T-79.3001 Logic in computer science: foundations **Spring 2007**
Exercise 6 ([Nerode and Shore, 1997], Chapter 8)
March 13–15, 2007

Solutions to demonstration problems

4. A few weeks ago a traffic light system was modeled. Transform the propositions specifying the behaviour of the system into clausuls and prove with resolution that both red lights are not on at the same time.

Solution. We transform the propositions into CNF and clauses. The last proposition in the table is the negation of statement "both red lights are not on at the same time", that is,

$$\neg(\neg(P1 \wedge P2)) \equiv P1 \wedge P2.$$

$Pi \vee Ki \vee Vi$		$\{Pi, Ki, Vi\}$
$Pi \rightarrow \neg Ki \wedge \neg Vi$	$\equiv \neg Pi \vee (\neg Ki \wedge \neg Vi)$ $\equiv (\neg Pi \vee \neg Ki) \wedge (\neg Pi \vee \neg Vi)$	$\{\neg Pi, \neg Ki\}, \{\neg Pi, \neg Vi\}$
$Ki \rightarrow \neg Pi \wedge \neg Vi$	$\equiv (\neg Ki \vee \neg Pi) \wedge (\neg Ki \vee \neg Vi)$	$\{\neg Pi, \neg Ki\}, \{\neg Ki, \neg Vi\}$
$Vi \rightarrow \neg Pi \wedge \neg Ki$	$\equiv (\neg Vi \vee \neg Pi) \wedge (\neg Vi \vee \neg Ki)$	$\{\neg Pi, \neg Vi\}, \{\neg Ki, \neg Vi\}$
$\neg(V1 \wedge V2)$	$\equiv \neg V1 \vee \neg V2$	$\{\neg V1, \neg V2\}$
$P1 \rightarrow (K2 \vee V2)$	$\equiv \neg P1 \vee K2 \vee V2$	$\{\neg P1, K2, V2\}$
$P2 \rightarrow (K1 \vee V1)$	$\equiv \neg P2 \vee K1 \vee V1$	$\{\neg P2, K1, V1\}$
$P1 \wedge P2$		$\{P1\}, \{P2\}$

We show that the set of clauses given in the table is unsatisfiable (empty clause $\Box$ means contradiction), which implies that $\neg(P1 \wedge P2)$ is derivable from the other clauses.

$$\begin{array}{lllll}
\{\neg P1, K2, V2\} \quad \{P1\} & & & & \\
\quad \{K2, V2\} & \{\neg P2, \neg K2\} & & & \\
& \quad \{\neg P2, V2\} & \{\neg P2, \neg V2\} & & \\
& & \quad \{\neg P2\} & \{P2\} & \\
& & & \quad \Box &
\end{array}$$

5. One successful application of expert systems has been analyzing the problem of which chemical syntheses are possible. Consider the following chemical reactions:

(1) $MgO + H_2 \rightarrow Mg + H_2O$

(2) $C + O_2 \rightarrow CO_2$

(3) $CO_2 + H_2O \rightarrow H_2CO_3$

a) Represent these rules and the assumptions that we have some MgO, H_2, O_2 and C by propositional logic formulas.

b) Give a resolution proof that we can get some H_2CO_3.

Solution. The chemical reactions can be formalized as implications, which can then be transformed into clausul form. The resulting clauses are:

(1)

$$\begin{aligned}
& MgO + H_2 \rightarrow Mg + H_2O \\
\Longrightarrow & MgO \wedge H_2 \rightarrow Mg \wedge H_2O \\
\Longrightarrow & \neg MgO \vee \neg H_2 \vee (Mg \wedge H_2O) \\
\Longrightarrow & (\neg MgO \vee \neg H_2 \vee Mg) \wedge (\neg MgO \vee \neg H_2 \vee H_2O)
\end{aligned}$$

The first reaction results in two clauses: : $\{\neg MgO, \neg H_2, Mg\}$ and $\{\neg MgO, \neg H_2, H_2O\}$.

