

Data Structures and Algorithms
(Algorithms and Complexity)
Exam
9:00 - 11:00, September 16, 2005

Prof. Dr. M. H. Böhlen

The exam consists of 4 exercises. For each exercise you can get 30 points. It is important that you argue for your answers and that you present your solutions in a readable form. Fully specify all algorithms that you use as part of your solutions.

Exercise 1

Assume an array $A[1..n]$ of integers. The array is *balanced* if

1. the sum of the elements in the left half is no more than twice the sum of the elements in the right half, and
2. the sum of the elements in the left half is at least half the sum of the elements in the right half, and
3. the left and right half are balanced.

Based on the above definition solve the following tasks:

1. For each of the following arrays decide if they are balanced:

$$\begin{aligned} A &= [5, 8, 10, 4, 7, 6, 7, 9, 5, 3] \\ B &= [5, 8, 3, 4, 6, 7, 9, 6, 5] \end{aligned}$$

2. Develop a C algorithm that decides if a given array is balanced. Explain the chosen data structures and algorithms. Identify and discuss the handling of border cases.
3. Determine the asymptotic complexity of your algorithm.

Exercise 2

A strongly connected component (SCC) of a directed graph $G = (V, E)$ is a maximal set of vertices $C \subseteq V$ such that from each vertex in C we can reach any other vertex in C .

A classical approach to determine the strongly connected components of a graph is to use two depth first searches (DFS). The first depth first search determines the order in which the nodes have to be visited in the next step. The second depth first search works with the transposed graph and constructs a forest where each tree represent a strongly connected component. The transposed graph G^T can be constructed by reversing all edges in the original graph G . The following abstract algorithm describes the computation of the strongly connected components.

StronglyConnectedComponents:

1. Call $DFS(G)$ to compute finishing times for each vertex of the graph
2. Consider the vertices in order of decreasing finishing times (as computed in step 1.) and call $DFS(G^T)$
3. Output the vertices of each tree in the depth first forest from step 2.

Consider the directed graph in Figure 1.

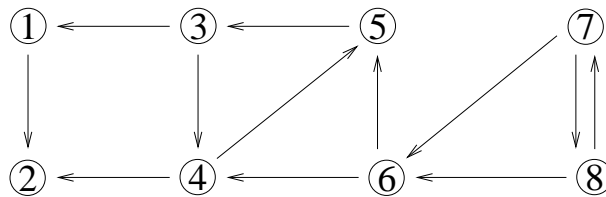


Figure 1: Directed Graph

1. Identify the strongly connected components of the graph in Figure 1.
2. Based on the abstract algorithm given above develop a concrete algorithm that prints the SCC of a graph G . Use pseudocode that can be translated to a C-like language directly. Specify and explain all parts of the algorithm.

Exercise 3

For each of the following functions determine the simplest asymptotically tight bound.

1. $30 \log n + \log 2^n$

2. $20^5 n^3 + 10n \log n + 5$
3. $\frac{5^2}{n+3^2}$

Consider the following recurrence:

$$\begin{aligned} T(1) &= 5 \\ T(n) &= n/3 + 3T(n/3), n > 1 \end{aligned}$$

1. What is the value of $T(n)$ for $n = 27$?
2. Solve the above recurrence exactly by finding a closed form expression for it (without $T(n)$ on the left side of the expression).
3. Use the substitution method to find and prove a tight asymptotic bound for $T(n)$.

Exercise 4

A group of universities is implementing a joint project on distributed multimedia and networking. Towards this goal a computer network shall be established that connects the participating universities through communication links that form a tree. A central file server that allows to share data shall be placed at one of the universities. Since the transmission time is dominated by the link setup and synchronization time, the cost of a data transfer is proportional to the highest number of links used in any peer to peer communication. Hence, it is desirable to choose a central location in the network for the file server. A central location minimizes the number of links from the server to any other university in the network.

For example, if we have universities a_1, a_2, a_3 , and a_4 with links (a_1, a_2) , (a_2, a_3) , and (a_2, a_4) the preferred location for the central file server is a_2 .

1. Suggest a solution to the problem. Precisely state the condition when a university can be hosting the central file server.
2. Design a general algorithm that, given a network of n universities, computes a candidate location for the file server.
3. Is it the case that a network of universities has exactly one candidate location for any n ? Prove uniqueness or give a counter example.
4. Give the worst-case complexity of your algorithm.