Logic: Domain Modeling /Proofs + Top-Down Proofs

CPSC 322 Lecture 21

Lecture Overview

- Recap
- Using Logic to Model a Domain (Electrical System)
- Reasoning/Proofs (in the Electrical Domain)
- Top-Down Proof Procedure

Soundness & completeness of proof procedures

- A proof procedure X is sound …
- A proof procedure X is complete …
- BottomUp for PDCL is …
- We proved this in general even for domains represented by **thousands of propositions** and corresponding **KB with millions of definite clauses** !

Learning Goals for today's class

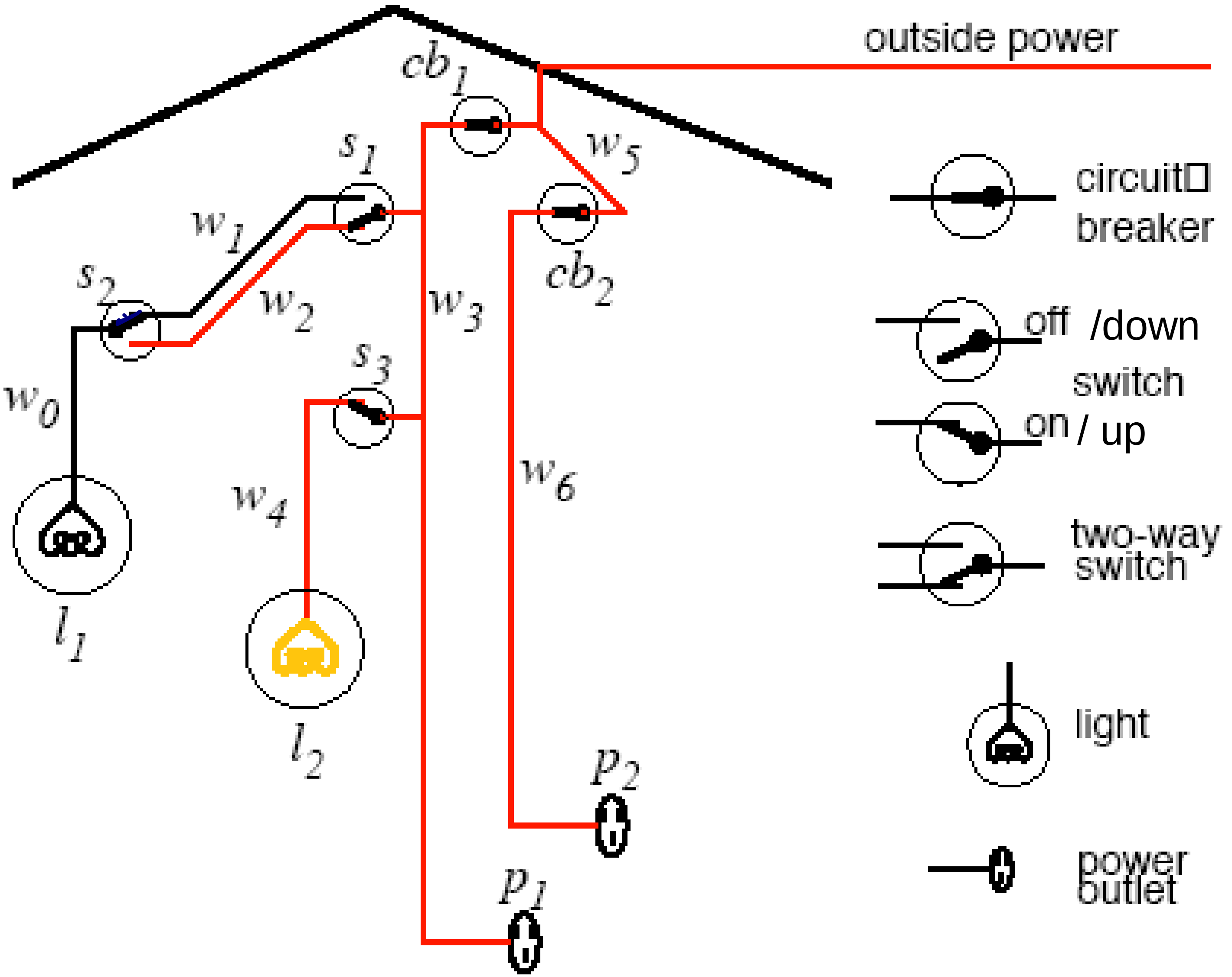
You can:

- Model a relatively simple domain with propositional definite clause logic (PDCL)
- Trace query derivation using SLD resolution rule of inference

