

Data Structures and Algorithms

Exam

Prof. Dr. M. Böhlen

September 21, 2007

14:00 - 16:00

The exam consists of 4 exercises. For each exercise you can get 25 points. It is important that you argue for your answers and that you present your solutions in a readable form. As auxiliary material you may use 1 A4 sheet with notes.

1 Algorithmic Correctness

Given the following algorithm that selects the k -th biggest integer in a given unsorted array A of length n . For example, if $A = (12, 4, 10, 8, 9, 33, 2, 18)$ then $SelectKth(A, 3) = 8$.

Input: Unsorted array $A[1..n]$ of integers and an integer $k \{1..n\}$.

Output: The k -th integer in A .

SelectKth(A, k):

```
for  $i := 1$  to  $k$  do
   $mini := i$ 
  for  $j := i + 1$  to  $Length[A]$  do
    if  $A[j] < A[mini]$  then
       $mini := j$ 
   $key := A[i]$ 
   $A[i] := A[mini]$ 
   $A[mini] := key$ 
return  $A[k]$ 
```

- 1a) Illustrate the content of array A after the execution of $SelectKth(A, 3)$.
- 1b) State a loop invariant for the inner for loop and show that it holds by discussing initialization, maintenance, and termination.
- 1c) State a loop invariant for the outer for loop and show that it holds by discussing initialization, maintenance, and termination.

2 Joining Lists

Let A_1 be a sorted array of length n_1 . Let A_2 be an unsorted array of length n_2 . A_1 and A_2 store integers and do not contain duplicates.

- 2a) Write an algorithm that prints the integers that are present in both arrays. The asymptotic complexity of your algorithm must be less than $O(n_1 \cdot n_2)$. The arrays may not be changed and your solution may not use more than a constant amount of additional memory.
- 2b) Determine the exact running time of your algorithm.
- 2c) Determine a tight asymptotic bound for the runtime of your algorithm.
- 2d) Discuss what happens if duplicates are present in the arrays. Illustrate the different cases by example. Assume an integer x occurs l times in A_1 and m times in A_2 . How would you have to modify your algorithm such that x is printed $\min(l, m)$ times?

3 Road Network

Several farms are connected with dirt roads as it shown in Figure 1. The edges are labeled with the distances between the farms given in kilometers. The municipality wants to connect the farms with new two-lane concrete roads, but wants to save on the materials. Each farm must be reachable from any other farm by concrete roads.

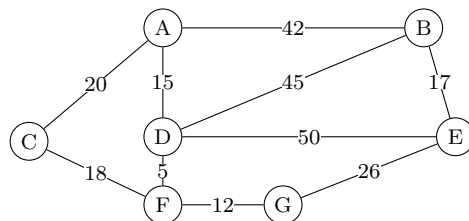


Figure 1: Farms and Connecting Roads

- 3a) Which algorithm should be used to select the roads that have to be upgraded. The selection of roads must satisfy the preferences of the municipality?
- 3b) Illustrate the chosen algorithm step-by-step on the example in Figure 1.
- 3c) State the asymptotic complexity of the algorithm.

4 Binary Number Multiplication

Assume n bit binary numbers x and y . Below we describe three different recursive divide and conquer algorithms to multiply two binary numbers. For each of the algorithms state a recurrence that describes the asymptotic complexity of the algorithm. Solve the recurrences and give a closed form expression for the asymptotic complexity.

For the analysis assume the following. Each addition takes linear time wrt to the length of the integers. A multiplication by 2^i is implemented as a shifting by i positions and takes linear time wrt i .

Algorithm 1 multiplies two binary numbers naively by multiplying each bit with each bit. This process is illustrated in the left part in the figure below.

$$\begin{array}{r}
 10100001 * 11001011 \\
 \hline
 10100001 \\
 10100001 \\
 00000000 \\
 10100001 \\
 00000000 \\
 00000000 \\
 10100001 \\
 10100001 \\
 \hline
 111111110101011
 \end{array}
 \qquad
 x * y = x * y_t + 2 * x * y_h$$

Let y_t be the last bit of y and let y_h be y without its last bit. For example with $y = 11001011$ we get $y_t = 1$ and $y_h = 1100101$. With this the naive multiplication of two binary numbers can be stated as shown in the formula to the right.

Algorithm 2 splits the numbers into smaller parts: $x = x_1x_2$ and $y = y_1y_2$. Assume that the numbers are split into parts of the same length. Thus, if x has n bits then x_1 and x_2 have $n/2$ bits. For example, for $x = 10100001$ we get $x_1 = 1010$ and $x_2 = 0001$. With this the multiplication can be performed as follows:

$$x * y = (x_1 * y_1) * 2^n + (x_1 * y_2 + x_2 * y_1) * 2^{n/2} + x_2 * y_2$$

Algorithm 3 is a further refinement of algorithm 2. Specifically, we rewrite $x_1 * y_2 + x_2 * y_1$ as $((x_1 + x_2) * (y_1 + y_2) - x_1 * y_1 - x_2 * y_2)$ and perform the multiplication as follows:

$$x * y = (x_1 * y_1) * 2^n + ((x_1 + x_2) * (y_1 + y_2) - x_1 * y_1 - x_2 * y_2) * 2^{n/2} + x_2 * y_2$$

Note that terms that occur multiple times only have to be computed once.