Eidgenössische
Technische
Hochschule Zürich

Ecole polytechnique f´ed´erale de Zurich
Politecnico federale di Zurigo
Federal Institute of Technology at Zurich

Department Informatik
Victoria Miebach
Julius Jung
David Minhea-Stefan
(kein offizielles Material)

Exam Preparation

Algorithmen und Datenstrukturen

December 16, 2024

DO NOT OPEN!

Student number: _____

Seat number: _____

Good luck!

(back of title page)

Theory Task T1: Complexity and O-notation.

In this problem, you have to provide **solutions only**. You do not need to justify your answer.

a) *Asymptotic notation quiz:* For each of the following claims, state whether it is true or false. Assume $n \geq 4$.

Claim	true	false
$n^5 + \ln(n) = \theta(n^5)$	☒	☐
$(e^n)^2 = e^{2n} \leq O(e^n)$	☐	☒
$\sum_{i=1}^{n} \sqrt{i} = \theta(n^{1.5})$	☒	☐
Suppose $a_1 = 4$ and $a_{i+1} = 2a_i$ for all $i \in \mathbb{N}$. It follows $a_n \leq O(n^2)$.	☐	☒

$\leq n \cdot \sqrt{n}$

$\geq \left(\frac{n}{2}\right) \cdot \sqrt{\left(\frac{n}{2}\right)}$

$a_n = 2^{n-1} \cdot 4$

$a_n = 2 \cdot a_{n-1}$
$= 2 \cdot 2 \cdot a_{n-2}$

b). Counting iterations: For the following code snippets, give the exact number of times f is called as a function of n. For the exact number, you are allowed to use summation sign in your expression. Then, derive an asymptotic bound for the number of times f is called in Θ-notation. Simplify your expression as much as possible (in particular without summation signs).

i) Snippet 1:

Algorithm 1

```
for i = 1,...,n do
    j ← 1
    while j ≤ n do
        f()
        j ← 2j
    f()
```

$\sum_{i=1}^{n}\left(1+\sum_{j=1}^{\lfloor \log_2(n) \rfloor + 1} 1\right) = n + \sum_{i=1}^{n}\sum_{j=1}^{\lfloor \log_2(n) \rfloor + 1} 1 = n + \sum_{i=1}^{n} \lfloor \log_2(n) \rfloor + 1 = 2n + n \cdot \lfloor \log_2(n) \rfloor = \Theta(n \log n)$

ii) Snippet 2:

Algorithm 2

```
for j = 1,...,n do
    for l = 1,...,100 do
        for k = j,...,n do
            f()
            f()
            f()
```

$1 \dots 3 = 3 - 1 + 1 = 3$

$j \dots n \quad n - j + 1 = n$

$1 \dots n \quad n - 1 + 1 = n$

$$\#\text{calls} = \sum_{j=1}^{n} 100 \cdot \sum_{k=j}^{n} 3 = \sum_{j=1}^{n} 300 \cdot (n-j+1) = 300 \cdot \sum_{j=1}^{n} n-j+1 = 300 \cdot \sum_{j=1}^{n} j$$
$$= 300 \cdot \frac{n \cdot (n+1)}{2} = 150n^2 + 150n = \Theta(n^2)$$

c) Sorting and searching algorithms: For parts i) and ii), there is only one correct choice. For parts (iii) there is at least one correct choice but there might be several ones. Please circle all correct answers.

i) There exists an array of length n for which BubbleSort needs time $O(n)$

(a) True
(b) False

ii) Every sorting algorithm takes at least time $\Omega(nlogn)$.

(a) True
(b) False

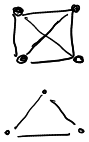
iii) Which of the following algorithms has runtime $O(nlogn)$ on [2,1,4,3,..., n,n-1]?

(a) BubbleSort
(b) Heapsort
(c) SelectionSort
(d) InsertionSort

Theory Task T2: Graphs.

