small as possible (in big-O notation) so that with probability at least$1 - \delta$. your strategy returns an x such that $(1 - \epsilon)k \leq \text{rank}(x) \leq (1 + \epsilon)k$.

3.1 solution:

the problem can be simplified as randomly select a number from a set of n distinct numbers.$rank^{-1}(k)$ is the number we want. $\Pr[Y_i \geq rank^{-1}(k)] = \frac{k-1}{n}$. then repeat for t times, we want the number of the elements in the set $\{Y_1, Y_2, ... Y_t\}$ that greater than $rank^{-1}(k)$. we call the number m. then the m-largest number in Y is the k-largest number in X. obviously m obey the Bernoulli trials. we define nonnegative Z represent the distribution of m. then:

$$Pr[Z = m] = C_t^m p^m \times (1 - p)^{t-m} \tag{1}$$

in this equation, p=$\frac{k-1}{n}$ denotes the possibility that we successfully fetched. then we get the expection $\mu = t \times p$.

$$\begin{aligned} \mu &= t \times p \\ &= t \times \frac{k-1}{n} \\ &= \frac{t \times (k-1)}{n} \end{aligned}$$

so the m-th largest number in Y is x. then we define: $\text{rank}_\text{X}(x)$is the rank of x in X, $\text{rank}_\text{Y}(x)$ is the rank of x in Y. thus

$$\begin{aligned} Pr[(1 - \epsilon)k \leq \text{rank}(x) \leq (1 + \epsilon)k] &= Pr[(1 - \epsilon)k \leq \text{rank}_\text{X}(x) \leq (1 + \epsilon)k] \\ &= Pr[(1 - \epsilon)\mu \leq \text{rank}_\text{Y}(x) \leq (1 + \epsilon)\mu] assume \\ &= Pr[(1 - \epsilon)\mu \leq m \leq (1 + \epsilon)\mu] \end{aligned}$$

$\text{W}_i = \frac{Z_i}{E(Z_i)}$, then $E(W) = 1$, $D(W) = \frac{D(Z)}{E^2(Z)} = \frac{1-p}{t \times p}$,

$$Pr[(1 - \epsilon)\mu \leq m \leq (1 + \epsilon)\mu] = Pr[|W - E(W)| \leq \epsilon] \geq 1 - \delta$$

due to Chebyshev's Inequality, then

$$\begin{aligned} Pr[|W - E(W)| \geq \epsilon] &\leq \frac{D(W)}{\epsilon^2} \\ &= \frac{1 - p}{\epsilon^2 \times tp} \end{aligned}$$

assume$p \geq \frac{1}{2}$, (if $p \leq \frac{1}{2}$, we can transfer this problem to find the k smallest number then$p \geq \frac{1}{2}$ still) then $t = \frac{1}{\epsilon^2 \times \delta}$

4 Problem 4

4.1 Solution:

1. (1) assume m=$\frac{n}{2}$, then the maximum load is also $\Theta\left(\frac{\log n}{\log\log n}\right)$. so the the maximum load $P \geq \Theta\left(\frac{\log n}{\log\log n}\right)$ for the remaining $\frac{n}{2}$, because the power of two choices is exponentially less than one choice. so the the maximum load $P \leq \Theta\left(\frac{\log n}{\log\log n}\right)$ then the asymptotically tight bound is $\Theta\left(\frac{\log n}{\log\log n}\right)$.
2. (2) this situation is more easier. I can view the situation as twice throw then get the sum. so the maximun load w.h.p is $\Theta\left(\frac{\log n}{\log\log n}\right) + \Theta(\log\log n) = \Theta\left(\frac{\log n}{\log\log n}\right)$
3. (3) i think this paradigm is also larger than throw $\frac{n}{2}$ balls into n bins. so the bound is also $\Theta\left(\frac{\log n}{\log\log n}\right)$.

