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 behavior of the system into clauses 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,

\[ \neg((P_1 \land P_2)) \equiv P_1 \land \neg P_2. \]

We show that the set of clauses given in the table is unsatisfiable (empty clause □ means contradiction), which implies that \( \neg(P_1 \land P_2) \) is derivable from the other clauses.

\[
\begin{align*}
P_1 \lor K_1 \lor V_1 & \quad \{P_1, K_1, V_1\} \\
P_1 & \rightarrow \neg K_1 \land \neg V_1 \equiv \neg P_1 \lor (\neg K_1 \land \neg V_1) \quad \{P_1, \neg K_1, \neg V_1\} \\
K_1 & \rightarrow \neg P_1 \land \neg V_1 \equiv \neg P_1 \lor (\neg K_1 \land \neg V_1) \quad \{-P_1, \neg K_1, \neg V_1\} \\
V_1 & \rightarrow \neg P_1 \land \neg K_1 \equiv \neg P_1 \lor (\neg V_1 \land \neg K_1) \quad \{-P_1, \neg V_1, \neg K_1\} \\
\neg(V_1 \land V_2) & \equiv \neg V_1 \lor \neg V_2 \quad \{-V_1, \neg V_2\} \\
P_1 \rightarrow \neg K_1 \land \neg V_1 \equiv \neg P_1 \lor \neg K_1 \lor \neg V_1 \quad \{\neg P_1, \neg K_1, \neg V_1\} \\
P_2 \rightarrow \neg K_1 \land \neg V_1 \equiv \neg P_2 \lor \neg K_1 \lor \neg V_1 \quad \{\neg P_2, \neg K_1, \neg V_1\} \\
P_1 \land P_2 & \quad \{P_1, P_2\} \\
\neg P_1, K_2, V_2 & \quad \{P_1\} \\
K_2, V_2 & \quad \{P_2, \neg K_2\} \\
\neg P_2, V_2 & \quad \{\neg P_2, \neg V_2\} \\
\neg P_2 & \quad \{\neg P_2\} \\
\boxed{} & \quad \{\neg P_1, K_2, V_2\} \\
\end{align*}
\]

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

1. \( \text{MgO} + \text{H}_2 \rightarrow \text{Mg} + \text{H}_2\text{O} \)
2. \( \text{C} + \text{O}_2 \rightarrow \text{CO}_2 \)
3. \( \text{CO}_2 + \text{H}_2 \rightarrow \text{H}_2\text{CO}_3 \)

a) Represent these rules and the assumptions that we have some MgO, H2, O2 and C by propositional logic formulas.
b) Give a resolution proof that we can get some H2CO3.

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

\[
\begin{align*}
\text{MgO} + \text{H}_2 & \rightarrow \text{Mg} + \text{H}_2\text{O} \\
\Rightarrow & \quad \text{MgO} \land \text{H}_2 \rightarrow \text{Mg} \land \text{H}_2\text{O} \\
& \quad \Rightarrow \quad \neg\text{MgO} \lor \neg\text{H}_2 \lor (\text{Mg} \land \text{H}_2\text{O}) \\
& \quad \Rightarrow \quad (\neg\text{MgO} \lor \neg\text{H}_2 \lor \text{Mg}) \land (\neg\text{MgO} \lor \neg\text{H}_2 \lor \text{H}_2\text{O}) \\
\end{align*}
\]

The first reaction results in two clauses: \( \{\neg\text{MgO}, \neg\text{H}_2, \text{Mg}\} \) and \( \{\neg\text{MgO}, \neg\text{H}_2, \text{H}_2\text{O}\} \).

