

General Information

- Time 08:00 - 12:00
- No books, internet, smartphones, notebooks etc allowed.
- In total, you can get 57 points and you need to get at least 28.5 points to pass the exam.
- There are statements “Prove *in detail* [...]”.
In detail means that arguments “as shown in the lecture” are not sufficient however and that a rigorous proof is required. However, you are allowed to apply the standard definitions like the axioms for matroids or terms like “connected”, “cycle”, etc.
- Make sure not to overlook any of the **8** problems.
On average, you have 30 minutes for each of the 8 problems.

- (a) Let $L = \{(ab)^n \mid n \in \mathbb{N}_{>0}\} \subseteq \Sigma^*$ with $\Sigma = \{a, b\}$ be the language that consists of all strings $s = s_1, \dots, s_\ell$ ($\ell > 0$ and ℓ even) of the form $s = ababab \dots ab$.
Provide a TM (in simplified finite state representation) that recognizes L . You can assume that all input strings are taken over the alphabet Σ . Briefly explain how your TM works for the input strings $s = ab$, $s = aba$, $s = ba$, and $s = ababab$.
- (b) When is a language TM-recognizable?

- (a) Let S be a finite set of size $|S| = N$ and $k \leq N$ be a positive integer. Show that the number of all subsets S' of S of size $|S'| = k$ is in $\mathcal{O}(N^k)$.
- (b) Provide the exact bounds for the time complexity $T(n)$ and space complexity $S(n)$ of the following program `policeThief(...)` in Θ -notation, assuming a unit-cost model.

Here is some more information about used commands.

- The command `length` returns the length of a given linked list.
- The input size is determined by $n := \text{length}(L)$ of the input list L .
- The function `append` adds the given element to the end of the list, thus extending it by 1 element.
- `list[i]` returns the key of the element corresponding to the i -th entry of the list `list`

Remark: You do **not** need to understand what the program computes in order to answer this question.

```

policeThief(list L, int n, int k)
1  int res := 0, i := 0
2  init empty list pol
3  init empty list thi
4  WHILE (i < n) DO
5      i := i + 1
6      IF (L[i] = P) THEN pol.append(i)
7      ELSE IF (L[i] = T) THEN thi.append(i)
8  int l := 0, r := 0
9  WHILE (l < length(thi) and r < length(pol)) DO
10     IF (|thi[l] - pol[r]| ≤ k) THEN
11         res := res + 1
12         l := l + 1
13         r := r + 1
14     ELSE IF (thi[l] < pol[r]) THEN l := l + 1
15     ELSE r := r + 1
16  return res
    
```

Answers of the form $T(n) \in \Theta(\cdot)$ and $S(n) \in \Theta(\cdot)$ are sufficient and proofs are not needed. In other words, you only need to answer:

- (a) Time complexity: Provide $f(n)$ such that $T(n) \in \Theta(f(n))$.
- (b) Space complexity: Provide $g(n)$ such that $S(n) \in \Theta(g(n))$.
- (c) Suppose that you have an algorithm that takes as input an undirected graph $G = (V, E)$ and that runs in $\mathcal{O}(|E|)$. Provide a class of connected undirected graphs $G = (V, E)$ for which this algorithm runs in $\mathcal{O}(|V|)$ time and where $|V|$ is not treated as a constant. Briefly explain your results.

Problem 3 (*Complexity*)

2+2+(4+2)=10p

- (a) Define the class NP.
- (b) Let B be a decision problem in the class P and A a decision problem such that $A \leq_p B$. Explain in a few words whether $A \notin \text{NP}$ or $A \in \text{NP}$.
- (c) Consider the following two problems:

CLIQUE:

Input: Given an undirected graph G and an integer $k \in \{1, \dots, |V(G)|\}$ *Question:* Does there exist a subgraph $H \subseteq G$ such that H is isomorphic to a complete graph K_ℓ on $\ell \geq k$ vertices?

