Exercise II, Algorithms 2024-2025

These exercises are for your own benefit. Feel free to collaborate and share your answers with other students. There are many problems on this set, solve as many as you can and ask for help if you get stuck for too long. Problems marked * are more difficult but also more fun :).

These problems are taken from various sources at EPFL and on the Internet, too numerous to cite individually.

1 (Exercise 2.3-4 in the book) We can express insertion sort as a recursive procedure as follows. In order to sort $A[1 \dots n]$, we recursively sort $A[1, \dots n-1]$ and then insert $A[n]$ into the sorted array $A[1 \dots n-1]$.

1a Write a recurrence for the worst-case running time of this recursive version of insertion sort.

1b Solve the recurrence to give a tight asymptotic bound (in terms of the Θ-notation).

2 Use the substitution method (mathematical induction) for solving the following recurrences

2a Show that when n is an exact power of 2, the solution of the recurrence

$$T(n) = \begin{cases} 2 & \text{if } n = 2, \\ 2T(n/2) + n & \text{if } n = 2^k, \text{for } k > 1 \end{cases}$$

is $T(n) = n \log_2 n$.

2b Consider the recurrence $T(n) = 8T(n/2) + \Theta(n^2)$.

- Draw the recurrence tree to make a qualified guess of the value of $T(n)$ in terms of the Θ-notation.
- (half a $*$) Then verify this guess using the substitution method. Ask the TA's for a hint if you get stuck for too long.

Solution: Hint: we should use $T(n) \leq dn^3 - d'n^2$ instead of only $T(n) \leq dn^3$ when proving that $T(n) = O(n^3)$.

3 (Exercise 4.5-1 in the book) Use the master method to give tight asymptotic bounds for the following recurrences.

3a $T(n) = 2T(n/4) + 1$.

3b $T(n) = 2T(n/4) + \sqrt{n}$.

3c $T(n) = 2T(n/4) + n$.

3d $T(n) = 2T(n/4) + n^2$.

4 Let $f(n)$ and $g(n)$ be the functions defined for positive integers as follows:

```
Function f(n):
 1: ans ← 0
 2: for i = 1, 2, ..., n − 1 do
 3:     for j = 1, 2, ..., n − i do
 4:         ans ← ans + 1
 5:     end for
 6: end for
 7: return ans
```

```
Function g(n):
 1: if n = 1 then
 2:     return 1
 3: else
 4:     return g(⌊n/2⌋) + g(⌊n/2⌋)
 5: end if
```

4a Find closed-form formulas of $f(n)$ and $g(n)$.

4b What is, in Θ-notation, the running time of these algorithms?

4c What is, in Θ-notation, the *running time* of algorithm $g(n)$ if we at line 4 replace $g(\lfloor n/2 \rfloor) + g(\lfloor n/2 \rfloor)$ by $2g(\lfloor n/2 \rfloor)$?

5 (Problem 2-1 in the book) Although merge sort runs in $\Theta(n \log n)$ worst-case time and insertion sort runs in $\Theta(n^2)$ worst-case time, the constant factors in insertion sort make it faster for small n. Thus, it makes sense to use insertion sort within merge sort when subproblems become sufficiently small. Consider a modification to merge sort in which n/k sublists of length k are sorted using insertion sort and then merged using the standard merging mechanism, where k is a value to be determined.

5a Show that the n/k sublists, each of length k, can be sorted by insertion sort in $\Theta(nk)$ worst-case time.

5b Show that the sublists can be merged in $\Theta(n \log(n/k))$ worst-case time.

5c Given that the modified algorithm runs in $\Theta(nk + n \log(n/k))$ worst-case time, what is the largest asymptotic value of k, as a function of n, for which the modified algorithm has the same asymptotic running time as standard merge sort?

5d How should k be chosen in practice?

6 (*, Problem 2-4 in the book) Let $A[1 : n]$ be an array of n distinct numbers. If $i < j$ and $A[i] > A[j]$, then the pair (i, j) is called an inversion of A.

6a List the five inversions of the array $\langle 2, 3, 8, 6, 1 \rangle$.

6b What array with elements from the set $\{1, 2, ..., n\}$ has the most inversions? How many does it have?

6c What is the relationship between the running time of insertion sort and the number of inversions in the input array? Justify your answer.

6d Give an algorithm that determines the number of inversions in any permutation on n elements in $\Theta(n \log n)$ worst-case time. (Hint: Modify merge sort.)