In the electrical domain, what if we predict that a light should be on, but observe that it isn’t? What can we conclude?

We will expand the definite clause language to include integrity constraints which are rules that imply false, where false is an atom that is false in all interpretations.

This will allow us to make conclusions from a contradiction.

A definite clause knowledge base is always consistent. This won’t be true with the rules that imply false.
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.
Negative Conclusions

- 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:

\[
\begin{align*}
KB &= \{ \begin{array}{l}
false \leftarrow a \land b. \\
a \leftarrow c. \\
b \leftarrow c.
\end{array} \} \\
KB \models \neg c.
\end{align*}
\]
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:

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

  $KB \models \neg c \lor \neg d.$
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

\[
KB = \begin{cases}
    false \leftarrow a \land b. \\
    a \leftarrow c. \\
    b \leftarrow d. \\
    b \leftarrow e.
\end{cases}
\]

- \{c, d\} is a conflict
- \{c, e\} is a conflict
- \{c, d, e, h\} is a conflict
Using Conflicts for Diagnosis

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:

\[
false \iff \text{dark}(L) \land \text{lit}(L).
\]
\[
false \iff \text{dead}(L) \land \text{live}(L).
\]

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

Suppose switches \(s_1\), \(s_2\), and \(s_3\) are all up:

\[up(s_1). \, up(s_2). \, up(s_3).\]
\[
lit(L) \leftarrow light(L) \& ok(L) \& live(L).
\]
\[
live(W) \leftarrow connected\_to(W, W_1) \& live(W_1).
\]
\[
live(outside) \leftarrow true.
\]
\[
light(l_1) \leftarrow true.
\]
\[
light(l_2) \leftarrow true.
\]
\[
connected\_to(l_1, w_0) \leftarrow true.
\]
\[
connected\_to(w_0, w_1) \leftarrow up(s_2) \& ok(s_2).
\]
\[
connected\_to(w_1, w_3) \leftarrow up(s_1) \& ok(s_1).
\]
\[
connected\_to(w_3, w_5) \leftarrow ok(cb_1).
\]
\[
connected\_to(w_5, outside) \leftarrow true.
\]
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.
Diagnoses

- 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)\},…
Meta-interpreter to find conflicts

\% \texttt{dprove}(G, D_0, D_1) \text{ is true if list } D_0 \text{ is an ending of list } D_1 \text{ such that assuming the elements of } D_1 \text{ lets you derive } G.

\begin{align*}
\texttt{dprove}(\text{true}, D, D). \\
\texttt{dprove}((A \& B), D_1, D_3) \leftarrow \\
\quad \texttt{dprove}(A, D_1, D_2) \land \texttt{dprove}(B, D_2, D_3). \\
\texttt{dprove}(G, D, [G|D]) \leftarrow \texttt{assumable}(G). \\
\texttt{dprove}(H, D_1, D_2) \leftarrow \\
\quad (H \leftarrow B) \land \texttt{dprove}(B, D_1, D_2). \\
\texttt{conflict}(C) \leftarrow \texttt{dprove}(false, [], C).
\end{align*}
Tricky Example

false \Leftarrow a.

a \Leftarrow b \& c.

b \Leftarrow d.

b \Leftarrow e.

c \Leftarrow f.

c \Leftarrow g.

e \Leftarrow h \& w.

e \Leftarrow g.

w \Leftarrow 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 $C = \{ \langle a, \{a\} \rangle : a$ 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 $\langle false, A_1 \rangle$ 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 \( \langle h, A' \rangle \in C \) 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
Integrity Constraints in Databases

Database designers can use integrity constraints to specify constraints that should never be violated.

Example: A student can't have two different grades for the same course.

\[
\text{false} \leftarrow \text{grade}(\text{St}, \text{Course}, \text{Gr}_1) \land \text{grade}(\text{St}, \text{Course}, \text{Gr}_2) \land \text{Gr}_1 \neq \text{Gr}_2.
\]

When false is derived, HOW can be used to debug the KB.