CSE548, AMS542: Analysis of Algorithms, Fall 2017 **Date:** Nov 17

Homework #3

(Due: Nov 29)

RandSelect-1(A, n, k)

(Inputs are an array A of $n > 0$ distinct elements from a totally ordered universe, and a positive integer $k \in [1, n]$. Output is the k-th smallest element of A.)

```
1.  if $k \le \frac{n}{\log n}$ or $k \ge n - \frac{n}{\log n}$ then
2.      compute the $k$-th smallest element $x$ of $A$ in $\mathcal{O}(n)$ time using a standard binary heap
3.      return $x$
4.  else
5.      choose $\lceil \log^2 n \rceil$ elements uniformly at random from $A[1:n]$ (with replacement)
6.      let $S$ be the set of elements chosen in step 5 after removing duplicates
7.      $q \leftarrow |S|$,  $s_0 \leftarrow -\infty$,  $s_{q+1} \leftarrow +\infty$
8.      sort the elements of $S$ in increasing order of value, and
        let $s_1, s_2, \ldots, s_q$ be those elements in sorted order
9.      let $B_i$ be a bin with range $(s_i, s_{i+1}]$, and count $c_i \leftarrow 0$, where $0 \le i \le q$
10.     for $j \leftarrow 1$ to $n$ do
11.         find bin $B_i$ such that $A[j] \in (s_i, s_{i+1}]$
12.         $c_i \leftarrow c_i + 1$
13.     find the smallest $p \in [0, q]$ such that $k \le \sum_{i=0}^{p} c_i$
14.     $T \leftarrow \emptyset$
15.     for $j \leftarrow 1$ to $n$ do
16.         if $A[j] \in (s_p, s_{p+1}]$ then $T \leftarrow T \cup \{A[j]\}$
17.     sort the elements of $T$ in increasing order of value
18.     find the $t$-th smallest element $x$ in the sorted version of $T$, where $t = k - \sum_{i=0}^{p-1} c_i$
19.     return $x$
```

Figure 1: A randomized algorithm for finding the k-th smallest element of a set of n elements chosen from a totally ordered universe.

Task 1. [60 Points] A Simple Randomized Selection Algorithm

Given an array $A[1:n]$ of n elements chosen from a totally ordered universe (e.g., from the set of real numbers) and an integer $k \in [1, n]$ the *selection problem* asks you to find the k-th smallest element in A. We assume for simplicity that all elements in A are distinct. Trivially, one can solve this problem by sorting the elements of $A[1:n]$ in increasing order of value in $\mathcal{O}(n \log n)$ time and reporting $A[k]$. We have seen in the class (see Lecture 6) how to solve this problem in $\Theta(n)$ time using a deterministic recursive divide-and-conquer algorithm.

This task asks you to analyze the running time of a simpler randomized selection algorithm (RandSelect-1) shown in Figure 1.

(*a*) [**5 Points**] Explain how you will implement step 2 of RandSelect-1.

(*b*) [**10 Points**] Prove that w.h.p. in n no element of A is chosen more than once in step 5 of RandSelect-1.

(*c*) [**20 Points**] Prove that in step 17 of RandSelect-1, $|T| = \mathcal{O}\left(\frac{n}{\log n}\right)$ holds w.h.p. in n.

(*d*) [**15 Points**] Give an upper bound (the best you can come up with) on the running time of RandSelect-1 that holds w.h.p. in n.

(*e*) [**10 Points**] Does RandSelect-1 ever fail to produce an answer? Why or why not? Is the answer it produces always guaranteed to be correct? Why or why not?

Task 2. [**75 Points**] **A Not So Simple Randomized Selection Algorithm**

This task asks you to analyze RandSelect-2 shown in Figure 2 which is a modified version of RandSelect-1 shown in Figure 1. Steps 1–9 of both algorithms are exactly the same. Differences start after step 9.

(*a*) [**10 Points**] Prove that after the loop in steps 11–12 terminates $|A'| = \Theta\left(\frac{n}{\log\log n}\right)$ holds w.h.p. in n.

(*b*) [**15 Points**] Let $\Delta = k_{\text{right}} - k_{\text{left}}$. Show that w.h.p. in n, $2\sqrt{m\log n} \le \Delta \le 2\sqrt{2m\log n}$.

(*c*) [**20 Points**] Let x_{left} (resp. x_{right}) be the k_{left}-th (resp. k_{right}-th) smallest element of A' after the termination of the loop in steps 11–12. Prove that w.h.p. in n, A does not have more than $4\left(\sqrt{2m\log n}\right)\log\log n$ elements with value in $[x_{\text{left}}, x_{\text{right}}]$.

(*d*) [**10 Points**] Prove that in step 22 of RandSelect-2, $|T| = \mathcal{O}\left(\frac{n}{\log n}\right)$ holds w.h.p. in n.

(*e*) [**10 Points**] Give an upper bound (the best you can come up with) on the running time of RandSelect-2 that holds w.h.p. in n.

