Logical Inference

Logic

- We want to tell our computers facts that are true of the world.
  "It is raining."
- Some of these facts specify how one thing is related to another.
  "It is raining implies it is wet."
- We want our computers to be able to infer what else must be true of the world.
  "It is wet."
- A *logic* is a system for inference from facts.

Propositional Logic

Syntax

- A *proposition* is something that is true or false.
- An *atomic proposition* or *atom* consists of a single *symbol*. (≈ boolean variable)
- A *compound proposition* is constructed from simpler propositions p and q using *logical operators* (≈ boolean expression):
  - $\neg p$ (read "not p")–*negation*
  - $p \wedge q$ (read "p and q")–*conjunction*
  - $p \vee q$ (read "p or q")–*disjunction*
  - $p \rightarrow q$ (read "p implies q")–*implication*
  - $q \leftarrow p$ (read "q if p")–*implication*
  - $p \leftrightarrow q$ (read "p iff q")–*equivalence*

[Note: I prefer using → to ←.]

Informal Semantics

Semantics maps between symbols and the world.

- Begin with a task domain.
- Choose symbols in the computer to denote propositions.
  Symbol ≈ variable name
- Tell the system knowledge about the domain.
  Knowledge ≈ code and inputs
- Ask the system true/false questions.
  Ask questions ≈ run a function
- The system should answer true, false or unknown as appropriate.
- You can interpret the answer because you know the meaning of the symbols.

Informal Semantics Example

- In computer: $sw\_up \land power \land unlit\_l1 \rightarrow l1\_broken$
- In user's mind: sw_up = switch is up,
  $power$ = there is power in,
  $unlit\_l1$ = light #1 isn't lit,
  $l1\_broken$ = light #1 is broken
- The computer doesn't know the meaning of the symbols.
- The user can interpret the symbols using their meaning.

Formal Semantics 1

- An *interpretation* I maps atoms to true or false.
- Based on how logical operators work, an interpretation maps each proposition to a truth value.
- Propositions may have different truth values in different interpretations.

p	q	$\neg p$	$p \land q$	$p \lor q$	$p \rightarrow q$	$q \leftarrow p$	$p \leftrightarrow q$
true	true	false	true	true	true	true	true
true	false	false	false	true	false	false	false
false	true	true	false	true	true	true	false
false	false	true	false	false	true	true	true

Formal Semantics 2

- A *knowledge base* is a set of propositions that the agent is given as being true.
- A *model* of knowledge base is an interpretation in which all the propositions in the knowledge base are true.
- If KB is a knowledge base and p is a proposition, KB *entails* p (written $KB \models p$) if p is true in every model of KB.
- $KB \models p$ means that no interpretation exists in which KB is true and p is false.
- If $KB \models p$ we also say p logically follows from KB, or p is a logical consequence of KB.

Examples

Simple Example

$KB = \{p \rightarrow q, \quad p, \quad s \rightarrow r\}$

	p	q	r	s	model?
I_1	true	true	true	true	
I_2	false	false	false	false	
I_3	true	true	false	false	
I_4	true	true	true	false	
I_5	true	true	false	true	

Which of p, q, r, s are entailed by KB?

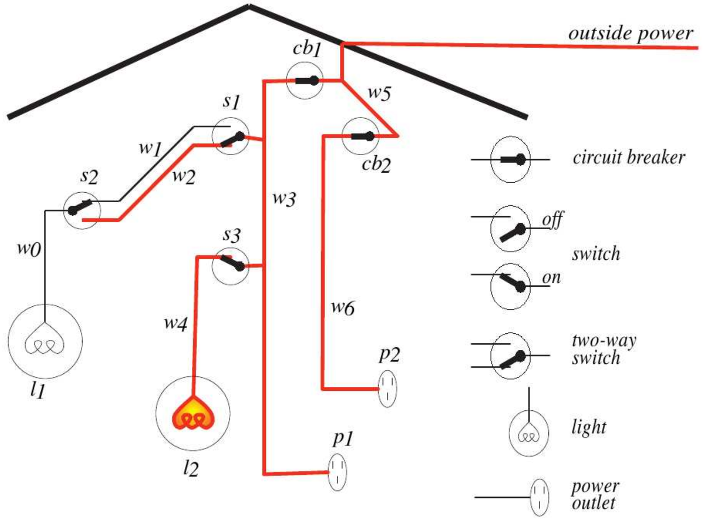
Simple Example

$KB = \{p \rightarrow q, \quad p, \quad s \rightarrow r\}$

	p	q	r	s	model of KB?
I_1	true	true	true	true	yes
I_2	false	false	false	false	no
I_3	true	true	false	false	yes
I_4	true	true	true	false	yes
I_5	true	true	false	true	no

Which of p, q, r, s are entailed by KB?
p and q

Electrical Environment



Representation

$light\_l1$
$light\_l2$
$down\_s1$
up_s2
up_s3
ok_l1
ok_l2
ok_cb1
ok_cb2
$live\_outside$

