Horn clauses

- An integrity constraint is a clause of the form

\[ \text{false} \leftarrow a_1 \land \ldots \land a_k \]

where the \( a_i \) are atoms and \( \text{false} \) is a special atom that is false in all interpretations.

- A Horn clause is either a definite clause or an integrity constraint.
Negations can follow from a Horn clause KB.

The negation of $\alpha$, written $\neg \alpha$ is a formula that
- is true in interpretation $I$ if $\alpha$ is false in $I$, and
- is false in interpretation $I$ if $\alpha$ is true in $I$.

Example:

$$KB = \begin{cases} 
false & \leftarrow a \land b. \\
   a & \leftarrow c. \\
b & \leftarrow c. 
\end{cases}$$

$KB \models \neg c.$
Disjunctive Conclusions

- Disjunctions can follow from a Horn clause KB.
- The disjunction of $\alpha$ and $\beta$, written $\alpha \lor \beta$, is true in interpretation $I$ if $\alpha$ is true in $I$ or $\beta$ is true in $I$ (or both are true in $I$).
- false in interpretation $I$ if $\alpha$ and $\beta$ are both false in $I$.

Example:

$$KB = \begin{cases} 
false & \iff a \land b. \\
a & \iff c. \\
b & \iff d. 
\end{cases}$$

$KB \models \neg c \lor \neg d.$
Questions and Answers in Horn KBs

- An assumable is an atom whose negation you are prepared to accept as part of a (disjunctive) answer.

- A conflict of $KB$ is a set of assumables that, given $KB$ imply $false$.

- A minimal conflict is a conflict such that no strict subset is also a conflict.
Example: If \( \{c, d, e, f, g, h\} \) are the assumables

\[
KB = \left\{ \begin{array}{l}
false \leftarrow a \land b.
\end{array} \right.
\]

\[
\begin{align*}
& a \leftarrow c. \\
& b \leftarrow d. \\
& b \leftarrow e. \\
\end{align*}
\]

\[\uparrow\] \( \{c, d\} \) is a conflict

\[\uparrow\] \( \{c, e\} \) is a conflict

\[\uparrow\] \( \{c, d, e, h\} \) is a conflict
Assume that the user is able to observe whether a light is lit or dark and whether a power outlet is dead or live.

A light can’t be both lit and dark. An outlet can’t be both live and dead:

\[
\text{false} \iff \text{dark}(L) \& \text{lit}(L).
\]

\[
\text{false} \iff \text{dead}(L) \& \text{live}(L).
\]

Make \textit{ok} assumable: \textit{assumable}(ok(X)).

Suppose switches \(s_1, s_2, \) and \(s_3\) are all up: \(up(s_1) \cdot up(s_2) \cdot up(s_3)\).
lit(L) ⇐ light(L) & ok(L) & live(L).

live(W) ⇐ connected_to(W, W₁) & live(W₁).

live(outside) ⇐ true.

light(l₁) ⇐ true.

light(l₂) ⇐ true.

connected_to(l₁, w₀) ⇐ true.

connected_to(w₀, w₁) ⇐ up(s₂) & ok(s₂).

connected_to(w₁, w₃) ⇐ up(s₁) & ok(s₁).

connected_to(w₃, w₅) ⇐ ok(cb₁).

connected_to(w₅, outside) ⇐ true.
If the user has observed $l_1$ and $l_2$ are both dark:

$$dark(l_1). \ dark(l_2).$$

There are two minimal conflicts:

$$\{ok(cb_1), \ ok(s_1), \ ok(s_2), \ ok(l_1)\} \text{ and}$$

$$\{ok(cb_1), \ ok(s_3), \ ok(l_2)\}.$$

You can derive:

$$\neg ok(cb_1) \lor \neg ok(s_1) \lor \neg ok(s_2) \lor \neg ok(l_1)$$

$$\neg ok(cb_1) \lor \neg ok(s_3) \lor \neg ok(l_2).$$

Either $cb_1$ is broken or there is one of six double faults.
A **consistency-based diagnosis** is a set of assumables that has at least one element in each conflict.

A **minimal diagnosis** is a diagnosis such that no subset is also a diagnosis.

Intuitively, one of the minimal diagnoses must hold. A diagnosis holds if all of its elements are false.

**Example:** For the proceeding example there are seven minimal diagnoses: \( \{ok(cb_1)\} \), \( \{ok(s_1), ok(s_3)\} \), \( \{ok(s_1), ok(l_2)\} \), \( \{ok(s_2), ok(s_3)\} \),…

© David Poole, Alan Mackworth, Randy Goebel, and Oxford University Press 1998-2002
Meta-interpreter to find conflicts

\% dprove(G, D₀, D₁) is true if list D₀ is an ending of list D₁
\% such that assuming the elements of D₁ lets you derive G.

\[\text{dprove}(\text{true}, D, D).\]
\[\text{dprove}((A \& B), D₁, D₃) \leftarrow\]
\[\text{dprove}(A, D₁, D₂) \land \text{dprove}(B, D₂, D₃).\]
\[\text{dprove}(G, D, [G|D]) \leftarrow \text{assumable}(G).\]
\[\text{dprove}(H, D₁, D₂) \leftarrow\]
\[\quad (H \leftarrow B) \land \text{dprove}(B, D₁, D₂).\]
\[\text{conflict}(C) \leftarrow \text{dprove}(\text{false}, [ ], C).\]
false ⇐ a.

a ⇐ b & c.

b ⇐ d.

b ⇐ e.

c ⇐ f.

c ⇐ g.

e ⇐ h & w.

e ⇐ g.

w ⇐ d.

assumable d, f, g, h.
Conclusions are pairs \( \langle a, A \rangle \), where \( a \) is an atom and \( A \) is a set of assumables that imply \( a \).

Initially, conclusion set \( C = \{\langle a, \{a\} \rangle : a \text{ is assumable}\} \).

If there is a rule \( h \leftarrow b_1 \land \ldots \land b_m \) such that for each \( b_i \) there is some \( A_i \) such that \( \langle b_i, A_i \rangle \in C \), then \( \langle h, A_1 \cup \ldots \cup A_m \rangle \) can be added to \( C \).

If \( \langle a, A_1 \rangle \) and \( \langle a, A_2 \rangle \) are in \( C \), where \( A_1 \subset A_2 \), then \( \langle a, A_2 \rangle \) can be removed from \( C \).

If \( \langle \text{false}, A_1 \rangle \) and \( \langle a, A_2 \rangle \) are in \( C \), where \( A_1 \subseteq A_2 \), then \( \langle a, A_2 \rangle \) can be removed from \( C \).
Bottom-up Conflict Finding Code

\[ C := \{ \langle a, \{a\} \rangle : a \text{ is assumable } \}; \]

repeat
    select clause “\( h \leftarrow b_1 \land \ldots \land b_m \)” in \( T \) such that
    \[ \langle b_i, A_i \rangle \in C \text{ for all } i \text{ and} \]
    there is no \( \langle h, A' \rangle \in C \text{ or } \langle \text{false}, A' \rangle \in C \)
    such that \( A' \subseteq A \) where \( A = A_1 \cup \ldots \cup A_m \);
\[ C := C \cup \{ \langle h, A \rangle \} \]
    Remove any elements of \( C \) that can now be pruned;
until no more selections are possible