Integrity Constraints

- 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

  $false \leftarrow a_1 \wedge \ldots \wedge a_k$

  where the a_i are atoms and *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 α, written $\neg\alpha$ is a formula that
  - is true in interpretation I if α is false in I, and
  - is false in interpretation I if α is true in I.
- Example:

$$KB = \left\{ \begin{array}{l} \mathit{false} \leftarrow a \wedge b. \\ a \leftarrow c. \\ b \leftarrow c. \end{array} \right\} \qquad KB \models \neg c.$$

Disjunctive Conclusions

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

$$KB = \left\{ \begin{array}{l} \mathit{false} \leftarrow a \wedge b. \\ a \leftarrow c. \\ b \leftarrow d. \end{array} \right\} \qquad KB \models \neg c \vee \neg d.$$

- It is always possible to find a model for a set of definite clauses.
- A set of Horn clauses can be unsatisfiable.
- The top-down and the bottom-up proof procedures can be used to prove inconsistency, by using false as the query: a Horn clause knowledge base is inconsistent if and only if false can be derived.

Reasoning from contradictions

- For many activities it is useful to know that some combination of assumptions is incompatible. For example:
  - it is useful in planning to know that some combination of actions an agent is trying to do is impossible;
  - it is useful in design to know that some combination of components cannot work together.
- In a diagnostic application it is useful to be able to prove that some components working normally is inconsistent with the observations of the system.
  - Consider a system that has a description of how it is supposed to work and some observations.
  - If the system does not work according to its specification, a diagnostic agent must identify which components could be faulty.

Questions and Answers in Horn KBs

- An assumable is an atom that can be assumed in a proof by contradiction. A proof by contradiction derives a disjunction of the negation of the assumables.
- With a Horn KB and explicit assumables, if the system can prove a contradiction from some assumptions, it can extract combinations of assumptions that cannot all be true.
- 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.

Conflict Example

Example: If $\{c, d, e, f, g, h\}$ are the assumables

$$KB = \left\{ \begin{array}{l} \mathit{false} \leftarrow a \land b. \\ a \leftarrow c. \\ b \leftarrow d. \\ b \leftarrow e. \end{array} \right\}$$

- $\{c, d\}$ is a conflict
- $\{c, e\}$ is a conflict
- $\{c, d, e, h\}$ is a conflict

Consistency-based diagnosis

- Making assumptions about what is working normally, and deriving what components could be abnormal, is the basis of consistency-based diagnosis.
  - Suppose a fault is something that is wrong with a system.
  - The aim of consistency-based diagnosis is to determine the possible faults based on a model of the system and observations of the system.
  - By making the absence of faults assumable, conflicts can be used to prove what is wrong with the system.

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:

  $false \leftarrow dark\_l_1 \,\&\, lit\_l_1.$

  $false \leftarrow dark\_l_2 \,\&\, lit\_l_2.$

  $false \leftarrow dead\_p_1 \,\&\, live\_p_2.$

- Assume the individual components are working correctly:

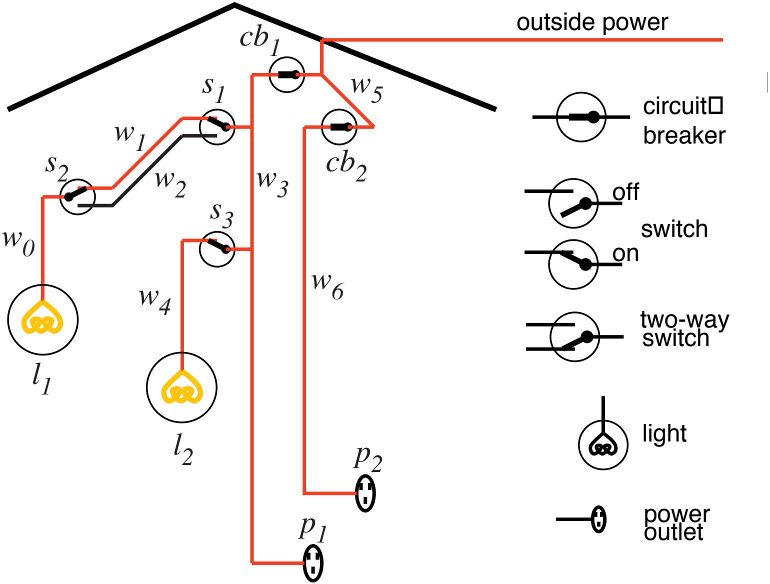
  $assumable\ ok\_l_1.$

  $assumable\ ok\_s_2.$

  ...

