

CS 6150: HW4 – Graphs, Randomized algorithms

Submission date: Wednesday, Nov 10, 2021 (11:59 PM)

This assignment has 5 questions, for a total of 50 points. Unless otherwise specified, complete and reasoned arguments will be expected for all answers.

Question	Points	Score
QuickSelect	6	
Sampling from a stream	6	
Walking on a path	12	
Birthdays and applications	12	
Checking matrix multiplication	14	
Total:	50	

Instructions. For all problems in which you are asked to develop an algorithm, write down the pseudocode, along with a rough argument for correctness and an analysis of the running time (unless specified otherwise). Failure to do this may result in a penalty. If you are unsure how much detail to provide, please contact the instructors on Piazza.

Question 1: QuickSelect [6]

Recall that given an (unsorted) array of **distinct** integers $A[0, 1, \dots, n - 1]$ and a parameter $1 \leq k \leq n$, the Selection problem asks to find the k th smallest entry of A . In class, we saw an algorithm that used a randomized implementation of ApproximateMedian, and showed that it leads to an $O(n)$ time algorithm. Let us now consider a different procedure, that is similar to QuickSort.

PROCEDURE QUICKSELECT(A, k)

1. If $|A| = 1$, return the only element
2. Select x from A uniformly at random
3. Form arrays B and C , containing the elements of A that are $< x$ and $> x$ respectively
4. If $|B| = (k - 1)$, return x , else if $|B| < (k - 1)$, return QUICKSELECT($C, k - |B| - 1$), else return QUICKSELECT(B, k)

Let $T(n)$ be defined as the **expected running time** of QuickSelect on an array of length n . Using the law of conditional expectation, prove that

$$T(n) \leq n + \sum_{j=1}^n \frac{1}{n} \max\{T(j - 1), T(n - j)\}.$$

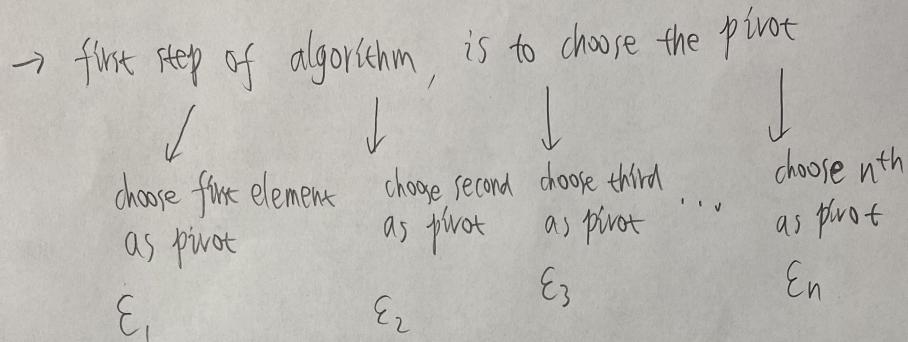
Using this along with $T(1) = 1$, prove that $T(n) \leq 4n$. Write down a description of all the events you use when you use conditional expectation.

(For the purposes of this question, you may ignore the additional $O(1)$ time for steps (1-2) and (4) of the procedure above.) [Hint: Follow the analysis for QuickSort seen in class, use induction.]

Side note. It is interesting to see that the constant term (the 4 in $4n$) above is much better than what we had for the deterministic algorithm we saw before. It turns out that there's a way of improving the constant further: instead of choosing x uniformly at random, we pick a small sample from the array and pick the sample median.

Answer: I write down my answer on paper, because it is really hard to write down these using latex.

