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 $A[0, 1, \ldots, n-1]$ and a parameter $1 \leq k \leq n$, the Selection problem asks to find the kth 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.

Solution. Verify this is correct.

Let X be the running time of the QuickSelect algorithm. Then $T(n) := \mathbb{E}[X]$. Let Y_j be the event that the jth element of A is chosen as the pivot. Since the pivot is chosen randomly, $Pr[Y_j] = 1/n$ for all $j \in [1, \ldots, n]$. By the law of conditional expectation,

$$T(n) = \sum_{j=1}^{n} \frac{1}{n} \mathbb{E}[X | Y_j]$$

Creating B and C requires n comparisons and results in a sub-problem of either size $(j-1)$ or size $(n-j)$, corresponding to the size of the B or C. Since we could recurse to either

B or C, we must consider the worst case and take the maximum of the two problem. Thus $\mathbb{E}[X|Y_j] \leq n + \max\{T(j-1), T(n-j)\}$ and

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

To prove $T(n) \leq 4n$, we use induction. The base case $T(1) = 1 \leq 4$ is given. Now we assume that $T(i) < 4i, \ \forall 1 < i < n$. From this assumption, we know

$$T(n) \leq n + \sum_{j=1}^{n} \frac{1}{n} \max\{T(j-1), T(n-j)\} \leq n + \sum_{j=1}^{n} \frac{1}{n} \max\{4(j-1), 4(n-j)\}$$

This gives the following, as we factor out $\frac{4}{n}$ and simplify the sum:

$$\begin{aligned}
T(n) &\leq n + \sum_{j=1}^{n} \frac{1}{n} \max\{4(j-1), 4(n-j)\} \\
&\leq n + \frac{4}{n} \sum_{j=1}^{n} \max\{(j-1), (n-j)\} \\
&\leq n + \frac{4}{n}(\Big(n-1+n-2+\cdots+n/2\Big) + \Big(n-1+n-2+\cdots+n/2\Big)) \\
&\leq n + \frac{4}{n} 2 \sum_{j=n/2+1}^{n} (j-1) = n + \frac{4}{n} 2 \sum_{j=1}^{n/2} (n-j) \\
&\leq n + \frac{4}{n} 2 (\frac{n^2}{2} - \sum_{j=1}^{n/2} j \\
&\leq n + \frac{8}{n} \Big(\frac{n^2}{2} - \frac{(n/2)(n/2+1)}{2}\Big) \\
&\leq 4n - 2 \leq 4n
\end{aligned}$$

Rubric:

- 1 point for understanding random variables and expected values
- 1 point for properly using the law of conditional expectation
- 1 point for showing the upper-bound of $T(n)$
- 3 points for proving $T(n) \leq 4n$

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 $\{0, 1, \ldots, n-1\}$ and return the ith element. Now suppose you have a *stream* of elements $a_1, a_2, \ldots$ (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.]

Solution. Let X_t be the event that a_t is chosen at time t. Then, for every $1 \leq i \leq N$, $\Pr[x = a_i] = \Pr[X_i \wedge \overline{X_{i+1}} \wedge \ldots \wedge \overline{X_N}]$, where $\overline{X}$ denotes the complement of event X. This is because if a_t is chosen at time t for some $i < t \leq N$, then x cannot be a_i (note that all elements are distinct).

This probability can be computed directly using $\Pr[X_t] = \dfrac{1}{t}$ and $\Pr[\overline{X_t}] = 1 - \Pr[X_t] = 1 - \dfrac{1}{t}$ for any t.

$$\begin{aligned}
\Pr[x = a_i] &= \Pr[X_i \wedge \overline{X_{i+1}} \wedge \ldots \wedge \overline{X_N}] \\
&= \Pr[X_i] \cdot \Pr[\overline{X_{i+1}}] \cdot \ldots \cdot \Pr[\overline{X_N}] \\
&= \frac{1}{i}\left(1 - \frac{1}{i+1}\right)\left(1 - \frac{1}{i+2}\right)\cdots\left(1 - \frac{1}{N}\right) \\
&= \frac{1}{i} \cdot \frac{i}{i+1} \cdot \frac{i+1}{i+2} \cdot \ldots \cdot \frac{N-1}{N} = \frac{1}{N}
\end{aligned}$$

Thus, the variable x stores a uniformly random sample from the stream in the end.

