Homework 1, Computational Complexity 2025

The deadline is 23:59 on Wednesday 8 October. Please submit your solutions on Moodle. Typing your solutions using LaTeX is strongly encouraged. The problems are meant to be worked on in groups of 1–2 students. Please submit only one writeup per team. You are strongly encouraged to solve these problems by yourself. If you must, you may use books or online resources to help solve homework problems, but you must credit all such sources in your writeup and you must never copy material verbatim.

1 Define a restriction of the SAT problem by

$$\mathrm{SAT}_k \coloneqq \{\langle\phi\rangle : \phi \text{ is a satisfiable CNF where each variable appears in at most } k \text{ clauses}\}.$$

1a Show that $\mathrm{SAT}_2 \in \mathsf{P}$.

1b Show that SAT_3 is NP-complete.

Solution:

1a For a CNF ϕ with all variables appearing in at most two clauses we can take any variable x and eliminate it from ϕ preserving satisfiability. Indeed $\phi(x_1, \ldots, x_n) = \psi(x_1, \ldots, x_n) \wedge \phi'(x_2, \ldots, x_n)$, where ψ is the conjunction of all clauses containing x_1 and ϕ' is the conjunction of all clauses that do not contain x_1. If ψ contains only one literal x_1 or $\neg x_1$, then we can assign x_1 the corresponding value to satisfy ψ, hence ϕ is satisfiable if and only if ϕ' is.

Therefore, the only interesting case is if both literals x_1 and $\neg x_1$ are contained in ψ. Since the variable x_1 can only be contained in two clauses, we have $\psi = (x_1 \vee A) \wedge (\neg x_1 \vee B)$, where A and B are clauses in variables $x_2, \ldots, x_n$. Then ψ' is satisfiable if and only if $\psi' \coloneqq A \vee B$ is: let $\rho\colon \{x_1, \ldots, x_n\} \to \{0, 1\}$ be an assignment satisfying ψ, then the restriction $\rho|_{\{x_2,\ldots,x_n\}}$ satisfies A if $\rho(x_1) = 0$ and B is $\rho(x_1) = 1$. In the other direction is ρ' satisfies $A \vee B$ let $\rho(x_1) \coloneqq 0$ if $\rho'(A) = 1$ and $\rho(x_1) \coloneqq 1$ if $\rho'(B) = 1$.

Overall, we have defined a mapping $\phi \mapsto \phi''$ such that ϕ'' has $n-1$ variables and is satisfiable iff ϕ is. Moreover, the total size of clauses in ϕ'' is smaller than the size of ϕ. Hence, we can eliminate all variables in the initial ϕ. The resulting formula is either empty, or has an empty clause, in the former case it is satisfiable, in the latter it is not.

1b Here, we also devise a mapping $\phi \mapsto \phi'$ such that ϕ is satisfiable iff ϕ' is and every variable appears in at most three clauses of ϕ'. Suppose we have a variable x that appears in $\ell > 3$ clauses, so

$$\phi(x, \vec{y}) = \bigwedge_{i \in [\ell]} (t_i \vee A_i(\vec{y})) \wedge \phi'(\vec{y}),$$

where $t_i \in \{x, \neg x\}$, A_i are clauses. We then replace each r_i with a literal of a fresh variable x_i with the same sign and enforce that for every satisfying assignment of the new formula all fresh variables have the same value:

$$\phi''(x_1, \ldots, x_\ell, \vec{y}) \coloneqq \bigwedge_{i \in [\ell]} (r_i \vee A_i(\vec{y})) \wedge \phi'(\vec{y}) \wedge \bigwedge_{i \in [\ell]} (x_i \implies x_{(i \bmod n)+1}),$$

where $r_i = x_i$ if $t_i = x$ and $r_i = \neg x_i$ if $t_i = \neg x$. Every satisfying assignment of ϕ'' must have $x_1 = x_2 = \cdots = x_\ell$ to satisfy the clauses $x_i \implies x_{(i \bmod n)+1}$, so assigning this value to x yields a satisfying assignment for ϕ. On the other hand if ρ is a satisfying assignment to ϕ, by defining $\rho'(x_i) \coloneqq \rho(x)$ and $\rho'(\vec{y}) \coloneqq \rho(\vec{y})$ we get a satisfying assignment for ϕ''. Therefore ϕ'' is satisfiable iff ϕ is. Each x_i appears in exactly 3 clauses. Then applying this mapping for all variables that appear in more than three clauses we get a CNF ϕ_{final}. We will apply the mapping at most n times and each application increases the size of the formula at most by n, so the size ϕ_{final} is polynomial in ϕ and n.