T_n : the running time of quickselect, on array of length n , distinct integers $[0, 1, \dots, n-1]$



each of this possibility is $\frac{1}{n}$, we have n choice

E_j : event that algorithm pick j th element as pivot

$$\Pr(E_j) = \frac{1}{n}$$

we need to find $E[T_n | E_j]$

$$E[T_n | E_j] = n + E[T_{j+1}] + E[T_{n-j}]$$

↓ time needed to do all the partition

$$\text{so } E[T] = E[T | E_1] \cdot \Pr(E_1) + E[T | E_2] \cdot \Pr(E_2) + \dots$$

$$E[T_n] = \sum_{j=1}^n \Pr(E_j) \cdot E[T_n | E_j] = \left(\sum_{j=1}^n \frac{1}{n} \cdot \max\{T(j+1), T(n-j)\} \right) + E(T | E_n) \Pr(E_n)$$

↓ event that j th element was chosen as pivot

This proof is also worst case, because it cover every possibility of every elements, since this is upper bound we could say

$$\text{Then, we have } T(n) \leq n + \sum_{j=1}^n \frac{1}{n} \max\{T(j+1), T(n-j)\}$$

↓ n is time needed to do all partition

$\max \{T(j), T(n-j)\}$ could also be understood as step(4) of quickselect, since step(4) take $O(1)$, and we could ignore, $T(n) \leq n + \sum_{j=1}^n \frac{n}{n} \cdot 1 \Rightarrow T(n) \leq n + n \cdot \frac{1}{n} \leq 2n$

Need this result for next prove

then we throw a floor function at $\left(\frac{n}{2}\right)$

then, we apply recursion on the algorithm

First step: running time = $T(2n) + T\left(\frac{n}{2}\right)$

Keep doing recursion

$$T\left(\frac{n}{2}\right) = \frac{2n}{2} + T\left(\frac{n}{4}\right)$$

$$T\left(\frac{n}{4}\right) = \frac{2n}{2} + T\left(\frac{n}{8}\right)$$

⋮

$$T(1) = 1$$

let us sum them up, $2n + \frac{2n}{2} + \frac{2n}{2} + \dots + 1 = 2n + n + \frac{n}{2} + \dots + 1 \leq 4n$ (upper bound)

then we say $T(n) \leq 4n$

Question 2: Sampling from a stream [6]

If you have an array of n elements, sampling one at random is easy: you choose an index i at random in $\{0, 1, \dots, n-1\}$ and return the i th element. Now suppose you have a *stream* of elements a_1, a_2, \dots (suppose they are all distinct for simplicity), and you don't know how many will arrive beforehand. Your goal is the following: at the end of the stream, you should output a random element from the stream.

The trivial algorithm is to store all the elements in an array (say a dynamic array), and in the end, output a random element. But it turns out that this can be done with very little memory.

Consider the following procedure: we maintain a special variable x , initialized to the first element of the array. At time t , upon seeing a_t , we set $x = a_t$ with probability $1/t$, otherwise we keep x unchanged.

Prove that in the end, the variable x stores a uniformly random sample from the stream. (In other words, if the stream had N elements, $\Pr[x = a_i] = 1/N$ for all i .)

[*Hint:* try doing a direct computation.]

Answer:

I have a stream of elements $[a_1, a_2, \dots, a_N]$,
they are all distinct,

goal: end of stream, output a random element from the stream
because every element is distinct, that means \Pr each element is equally probable.

base on the procedure from question,

for example, I have a stream of elements $[2, 1, 4, 6, 8, 9]$
initialized $X=2$ at $t=1$, $X=a_1$, with probability $\frac{1}{1}$,

then $t=2$, $X=a_2$, $X=1$, with probability $\frac{1}{2}$,

then $t=3$, $X=a_3$, $X=4$, with probability $\frac{1}{3}$

⋮
to last $t=6$, $X=a_6$, $X=9$, with probability $\frac{1}{6}$.

to be saying, every element is distinct, is equally probable

this means, (this is uniformly) that is to say,

in the end, the last element is also uniformly with \Pr ,

because X change every time.

