Algorithm Design, Anal. & Imp., Homework 1

Out: Jan 22, 2015, Due: Feb 04, 2015 (beginning of the class), Total: 80

Note:

- This homework will carry 8 points towards your final score
- Please answer all the questions below. Please type or write legibly. Submit your homework in class at the beginning of the class.
- For any solution to an algorithm problem, you always need to provide correctness proof and complexity analysis, by default.
- Homeworks are individual work, please do not collaborate with others inside or outside of the class. Do not take help from solutions available online.
- Please post your questions on piazza forum; the instructor or the TAs will help you actively. Also use the office hours of instructor and the TAs.

Questions

1: Sort the following functions by increasing order of complexity; if applicable, you need to prove stronger results that involve "small-o" or "small-ω" notations. Also, show your work with detailed explanation, just writing the order will not provide you any credit.
$n!$, 2^n, $n^{\lg n}$, $n \lg n$, $n\sqrt{n}$ (10)

2: Use Integration bound to prove the followings (6)

a.

$$\sum_{k=2}^{n-1} k \lg k \leq \frac{1}{2} n^2 \lg n - \frac{n^2}{4}$$

b.

$$\lg(n!) = O(n \lg n)$$

3: Solve the problem C.4-6 (CLRS Page 1207) (10)

4: Prove that in the array P in procedure PERMUTE-BY-SORTING (CLRS Page 125), the probability that all elements are unique is at least $1 - 1/n$. (hint: we are not looking for the exact probability, rather we are looking for a lower bound, so you can simplify your probability expression by choosing a suitable lower bound, the inequality $(1-x)^n \geq 1 - nx$ can be useful) (10)

5: Selection sort algorithm is Given below. Prove the correctness of this algorithm using loop invariant method. Considering uniform permutation, find the expected number of comparisons and the expected number of swap operations. (10)

```
SELECTION-SORT(A)
n = A.size()
for i = 1 to n − 1:
        m = i
        for j = i + 1 to n:
                if A[m] > A[j]
                        m = j
        if m > i
                SWAP(A[i], A[m])
```

6: Bubble sort algorithm is given below. Prove the correctness of this algorithm using loop invariant method. Considering uniform permutation, find the expected number of comparisons, and the expected number of swap operations. (10)

```
BUBBLE-SORT(A)
n = A.size()
for i = 1 to n − 1:
        for j = n downto i + 1:
                if A[j − 1] > A[j]
                        SWAP(A[j − 1], A[j])
```

7: Suppose we want to create a random subset of size m $(0 \leq m \leq n)$ of the set $\{1, 2, 3, \ldots, n\}$, where each m-subset is equally likely to be created. For this, we write the following recursive procedure:

```
RANDOM-SAMPLE(m, n)
if m == 0
      return ∅
else
      S = RANDOM-SAMPLE(m − 1, n − 1)
      i = RANDOM(1, n)
      if i ∈ S
            S = S ∪ {n}
      else
            S = S ∪ {i}
      return S
```

Now answer the following question:

- **a.** Prove that the above algorithm is correct, i.e, the probability of creating each of the m-size subset is equal. (7)
- **b.** Perform a complexity analysis of this algorithm. (3)

8. For the following recurrences, find a good asymptotic upper bound using recursion tree method (6)

- **a.** $T(n) = 4T(n/2) + n$
- **b.** $T(n) = 2T(n/2) + n \lg n$

9: For the following recurrences, find a good asymptotic upper bound using Master method (if applies) and substitution method (8)

- **a.** $T(n) = 4T(n/2) + n^2\sqrt{n}$
- **b.** $T(n) = 4T(n/3) + n \lg n$
- **c.** $T(n) = 2T(n/2) + n \lg n$
- **d.** $T(n) = 2T(n/4) + \sqrt{n}$