5 Problem 5

5.1 Solution:

1. (1)

$$\begin{aligned} Pr[X \geq t] &= Pr[(\mathrm{e}^{\lambda X}) \geq (\mathrm{e}^{\lambda t})], for all \lambda \geq 0 \\ &\leq \frac{E(\mathrm{e}^{\lambda X})}{\mathrm{e}^{\lambda t}}, due to generalized Markov' s inequality. \\ &= \mathrm{e}^{\ln \frac{E(\mathrm{e}^{\lambda X})}{\mathrm{e}^{\lambda t}}} \\ &= \mathrm{e}^{-(\lambda t - \Psi_X(\lambda))} \end{aligned}$$

then for all $\lambda \geq 0$,$Pr[X \geq t] \leq \mathrm{e}^{-(\lambda t - \Psi_X(\lambda))}$,so$Pr[X \geq t] \leq \min \mathrm{e}^{-(\lambda t - \Psi_X(\lambda))}$, then $\Psi_X^*(t) := \sup_{\lambda \geq 0}(\lambda t - \Psi_X(\lambda))$

2. $let f(\lambda) = \lambda t - \Psi_X(\lambda)$, given a fixed t, the function get it's extreme value when $f'(\lambda)$, that is $\Psi_X'(\lambda) = t$
3. Normal random variables we can get $\mathbb{E}\left[\mathrm{e}^{\lambda X}\right] = \mathrm{e}^{\frac{2\lambda\mu - \lambda^2\sigma}{2\sigma}}$, then $\Psi_X(\lambda) = \frac{2\lambda\mu - \lambda^2\sigma}{2\sigma}$. the detailed proof is in the supporting materials.

6 Problem 6

A boolean code is a mapping $C : \{0,1\}^k \to \{0,1\}^n$.Each $x \in \{0,1\}^k$ is called a message and $y = C(x)$is called a codeword. The code rate r of a code C is $r = \frac{k}{n}$. A boolean code $C : \{0,1\}^k \to \{0,1\}^n$ is a linear code if it is a linear transformation, i.e. there is a matrix $A \in \{0,1\}^{n\times k}$ such that$C(x) = Ax$for any $x \in \{0,1\}^k$, where the additions and multiplications are defined over the finite field of order two, $(\{0,1\}, +_{\bmod 2}, \times_{\bmod 2})$,The distance between two codewordy_1,y_2,is defined as the Hamming distance between them. Formally, $d(y_1, y_2) = \|y_1 - y_2\|_1 =$

$\sum_{i=1}^{n} |y_1(i) - y_2(i)|$The distance of a code C is the minimum distance between any two codewords. Formally, $d = \min_{\substack{x_1, x_2 \in \{0,1\}^k \\ x_1 \neq x_2}} d(C(x_1), C(x_2))$. Usually we want to make both the code rate r and the code distance d as large as possible, because a larger rate means that the amount of actual message per transmitted bit is high, and a larger distance allows for more error correction and detection. Use the probabilistic method to prove that there exists a boolean code $C : \{0,1\}^k \to \{0,1\}^n$,of code rate r and distance$\left(\frac{1}{2} - \Theta\left(\sqrt{r}\right)\right) n$Try to optimize the constant in $\Theta(\cdot)$ Prove a similar result for linear boolean codes.

6.1 solution:

1. first we want to prove that: for random i,j, $d(y_i, y_j) = \|y_i - y_j\|_1 = \sum_{k=1}^{n} |y_i(k) - y_j(k)|$ which is the Hamming distance between y_i andy_j, the expectation of d is $\frac{n}{2}$.the detailed proof is in the supporting material. then we define e_{ij} is the comparation between y_i andy_j. then we can obviously find that there are $\frac{(2^k-1)\times 2^k}{2}$ edges. that is, there are $\frac{(2^k-1)\times 2^k}{2}$ comparations in the whole $\{0,1\}^n$ space. then we can define $X : X_1, X_2, ...X_{\frac{(2^k-1)\times 2^k}{2}}$, represent the random event. we can use the Markov's Inequailty to get a bound for a X_i. and the problem is to prove the possibility of the $\frac{(2^k-1)\times 2^k}{2}$ union sets is lower than 1, which is proved in the supporting material.
2. the problem is similar to question 1.