If the stream had N elements, with every element is distinct,

the probability of each element depends on length of stream,
which is N , $\Pr(\text{each element}) = \frac{1}{N}$

that is to say, if the stream had N elements,

$\Pr[X=a_i] = \frac{1}{N}$ for all i , because each element has equally probability

Question 3: Walking on a path [12]

Consider a path of length n comprising vertices v_0, v_1, \dots, v_n . A particle starts at v_0 at $t=0$, and in each time step, it moves to a **uniformly random neighbor** of the current vertex. Thus if it is at v_s at time t for some $s > 0$, then at time $(t+1)$, it moves to v_{s+1} or v_{s-1} with probability $1/2$ each. (If it is at v_0 , the only neighbor is v_1 and so it moves there.) The particle gets "absorbed" once it reaches v_n and the walk stops.

Define $T(i)$ as the expected number of time steps taken by a particle starting at i to reach v_n . By definition, $T(n) = 0$.

- (a) [5] Prove that $T(0) = 1 + T(1)$, and further, that for any $0 < s < n$, $T(s) = 1 + \frac{T(s-1) + T(s+1)}{2}$.

Answer:

from the question, a path of length n comprising vertices, v_0, v_1, \dots, v_n , $T(i)$ define as expected number of time steps by a particle starting at i to reach v_n . By definition, $T(n) = 0$, which means, the last step $T(n) = 0$, so that $T(0)$ should be, = how many steps we need, for example, if we need 10 steps to reach v_n , $T(0) = 10$, so $T(0) = 1 + T(1) \rightarrow T(0) - T(1) = 1$

this 1 means, because at v_0 , the only neighbor is v_1 , it moves to v_1 for sure, and also, this is uniformly neighbor and at time $(t+1)$ to move to next vertex, so it takes 1 step to move to next vertex, we get $T(0) - T(1) = 1$, meaning, take 1 step to move to next vertex, we get $T(0) = 1 + T(1)$

For any $0 < s < n$, $T(s) = 1 + \frac{T(s-1) + T(s+1)}{2}$

Step of this algorithm, is to choose which way to go

each way have probability of $\frac{1}{2}$, either go v_{s+1} , or go v_{s-1} , $\Pr(\text{which way}) = \frac{1}{2}$

we need to find $E[T_s | \text{which way to go}]$

\downarrow
 $(1 + E[T_{s-1}] + E[T_{s+1}]) / 2$, 1 is first step needed

\downarrow
 $E[T] = E[T|v_{s+1}] \cdot \Pr(\text{which way}) + E[T|v_{s-1}] \cdot \Pr(\text{which way})$

so $T(s) = 1 + T(s-1) \cdot \frac{1}{2} + T(s+1) \cdot \frac{1}{2} = \frac{(1 + T(s-1) + T(s+1))}{2}$

- (b) [5] Use this to prove that $T(s) = (2s + 1) + T(s + 1)$ for all $0 \leq s < n$, and then find a

closed form for $T(0)$. [Hint: Use induction.]

Answer:

(b) to prove $T(s) = (2s+1) + T(s+1)$ for all $0 \leq s \leq n$

check $T(0)$ first, we

$$\text{when } s=0 \quad T(0) = (2 \cdot 0 + 1) + T(0+1)$$

$$T(0) = 1 + T(1), \text{ check,}$$

↓ proved in part (a) of this question.

$$\text{when } n \in \mathbb{N}, T(s) = 1 + \frac{T(s-1) + T(s+1)}{2}, \text{ also proved in part (a)}$$

$$2T(s) = 2 + T(s-1) + T(s+1) \Rightarrow T(s) = 2 + \underbrace{T(s-1) - T(s)}_{\downarrow} + T(s+1)$$