(2)

$$\begin{aligned}
& C + O_2 \rightarrow CO_2 \\
\Longrightarrow & C \wedge O_2 \rightarrow CO_2 \\
\Longrightarrow & \neg C \vee \neg O_2 \vee CO_2 \\
\Longrightarrow & \{\neg C, \neg O_2, CO_2\}
\end{aligned}$$

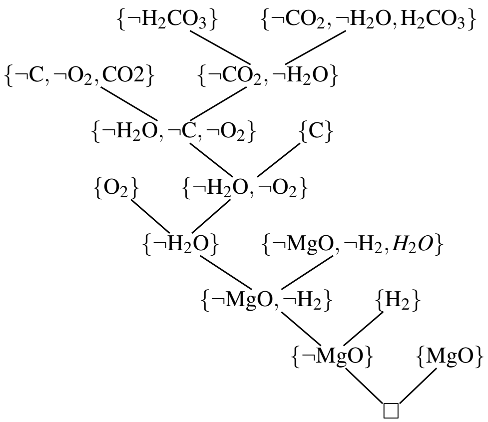
(3)

$$\begin{aligned}
& CO_2 + H_2O \rightarrow H_2CO_3 \\
\Longrightarrow & CO_2 \wedge H_2O \rightarrow H_2CO_3 \\
\Longrightarrow & \neg CO_2 \vee \neg H_2O \vee H_2CO_3 \\
\Longrightarrow & \{\neg CO_2, \neg H_2O, H_2CO_3\}
\end{aligned}$$

The elements availabe at the start are:

$$\mathrm{MgO} \wedge \mathrm{H_2} \wedge \mathrm{O_2} \wedge \mathrm{C}$$
$$\Longrightarrow \{\mathrm{MgO}\}, \{\mathrm{H_2}\}, \{\mathrm{O_2}\}\{\mathrm{C}\}$$

We denote the above set of clauses with Σ. now we want to prove that $\Sigma \models \mathrm{H_2CO_3}$. The proof is constructed by showing that $\Sigma \cup \{\neg \mathrm{H_2CO_3}\}$ is unsatisfiable.



6. Construct a deterministic Turing machine that counts the successor of a given binary number.

Solution. The solution is obtained from "Computational Complexity" by C. Papadimitriou. A deterministic Turing machine is a quadruple $\langle A, S, s_0, t \rangle$, where

- A is the alphabet,
- S is the set of states,
- $t : S \times A \to S \times A \times \{\to, \leftarrow, \downarrow\}$ is the state transition function
- $s_0 \in S$ is the start state.

For our machine we have $S = \{s\}$, $A = \{0, 1\}$, $s_0 = s$ and the state transition function is given in the following table:

$p \in S$	$\sigma \in A$	$t(p, \sigma)$
s	0	$(h, 1, -)$
s	1	$(s, 0, \to)$
s	$\sqcup$	$(h, 1, -)$
s	$\triangleright$	$(s, \triangleright, \to)$

With input 1101 the computation goes as follows: $(s, \triangleright, 1101) \xrightarrow{M} (s, \triangleright 0, 101) \xrightarrow{M} (s, \triangleright 00, 01) \xrightarrow{M} (h, \triangleright 001, 1)$.

7. Show the problem of 3-coloring a graph is in the class **NP** by reducing it into the propositional satisfiability problem.

Solution. The problem of 3-coloring a graph is as follows: "give a graph G, is there a way to color the nodes in G using 3 colors so that no two adjacent nodes have same color?"

Let $N = \{n_1, n_2, \ldots, n_m\}$ be the set of nodes and $E \subseteq N \times N$ the set of edges.

For each node n_i we take atomic propositions $R_{n_i}, G_{n_i}, B_{n_i}$ to denote that node n_i is colored red, green or blue, respectively.

Each node is colored with some color, that is, $R_{n_i} \vee G_{n_i} \vee B_{n_i}$, for each n_i.

No node is colored with two different colors, that is,

$$(R_{n_i} \to (\neg G_{n_i} \wedge \neg B_{n_i})) \wedge (G_{n_i} \to (\neg R_{n_i} \wedge \neg B_{n_i})) \wedge (B_{n_i} \to (\neg R_{n_i} \wedge \neg G_{n_i})),$$

for each n_i.

Finally, two adjacent color can't have same color, that is,

$$(R_n \to \neg R_m) \wedge (G_n \to \neg G_m) \wedge (B_n \to \neg B_m),$$

for each $(n, m) \in E$.

Now, if we take the conjunction of all these propositions (denoted by ϕ), then ϕ is satisfiable iff the graph has a 3-coloring (the proof is omitted).