Algorithmics Problem Sheet 2

Jan Kurkofka, Will Turner

Summer 2024

This problem sheet is due on Monday 22nd April. The tutorial will be 14:30-16:00 in LAM-2090. Please have a go at as many problems as you can. We have removed the points system to make the problem sheet more relaxed, and it is also now non-compulsory. As well as going through the problems, the session on Monday can be used to answer any other questions you have about the course.

Problem 1

Recall that lg denotes the binary algorithm, so $\lg n = \log_2 n$. Suppose there are algorithms called ALG1 and ALG2. On an input of size n, ALG1 takes $164n \lg n$ steps to terminate (come to a conclusion), and ALG2 takes $10n^2$ steps. At what sizes of input, does ALG1 terminate more quickly than ALG2?

Problem 2

The following pseudocode describes the algorithm BUBBLESORT, which takes an array $A[1:n]$ of numbers as an input and turns it into a sorted version.

Require: integer $n \geqslant 1$, array $A[1:n]$
for $i = n$ downto 1 **do**
    for $j = i \dots i-1$ **do**
        if $A[j] > A[j+1]$ **then**
            exchange the values in $A[j]$ and $A[j+1]$

By hand, perform bubble sort on the array $A[1:5] = [7, 1, 2, 4, 3]$. Write out $A[1:5]$ after each iteration of the inner for-loop (the one indexed by j).

Problem 3

Consider the following algorithm called sum-array, which computes the sum of all of the numbers in the array.

Require: integer $n \geqslant 1$, array $A[1:n]$
sum $\leftarrow 0$
for $i = 1 \dots n$ **do**
    sum $\leftarrow$ sum $+A[i]$
return sum

Suppose we don't care about the exact value of the sum if it goes above 70. Modify the above pseudocode so that the algorithm terminates with the output "70" if the sum ever goes above the value 70.

Problem 4

Write a short piece of pseudocode that describes how to cook your favourite food. If you can, include a loop (for example, you might need to repeatedly add salt until the food is salty enough).

Problem 5

The decimal number 9 has binary representation 101. Convert the decimal number 42 to a binary representation.

Now write a short piece of pseudocode which reverses this process: Given a binary representation of a number, as an array $A[1:n]$ of 1s and $0s$, output the value of this as a (normal) decimal number.

Problem Sheet 1

This problem sheet is due on Monday 15th April. The tutorial will be 14:30-16:00 in LAM-2090. Please have a go at as many problems as you can. Each question solved gives you the number of points specified. A grade of 100% can be achieved by getting $10 = 16 - 6$ points. Additional points *do not* carry over to the next problem sheet.

Problem 1 — 1 point

Answer in one or two sentences: Other than speed (time efficiency), what other measures of efficiency might you need to consider in a real-world setting?

Problem 2 — 1 point

Answer in one or two sentences: Suggest a real-world problem in which only the best solution will do. Then come up with one for which an approximate solution is good enough.

Problem 3 — 2 points

Answer the following: How could we modify any sorting algorithm so that it always has a good **best-case** running time?

Problem 4 — 3 points

Consider the following method for sorting the numbers in an array $A[1:n]$: Find the smallest element of $A[1:n]$ and exchange it with $A[1]$. Then find the smallest element of $A[2:n]$ and exchange it with $A[2]$. Then exchange the smallest element in $A[3:n]$ with $A[3]$. Continue in this way until the array is sorted. This algorithm is known as *selection sort*.

Complete the following: Write the pseudocode for selection sort. Give the best-case and the worst-case running times in Θ-notation.

Problem 5 — 3 points

Consider the following computational problem: Input an integer n and two n-bit binary numbers, stored as two arrays $A[0:n-1]$ and $B[0:n-1]$. Output the sum of the two numbers as an $(n+1)$-bit binary number, stored as an array $C[0:n]$.

Complete the following: Write an algorithm in pseudocode that solves the above problem. Work out its worst-case running time.

Problem 6 — 3 points

Consider the following algorithm called sum-array, which computes the sum of all of the numbers in the array.

Require: integer $n \geqslant 1$, array $A[1:n]$

sum $\leftarrow 0$
for $i = 1 \ldots n$ **do**
    sum $\leftarrow$ sum $+A[i]$
return sum

Complete the following: State a loop invariant for this procedure. Use it to show that the procedure does what it's meant to.

Problem 7 **3 points**

Code! Implement insertion sort in Python (or a language of your choice).