$$T(s+1) = 1$$

$$2s-1$$

$$= 2s+1 + T(s+1)$$

Then find the closed form for $T(0)$

$$T(s) = 2s+1 + T(s+1), \text{ with } T(n) = 0, \text{ the last step}$$

$$s=n-1 \Rightarrow T(n-1) = 2(n-1) + 1 + T(n) = 2(n-1) + 1 = 2n-1$$

$$s=n-2 \Rightarrow T(n-2) = 2(n-2) + 1 + T(n-1)$$

$$= 2(n-2) + 1 + (2n-1) + 1 = 4n-4$$

$$s=n-3 \Rightarrow T(n-3) = 2(n-3) + 1 + T(n-2) = 2n-6+1+4n-4$$

$$= 6n-9$$

$$s=n-4 \Rightarrow T(n-4) = 2(n-4) + 1 + T(n-3) = 2n-8+1+6n-9$$

$$= 8n-16$$

$$s=n-5 \Rightarrow T(n-5) = 2(n-5) + 1 + T(n-4) = 2n-10+1+8n-16$$

$$= 10n-25$$

we have a pattern

$$s=0 \Rightarrow T(0) = (2 \cdot (n-s)) \cdot n - (n-s)^2 \text{ for all } 0 \leq s \leq n$$

- (c) [2] Give an upper bound for the probability that the particle walks for $> 4n^2$ steps without getting absorbed.

Answer:

$$E[n] = 2(n-5) \cdot n - (n-5)^2 \Rightarrow n^2 - 5^2$$

we want the probability that walk for $> 4n^2$

steps without getting absorbed

we need at least have $E[n] \cdot 4$ to get $4n^2$

so probability at least $\left(\frac{1}{2} \cdot 4 = \frac{1}{8}\right)$, (upper bound)

Question 4: Birthdays and applications [12]

Suppose we have n people, each of whom has their birthday on a random day of the year.
Suppose **there are m days in a year**, and let us pretend that this is some parameter.

- (a) [5] What is the expected number of pairs (i, j) with $i < j$ such that person i and person j have the same birthday? For what value of n (as a function of m) does this number become 1?

Answer:

Suppose we have n people, each of them have their birthday on a random day of year, there are m days in a year,

(a) first, we have m days of a year, and this is uniform distribution

• with m days, $\Pr(\text{particular pair share a birthday}) = \frac{1}{m}$

• with n people, there are $\binom{n}{2} = \frac{n(n-1)}{2}$ pairs of people,

• so $E[\text{number of pairs have same birthday}] = \frac{1}{m} \cdot \frac{n(n-1)}{2} = \frac{n^2-n}{2m}$

• Then find the value of n the number become 1,

We need $E \geq 1$ which mean $\frac{n^2-n}{2m} \geq 1$

$$\text{Solve equation } \frac{n^2-n}{2m} = 1 \Rightarrow n^2 - n - 2m = 0$$
$$n = \frac{1 + \sqrt{1+8m}}{2} \text{ or } \frac{1 - \sqrt{1+8m}}{2} \text{ (ignore)}$$

only answer

- (b) [7] This idea has some nice applications in CS, one of which is in estimating the “support” of a distribution. Suppose we have a radio station that claims to have a library of one million songs, and suppose that the radio station plays these songs by picking, at each step a uniformly random song from its library (with replacement), playing it, then picking the next song, and so on.

Suppose we have a listener who started listening when the station began, and noticed that among the first 200 songs, there was a repetition (i.e., a song played twice). Prove that the probability of this happening (conditioned on the library size being a million songs) is < 0.05 . Note that this gives us “reasonable doubt” about the station’s claim that its library has a million songs.

Hint: Compute the probability of the complementary event —that all songs would be distinct— and prove that it must be large. You may use the inequality $(1 - x)^n \geq 1 - nx$ (for $x > 0$ and a positive integer n) without proof.

[This idea has many applications in CS, for estimating the size of sets without actually enumerating them.]