- Suppose switches s_1, s_2, and s_3 are all up:
  $up\_s_1.\ up\_s_2.\ up\_s_3.$

Electrical Environment



Representing the Electrical Environment

$light\_l_1.$

$light\_l_2.$

$up\_s_1.$

$up\_s_2.$

$up\_s_3.$

$live\_outside.$

$lit\_l_1 \leftarrow live\_w_0 \wedge ok\_l_1.$

$live\_w_0 \leftarrow live\_w_1 \wedge up\_s_2 \wedge ok\_s_2.$

$live\_w_0 \leftarrow live\_w_2 \wedge down\_s_2 \wedge ok\_s_2.$

$live\_w_1 \leftarrow live\_w_3 \wedge up\_s_1 \wedge ok\_s_1.$

$live\_w_2 \leftarrow live\_w_3 \wedge down\_s_1 \wedge ok\_s_1.$

$lit\_l_2 \leftarrow live\_w_4 \wedge ok\_l_2.$

$live\_w_4 \leftarrow live\_w_3 \wedge up\_s_3 \wedge ok\_s_3.$

$live\_p_1 \leftarrow live\_w_3.$

$live\_w_3 \leftarrow live\_w_5 \wedge ok\_cb_1.$

$live\_p_2 \leftarrow live\_w_6.$

$live\_w_6 \leftarrow live\_w_5 \wedge ok\_cb_2.$

$live\_w_5 \leftarrow live\_outside.$

- If the user has observed l_1 and l_2 are both dark:

  $dark\_l_1.\ dark\_l_2.$

- There are two minimal conflicts:

  $\{ok\_cb_1, ok\_s_1, ok\_s_2, ok\_l_1\}$ and

  $\{ok\_cb_1, ok\_s_3, ok\_l_2\}$.

- You can derive:

  $\neg ok\_cb_1 \vee \neg ok\_s_1 \vee \neg ok\_s_2 \vee \neg ok\_l_1$

  $\neg ok\_cb_1 \vee \neg ok\_s_3 \vee \neg ok\_l_2.$

- Either cb_1 is broken or there is one of six double faults.

Diagnoses

- Given the set of all conflicts, a user can determine what may be wrong with the system being diagnosed.
- Some of the questions that a user may want to know are whether all of the conflicts could be accounted for a by a single fault or a pair of 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: $\{ok\_cb_1\}$, $\{ok\_s_1, ok\_s_3\}$, $\{ok\_s_1, ok\_l_2\}$, $\{ok\_s_2, ok\_s_3\}$,...

Recall: top-down consequence finding

To solve the query $?q_1 \wedge \ldots \wedge q_k$:

$ac :=$ "$yes \leftarrow q_1 \wedge \ldots \wedge q_k$"
repeat
  select atom a_i from the body of ac;
  choose clause C from KB with a_i as head;
  replace a_i in the body of ac by the body of C
until ac is an answer.

Implementing conflict finding: top down

- Query is *false*.
- Don't select an atom that is assumable.
- Stop when all of the atoms in the body of the generalised query are assumable:
    - this is a conflict

Example

$false \leftarrow a.$

$a \leftarrow b \,\&\, c.$

$b \leftarrow d.$

$b \leftarrow e.$

$c \leftarrow f.$

$c \leftarrow g.$

$e \leftarrow h \,\&\, w.$

$e \leftarrow g.$

$w \leftarrow f.$

assumable $d, f, g, h.$

Example

light-l1. light-l2. live-outside.
live-l1 ← live-w0.
live-w0 ← live-w1, up-s2, ok-s2.
live-w0 ← live-w2, down-s2, ok-s2.
live-w1 ← live-w3, up-s1, ok-s1.
live-w2 ← live-w3, down-s1, ok-s1.
live-l2 ← live-w4.
live-w4 ← live-w3, up-s3, ok-s3.
live-p1 ← live-w3.
live-w3 ← live-w5, ok-cb1.
live-p2 ← live-w6.
live-w6 ← live-w5, ok-cb2.
live-w5 ← live-outside.
lit-l1 ← light-l1, live-l1, ok-l1.
lit-l2 ← light-l2, live-l2, ok-l2.
false ← dark-l1, lit-l1.
false ← dark-l2, lit-l2.
assumable ok-cb1,ok-cb2,ok-s1,· · · .