2 Let $G = (V, E)$ be an undirected graph. A set of vertices $D \subseteq V$ is *dense* if every vertex $v \in V \setminus D$ has a neighbour in D (that is, $\{v, u\} \in E$ for some $u \in D$). Prove the following NP-complete:

$$\textsc{DenseSet} = \{\langle G, k\rangle : G \text{ has a dense set of size at most } k\}.$$

Solution: We need to show two things: That DenseSet is in NP and that DenseSet is NP-hard.

DenseSet is in NP: The polynomial time verifier will take $(G = (V, E), k)$ as the problem instance and a set D as the certificate. The algorithm will check if:

1. $D \subseteq V$
2. $|D| \leq k$
3. For every vertex $v \in V \setminus D$, v has at least one edge to D.

The running time is $O(|V||E|)$, which is polynomial in the input.

DenseSet is NP-hard: We reduce the known NP-complete problem VertexCover to DenseSet.

Reduction: Given a graph $G = (V, E)$ and a number k, we define the instance $G' = (V', E')$ of DenseSet as follows: Remove any isolated vertices. For each edge $(u, v) \in E$, create a new vertex w_{uv} and add the edges (w_{uv}, u) and (w_{uv}, v). Let $V' = V \cup \{w_{uv} : (u, v) \in E\} \setminus \{\text{isolated vertices in } V\}$ and let $E' = E \cup \{(w_{uv}, u), (w_{uv}, v) : (u, v) \in E\}$. This is clearly a polynomial-time reduction.

Claim: G has a vertex cover of size k if and only if G' has a dense set of size k.
Proof of claim:
[$\implies$]: Suppose that S is a vertex cover of size k for G. Then $D = S$ is a dense set in G'. Indeed, given a vertex $v' \in V'$, if v' is one of the "old" vertices from V, then v' must have at least one edge $(u, v') \in E$. Since $S = D$ is a vertex cover, we have that either $v' \in D$, or $u \in D$, as required. If instead $v' = w_{uv}$ is a vertex of the new type, then w_{uv} has edges to both u and v, and either $u \in D$ or $v \in D$ (because D is a vertex cover in G), so w_{uv} has an edge to an element of D.

[$\Longleftarrow$]: Suppose that D is a dense set of size k in G'. Then we can create a vertex cover S for G as follows: For each vertex $w_{uv} \in D$ of the new type, replace w_{uv} by u or v. Let D' be the resulting set. Note that D' is still a dense set in G', and the size has not increased. Moreover, we have that $D' \subseteq V$ by construction. Let $S = D'$. We claim that S is a vertex cover in G. Indeed, suppose for contradiction that there exists an edge $(u, v) \in E$ such that $u \notin S$ and $v \notin S$. Then w_{uv} does not have any neighbours in D', and $w_{uv} \notin D'$ (by construction of D'), which contradicts that D' is a dense set in G'.

3 A *monotone* boolean function $f\colon \{0,1\}^n \to \{0,1\}$ is one that has the following property: If one of the input bits changes from 0 to 1, the value of the function cannot change from 1 to 0. Show that f is monotone if and only if it can be computed by a circuit with only $\vee$ and $\wedge$ gates.

Solution: First let us prove that every monotone circuit computes a monotone function. We can do it by induction on the size of the circuit C computing f. The base case where C has no gates is trivial. Then suppose $|C| = k > 1$. By definition $C(x) = C_1(x) \diamond C_2(x)$ where $\diamond \in \{\vee, \wedge\}$ and $|C_i| < k$ for $i \in \{1, 2\}$. By the induction hypothesis we have that C_1 and C_2 compute monotone functions f_1 and f_2 respectively. Consider any $x \leq y \in \{0,1\}^n$, we need to show that $f(x) \leq f(y)$. We have that $f_i(x) \leq f_i(y)$ for $i \in \{1, 2\}$. Then $(f_1(x), f_2(x)) \leq (f_1(y), f_2(y))$. Since $\diamond$ is monotone, we have $f_1(x) \diamond f_2(x) \leq f_1(y) \diamond f_2(y)$.

Suppose now that $f\colon \{0,1\}^n \to \{0,1\}$ is monotone. Then

$$f(x) = \bigvee_{s \in \{0,1\}^n\colon f(s)=1} (x \geq s) = \bigvee_{s \in \{0,1\}^n\colon f(s)=1} \bigwedge_{i \in [n]\colon s_i = 1} x_i.$$

Indeed if $f(x) = 1$, then the term corresponding to x on the right-hand side is going to be satisfied. On the other hand if for $x \in \{0,1\}^n$ a term for s is satisfied, it means that there exists $s \leq x$ such that $f(s) = 1$. Then by monotonicity $f(x) = 1$.