Homework #3
Introduction to Algorithms/Algorithms 1
600.363/463
Spring 2013

Due on: Tuesday, Feb 19th, 5pm
Late submissions: will NOT be accepted
Format: Please start each problem on a new page.
Where to submit: On blackboard, under student assessment
Please type your answers; handwritten assignments will not be accepted.
To get full credit, your answers must be explained clearly,
with enough details and rigorous proofs.

January 21, 2013

1 Problem 1 (20 points)

Assume that there are N robots $R_1, \ldots, R_N$ and N tasks, $T_1, \ldots, T_N$. Typically, robot R_i performs task T_i. Also, the power of robots grows with their index. Thus, robot R_i can perform any task T_j without a failure for $j \leq i$ but it will fail if $i < j$. As a result of a program bug, all tasks have been permuted randomly and then assigned to robots. That is, R_i performs task $T_{\pi(i)}$, where π is a random permutation of the numbers $\{1, 2, \ldots, N\}$.

1. What is the expected number of robots that perform their original tasks?
2. What is the expected number of failures?

2 Problem 2 (20 points)

2.1 (10 points)

Resolve the following recurrences. Use Master theorem, if applicable. In all examples assume that $T(1) = 1$. To simplify your analysis, you can assume that $n = a^k$ for some a, k.

1. $T(n) = 5T(n/2) + \sqrt{n}$
2. $T(n) = T(n/2) + 10$
3. $T(n) = 200T(\sqrt{n}) + n$
4. $T(n) = 12T(n/12) + n^2$
5. $T(n) = T(n/200) + n^{200}$
6. $T(n) = n + T(n-1)$
7. $T(n) = 50T(n/45) + n^3$
8. $T(n) = \sqrt{n}T(n/2)$
9. $T(n) = 5T(n/4) + n$
10. $T(n) = 9T(n/3) + n^2$

2.2 (10 points)

Imagine abstract problem A with the input of size n. You and your friends came up with the following four algorithms that solve A:

1. Algorithm X divides A into 5 subproblems of half the size, recursively solves each subproblem and then combines the solutions in quadratic time.
2. Algorithm Y divides A into 1 subproblem of size $n - 2$, recursively solves the subproblem and then derives the solution in linear time.
3. Algorithm Z divides A into 2 subproblems of size $n - 1$, recursively solves each subproblem and then combines the solutions in constant time.
4. Algorithm W divides A into 100 subproblems of size $n/1000$, recursively solves each subproblem and then combines the solutions in linear time.

Which algorithm you should choose and why?

3 Optional Exercises

Solve the following problems and exercises from CLRS: 4-3, 4-1, 7-3.