{false}
{dark-l1,lit-l1}
{lit-l1}
{light-l1,live-l1, ok-l1}
{live-l1, ok-l1}
{live-w0, ok-l1}
{live-w1, up-s2, ok-s2, ok-l1}
{live-w3, up-s1, ok-s1, up-s2, ok-s2, ok-l1}
{live-w5, ok-cb1, up-s1, ok-s1, up-s2, ok-s2, ok-l1}
{live-outside, ok-cb1, up-s1, ok-s1, up-s2, ok-s2, ok-l1}
{ok-cb1, up-s1, ok-s1, up-s2, ok-s2, ok-l1}
{ok-cb1, ok-s1, up-s2, ok-s2, ok-l1}
{ok-cb1, ok-s1, ok-s2, ok-l1}.

Bottom-up Conflict Finding

- 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 $h \leftarrow b_1 \wedge \ldots \wedge b_m$ 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 $\langle \mathit{false}, A_1 \rangle$ 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

$C := \{\langle a, \{a\}\rangle : a$ is assumable $\}$;
repeat
  select clause "$h \leftarrow b_1 \wedge \ldots \wedge b_m$" in T such that
    $\langle b_i, A_i\rangle \in C$ for all i and
    there is no $\langle h, A'\rangle \in C$ or $\langle \mathit{false}, A'\rangle \in C$
      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

Example

light-l1. light-l2. live-outside.
live-l1 $\leftarrow$ live-w0.
live-w0 $\leftarrow$ live-w1, up-s2, ok-s2.
live-w0 $\leftarrow$ live-w2, down-s2, ok-s2.
live-w1 $\leftarrow$ live-w3, up-s1, ok-s1.
live-w2 $\leftarrow$ live-w3, down-s1, ok-s1.
live-l2 $\leftarrow$ live-w4.
live-w4 $\leftarrow$ live-w3, up-s3, ok-s3.
live-p1 $\leftarrow$ live-w3.
live-w3 $\leftarrow$ live-w5, ok-cb1.
live-p2 $\leftarrow$ live-w6.
live-w6 $\leftarrow$ live-w5, ok-cb2.
live-w5 $\leftarrow$ live-outside.
lit-l1 $\leftarrow$ light-l1, live-l1, ok-l1.
lit-l2 $\leftarrow$ light-l2, live-l2, ok-l2.
false $\leftarrow$ dark-l1, lit-l1.
false $\leftarrow$ dark-l2, lit-l2.
assumable ok-cb1,ok-cb2,ok-s1,$\cdots$.

$\{\langle$ok-l1,$\{$ok-l1$\}\rangle$,$\langle$ok-l2,$\{$ok-l2$\}\rangle$, $\cdots\}$.

$\langle$live-outside,$\{\}\rangle$
$\langle$connected-to-w5,outside,$\{\}\rangle$
$\langle$live-w5,$\{\}\rangle$
$\langle$connected-to-w3,w5,$\{$ok-cb1$\}\rangle$
$\langle$live-w3,$\{$ok-cb1$\}\rangle$
$\langle$up-s3,$\{\}\rangle$
$\langle$connected-to-w4,w3,$\{$ok-s3$\}\rangle$
$\langle$live-w4,$\{$ok-cb1,ok-s3$\}\rangle$
$\langle$connected-to-l2,w4,$\{\}\rangle$
$\langle$live-l2,$\{$ok-cb1,ok-s3$\}\rangle$
$\langle$light-l2,$\{\}\rangle$
$\langle$lit-l2,$\{$ok-cb1,ok-s3,ok-l2$\}\rangle$
$\langle$dark-l2,$\{\}\rangle$
$\langle$false,$\{$ok-cb1,ok-s3,ok-l2$\}\rangle$.

Thus, the knowledge base entails:
$\neg$ok-cb1 $\wedge$ $\neg$ok-s3 $\wedge$ $\neg$ok-l2.