Universität Leipzig
Institute of Computer Science
Winter term 2026/2027

Andreas Maletti

Exercises for the lecture

Foundations of Complexity Theory

Solutions to Exercise Sheet 4

for the exercise on November 11, 2026

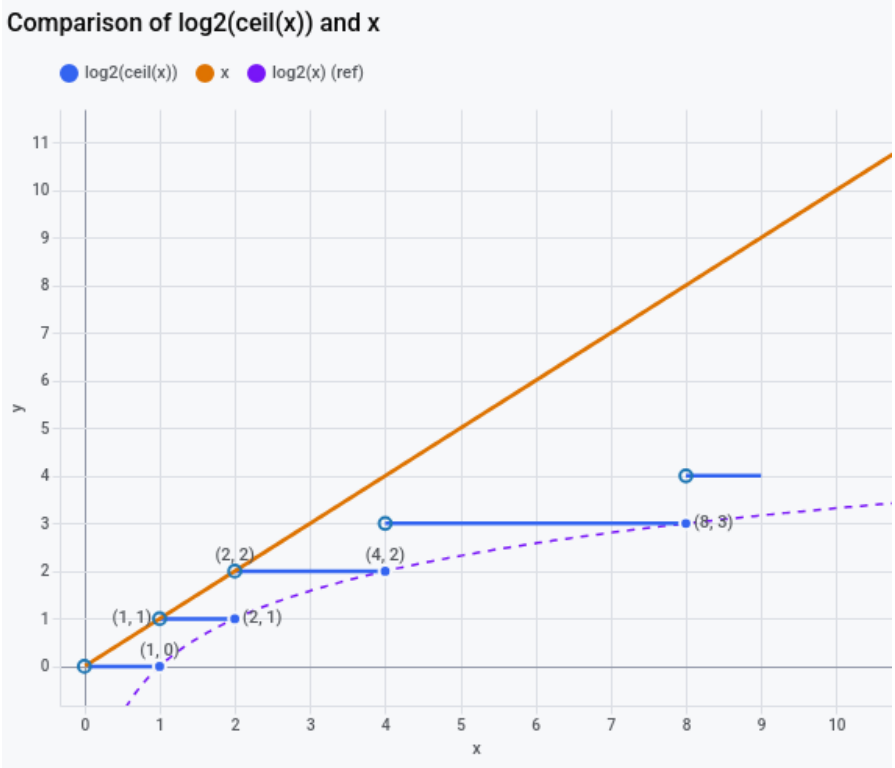
Exercise 4.1

Prove $\mathbf{P} \subsetneq \mathrm{DTIME}(O(2^x)) \subseteq \mathbf{EXPTIME}$ (page 15 of 4th lecture)

Solution: Both inclusions are trivial. For strictness, we consider the function $f : \mathbb{N} \to \mathbb{N}$ with $f(n) = \lceil (\sqrt{2})^n \rceil$ for every $n \in \mathbb{N}$. We have $\lim_{n\to\infty} \frac{n^i}{2^{n/2}} = 0$ and thus also $\lim_{n\to\infty} \frac{n^i}{\lceil 2^{n/2} \rceil} = 0$ for all $i \in \mathbb{N}$ and thus $\mathbf{P} \subseteq \mathrm{DTIME}(o(f))$. By the time-hierarchy theorem (§3.5)

$$\begin{aligned}\mathbf{P} &\subseteq \mathrm{DTIME}(o(f)) \subsetneq \mathrm{DTIME}(O(f \log f)) = \mathrm{DTIME}(O(\lceil 2^{x/2} \rceil \log(\lceil 2^{x/2} \rceil)))\\ &\overset{\dagger}{\subseteq} \mathrm{DTIME}(O(2 \cdot 2^{x/2} \cdot 2^{x/2})) = \mathrm{DTIME}(O(2 \cdot 2^x)) = \mathrm{DTIME}(O(2^x)) \ ,\end{aligned}$$

where at † we used $r \geq \log\lceil r \rceil$ for all $r \in \mathbb{R}_+$.



Exercise 4.2

Prove that for propositional formulae in disjunctive normal form we can test satisfiability in polynomial time.

Solution: A formula F in disjunctive normal form is a set of sets of literals and satisfiable iff (if and only if) one set $L \in F$ of literals is satisfiable. Since the literals in L are conjunctively connected, the set L is satisfiable iff there exists no variable x such that $\{x, \neg x\} \subseteq L$. This can easily be checked in quadratic time in the size of the formula by considering each set $L \in F$ of literals and considering each variable $x \in L$.

