Design and Analysis of Algorithms 2013 (Home work 3)

October 14, 2013

- Due on, October 30, before 10 a.m.
- Late home works will not be accepted.
- Please give precise arguments for all statements that you write.
- Please do not hesitate to contact me if you do not understand the problems.
- Collaboration is discouraged, but not prohibited. It is recommended that you try to solve the problems on your own. You can discuss the questions with your colleagues but you should not copy solutions. Always write down your own answers. If copying is detected that may immediately lead to a grade less than 7. (**This would be followed strictly**)
- Credits would be given to partial solutions also.
- When you write an algorithm, you should briefly discuss the main idea of your algorithm, then write a pseudo code, argue about its correctness and state and prove the running time of your algorithm.
- The answers should be typed or written clearly and a hard copy is to be submitted.

1. [**10 points**] Give asymptotic tight bounds for $T(n)$ for the following cases. Assume that $T(n)$ is a constant if $n \leq 2$.
   (a) $T(n) = 2T(n/2) + n^3$.
   (b) $T(n) = 7T(n/3) + n^2$.
   (c) $T(n) = 3T(n/2) + n \lg n$.
   (d) $T(n) = 2T(n-1) + 1$.
2. [**20 points**] Give asymptotic tight bounds for $M(n)$ for the following cases (i.e., find $f(n)$ such that $M(n) = \Theta(f(n))$). Justify your answer, i.e., you should show by the substitution method, why you think your bound is correct.

   (a) $M(n) = M(n/4) + M(n/2) + n^2$.
   (b) $M(n) = 2M(n/2) + n \lg n$.

3. **[10 points]**You are given an array A with n distinct elements. You are also told that the sequence of values $A[1], A[2], \ldots, A[n]$ is unimodal: For some index p between 1 and n, the values in the array entries increase up to a position p in A and then decrease the remainder of the way until position n. Give a $O(\lg n)$ algorithm to find the maximum of the array.

4. **[10 points]**Given an array A, we call a pair (i, j) as a *significant inversion* if $i < j$ and $A[i] > 2A[j]$. Give a $O(n \lg n)$ algorithm to count the significant inversions in A.

5. **[10 points]** Give a divide and conquer algorithm to multiply two polynomials of degree $n - 1$ which run in time $\Theta(n^{\lg 3})$.

6. **[10 points]** Given two sorted arrays A and B such that all elements in A and B are distinct and each contain n elements. Give an $O(\lg n)$ algorithm to find the median of all the elements present in A and B.

7. **[10 points]** Given an array A with n elements we would like to know whether there is an element x in A which occurs more than $n/2$ times. Give an $O(n)$ algorithm to do this.

8. **[10 points]** Given two arrays A and B of size n and a number c, design an algorithm which decides whether there exists $i, j \in \{1, 2, \ldots, n\}$, such that $A[i] + B[j] = c$. Your algorithm should run in $O(n \log n)$ time.

9. **[20 points]**Consider the following sorting algorithm called StrangeSort:

```
StrangeSort($A, i, j$)

1. if $A[i] > A[j]$;
2.     exchange $A[i] \leftrightarrow A[j]$
3. if $i + 1 \geq j$
4.     return
5. $k \leftarrow \lfloor (j - i + 1)/3 \rfloor$
6. StrangeSort($A, i, j - k$)
7. StrangeSort($A, i + k, j$)
8. StrangeSort($A, i, j - k$)
```

   (a) Argue that if $n =$ length$[A]$, then StrangeSort$(A, 1, n)$ correctly sorts the input array $A[1, 2, \ldots, n]$.
   (b) Give a recurrence for the worst case running time of StrangeSort and a tight asymptotic bound on the worst case running time.