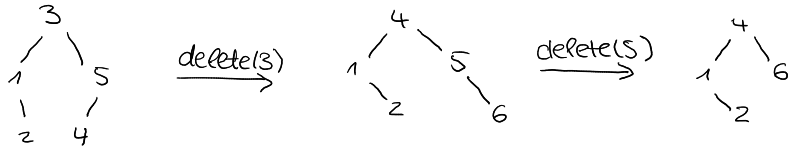
a) *Graph quiz:* For each of the following claims, state whether it is true or false. Assume $G = (V, E)$ with $|V| = n$ and $|E| = m$.

Claim	true	false
If G is a tree, then G contains n-1 edges.	☒	☐
If G has a Hamiltonian cycle, then G has an Eulerian walk.	☐	☒
If G contains a closed walk of length 6, then there exists a cycle of length 6.	☐	☒
G is a tree if and only if G is connected and contains no cycle.	☒	☐
To find out whether any two nodes u and v in a graph are ~~connected,~~ adjacent an adjacency list achieves the same or better access time than an adjacency matrix.	☐	☒

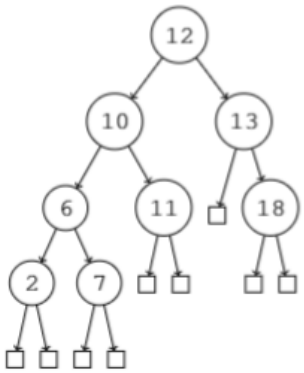


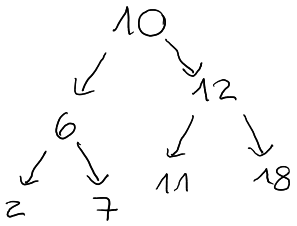
b) *Binary trees:*

i) Draw the binary search tree after following operations: insert(3), insert(1), insert(5), insert(2), insert(4), delete(3), insert(6), delete(5).

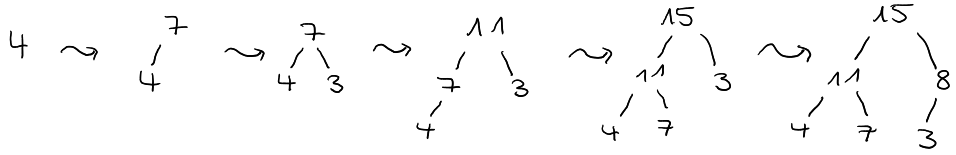


ii) Draw the AVL-tree obtained from the following tree by performing delete(13).

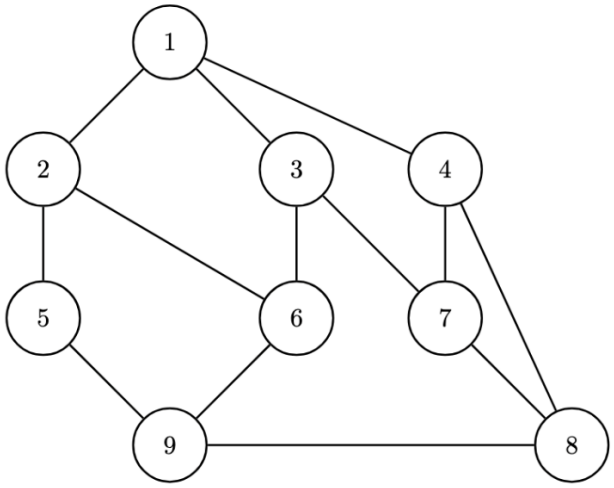




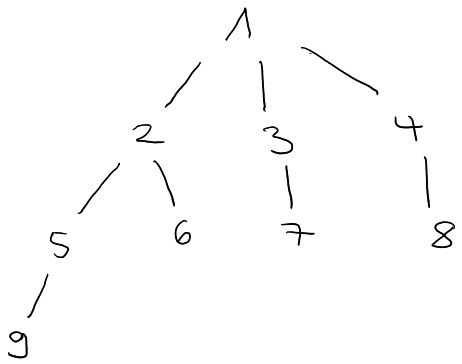
iii) Draw the max-heap obtained from inserting the keys 4, 7, 3, 11, 15, 8 in this order into an empty heap. It suffices to only draw the final heap. If there are several solutions, it is enough to draw only one of them.