Obviously this problem can even be solved in **L**. Using the same argument as above, we can mark a literal inside a clause with the help of a pointer into the input. Obviously this only requires logarithmic space. Initially this marking is on the first literal of the first clause. We read the marked literal x, store it into the state, walk left to the opening parenthesis of the clause and check the whole clause for any occurrence of $\neg x$ (with the end of clause being the closing parenthesis and $x = \neg\neg x$). If such an occurrence is found (i.e., the current clause is unsatisfiable), then we advance the marked literal to the first literal in the next clause; if there is no additional clause, then we reject. If we do not find such an occurrence, then we advance the marking to the next literal in the current clause; if there is no additional literal in the current clause, then we accept.

Exercise 4.3

Prove that 2-SAT $\in$ **NL**; i.e. satisfiability of propositional formulae in conjunctive normal form with at most 2 literals per disjunction.

Solution: Let F be a conjunctive normal form such that $|D| \leq 2$ for all disjunctions $D \in F$. Let L be the set of literals over variables that occur in F, and let $\neg\neg\ell = \ell$ for all $\ell \in L$. We construct the implication graph $G = (L, E)$ with

$$E = \{(\ell, \ell') \in L \times L \mid \{\neg\ell, \ell'\} \in F\} \ .$$

If $(\ell, \ell') \in E$, then also $(\neg\ell', \neg\ell) \in E$. Hence G has at most $2\,|F|$ vertices and edges. If there exists vertex $\ell \in L$ such that $\neg\ell$ is reachable from ℓ and ℓ reachable from $\neg\ell$ in G, then F is unsatisfiable. We prove this by case distinction. Let $(\ell_0, \ell_1), (\ell_1, \ell_2), \ldots, (\ell_{n-1}, \ell_n) \in E$ and $(\ell'_0, \ell'_1), (\ell'_1, \ell'_2), \ldots, (\ell'_{m-1}, \ell'_m) \in E$ with $\ell_0 = \ell'_m = \ell$ and $\ell_n = \ell'_0 = \neg\ell$, and let I be an arbitrary interpretation.

- Suppose that $I(\ell)$ is false. Then $I(\ell'_0)$ is true and hence also $I(\ell'_i)$ must be true for all i in order to make $I(F)$ true. However, $I(\ell'_m) = I(\ell)$ is false, which forces $I(F)$ to be false.
- Suppose that $I(\ell)$ is true. Then $I(\ell_0)$ is true and hence also $I(\ell_i)$ must be true for all i in order to make $I(F)$ true. However, $I(\ell_n) = I(\neg\ell)$ is false, which again forces $I(F)$ to be false.

In the other direction we have that $\neg\ell$ is not reachable from ℓ or ℓ is not reachable from $\neg\ell$ for each vertex $\ell \in L$. Let

$$P = \{\ell \in L \mid (\neg\ell, \ell) \in E^*\} \qquad \text{and} \qquad N = \{\ell \in L \mid (\ell, \neg\ell) \in E^*\} \ ,$$

where E^* is the reflexive, transitive closure of E (i.e., the reachability relation). Obviously $P \cap N = \emptyset$ by our assumption. Additionally, let P^* (resp. N^*) be the closure of P (resp. N) under successors (resp. predecessors) in G; i.e.

$$P^* = \{\ell \in L \mid \exists \ell' \in P \colon (\ell', \ell) \in E^*\} \qquad \text{and} \qquad N^* = \{\ell \in L \mid \exists \ell'' \in N \colon (\ell, \ell'') \in E^*\} \ .$$

We claim that also $P^* \cap N^* = \emptyset$. To prove this claim via contradiction, suppose there exists $\ell \in P^* \cap N^*$. Then there exist $\ell' \in P$ and $\ell'' \in N$ such that $(\ell', \ell) \in E^*$ and $(\ell, \ell'') \in E^*$. Additionally, we have $(\neg\ell', \ell') \in E^*$ and $(\ell'', \neg\ell'') \in E^*$ by $\ell' \in P$ and $\ell'' \in N$. Hence

$$(\ell', \neg\ell'') \in E^* \qquad \text{as well as} \qquad (\neg\ell'', \neg\ell) \in E^* \qquad \text{and} \qquad (\neg\ell, \neg\ell') \in E^* \ .$$

Thus we also have $(\ell', \neg\ell') \in E^*$, which together with $(\neg\ell', \ell') \in E^*$ yields the contradiction.

We proved that $P^* \cap N^* = \emptyset$. Now we construct a model I by

$$I(\ell) = \begin{cases} 1 & \text{if } \ell \in P^* \\ 0 & \text{if } \ell \in N^* \\ |\ell| & \text{otherwise} \end{cases}$$

where $|v| = 0$ and $|\neg v| = 1$ for all atoms v. Obviously I is a model of F.

