CSCI3160: Regular Exercise Set 1

Prepared by Yufei Tao

Problem 1. Recall that our RAM model has been extended with an atomic operation RANDOM(x, y) which, given integers x, y, returns an integer chosen uniformly at random from $[x, y]$. Suppose that you are allowed to call the operation *only* with $x = 1$ and $y = 128$. Describe an algorithm to obtain a uniformly random number between 1 and 100. Your algorithm must finish in $O(1)$ expected time.

Problem 2*. Suppose that we enforce an even harder constraint that you are allowed to call RANDOM(x, y) *only* with $x = 0$ and $y = 1$. Describe an algorithm to generate a uniformly random number in $[1, n]$ for an arbitrary integer n. Your algorithm must finish in $O(\log n)$ expected time.

Problem 3. Consider the following algorithm to find the greatest common divisor of n and m where $n \le m$:

> **algorithm** $GCD(n, m)$
>   **if** $n = 0$ **then**
>   **return** m
> $m = m - n$
>   **if** $n \le m$ **then return** $GCD(n, m)$
>   **else return** $GCD(m, n)$

Prove:

1. The time complexity of the algorithm is $O(m)$.
2. The time complexity of the algorithm is $\Omega(m)$.

Problem 4. For the k-selection problem, consider an input array A that has $n = 120$ elements. Our randomized algorithm selects a number v, and recurse into a smaller array A' if the rank of v is within $[n/3, 2n/3] = [40, 80]$. For $k = 20$, what is the probability that the size of A' is at most 60?

Problem 5 (A Simpler Randomized Algorithm for k-Selection, but with a More Tedious Analysis).** In the k-selection problem, we have an array S of n distinct integers (not necessarily sorted). We would like to find the k-th smallest integer in S where $k \in [1, n]$. Here is another way of solving it using randomization. If $n = 1$, then we simply return the only element in S. For $n > 1$, we proceed as follows:

- Randomly pick an integer v in S, and obtain the rank r of v in S.
- If $r = k$, return v.
- If $r > k$, produce an array S' containing the integers of S that are smaller than v. Recurse by finding the k-th smallest in S'.
- Otherwise, produce an array S' containing the integers of S that are larger than v. Recurse by finding the $(r - k)$-th smallest in S'.

Prove that the above algorithm finishes in $O(n)$ expected time.