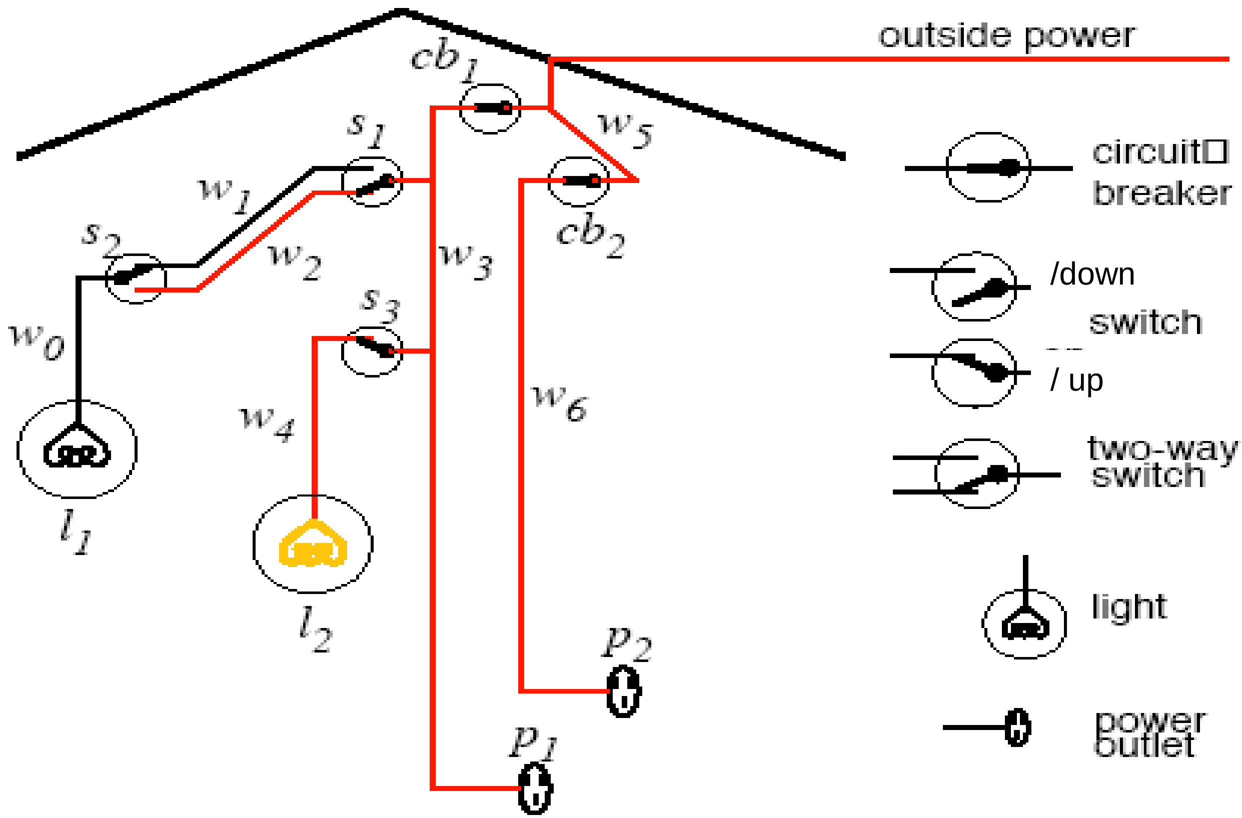
Lecture Overview

- Recap
- Using PDCL Logic to Model a Domain (Electrical System)
- Reasoning/Proofs (in the Electrical Domain)
- Top-Down Proof Procedure

