Exam “Algorithms and Computability” F25

- The exam paper has 4 pages with 6 questions. The maximal number of points is 90. You have 180 minutes for the exam, i.e., roughly two minutes per point.
- Carefully read each question before you answer it.
- You may bring a (single-sided) handwritten A4 piece of notes to the exam, no other aids are allowed.
- If you believe there is an error on the exam paper or that you are missing information then clearly write down the assumptions you are making.
- Submit only one `.pdf` file.
- Good luck!

Question 1 — 10 Points

Let $L \subseteq \{a, b\}^*$ be the language of words that contain exactly two a, i.e., $aa \in L$, $abba \in L$, and $bababbb \in L$, but $a \notin L$, $bababa \notin L$, and $bb \notin L$.

Construct a one-tape halting DTM accepting L by specifying all its components in full detail. Also describe how the machine works.

Question 2 — 12 + 6 + 2 Points

Let $G = (V, E)$ be an undirected graph.

- G is 3-colorable, if there is a function $c \colon V \to \{R, G, B\}$ such that we have $c(v) \neq c(v')$ for all $(v, v') \in E$ with $v \neq v'$, i.e., we can color the vertices of G by the colors red, green, and blue so that different vertices that are connected by an edge have different colors.
- G is 4-colorable, if there is a function $c \colon V \to \{C, M, Y, K\}$ such that we have $c(v) \neq c(v')$ for all $(v, v') \in E$ with $v \neq v'$, i.e., we can color the vertices of G by the colors cyan, magenta, yellow, and black so that different vertices that are connected by an edge have different colors.

Let

$$3\text{COL} = \{G \mid G \text{ is a 3-colorable undirected graph}\}$$

and

$$4\text{COL} = \{G \mid G \text{ is a 4-colorable undirected graph}\}.$$

1. Show that 3COL is polynomial-time reducible to 4COL. To this end, first spell out exactly what you need to show.
2. Show that 4COL is in NP.
3. We know from the lecture that 3COL is NP-complete. What can we conclude about the complexity of 4COL? Argue your answer.

Question 3 — 8 + 2 Points

Let $G = (V, E)$ be an undirected graph. A set $D \subseteq V$ of vertices is a dominating set if for every vertex $v \in V \setminus D$ there is a vertex $v' \in D$ such that $(v, v') \in E$, i.e., each vertex of G is either in the dominating set or connected by an edge to a vertex in the dominating set.

For example, in the graph for Exercise 6, $\{E, F, H, K\}$ is a dominating set of size four, but $\{F, K, L\}$ is not a dominating set, since vertex E is neither in $\{F, K, L\}$ nor connected to a vertex in $\{F, K, L\}$.

We define

$$\text{DOMSET} = \{(G, k) \mid G \text{ is an undirected graph with a dominating set of size } k\}.$$

1. Show that DOMSET is expressible as a 0-1 Integer Linear Program, i.e., show that every pair (G, k) can (in polynomial-time) be turned into a 0-1 Integer Linear Program that has a solution if and only if $(G, k) \in \text{DOMSET}$.

   Hint: First, explain which decision variables you use and how to interpret them. Also note that you do not need an objective function, as DOMSET is a decision problem.

2. In the optimization variant of DOMSET, is one interested in maximizing or minimizing the size of a dominating set? Explain your answer.

Question 4 — 14 Points

Apply the backtracking algorithm for the knapsack problem to the following instance: Size $W = 100$, threshold $T = 20$, and the following items:

item	1	2	3	4
weight	60	20	40	20
value	8	5	10	6

Show the tree of partial solutions until a valid solution is found or until the algorithm terminates without a valid solution.

Question 5 — 11 + 11 Points

Apply the DPLL algorithm to the following formulas to determine whether they are satisfiable or unsatisfiable.

Illustrate the execution of the algorithm using a tree structure, i.e., showing the simplified formulas and the rules applied.

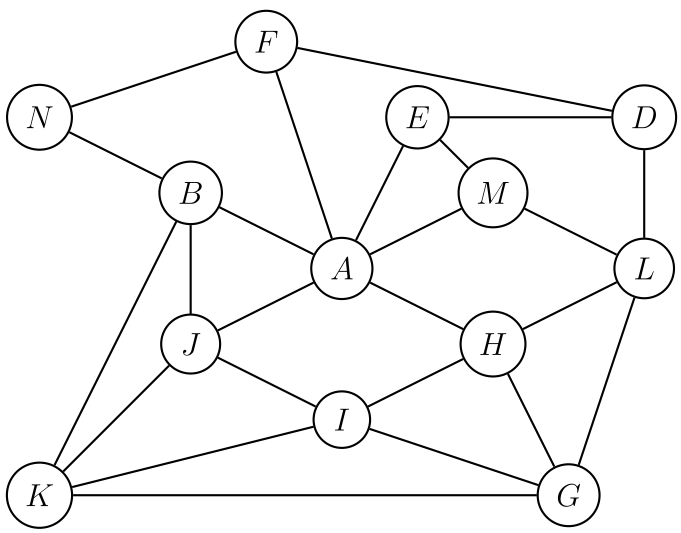
You may apply the splitting rule to any variable, if no other rules are applicable.

1. $\varphi_1 = (X \vee \neg Y \vee \neg W) \wedge (\neg X \vee \neg Y \vee \neg W) \wedge (Y \vee W) \wedge (W \vee Z) \wedge (W \vee \neg W)$
2. $\varphi_2 = (\neg W \vee \neg Y \vee \neg V) \wedge (W \vee \neg Y) \wedge (W \vee \neg Z) \wedge (Y \vee \neg X) \wedge (\neg V \vee Z) \wedge W \wedge (V \vee X) \wedge (V \vee \neg X) \wedge (\neg Z \vee X)$

Hint: You can use & for $\wedge$, | for $\vee$, and ! for $\neg$ if it is more convenient.

Question 6 — 14 Points

Apply the approximation algorithm for vertex cover to the following graph.



Show for each iteration the edge selected in this iteration (in any order), the intermediate value of the set `C`, and the newly-covered edges in this iteration.