\[
\begin{align*}
\text{C} + \text{O}_2 & \rightarrow \text{CO}_2 \\
\Rightarrow & \quad \text{C} \land \text{O}_2 \rightarrow \text{CO}_2 \\
& \quad \Rightarrow \quad \neg\text{C} \lor \neg\text{O}_2 \lor \text{CO}_2 \\
& \quad \Rightarrow \quad \{\neg\text{C}, \neg\text{O}_2, \text{CO}_2\} \\
\end{align*}
\]

\[
\begin{align*}
\text{CO}_2 + \text{H}_2 & \rightarrow \text{H}_2\text{CO}_3 \\
\Rightarrow & \quad \text{CO}_2 \land \text{H}_2 \rightarrow \text{H}_2\text{CO}_3 \\
& \quad \Rightarrow \quad \neg\text{CO}_2 \lor \neg\text{H}_2 \lor \text{H}_2\text{CO}_3 \\
& \quad \Rightarrow \quad \{\neg\text{CO}_2, \neg\text{H}_2, \text{H}_2\text{CO}_3\} \\
\end{align*}
\]
The elements available at the start are:
\[ \text{MgO} \land H_2 \land O_2 \land C \]
\[ \implies \{\text{MgO}\}, \{H_2\}, \{O_2\}, \{C\} \]

We denote the above set of clauses with \( \Sigma \). Now we want to prove that \( \Sigma \models H_2\text{CO}_3 \). The proof is constructed by showing that \( \Sigma \cup \{\neg H_2\text{CO}_3\} \) is unsatisfiable.

\[
\{\neg H_2\text{CO}_3\} \quad \{\neg\text{CO}_2, \neg\text{H}_2\text{O}, \text{H}_2\text{CO}_3\}\]
\[
\{\neg\text{C}, \neg\text{O}_2, \text{CO}_2\} \quad \{\neg\text{CO}_2, \neg\text{H}_2\text{O}\}\]
\[
\{\neg\text{H}_2\text{O}, \neg\text{C}, \neg\text{O}_2\} \quad \{\text{C}\}\]
\[
\{\text{O}_2\} \quad \{\neg\text{H}_2\text{O}, \neg\text{O}_2\}\]
\[
\{\neg\text{H}_2\text{O}\} \quad \{\neg\text{MgO}, \neg\text{H}_2\text{O}\}\]
\[
\{\neg\text{MgO}, \neg\text{H}_2\}\quad \{\text{H}_2\}\]
\[
\{\neg\text{MgO}\} \quad \{\text{MgO}\}\]

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 \( \langle A, S, s_0, \tau \rangle \), where

- \( A \) is the alphabet,
- \( S \) is the set of states,
- \( \tau : S \times A \rightarrow S \times A \times \{\rightarrow, \leftarrow, \downarrow\} \) 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:

<table>
<thead>
<tr>
<th>( p \in \Delta )</th>
<th>( \sigma \in A )</th>
<th>( \tau(p, \sigma) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( s )</td>
<td>0</td>
<td>( (h, 1, \rightarrow) )</td>
</tr>
<tr>
<td>( s )</td>
<td>1</td>
<td>( (s, 0, \rightarrow) )</td>
</tr>
<tr>
<td>( s )</td>
<td>( \neg )</td>
<td>( (h, 1, \rightarrow) )</td>
</tr>
<tr>
<td>( s )</td>
<td>( \neg )</td>
<td>( (s, \neg, \rightarrow) )</td>
</tr>
</tbody>
</table>

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

7. Show the problem of 3-coloring a graph is in the class \( \text{NP} \) 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, \( R_{n_i} \lor G_{n_i} \lor B_{n_i} \) for each \( n_i \).

No node is colored with two different colors, that is,
\[
(R_{n_i} \lor \neg G_{n_i} \lor \neg B_{n_i}) \land (G_{n_i} \lor \neg R_{n_i} \lor \neg B_{n_i}) \land (B_{n_i} \lor \neg R_{n_i} \lor \neg G_{n_i}),
\]
for each \( n_i \).

Finally, two adjacent color can’t have same color, that is,
\[
(R_{n_i} \lor \neg R_{n_{im}}) \land (G_{n_i} \lor \neg G_{n_{im}}) \land (B_{n_i} \lor \neg B_{n_{im}}),
\]
for each \( (n_i, m) \in E \).

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