$live\_w0 \land ok\_l1 \rightarrow lit\_l1$
$live\_w1 \land up\_s2 \rightarrow live\_w0$
$live\_w2 \land down\_s2 \rightarrow live\_w0$
$live\_w3 \land up\_s1 \rightarrow live\_w1$
$live\_w3 \land down\_s1 \rightarrow live\_w2$
$live\_w4 \land ok\_l2 \rightarrow lit\_l2$
$live\_w3 \land up\_s3 \rightarrow live\_w4$
$live\_w3 \rightarrow live\_p1$
$live\_w5 \land ok\_cb1 \rightarrow live\_w3$
$live\_w6 \rightarrow live\_p2$
$live\_w5 \land ok\_cb2 \rightarrow live\_w6$
$live\_outside \rightarrow live\_w5$

Proof Procedures

Proofs

- A *proof* is a derivation that a proposition logically follows from a knowledge base.
- Given a proof procedure, $KB \vdash p$ means p can be *derived* or *proved* from KB.
- Recall $KB \models p$ means KB entails p, that p is true in all models of KB.
- A proof procedure is *sound* if $KB \vdash p$ only if $KB \models p$. Anything that is proved is also entailed.
- A proof procedure is *complete* if $KB \models p$ then also $KB \vdash p$. Everything that is entailed can be proved.

Brute Force Inference

- Enumerate all interpretations.
- Determine which interpretations are models of the KB.
- Determine which atoms (and any other propositions of interest) are true in all models (or false in all models).
- This is $\Omega(2^n)$ where n is the number of atoms.

CSP Inference

- Set up KB as a CSP. Each atom is a variable with two possible values. Each proposition in the KB is a constraint.
- Solutions of CSP = models of KB.
- Run arc consistency/domain splitting.
- Don't stop after finding one CSP solution (KB model). Find them all.
- Determine which atoms are true in all models (or false in all models).
- This is still potentially exponential, but more efficient than brute force.
- See Section 4.6.1.

Definite Clause Inference

- Suppose all propositions in KB are *definite clauses*, either:
  - an atom (e.g., an observation), or
  - of the form $p \rightarrow q$, where p and q are atoms (e.g., a rule about the behavior of the world)
  - of the form $p_1 \wedge \ldots \wedge p_k \rightarrow q$, where q and each p_i are atoms
- Running CSP inference is efficient (linear in the length of the KB).
- See Section 5.2.

Definite Clause Example

$KB = \{a,\ \ b,\ \ a \rightarrow c,\ \ b \wedge c \rightarrow d,\ \ d \wedge e \rightarrow f\}$

Know a and b.

Derive c from a and $a \rightarrow c$.

Derive d from b and c and $b \wedge c \rightarrow d$

Cannot derive e or f.

Proof by Contradiction

- Suppose we want to determine if $KB \models p$.
- Let $KB' = KB \cup \{\neg p\}$
- Determine that no model exists for KB'.
- Conclude that $KB \models p$.
- Should probably show that KB has at least one model.

Inference Rules

- *Modus ponens* is an inference rule. If p is true, and if $p \rightarrow q$ is true, then q is true.
- That is, if $KB \models p$ and $KB \models p \rightarrow q$, then $KB \models q$.
- *Resolution inference rule* (really, two rules)
  - If $KB \models p \vee q$ and $KB \models \neg p$, then $KB \models q$.
  - If $KB \models p \vee q$ and $KB \models \neg p \vee r$, then $KB \models q \vee r$.
- Remember p and q and r can be any propositions, not just atoms.

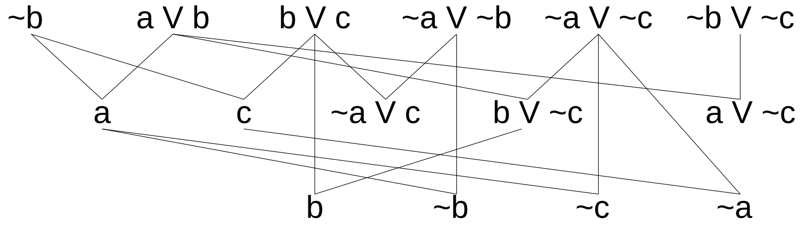
Resolution Theorem Proving

- *Resolution theorem proving* is a sound and complete inference procedure for propositional logic.
- Transform the KB to *conjunctive normal form*, meaning each propositions in the KB is of the form l or $l_1 \vee \ldots \vee l_k$, where each l_i is a *literal*, an atom or the negation of an atom.
- To show $KB \models p$, let $KB' = KB \cup \{\neg p\}$, and ensure KB' is in CNF.
- Proof is by deriving a contradiction, derive both a and $\neg a$ for some atom a.
- Worst-case exponential-time. Lots of approaches to reduce the exponential.

Resolution Example

$KB = \{a \vee b,\ \ b \vee c,\ \ \neg a \vee \neg b,\ \ \neg a \vee \neg c,\ \ \neg b \vee \neg c\}$

To prove $KB \models b$, add $\neg b$ and prove a contradiction using the resolution inference rule.



~b a V b b V c ~a V ~b ~a V ~c ~b V ~c

a c ~a V c b V ~c a V ~c

b ~b ~c ~a