
Problem 7.24. Let

$$CNF_k = \{\langle\phi\rangle \mid \phi \text{ is a satisfiable cnf-formula where each variable appears in at most } k \text{ places}\}.$$

Part a. Show that $CNF_2 \in P$.

Proof. We show that $CNF_2 \in P$ by presenting a polynomial time algorithm that decides CNF_2. A polynomial time algorithm M for CNF_2 operates as follows.

$M =$ "On input $\langle\phi\rangle$, ϕ is a cnf-formula where each variable appears in at most 2 places:

1. Construct ϕ' by rewriting ϕ as OR of ANDs using distributive laws.
2. If all conjunctions in ϕ' contain a contradiction, then *reject*.
3. *Accept*."

□

Let $x_1, x_2, \cdots, x_l$ be the variables of ϕ. The maximum number of clauses in ϕ can be at most $2l$. The distributive laws, as described in Chapter 0, state that we can replace an AND of ORs with an equivalent OR of ANDs. Doing so may significantly increase the size of each subformula, but it creates at most polynomial number of conjunctions.

Part b. Show that CNF_3 is NP-complete.

Proof. To show that CNF_3 is NP-complete, we must show that it is in NP and that all NP-problems are polynomial time reducible to it. The first part is easy; a certificate is simply a satisfying assignment. To prove the second part, we show that $3SAT$ is polynomial time reducible to CNF_3. The reduction converts a 3cnf-formula ϕ into a cnf-formula ϕ', where each variable appears in at most 3 places, so that ϕ is satisfiable, iff ϕ' is satisfiable.

We show that an equivalent cnf-formula ϕ' can be constructed for every 3cnf-formula ϕ in polynomial time. Such a ϕ' contains additional variables and clauses, so that no variable is used more than 3 times. The following methods can be used to reduce the number of times a variable is used, while retaining satisfiability.

1. Use identity of $\vee$[1] to rewrite $(a \vee a \vee \cdots)$ as $(a \vee \cdots)$.
2. Use law of excluded middle[2] to rewrite $(a \vee \overline{a} \vee \cdots)$ as $(b \vee \overline{b} \vee \cdots)$, where b is a single additional variable.

[1] $p \vee p \equiv p$
[2] $p \vee \overline{p} \equiv true$

3. If a literal appears in more than one clause

$$(a \vee \cdots) \wedge (a \vee \cdots) \wedge \cdots \wedge (a \vee \cdots),$$

then rewrite the clauses as

$$(a \vee \overline{z}) \wedge (z \vee \cdots) \wedge (z \vee \cdots) \wedge \cdots \wedge (z \vee \cdots),$$

where z is a new variable for each application of this method.

Similarly, an equivalent 3cnf-formula ϕ can be construct for every cnf-formula ϕ'.

1. In ϕ', replace any clause c_i that contains more than 3 literals

$$(a_i \vee b_i \vee c_i \vee d_i \vee \cdots),$$

with the two clauses

$$(a_i \vee b_i \vee \overline{z_i}) \wedge (z_i \vee c_i \vee d_i \vee \cdots),$$

where z_i is a new variable for each clause c_i. Repeatedly apply this method until all clauses contain at most 3 literals.

2. Rewrite (a_i) as $(a_i \vee a_i \vee a_i)$.

3. Rewrite $(a_i \vee b_i)$ as $(a_i \vee a_i \vee b_i)$.

□