Models and entailment

A model is an assignment of Boolean values to the Boolean variables that define the domain of interest. In propositional logic the number of models is finite, and a formula can be described by a truth-table, exponential in the number of variables. In first-order logic the number of models is infinite. In both cases a Boolean formula specifies a set of models that satisfy it.

A knowledge base is a collection of Boolean formulas. (It can also be viewed as a single Boolean formula, constructed as a conjunction of all the formulas in the knowledge base.) It specifies a set of models that satisfy all of its formulas. The entailment relation is defined as follows. We say that the knowledge base KB entails the formula \( \beta \) (or that \( \beta \) is entailed by KB) if \( \beta \) is true in all models of KB. The mathematical notation is:

\[
KB \models \beta
\]

This is sometimes also written as:

\[
KB \Rightarrow \beta
\]

Proving entailment

Using truth table

In propositional calculus, when given KB and \( \beta \) we can prove that \( KB \models \beta \) by the truth-table approach. Create the truth table of \( KB \) and verify that whenever KB is true, \( \beta \) is also true.

Example:

\[
KB = x \land y, \quad \beta = x \lor y
\]

<table>
<thead>
<tr>
<th>x</th>
<th>y</th>
<th>KB</th>
<th>\beta</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

Since \( \beta \) is true whenever KB is true (the last row) it follows that \( KB \models \beta \).

Proof by negation (refutation)

A similar truth-table approach is to negate \( \beta \), add it to KB, and then show that there are no models that satisfy the result. Formally we create a new knowledge base NKB as follows:

\[
NKB = KB \land \neg \beta
\]

A truth table can then be used to show that NKB has no satisfying models.

Example:

\[
KB = x \land y, \quad \beta = x \lor y, \quad \neg \beta = \overline{x} \land \overline{y}, \quad NKB = (x \land y) \land (\overline{x} \land \overline{y})
\]

<table>
<thead>
<tr>
<th>x</th>
<th>y</th>
<th>KB</th>
<th>\beta</th>
<th>NKB</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

Since NKB has no satisfying models it follows that \( KB \models \beta \).
Using search

The problem with the truth-table approach is that it is exponential. For a more efficient proof by negation we can use a search technique, searching for a model that satisfies NKB. There are no search techniques that are guaranteed to always work in polynomial time. But there are several techniques that work reasonably fast for some real-world applications.

Proof by inference

A proof by inference of the formula \( \beta \) from the knowledge base KB is set of formulas \{f_1, f_2, \ldots, f_n\}, where each \( f_i \) is either in KB or can be inferred from \{f_1, f_2, \ldots, f_{i-1}\} using rules of inference. There are many rules of inference that can be used with Propositional logic; here are the most common:

\[
\begin{align*}
(A) & \quad \text{Modus Ponens:} & \alpha & \rightarrow \beta, \quad \alpha & \quad & \beta \\
(B) & \quad \text{AND-Elimination:} & \alpha_1 \land \alpha_2 \land \ldots \land \alpha_n & \quad & \alpha_i \\
(C) & \quad \text{AND-Introduction:} & \alpha_1, \alpha_2, \ldots, \alpha_n & \quad & \alpha_1 \land \alpha_2 \land \ldots \land \alpha_n \\
(D) & \quad \text{OR-Introduction:} & \alpha_i & \quad & \alpha_1 \lor \alpha_2 \lor \ldots \lor \alpha_n \\
(E) & \quad \text{Double-Negation Elimination:} & \neg \neg \alpha & \quad & \alpha \\
(F) & \quad \text{Unit Resolution:} & \alpha \lor \beta, \quad \neg \beta & \quad & \alpha \\
(G) & \quad \text{Resolution:} & \alpha \lor \beta, \quad \neg \beta \lor \gamma & \quad & \alpha \lor \gamma \\
\end{align*}
\]

or equivalently:

\[
\begin{align*}
& \neg \alpha \rightarrow \beta, \quad \beta \rightarrow \gamma \\
& \neg \alpha \rightarrow \gamma
\end{align*}
\]

Example: KB = \( x \land y \), \( \beta = x \lor y \)

1. \( x \land y \) \quad \text{in \( KB \)}
2. \( x \) \quad \text{from 1 using B}
3. \( x \lor y \) \quad \text{from 2 using D}
**Example:** \( KB = S, (S \lor P) \rightarrow (Q \land R), X. \) Show: \( Q \land X. \)

<table>
<thead>
<tr>
<th>Step</th>
<th>Statement</th>
<th>Source</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>( S )</td>
<td>in KB</td>
</tr>
<tr>
<td>2.</td>
<td>( S \lor P )</td>
<td>from 1 using OR introduction</td>
</tr>
<tr>
<td>3.</td>
<td>( (S \lor P) \rightarrow (Q \land R) )</td>
<td>in KB</td>
</tr>
<tr>
<td>4.</td>
<td>( (Q \land R) )</td>
<td>from 2,3 using Modus Ponens</td>
</tr>
<tr>
<td>5.</td>
<td>( Q )</td>
<td>from 4 using AND elimination</td>
</tr>
<tr>
<td>6.</td>
<td>( X )</td>
<td>in KB</td>
</tr>
<tr>
<td>7.</td>
<td>( Q \land X )</td>
<td>from 5,6 using AND introduction</td>
</tr>
</tbody>
</table>