Problem Set 5

Ekkapot Charoenwanit
Efficient Algorithms

September 10, 2020

Problem 5.1. Binary Insertion Sort

1) In the vanilla version of insertion sort we discussed in the lecture, at any iteration i, we have to traverse the entire subarray $A[1...i-1]$ in the worst case to look for the correct position to place the key $A[i]$ into. How do you apply binary search to improve this? Also analyze the running time in the worst case of your binary insertion sort algorithm.

2) In your binary insertion sort algorithm that you proposed in 1), how many comparisons do you need in the worst case and how many swaps do you need in the worst case?

Problem 5.2. Quick Sort

1) What is the running time of quick sort when all the array elements have the same value?

2) Modify the algorithm of quick sort presented in the lecture so that it sorts an array in non-increasing order.

3) Under what situations does the running time of quick sort end up in the worst case?

Hint: Look at the recurrence of the running time of quick sort and see what causes such a worst-case situation.

Problem 5.3. Strassen's Algorithm

1) How do you apply Strassen's Algorithm to multiply two $n \times n$ matrices where n is not an exact power of two?

2) What is the running time $T(n)$ of the modified Strassen's algorithm in asymptotic notation?

Problem 5.4. Recurrences and Master Theorem

1) Write a recurrence for the number of scalar additions for Strassen's Algorithm using asymptotic notation (not in the exact form).

2) Solve the recurrence using the recursion tree method and confirm your answer with Master theorem.