(*f*) [**10 Points**] Does RandSelect-2 ever fail to produce an answer? Why or why not? Is the answer it produces always guaranteed to be correct? Why or why not?

```
RandSelect-2( $A$, $n$, $k$ )
(Inputs are an array $A$ of $n > 0$ distinct elements from a totally ordered universe, and a positive integer $k \in [1, n]$.
Output is the $k$-th smallest element of $A$.)
 1. if $k \leq \frac{n}{\log n}$ or $k \geq n - \frac{n}{\log n}$ then
 2.     compute the $k$-th smallest element $x$ of $A$ in $\mathcal{O}(n)$ time using a standard binary heap
 3.     return $x$
 4. else
 5.     choose $\lceil \log^2 n \rceil$ elements uniformly at random from $A[1:n]$ (with replacement)
 6.     let $S$ be the set of elements chosen in step 5 after removing duplicates
 7.     $q \leftarrow |S|$,  $s_0 \leftarrow -\infty$,  $s_{q+1} \leftarrow +\infty$
 8.     sort the elements of $S$ in increasing order of value, and
        let $s_1, s_2, \ldots, s_q$ be those elements in sorted order
 9.     let $B_i$ be a bin with range $(s_i, s_{i+1}]$, and count $c_i \leftarrow 0$, where $0 \leq i \leq q$
10.     $A' \leftarrow \emptyset$
11.     for $j \leftarrow 1$ to $n$ do
12.         $A' \leftarrow A' \cup \{A[j]\}$ with probability $\frac{1}{\log \log n}$
13.     for each $z \in A'$ do
14.         find bin $B_i$ such that $z \in (s_i, s_{i+1}]$
15.         $c_i \leftarrow c_i + 1$
16.     $m \leftarrow |A'|$,  $k' \leftarrow m \times \frac{k}{n}$,  $k_{\text{left}} \leftarrow k' - 2\sqrt{k' \log n}$,  $k_{\text{right}} \leftarrow k' + 2\sqrt{(m - k') \log n}$
17.     find the smallest $l \in [0, q]$ such that $k_{\text{left}} \leq \sum_{i=0}^{l} c_i$
18.     find the smallest $r \in [0, q]$ such that $k_{\text{right}} \leq \sum_{i=0}^{r} c_i$
19.     $T \leftarrow \emptyset$
20.     for $j \leftarrow 1$ to $n$ do
21.         if $A[j] \in (s_l, s_{r+1}]$ then $T \leftarrow T \cup \{A[j]\}$
22.     sort the elements of $T$ in increasing order of value
23.     find the $t$-th smallest element $x$ in the sorted version of $T$, where $t = k - \sum_{i=0}^{l-1} c_i$
24.     return $x$
```

Figure 2: Another randomized algorithm for finding the k-th smallest element of a set of n elements chosen from a totally ordered universe.

Task 3. [45 Points] Flipping Coins in NCS 220

This task has been picked directly from the nightmares of a SBUCS PhD student who took his RPE[1] in NCS 220. As you may have already guessed, the exam didn't go well. ☹ He was highly competent in the subject matter and could have aced the exam easily. But something totally unexpected got in his way – he simply couldn't figure out how to activate both the projector and the projection screen on the same side of the room. He could activate either the left projector and the right screen or the right projector and the left screen or both projectors and no screen or no projector and both screens or no projector and no screen. But no logical combination of pressing the buttons on the control panel gave him any useful projector/screen combination. Seemed like a conspiracy of some sort. Luckily, he managed to pass the exam (though barely). But this traumatic experience triggered recurring nightmares which continued till the day of his defense.

"Day after day it reappears

Night after night my heartbeat shows the fear

Ghosts appear and fade away"[2]

He dreamt of an NCS 220 with a ridiculously complicated control panel. There were thousands of buttons with thousands of equipment to control. Pressing (i.e., turning on) a button B_i activated a subset $S_i^{(\text{on})}$ of the equipment, but pressing it again (i.e., turning it off) activated another subset $S_i^{(\text{off})}$. The two subsets were disjoint, i.e., $S_i^{(\text{on})} \cap S_i^{(\text{off})} = \emptyset$. When B_i activated $S_i^{(\text{off})}$ it deactivated $S_i^{(\text{on})}$ and vice versa. Between 3 and 5 buttons could be used to activate/deactivate each equipment. For example, equipment E_{137} could be activated by either turning B_{29} "on" or turning B_{751} "off" or turning B_{335} "off" or turning B_{4018} "on". Turning one button "on" or "off" to activate one particular equipment deactivated a number of other equipment that were not already activated by other buttons.

The dream turned into a nightmare because for no good reason someone in his dream challenged him to activate more than 95% of the equipment simultaneously! He was told that exactly 10% of the equipment had 3 control buttons each, 20% had 4 each, and each of the remaining 70% had 5 control buttons. Each button controlled at least one equipment.

Failing to device any deterministic algorithm for solving the problem, he came up with a simple randomized algorithm which is as follows. For each button flip a fair coin. Turn the button "on" if the coin turns up heads and turn it "off" otherwise. Let's call this algorithm $\mathcal{RAND}$-$\mathcal{NCS}$.

Let n be the total number of equipment controlled by the panel.

(*a*) [**5 Points**] Prove that the expected number of equipment activated by $\mathcal{RAND}$-$\mathcal{NCS}$ is $\frac{61}{64}n$ (i.e., 95.3125% of n).

(*b*) [**15 Points**] Prove that the probability that $\mathcal{RAND}$-$\mathcal{NCS}$ at least $\frac{61}{64}n$ equipment is at least $\frac{1}{64n}$.

[1] Research Proficiency Exam

[2] Colin Hay; Overkill (Men at Work)

(*c*) [**10 Points**] Use your results from part 3(*b*) to design an algorithm that activates at least $\frac{61}{64}n$ equipment w.h.p. in n. What is the running time of your algorithm?

(*d*) [**10 Points**] Given any $\epsilon \in \left(0, \frac{61}{64}\right)$, prove that $\mathcal{RAND}$-$\mathcal{NCS}$ activates at least $\left(\frac{61}{64} - \epsilon\right) n$ equipment with probability at least ϵ.

(*e*) [**5 Points**] Use your results from part 3(*d*) to design an algorithm that activates at least $\left(\frac{61}{64} - \epsilon\right) n$ equipment w.h.p. in n. What is the running time of your algorithm?