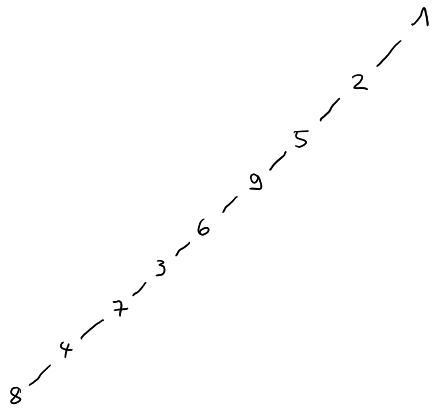
c) *Breadth/depth-first search:* Consider the following undirected graph:



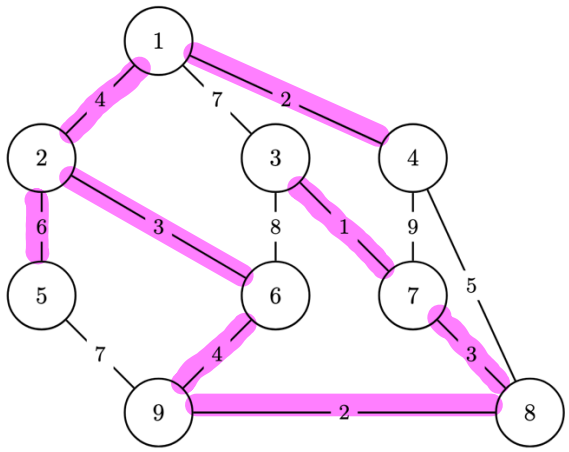
i) Draw the breadth-first tree resulting from a breadth-first search starting from vertex 1. Process the neighbours of a vertex in increasing order.



ii) Draw the depth-first tree resulting from a depth-first search starting from vertex 1. Process the neighbours of a vertex in increasing order.



d) *Minimum Spanning Tree:* Consider the following graph:

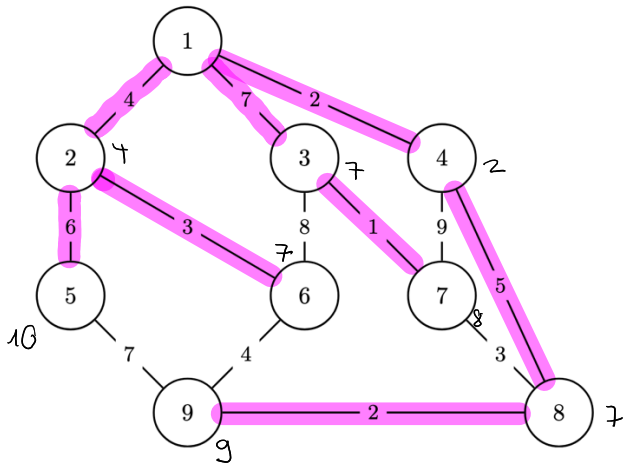


i) Find a minimum spanning tree. You can either highlight its edges in the picture above, or draw the minimum spanning tree below. (If there are more than one minimum spanning trees, it suffices to find just one of them.)

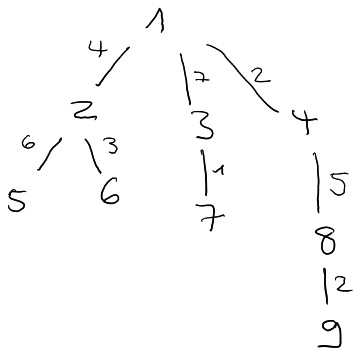
ii) List all positive edge weights you could replace edge {2,5} such that it is still included in at least one minimum spanning tree.