Since GAP $\in$ **NL** we can guess vertex ℓ and check reachability of $\neg\ell$ from ℓ and reachability of ℓ from $\neg\ell$ in G in nondeterministic log-space. Hence 2-SAT $\in$ **co-NL** $=$ **NL** (by Immerman-Szelepcsényi).

Let us discuss how to solve this without reference to GAP. The graph accessibility problem (GAP) is the reachability problem in a graph and can be solved in nondeterministic logarithmic space by storing 2 references to vertices: the current and the next vertex c and s, respectively. Initially the current vertex c is set to the start vertex. In each iteration (of at most as many as the graph has vertices) we check whether the current vertex is the target vertex. If so, then we simply accept. Otherwise we nondeterministically select a vertex s and check whether (c, s) is an edge in the graph; i.e., whether s is indeed a successor of c. For our implication graph, which we never store in full size, $(c, s) \in E$ iff $\{\neg c, s\} \in F$. In other words, the edge (c, s) exists in the implication graph if and only if the clause $\{\neg c, s\}$ is part of the formula. Clearly, this can be checked by inspecting the input formula for the clauses $(\neg c \vee s)$ or $(s \vee \neg c)$ and thus requires no additional space. If the required edge exists, then we set the current vertex to s and enter another iteration.

If the required edge is not present in the graph, then we reject. If we complete all iterations, then we reject as well.

To validate unsatifiability, we can nondeterministically select a vertex $\ell \in L$ and check reachability of $\neg\ell$ from ℓ and vice versa in the implication graph G. If we identify a suitable vertex, for which both reachability conditions are fulfilled (i.e., our nondeterministic algorithm

succeeds), then the formula is obviously unsatisfiable. If no such vertex is found (no accepting computation), then the formula is satisfiable. Thus this approach actually solves (accepts) unsatisfiability (instead of satisfiability), so we obtain UNSAT $\in$ **NL** and SAT $=$ co-UNSAT $\in$ **co-NL**. Since we deal with nondeterministic approaches, these two complementary problems (satisfiability and unsatisfiability) are not necessarily interchangeable. However, we have **co-NL** $=$ **NL** by Immerman-Szelepcsényi, which proves that SAT $=$ co-UNSAT $\in$ **co-NL** $=$ **NL**.

Exercise 4.4

Show the following statement:

Let $s, s_c \colon \mathbb{N} \to \mathbb{N}$ be space-constructible with $s(n) \geq \log n$ and $s_c(n) = s(n) + c$ for all $n, c \in \mathbb{N}$. For every s-space-bounded Turing machine, there exists a constant $c \in \mathbb{N}$ and an equivalent s_c-space-bounded Turing machine (i.e., one that accepts the same language) that terminates on each input.

Demonstrate the basic working principle of the constructed terminating Turing machine.
You need not prove its correctness.

Solution: By Lemma §1.7 such a TM allows at most $2^{O(s)}$ configurations and can thus take at most that many steps before repeating a configuration. Repeated configurations indicate an endless loop for deterministic TM and are generally useless for acceptance (for any computation the removal of any sequence of steps between repeating configurations yields another computation with the same acceptance behavior). Thus for sufficiently long inputs we implement a counter for the number of steps that counts to at most $2^{k \cdot s(n)}$ for some $k \in \mathbb{N}$, which is computable due to the space-constructibility of s. Shorter inputs are directly accepted or rejected based on memorized information. The space-constructibility of s shows that a unary representation $0^{s(n)}$ of $s(n)$ can be computed in space bounded by $s(n)$. We put k such blocks of 0s in sequence and add a single bit 1 in front of the resulting sequence $0^{k \cdot s(n)}$ to obtain a binary representation of $2^{k \cdot s(n)}$. Additional space $\log(2^{k \cdot s(n)}) = k \cdot s(n)$ is required for this counter. By linear tape-compression (Theorem §2.5) we can reduce the space requirement to $s = s_0$. If we reach the maximal value of the counter, then we certainly repeated a configuration and can thus reject. The such constructed Turing machine obviously terminates for each input.

The more suitable result of Lemma §1.7 for this task is that there exist $a, b \in \mathbb{N}$ such that at most $a \cdot n \cdot b^{s(n)}$ configurations exist for an input of length n. The remaining proof proceeds in much the same fashion, but the distinction into short and long inputs can be avoided in this case, since the bound is absolute (instead of the asymptotic bound provided by the O-notation).