Answer:

(b) because library has 1000 000 songs,
 these songs was played by picking each step
 a uniformly random song from library

$\Pr(\text{not play same songs in 200 tries})$

$$= \sum_{j=1}^{200} \frac{1000000-j}{1000000} = \frac{1999799}{10000}$$

$$\Pr(\text{play same songs in 200 tries}) = 200 - \frac{1999799}{10000}$$

$0.0201 < 0.05$, check

$$= 0.0201$$

because this is

complementary events, and there are 200 songs played

$$\text{we use } 200 \cdot 1 - \sum_{j=1}^{200} \frac{1000000-j}{1000000} = 0.0201$$

This is "reasonable doubt", the probability of this happen = 0.0201,
 which is really low, which we doubt, Are there really 1 million songs

Question 5: Checking matrix multiplication [14]

Matrix multiplication is one of the classic algorithmic problems. Consider the problem of multiplying two $n \times n$ matrices over the field \mathbf{F}_2 (i.e., we have matrices with entries 0/1, and we perform all computations modulo 2; e.g., $0*0 + 1*1 + 1*1 + 1*0 = 0$).

The best known algorithms here are messy and take time $O(n^{2.36\dots})$. However, the point of this exercise is to prove a simpler statement. Suppose someone gives a matrix C and claims that

$C = AB$, can we quickly verify if the claim is true?

- (a) [5] First prove a warm-up statement: suppose a and b are any two 0/1 vectors of length n , and suppose that $a \neq b$. Then, for a random binary vector $x \in \{0, 1\}^n$ (one in which each coordinate is chosen uniformly at random), prove that $\Pr[\langle a, x \rangle \neq \langle b, x \rangle \pmod{2}] = 1/2$. [In other words, with a probability 1/2, we can “catch” the fact that $a \neq b$.]

Answer:

(a)

a and b are any two 0/1 vectors of length n ,
and suppose that $a \neq b$, a random binary vector $x \in \{0, 1\}^n$

In this case, I will just imagine work over the reals,
the question have “matrices” are over Field F_2 ,
so the computations should also carried out over the Field F_2

but, proofs remain unchanged,

proof: suppose, $a \neq b$, let $\alpha = \sum_{i \in \ell} a_i x_i$
 $\beta = \sum_{i \in \ell} b_i x_i$

We can write $ax = \alpha + a_i x_i$

$bx = \beta + b_i x_i$

this gives $ax - bx = (\alpha - \beta) + (a_i - b_i)x_i$

then invoke the Principle of Deferred Decision

to assume that we have first picked all the values x_j for $j \notin \ell$,

then we can write

$$\Pr[ax - bx = 0] = \Pr[x_i = \frac{\alpha - \beta}{a_i - b_i}] \leq \frac{1}{2}$$

where we use the fact that $(\alpha - \beta)/(a_i - b_i)$

can only be either 0 or 1, and a randomly

chosen x_i will take that value with probability at most half

- (b) [6] Now, design an $O(n^2)$ time algorithm that tests if $C = AB$ and has a success probability $\geq 1/2$. (You need to bound both the running time and probability.)

Answer:

(b) the algorithm will be, compare ABx with Cx ,
this takes $O(n^2)$ time, since we can first compute $y = Bx$
and then compute $Ay = ABx$, each matrix vector product
takes only $O(n^2)$ time

then if $ABx = Cx$, output yes, if not, output no

if $AB \neq C$, algorithm fail with probability at most $\frac{1}{2}$

if $AB \neq C$, then there is at least one row in AB ,
say $(AB)_i$, differ from corresponding row $(C)_i$ in C ,
the probability that $a_i \cdot x = b_i \cdot x$ is at most $\frac{1}{2}$,
For the algorithm, to output yes, we must have $a_i \cdot x = b_i \cdot x$,
so the probability of failure for the algorithm is at most $\frac{1}{2}$

(c) in order to increase the success probability to $\frac{7}{8}$,
we need to have more numbers in matrices
instead of only 0, 1

- (c) [3] Show how to improve the success probability to $7/8$ while still having running time $O(n^2)$.

Answer: Answer was included in part (b).