1, 2, 3, 4, 5, 6, 7

e) *Shortest Path Tree:* Consider the same graph as in d). Here again drawn for convenience



i) Find a shortest-path tree rooted at vertex 1 (e.g., the output of Dijkstra's algorithm if we were to start from vertex 1). You can either highlight its edges in the picture above, or draw the shortest-path tree below.



Theory Task T3: Algorithm Design

a) You are given an array $\mathbf{A} = [a_1, a_2, ..., a_n]$ of integers. We want to find the subarray S of length max. k such that the sum of all elements in S is maximized.

Your task is to design a dynamic programming algorithm that, given an array **A** of length **n** and an integer $\mathbf{k} \leq n$, returns the maximum sum of a subarray of at most k. Your algorithm should have a time complexity of $\mathbf{O(n \cdot k)}$.

In your solution, address the following aspects:

1. *Dimensions of the DP table*: What are the dimensions of the *DP* table?
2. *Subproblems*: What is the meaning of each entry?
3. *Recursion*: How can an entry of the table be computed from previous entries? Justify why your recurrence relation is correct. Specify the base cases of the recursion, i.e., the cases that do not depend on others.
4. *Calculation order*: In which order can entries be computed so that values needed for each entry have been determined in previous steps?
5. *Extracting the solution*: How can the solution be extracted once the table has been filled?
6. *Running time*: What is the running time of your solution?

Variante: max. subset sum mit ≤k Elementen

1) $(n+1) \times (k+1)$

2) $DP[i,j] :=$ max. subset sum with j elements from $a_1, \dots, a_i$

3) Base Case: $DP[i,0] = 0$ for $0 \leq i \leq n$

$DP[i,j] = \max\{DP[i-1,j], DP[i-1,j-1] + a_i\}$

4) for $i \leftarrow 0 \dots n$
  for $j \leftarrow i \dots n$
    compute $DP[i,j]$

5) $DP[n,k]$

6) compute $(n+1) \cdot (k+1)$ entries each in $O(1)$
→ $O(n \cdot k)$

b) The Department for the Education of Young Witches and Wizards at the Ministry of Magic has received many owls with letters of complaint. The parents of various wizarding families are unhappy that the Hogwarts Express only leaves from London and therefore all families, regardless of where they live, have to take their children to London to catch the train. So the Ministry has decided to expand the stops. Stops are to be set up in many more cities. As Algorithms and Data Structures will unfortunately not be taught at Hogwarts until next year, the wizards are currently still reliant on your help to plan the expansion of the Hogwarts Express line efficiently.

Remark: When asked to describe an algorithm, you need to address the following aspects:

1. the graph algorithm that you use as a subroutine in your solution,
2. the construction of the graph that you run this algorithm on,
3. the correctness of your solution, with a short justification (i.e., how and why does running the chosen graph algorithm on the graph you constructed solve the original problem),
4. the total running time of your solution, with a short justification.

Task 1: The Hogwarts Express can run on some of the Muggle tracks (tracks built by humans). However, these are not enough to connect all the towns. Therefore, the Ministry suggests a bunch of other connections, each of which requires a certain amount of wizarding power to conjure. Now the ministry turns to you to find out which routes should be conjured up so that all stops are connected and as little magic power as possible is used. You will receive:

a. A number of stops H

b. A number of existing connections between stops E

c. A set of possible connections EPoss between stops that can still be conjured up, each with a positive real number that corresponds to the magic power to build this route.

Describe an algorithm that outputs a rail network which connects all stops and needs the least amount of conjuring power. You can assume that it is possible to connect all stops with the proposed connections EPoss.

ausführlicher in der Prüfung!