Rubric:

- 3 points for reasoning
- 3 points for computing $\Pr[x = a_i]$

Question 3: Walking on a path .... **[12]**

Consider a path of length n comprising vertices $v_0, v_1, \ldots, v_n$. 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 $0 < s < n$, $T(s) = 1 + \frac{T(s-1)+T(s+1)}{2}$.

Solution.

Assume the particle is at v_s, let X_s be the number of steps to reach the end from v_s, and let F_R be the event the particle moves to right and F_L be the event the particle moves to the left. We can see that since both events are equally likely, $\mathbb{P}(F_R) = \mathbb{P}(F_L) = \frac{1}{2}$. If the particles moves to the right, then $X_s = 1 + X_{s+1}$ and if the particle moves to the left, then $X_s = 1 + X_{s-1}$. Then using the law of conditional expectation,

$$\begin{aligned}T(s) &= E(X_s) = E(X_s|F_R)\mathbb{P}(F_R) + E(X_s|F_L)\mathbb{P}(F_L)\\ &= E(1+X_{s+1})\frac{1}{2} + E(1+X_{s-1})\frac{1}{2} = (1+T(s+1))\frac{1}{2} + (1+T(s-1))\frac{1}{2}\\ &= 1 + \frac{T(s-1)+T(s+1)}{2}\end{aligned}$$

Rubric:

- 2 points for correct analogy.
- 3 points for proof explanation.

(b) [**5**] Use this to prove that $T(s) = (2s+1) + T(s+1)$ for all $0 \le s < n$, and then find a closed form for $T(0)$. [*Hint:* Use induction.]

Solution.

We can show this using induction.

Base case: Setting $s = 0$, we have $T(0) = 1 + T(1)$, which is given to be true.

Assume the statement is true for $s - 1 < n - 1$, i.e. $T(s-1) = (2(s-1)+1) + T(s)$, then we can see that since $s < n$,

$$\begin{aligned}T(s) = 1 + \frac{T(s-1)+T(s+1)}{2} &= 1 + \frac{(2(s-1)+1) + T(s) + T(s+1)}{2}\\ 2T(s) &= 2 + (2(s-1)+1) + T(s) + T(s+1)\\ T(s) &= (2s+1) + T(s+1)\end{aligned}$$

Therefore, by induction, $T(s) = (2s+1) + T(s+1)$ for all $0 \le s < n$

Given $T(s) - T(s+1) = (2s+1)$, summing from $s = 0$ to $n-1$,

$$\begin{aligned}T(0) = T(0) - T(n) &= \sum_{s=0}^{n-1} (T(s) - T(s+1))\\ &= \sum_{s=0}^{n-1} 2s + 1 = 2\left(\frac{n(n-1)}{2}\right) + n = n^2\end{aligned}$$

Rubric:

- 2 points for proof explanation
- 3 points for giving a closed form for T(0)

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

Solution.

Using Markov's inequality, if X is the number of steps the particle walks, $E(x) = T(0)$ and therefore,

$$\mathbb{P}(X > 4n^2) \le \mathbb{P}(X \ge 4n^2) \le \frac{T(0)}{4n^2} = \frac{1}{4}$$

Rubric:

- 2 points for giving an 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?

Solution.

If Y_{ij} is the random variable that indicates persons i and j have the same birthday, then all is asked is to compute the expected value of the sum over all pairs of Y_{ij}. They may not be independent, but to compute expectation we do not need independence (because of linearity of expectation). Y_{ij} will be 1 if i and j share the same birthday and 0 otherwise.

Now, we have to find the probability of 2 people who share the same birthday. For each pair, the first person can have any day as their birthday. However, if we want to find the probability of the second person having the same birthday as the first one, then the second person has a probability of $\frac{1}{m}$ (where m is all the days in a a year), i.e., we do not care what day is the birthday of the first person, we just want to see the probability of the second person sharing the same birthday with the first person which is $\frac{1}{m}$.

