Prof. Dr. Susanne Albers
Dr. Suzanne van der Ster
Dario Frascaria
Lehrstuhl für Theoretische Informatik
Institut für Informatik
Technische Universität München

Fall Semester
November 2, 2015

Randomized Algorithms

Exercise Sheet 3

Due: November 9, 2015
at 10:15, in class

Exercise 3.1 (10 points)
I am playing in a tennis tournament, and I am up against a player I have never played before. I consider three possibilities for my prior model: we are equally talented, and each of us is equally likely to win each game; I am slightly better, and therefore I win each game independently with probability 0.6; or he is slightly better, and thus he wins each game independently with probability 0.6. Before we play, I think that each of these three possibilities is equally likely.

In our match we play until one player wins three games. I win the second game, but my opponent wins the first, third, and fourth game. After this match, in my posterior model with what probability should I believe that my opponent is slightly better than I am?

Exercise 3.2 (10 points)
Consider the Randomized Selection algorithm (or Find algorithm) presented in class for finding the k-th smallest element in an array with n distinct elements. At each step, the algorithm goes from a sub-problem of size m to a sub-problem of size $m - X$, where X is a random variable.

(a) Show that $E[X] \geq g(m)$, where $g(m) = \frac{m}{4}$.

In class, the fact that in each recursive call to the algorithm, the size of the sub-problem studied is decreased by a factor of at least $\frac{1}{4}$ (i.e., $E[X] \geq \frac{m}{4}$) was used to show that the expected number of recursive calls performed by the algorithm is at most $4 \ln n$.

(b) Now use both facts to show that the expected running time of Randomized Selection is $O(n)$.

Exercise 3.3 (10 points)
Consider again the Randomized Selection algorithm. In the previous exercise you showed that the expected running time of the Randomized Selection algorithm is $O(n)$. Recall that one important fact in the analysis of the algorithm is that in each recursive call to the algorithm the expected reduction in the problem size is a factor of at least $1/4$.

Now recall the Randomized Quicksort algorithm from Week 1. Observe that this algorithm works very similar to Randomized Selection. When confronted with a sub-problem of size m, it also picks a random element e and partitions the other elements into two subsets, those that are bigger and those that are smaller than e. However, as you remember from Week 1, the running time of Randomized Quicksort is $O(n \log n)$ and not $O(n)$.

Explain what the difference between the Randomized Selection and Randomized Quicksort algorithms is that causes different running times. Then verify, by using probabilistic recurrence, that the running time of Randomized Quicksort is $O(n \log n)$.
Hint: Denote by $T(n)$ the running time of Randomized Quicksort for an array of n elements. Write down a recurrence relation and show that the recurrence holds for $T(n) \leq c_1 n + c_2 n \log n$, for some constants $c_1, c_2 > 0$.
Note that copying the proof given in Week 1 for the running time of Randomized Quicksort will not yield any points.

Exercise 3.4 (10 points)
Show that $\mathbf{P} \subseteq \mathbf{RP} \subseteq \mathbf{NP}$.

Recall that the complexity class **NP** contains the languages that can be verified by a polynomial-time algorithm. Specifically, **NP** consists of every language L which has a polynomial-time verification algorithm A such that for every string x

$$x \in L \Rightarrow \exists \text{ certificate } y \text{ such that } A(x, y) \text{ accepts}$$

$$x \notin L \Rightarrow \forall \text{ certificate } y,\ A(x, y) \text{ rejects}$$