Exercise Sheet 7
COMS10018 Algorithms 2024/2025

Reminder: $\log n$ denotes the binary logarithm, i.e., $\log n = \log_2 n$.

1 Countingsort and Radixsort

1. We use Countingsort to sort the following array A:

4	2	2	0	1	4	2

   Answer the following questions:

   (a) What is the state of the auxiliary array C after the second loop of the algorithm?

   (b) What is the state of C after each iteration i of the third loop?

2. Illustrate how Radixsort sorts the following binary numbers:

   100110 101010 001010 010111 100000 000101

3. Radixsort sorts an array A of length n consisting of d-digit numbers where each digit is from the set $\{0, 1, \dots, b\}$ in time $O(d(n + b))$.

   We are given an array A of n integers where each integer is *polynomially bounded*, i.e., each integer is from the range $\{0, 1, \dots, n^c\}$, for some constant c. Argue that Radixsort can be used to sort A in time $O(n)$.

   Hint: Find a suitable representation of the numbers in $\{0, 1, \dots, n^c\}$ as d-digit numbers where each digit comes from a set $\{0, 1, \dots, b\}$ so that Radixsort runs in time $O(n)$. How do you chose d and b?

2 Loop Invariant for Radixsort

Radixsort is defined as follows:

Require: Array A of length n consisting of d-digit numbers where each digit is taken from the set $\{0, 1, \dots, b\}$
for $i = 1, \dots, d$ **do**
  Use a stable sort algorithm to sort array A on digit i
end for

(least significant digit is digit 1)

In this exercise we prove correctness of Radixsort via the following loop invariant:

> At the beginning of iteration i of the for-loop, i.e., after i has been updated in Line 1 but Line 2 has not yet been executed, the following holds:
>
> The integers in A are sorted with respect to their last $i-1$ digits.

1. *Initialization:* Argue that the loop-invariant holds for $i = 1$.
2. *Maintenance:* Suppose that the loop-invariant is true for some i. Show that it then also holds for $i+1$.

   Hint: You need to use the fact that the employed sorting algorithm as a subroutine is stable.
3. *Termination:* Use the loop-invariant to conclude that A is sorted after the execution of the algorithm.

3 Recurrences: Substitution Method

1. Consider the following recurrence:

   $$T(1) = 1 \text{ and } T(n) = T(n-1) + n$$

   Show that $T(n) \in O(n^2)$ using the substitution method.
2. Consider the following recurrence:

   $$T(1) = 1 \text{ and } T(n) = T(\lceil n/2 \rceil) + 1$$

   Show that $T(n) \in O(\log n)$ using the substitution method.

   Hint: Use the inequality $\lceil n/2 \rceil \le \frac{n}{\sqrt{2}} = \frac{n}{2^{\frac{1}{2}}}$, which holds for all $n \ge 2$. Use $n = 2$ as your base case.

4 Optional and Difficult Questions

Exercises in this section are intentionally more difficult and are there to challenge yourself.

4.1 Algorithmic Puzzle: Maxima of Windows of length $n/2$

We are given an array A of n positive integers, where n is even. Give an algorithm that outputs an array B of length $n/2$ such that $B[i] = \max\{A[j] \ , \ i \le j \le i + n/2 - 1\}$. Can you find an algorithm that runs in time $O(n)$?