The total number of all possible pairs is $\binom{n}{2}$. Now if $E[X]$ is the expected number of pairs, then using linearity of expectation,

$$E[X] = \sum_{ij \mid i<j} \frac{1}{m} = \frac{\binom{n}{2}}{m}$$

Now, we have $\frac{\binom{n}{2}}{m} = 1 \rightarrow \frac{n(n-1)}{2m} = 1 \rightarrow n^2 - 2m - n = 0$

Since the number of the people cannot be negative the answer will be:

$$n = \frac{1 + \sqrt{1 + 8m}}{2}$$

Rubric:

- 1 point for getting the probability of 2 people having the same birthday correct.
- 1 point for getting total number of possible pairs.
- 1 point for using linearity of expectation.
- 2 points for calculating the expected value
- 1 point for getting n.

(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.]

Solution.

Method 1:

From 4.a we reasoned that to determine the probability of two songs (birthdays) occurring on the same day is dependent on the number of possible pairs between two elements and likelihood of overlap. With m being the size of our sample space and n the number of subset samples we can see that there are a total of $\binom{n}{2}$ possible unique pairs. In our case there are $\binom{200}{2}$. Labeling all $\binom{n}{2}$ (i,j) pairs by index k from 1 to $\binom{n}{2}$ and X_k to be the random variable in which selected songs i and j are identical for k corresponding to pair (i,j) then the likelihood of this event occurring is given by $P(X_k = 1) = \frac{1}{m} = \frac{1}{10^6}$ and $E(X_k) = P(X_k = 1) = \frac{1}{10^6}$. Let Z be the number of such pairs, then we can see that $Z = \sum_{k=1}^{\binom{n}{2}} X_k$, and therefore using linearity of expectation,

$$\begin{aligned}
E[Z] =& E[\sum_{k=1}^{\binom{n}{2}} X_k] = \sum_{k=1}^{\binom{n}{2}} E[X_k] \\
=& \sum_{k=1}^{\binom{n}{2}} P(X_k = 1) \\
=& \sum_{k=1}^{\binom{n}{2}} \frac{1}{m} \\
=& \sum_{k=1}^{\binom{200}{2}} \frac{1}{10^6} \\
=& \binom{200}{2} \frac{1}{10^6}
\end{aligned}$$

Given $Z \geq 0$ by Markov's inequality we see that $P(Z \geq 1) \leq E(Z)$ will give us a bound on the probability that at least two of the songs out of 200 were the same. From this we see that $P(Z \geq 1) \leq \binom{200}{2}\frac{1}{10^6} = \frac{200 \cdot 199}{2 \cdot 10^6} = 0.0199$.

Rubric:

- 1 point for getting the probability of 2 songs being the same.
- 1 point for getting total number of possible pairs.
- 2 point for defining Z.
- 1 point for using linearity of expectation.
- 1 point for getting expected value of Z.
- 1 point for using Markov's inequality to bound the probability.

Method 2:

The probability of two songs being identical as previously discussed is given by the likelihood a selected song will be identical, or $\frac{1}{10^6}$. The likelihood two songs not being the same is then $(1 - \frac{1}{10^6})$. Similarly of from three songs selected the likelihood two are the same is given by the likelihood the first selected was identical multiplied by the likelihood the second selected was identical, or $(1 - \frac{1}{10^6}) \cdot (1 - \frac{2}{10^6})$. Out of n songs the probability of none being the same is then a product of the probability of 1 songs being the same, 2 songs, ... , for each i up to $n-1$ songs being the same for each independent event, totalling $(n-1)$ events due to the fact we are considering a song with respect to all other samples. More formally $P(X_n \neq 1) = \prod_{i=1}^{n-1} \big(1 - P(X_i = 1)\big) = \prod_{i=1}^{n-1} (1 - i\frac{1}{10^6})$ which in our case is equal to $\prod_{i=1}^{199} \big(1 - \frac{i}{10^6}\big)$. Taking the largest likelihood for our bound we see that

$$\begin{aligned}
P(X_n \neq 1) =& \prod_{i=1}^{199} \big(1 - \frac{i}{10^6}\big) \\
\geq& \prod_{i=1}^{199} \big(1 - \frac{199}{10^6}\big) && \text{since } (1 - \tfrac{i}{10^6}) \text{ minimized for } i = 199 \\
=& (1 - \frac{199}{10^6})^{199} \\
\geq& (1 - \frac{199^2}{10^6}) && \text{using our hint}
\end{aligned}$$

The probability of this not occurring is then 0.96 and therefore a likelihood of 0.04 that two of the songs would have been the same.