SUBGRAPHISOMORPHISM:

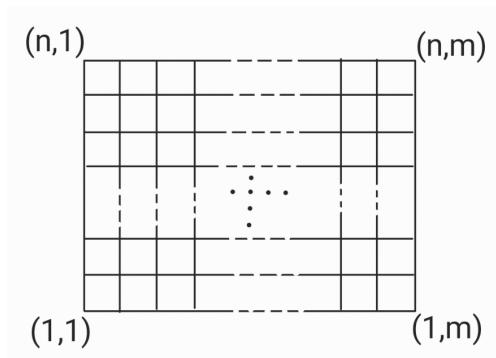
Input: Two undirected graphs H and G .*Question:* Is there an isomorphism φ from H to some subgraph $G' \subseteq G$?

- (i) Assume that we know that CLIQUE is NP-complete. Prove *in detail* that SUBGRAPHISOMORPHISM is NP-complete (by reduction from CLIQUE).
- (ii) Explain if the latter result implies that the problem of testing whether two graphs G and H are isomorphic is NP-hard.

Problem 4 (*Dynamic Programming*)

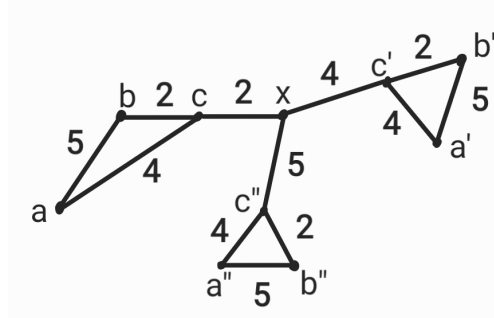
3+3 = 6p

Shown is an $n \times m$ grid where the vertices are labeled with corresponding coordinates (i, j) , $1 \leq i \leq n$, $1 \leq j \leq m$.



- (a) Give a recursive formula for computing the number of simple shortest paths from $(1, 1)$ to (n, m) . Explain shortly why your formula works.
- (b) What is the number of simple shortest paths from the origin $(1, 1)$ to the vertex with coordinates $(3, 4)$?

- (a) Consider the following edge-weighted graph G :



Draw into the graph G (or provide the edge set of) a minimum spanning tree that you find by applying Kruskal's algorithm. How many spanning trees does G have?

In the following, E is a finite set and $\mathbb{F}$ a collection of subsets of E .

- (b) Give the properties that must be satisfied for $(E, \mathbb{F})$ to be a matroid.
 (c) Suppose that $(E, \mathbb{F})$ is an independent system and assume that $I = \{a, b, c\}$ and $J = \{x, a, 0, 1\}$ are two (but not necessarily all) maximal independent sets of $(E, \mathbb{F})$.
 Prove *in detail* that $(E, \mathbb{F})$ is not a matroid.

- (a) Given is the branching vector $(1, 1, 2)$. Provide a bound for the number of leaves in a search tree to which this branching vector applies.
 (b) Prove that the Vertex Cover Problem is FPT.
Hint: Bounded Search Tree

- (a) Give the definition of a ρ -approximation algorithm for an optimization problem.
 (b) In the lecture, the following approximation algorithm for the Vertex Cover Problem was provided:

```

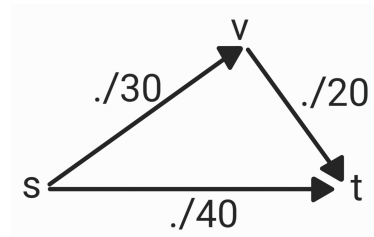
GREEDY_VC (graph  $G$ )
1:  $C = \emptyset$ ,  $E' = E(G)$ 
2: while  $E' \neq \emptyset$  do
3:    $e = \{u, w\}$  some edge in  $E'$ 
4:    $C = C \cup \{u, w\}$ 
5:   Remove all edges incident to  $u$  and  $w$  from  $E'$ 
6: return  $C$ 
    
```

Prove *in detail* that GREEDY_VC is a *polynomial-time* 2-approximation algorithm.
Hint: The edges that have been chosen in Line 3 are vertex disjoint.

Problem 8 (*Maximum Flow*)

4p

Given is the following flow network $G = (V, E)$ together with capacities along its edges. Non-edges (u, v) have capacity $c(u, v) = 0$.



Apply the Ford-Fulkerson-Algorithm to compute a maximum flow in G . Draw for each individual step the residual network and provide the value of a maximum flow.