Exercise V, Computational Complexity 2025

These exercises are for your own benefit. Feel free to collaborate and share your answers with other students. Solve as many problems as you can and ask for help if you get stuck for too long. Problems marked * are more difficult but also more fun :).

Space Complexity

1 Basics:

(a) Prove that TQBF $\in$ PSPACE by designing a polynomial-space algorithm for it.
(b) Prove that NL $\subseteq$ P. *(Hint: Construct the whole configuration graph for an* NL *algorithm.)*

2 Consider the following single-player *pebble game* played on an directed acyclic graph $G = (V, E)$ with a unique *sink vertex* $v^* \in V$ (out-degree 0). There is a set of pebbles that can be placed on the vertices in V. The game proceeds in rounds. At start, the graph is empty of pebbles. In each round we perform either of the following pebbling moves:

(a) We may place a pebble on any vertex whose predecessors are pebbled. In particular, we may always place a pebble on a *source vertex* (in-degree 0).
(b) We may always remove a pebble from any vertex.

The goal is to pebble the sink v^* while minimising the number of pebbles used (maximum number of pebbles in any configuration during the game).

In the PEBBLE problem, we are given $\langle G, k \rangle$ as input and want to determine if the sink can be pebbled using at most k pebbles. Show that PEBBLE $\in$ PSPACE. Do you think PEBBLE $\in$ NP?

3 Show that the following parsing problems are in L.

(a) The input is a string of parentheses $x \in \{(,)\}^*$. We wish to know whether x is *properly nested*; for instance, `(())()` and `()()()` are allowed but `())` is not.

(b) The input is a string $x \in \{(,), [,]\}^*$. We again wish to know whether x is properly nested; for instance, `([])[]` and `[()[]]` are allowed but `[(])` is not.

(Hint: Show that one only needs to check a simple condition for each substring of x.)

4 In the context of the previous problem, suppose the TM is only allowed to scan the input x once from left to right—much like a deterministic finite automaton. Show that (a) can still be done with memory $O(\log n)$, but (b) requires memory $\Theta(n)$.

(Hint: What does the TM need to remember when it sweeps past the middle symbol of x?)

5 The *alternating path* game is played by two competing players, Alice and Bob, on a directed graph $G = (V, E, v^*)$ where v^* is a distinguished vertex. The players alternate in constructing a directed path $v_0, v_1, \ldots$ starting at $v_0 = v^*$: Alice chooses v_1 as one of the out-neighbours of v_0, then Bob chooses v_2 as one of the out-neighbours of v_1, and so on the players alternate. The first player who cannot choose a previously unvisited vertex loses the game—that is, the player is forced to choose a previously visited vertex, or the path has terminated at a sink.

In the AltPath problem, the input is the graph G and the goal is to decide if Alice has a winning strategy for the alternating path game on G. Show that AltPath is PSPACE-complete by giving a reduction from TQBF.

(Hint: Suppose $\exists x_1 \forall x_2 \cdots Q_n x_n \colon \varphi(x)$ is the TQBF instance where φ is a 3-CNF. Construct G so that it has one vertex for every literal x_i and $\bar{x}_i$, and connect them by directed edges so that Alice (resp. Bob) gets to choose, for each existentially (universally) quantified x_i, whether to visit vertex labelled x_i or $\bar{x}_i$. After a path has determined a truth assignment to x by visiting appropriate literal vertices, check whether it satisfies the CNF formula by including in G one node each clause in φ.)