Algorithms Analysis and Design: 2620301
Chongqing University of Technology — Oct 13, 2019
Lecturer Gou Guanglei — Handout 1

Problem Set 1

This problem set is due **Next Sunday at 11:59AM**.
Solutions should be turned in PDF form using LaTeX or word document.

Exercises are for extra practice and should not be turned in.
Exercises:

- Implement Insertion-sort.
- Implement Binary Search.
- Implement Merge-sort.

---

1. **(25 points)** Suppose you are choosing between the following three algorithms:

   Algorithm A solves problems by dividing them into five subproblems of half the size, recursively solving each subproblem, and then combining the solutions in linear time.

   Algorithm B solves problems of size n by recursively solving two subproblems of size $n-1$ and then combining the solutions in constant time.

   Algorithm C solves problems of size n by dividing them into nine subproblems of size $n/3$, recursively solving each subproblem, and then combining the solutions in $O(n^2)$ time.

   What are the running times of each these algorithms(in big-O) notation), and which would you choose?

2. **(25 points)** Asymptotic analysis

   (a) **15pts** Solve the recurrence $T(n) = 3T(n/4) + O(\sqrt{n})$. What is the general k-th term in this case? And what value of k should be plugged in to get the answer?

   (b) **10pts** Now try the recurrence $T(n) = T(n-1) + O(1)$, a case which is not covered by the master theorem. Can you solve this too?

3. **(22 points)** Solve the following recurrence relation and give a θ bound for each of them.

   (a) $T(n) = 2T(n/3) + 1$

   (b) $T(n) = 6T(n/5) + n$

(c) $T(n) = 7T(n/7) + n$

(d) $T(n) = 4T(n/2) + n^2$

(e) $T(n) = 8T(n/2) + n^3$

(f) $T(n) = 49T(n/25) + n^{3/2}logn$

(g) $T(n) = T(n-1) + 2$

(h) $T(n) = 6T(n-1) + n^c$, where $c \geq 1$ is a constant

(i) $T(n) = 6T(n-1) + c^n$,where $c \geq 1$ is some constant

(j) $T(n) = 2T(n-1) + 1$

(k) $T(n) = T(\sqrt{n}) + 1$

4. **(23 points) Unimodal Search**

An array $A[1...n]$ is **unimodal** if ti consists of an increasing sequence followed by a decreasing sequence, or more precisely, if there is an index $m \in \{1, 2, ..., n\}$ such that

- $A[i] < A[i+1]$ for all $1 \leq i < m$, and
- $A[i] > A[i+1]$ for all $m \leq i < n$

In particular, $A[m]$ is maximum element, and it is the unique "local maximum" element surrounded by smaller elements($A[m-1]$ and $A[m+1]$).

(a) 10ptsGiven an algorithm to compute the maximum element of a unimodal input array $A[1, ..., n]$ in $O(lgn)$ time.

(b) 13pts Prove the bound on its running time.