Rubric:

- 1 point using complement rule of probability
- 2 points for finding the probability of n songs with no repetition.
- 2 points for bounding the product (replace i with 199)
- 1 point for calculating the probability
- 1 point for calculating the complement

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 $O(n^{2.36\cdots})$. 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 $a \neq b$.]

Solution.

The inner products of a and b with x amount to a summation of 1's. From this our intuition tells us that the inequality of $\langle a, x\rangle \neq \langle b, x\rangle \pmod 2$ will be dependent on whether or not the inner products of x with a and with b are odd or even valued. We can exploit this by rephrasing the problem due to the expression $\langle a, x\rangle = \langle b, x\rangle($ mod 2) being isomorphic to creating an equivalence $\langle d, x\rangle = 0$ mod 2 where vector d is constructed element-wise such that $d_i \in d$ at index i is such that $d_i = 0$ if $a_i = b_i$ and $d_i = 1$ if $a_i \neq b_i$. We note that the inner product of d with x is odd for half of the random binary vectors x and similarly even for the other half. This is true because for each $x_i \in \{0, 1\}$ in $x \in \{0, 1\}^n$ we would have $d_i = 1$ only if $a_i \neq b_i$ and due to the equal probability that x_i will either be 0 or 1 we as a result have with probability $\frac{1}{2}$ that $\langle d, x\rangle$ is either even or odd. In full the probability of $\langle a, x\rangle \neq \langle b, x\rangle$ (mod 2) is $\frac{1}{2}$ because $\Pr[\langle d, x\rangle = 0(\text{mod } 2)] = 1/2$.

Rubric:

- 1 point for accounting for all elements in vectors a and b
- 3 points for clear, logical argument
- 1 point for reaching 1/2 probability

(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.)

Solution.

See Algorithm 1

Algorithm 1 Randomised Matrix Equality Test

```
Input: Matrix $C$, $A$ and $B$ all in $\mathbf{F}_2$ of size $n \times n$.
Output: Boolean, True or False if $C = AB$ with probability 1/2.
Randomly Initialise vector $x$ with elements $x_i \in \{0, 1\}$
Compute $C' = Cx$                                   ▷ $O(n^2)$
Compute $B' = Bx$                                   ▷ $O(n^2)$
Compute $(AB)' = AB'$        ▷ equivalent to $ABx$ by matrix associative, $O(n^2)$
function RandomisedEquivalenceCheck($C'$, $(AB)'$)
    for $i \in \{0, \cdots n\}$ do
        if $C'[i] \neq (AB)'[i]$ then                ▷ $O(n)$
            Return False
    Return True
Return RandomisedEquivalenceCheck($C'$, $(AB)'$)
Total Complexity: $O(n) + 3O(n^2) = O(n^2)$
```

By part (a) we know the probability of success will be 1/2 if C and AB differ in their respective i'th elements due to a probability of 1/2 that Cx and ABx will differ as odd or even in their ith elements.

Rubric:

- 2 points for correct algorithm
- 2 points for $O(n^2)$ runtime with explanation
- 2 points for argument of 1/2 probability of success (can simply cite part a)

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

Solution.

A robust strength of randomised algorithms is the ability to improve running time by adding random components and then improve probability of success through multiple iterations of the randomised algorithm without damaging running time. In this case to get a probability of success of $\frac{7}{8}$ we would need top only run the algorithm three times. This is due to the fact that for each run we have probability of success of $\geq \frac{1}{2}$. As a result if in fact $C \neq AB$ then probability of $\Pr[\langle C, x\rangle = \langle AB, x\rangle \pmod 2] \leq (1/2)^3 \leq 1/8$ in which the algorithm passes after three runs. We then have probability of determining correctly that $C \neq AB$ successfully with probability 7/8 and only increase our running time by a factor of three.

Rubric:

- 1 points for repeating experiment 3 or more times.
- 2 points for logical explanation