1) Kruskal

2) weighted graph
   vertices ≙ stops
   edges ≙ existing connections with weight 0
   + possible connections with needed magic power as weight

3) use MST algorithm Kruskal, all edges in MST that are from EPoss are the routes that need to be added to keep construction as cheap as possible

4) runtime of Kruskal: $O(m \log m + n \log n)$

Task 2: Now the wizards want to find the fastest route from a stop H to Hogwarts. The problem is that some of the Muggle tracks are at some times used by other trains, so the Hogwarts Express cannot run along them. More precisely, for each section of track built by Muggles, there is a set of times at which a train runs on this route. These times are always every quarter of an hour, i.e. they can only be of the form hh:00, hh:15, hh:30 and hh:45. The route section is then not passable for the entire quarter of an hour.

Given

a. The set of stops,

b. All route sections between stops,

c. A weight function that states how long it takes to travel on a section (to simplify the task you can assume these times to be multiples of 15 minutes)

d. A function that assigns a set of blocking times to each route section according to the above format

design an algorithm that calculates whether the Hogwarts Express can make it to Hogwarts by 6pm if it leaves from stop H at 10am.

idea: graph layering again and use Dijkstra

Task 3: The wizards have now commissioned a specialist to build the track who manages to conjure up such good tracks that the train completes them earlier than it departs. Explain under which assumption it is now possible to develop a reliable algorithm that solves the problem from task 2 and describe the changes you have to make to your solution to task 2.

same graph as in task 2 but also with negative edges
→ use of Bellman-Ford
→ necessary assumption: no negative cycles

Theory Task T4: Proofs

a) Let $a \neq 1$ be a real number. Prove by mathematical induction that for every non-negative integer n,

$$\Sigma_{i=0}^{n} a^i = \frac{a^{n+1} - 1}{a - 1}$$

Base Case: $n=0$ $\sum_{i=0}^{0} a^i = a^0 = 1$ $\quad \frac{a^{0+1}-1}{a-1} = 1$ ✓

Induction Hypothesis: $\sum_{i=0}^{k} a^i = \frac{a^{k+1}-1}{a-1}$ holds for some non-negative integer k

Induction Step: $\sum_{i=0}^{k+1} a^i = \sum_{i=0}^{k} a^i + a^{k+1} \overset{I.H.}{=} \frac{a^{k+1}-1}{a-1} + a^{k+1}$

$$= \frac{a^{k+1} - 1 + (a-1)a^{k+1}}{a-1}$$

$$= \frac{a^{k+1} - 1 + a^{k+2} - a^{k+1}}{a-1}$$

$$= \frac{a^{k+2} - 1}{a-1}$$

Thus, by the principle of mathematical induction it can be proven that $\sum_{i=0}^{n} a^i = \frac{a^{n+1}-1}{a-1}$ holds for all non-negative integers n. ▨

Why does your proof fail for $a = 1$?

division by zero

b) Prove the following statement:

$G = (V, E)$ *is a tree.* $\Leftrightarrow$ *For all* $u, v \in V$ *there exists exactly one* $u - v - Path.$

!muss in der Prüfung ausführlicher!

"$\Rightarrow$"
- da G Baum, ist G zusammenhängend
- es gibt mind. einen u-v-Pfad für alle $u,v \in V$
- es darf nur genau einen u-v-Pfad für alle $u,v \in V$ geben, da G sonst einen Kreis hätte und kein Baum wäre

"$\Leftarrow$"
- G ist zusammenhängend
- Annahme: G besitzt Kreis
  → Widerspruch, da es dann zwei verschiedene u-v-Pfade gibt
- G ist Baum, da zusammenhängend und kreisfrei

Extra space for notes. If you write any answers here that should be graded, please indicate clearly to which task your notes belong, and make a note in the main body of the exam.

Extra space for notes. If you write any answers here that should be graded, please indicate clearly to which task your notes belong, and make a note in the main body of the exam