

Algorithms and Complexity

Exam

10:00 - 13:00

February 7, 2005

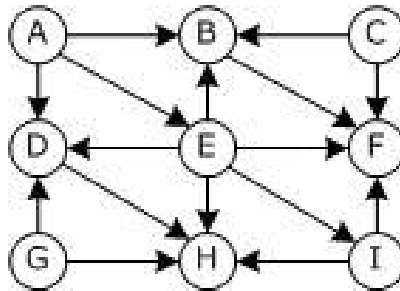
Prof. M. H. Böhlen

The exam consists of 6 exercises. That means you have roughly half an hour per exercise if you solve all of them. Initially, read all exercises carefully. The first exercise is a multiple choice exercise. Solve as many exercises as possible.

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. During the exam no auxiliary material, except a simple pocket calculator, is permitted.

1 Multiple Choice Exercises (22 points)

1.1 Consider the graph below. A topological sorting was run on this graph.



Which of the following orderings of vertices are correct topological sorting of the initial graph?

- a) (G, C, A, E, I, D, H, B, F)
- b) (G, C, H, D, I, E, A, B, F)
- c) (G, C, A, E, D, I, H, B, F)
- d) (D, E, G, C, A, I, H, B, F)

1.2 Assume array $A = (6 \ 2 \ 15 \ 11 \ 81 \ 60 \ 4 \ 39 \ 8 \ 53)$. Which of the following arrays represent valid heaps for A ?

- a) (81 53 60 11 2 15 4 39 8 6)
- b) (81 53 60 39 6 15 4 11 8 2)
- c) (2 53 60 39 6 15 4 11 8 81)
- d) (81 53 60 8 6 15 4 11 39 2)

1.3 Assume a directed graph. Which of the following statements about the depth first search (DFS) algorithm are true?

- a) DFS computes a forest.
- b) DFS computes a tree.
- c) DFS computes the connected components of the graph.
- d) The running time of the DFS algorithm is $O(|V|^2)$.

1.4 List five sorting algorithms together with their worst case asymptotic complexity when run on an array with n elements.

Sorting algorithm	Worst case complexity

1.5 The following strings represent two DNA strands:

$S_1 = \text{ATCGGTCG}$

$S_2 = \text{GTAGTTC}$

Which of the following strings are longest common subsequences (LCS) of S_1 and S_2 ?

- a) TC
- b) GT
- c) ATCTG
- d) ATTC

2 Recurrence (16 points)

Consider the following recurrence:

$$T(1) = 1$$

$$T(n) = T(n - 1) + 2n + 1 \quad (n \geq 2)$$

- What is the value of $T(n)$ for $n = 2, 3, 4$?
- Solve the above recurrence exactly by finding a closed form expression for it (without $T(n)$ on the right side of the expression).
- Find the simplest $f(n)$ such that $T(n) = \Theta(f(n))$.

3 Graphs (20 points)

Consider a directed graph $G = (V, E)$. Each edge $(u, v) \in E$ has an associated value $p(u, v)$, which is a real number in the range $0 \leq p(u, v) \leq 1$. $p(u, v)$ represents the reliability of the communication channel from vertex u to vertex v , and is expressed as the probability that the channel from u to v will not fail. If $p(u, v) = 0$ the channel is unreliable, whereas if $p(u, v) = 1$ the channel is reliable. Assume that the probabilities are independent: given a path $(u, z) = (u, v)(v, z)$ then $p(u, z) = p(u, v) \cdot p(v, z)$.

Give an efficient algorithm to find and print out the most reliable path between two given vertices. Use Dijkstra's shortest path algorithm as a basis for your solution.

4 Dynamic Data Structures, C (20 points)

As an illustrative example application consider a queue in a post office with persons waiting to be serviced. When a new person arrives s/he takes the place at the tail of the queue. The first person served is always the one at the head of the queue. The queue may grow up to MAX persons.

This type of application can be modeled with a FIFO (first in first out) queue. Choose a dynamic data structure to model a FIFO queue and give appropriate C declarations. Give C functions that implement the following operations. The asymptotic complexity of each operation shall be constant.

1. `ISEMPTY()` to check if the queue is empty.
2. `ISFULL()` to check if the persons in the queue fill up all available places.
3. `GET()` to return the name of the first person in the queue and dequeue the person.
4. `PUT(NAME)` to enqueue a new person.

Assume that names are single characters.

5 Dynamic Programming (20 points)

A recursive solution for the k th Fibonacci number is given as follows:

```
FibRec(k)
  if (k<2) then return k
  else return FibRec(k-1) + FibRec(k-2)
```

1. Determine the complexity of the recursive solution given above.
2. Use dynamic programming techniques to compute the k th Fibonacci number. Give a dynamic programming algorithm to compute the k th Fibonacci number and determine its complexity.
3. Use the memoization technique to compute the k th Fibonacci number. Give a memoization algorithm to compute the k th Fibonacci number and determine its complexity.

6 Binary Search Tree (25 points)

A binary search tree is a binary tree with the following properties:

1. Each node stores a search key k .
2. Keys stored in the left subtree of a node with key k are less than or equal to k .
3. Keys stored in the right subtree of a node with key k are greater than or equal to k .

Assume a binary search tree with positive integers as keys. The binary search tree is defined as follows:

```
struct node {
  int key;
  struct node* left;
  struct node* right;
}

struct node* root = NULL;
```

Given an integer x , write a function that finds the successor of x in the binary search tree. Give and explain example BST trees that illustrate all the different cases that have to be handled. If x is not in the tree or does not have a successor then -1 shall be returned.