Electrical Environment



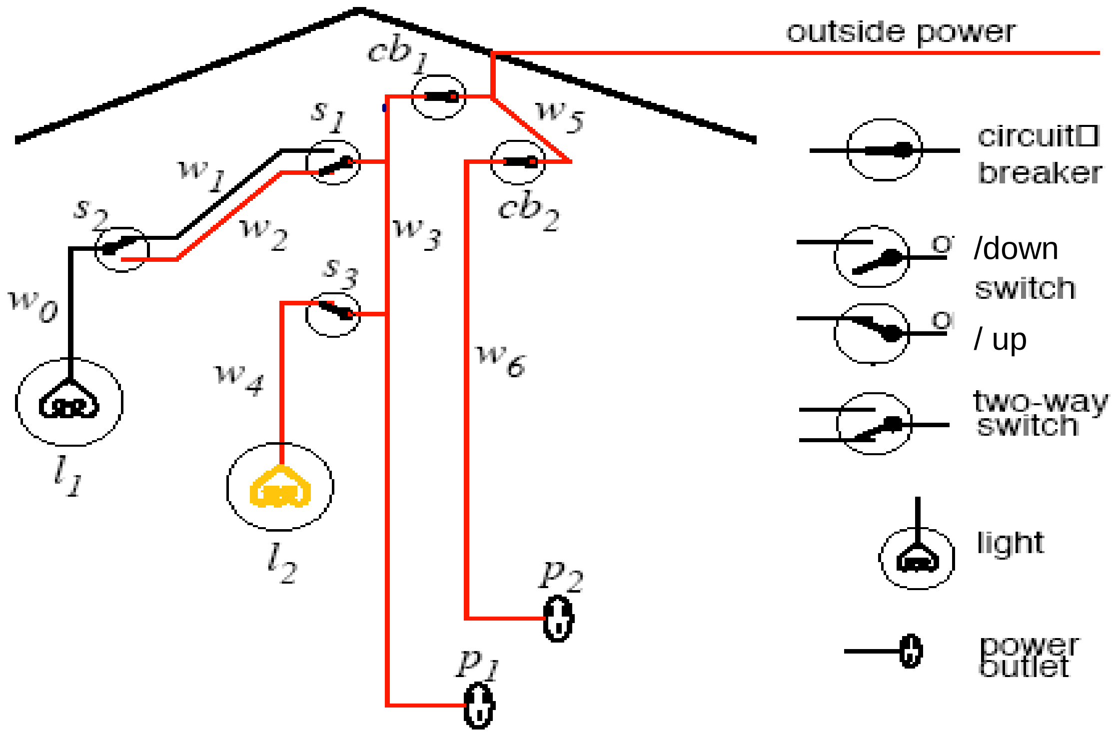
Let's define relevant propositions



- For each wire *w*
- For each circuit breaker *cb*
- For each switch *s*
- For each light *l*
- For each outlet *p*

How many interpretations?

Let’s now tell system knowledge about how the domain works



live_l₁ ←

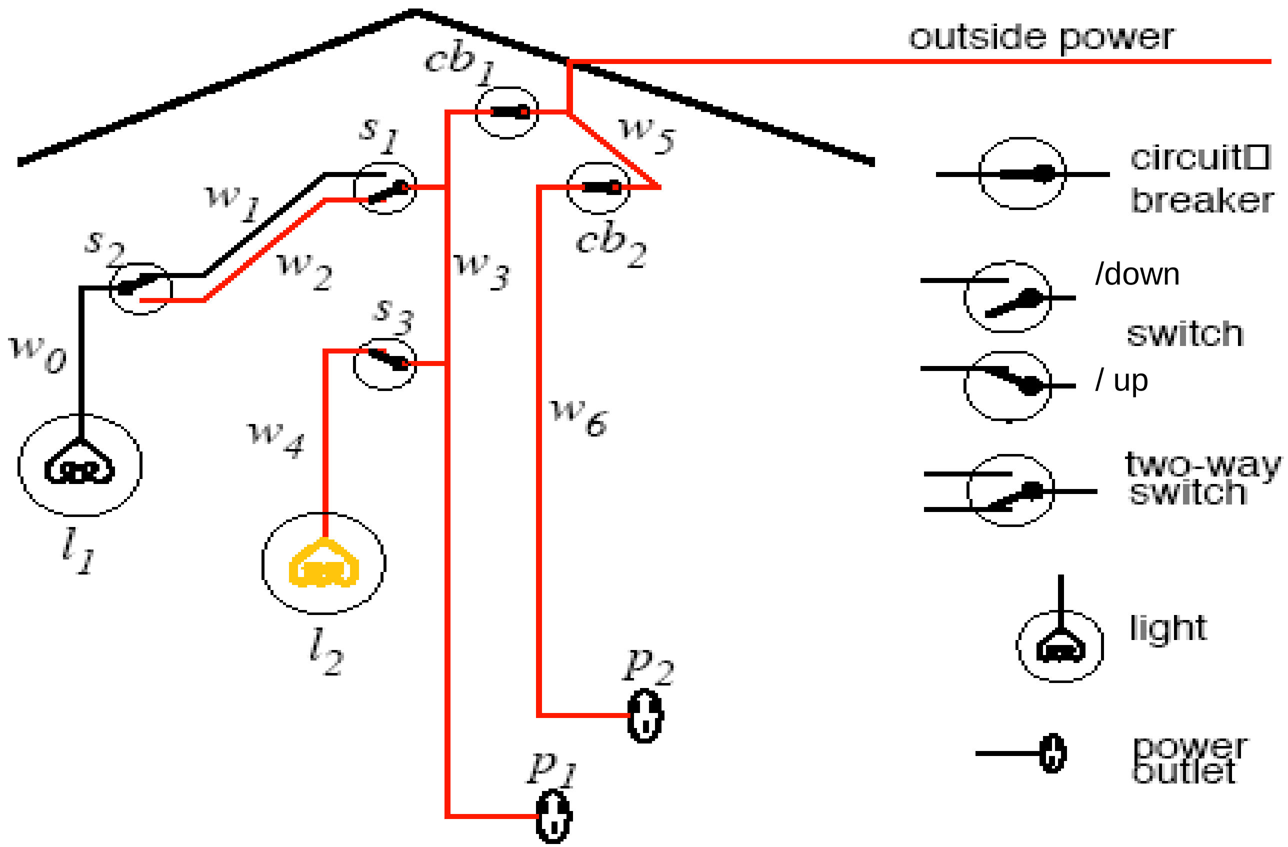
$live\_l_1 \leftarrow$

$live\_w_0 \leftarrow$

$live\_w_0 \leftarrow$

$live\_w_1 \leftarrow$

More on how the domain works….



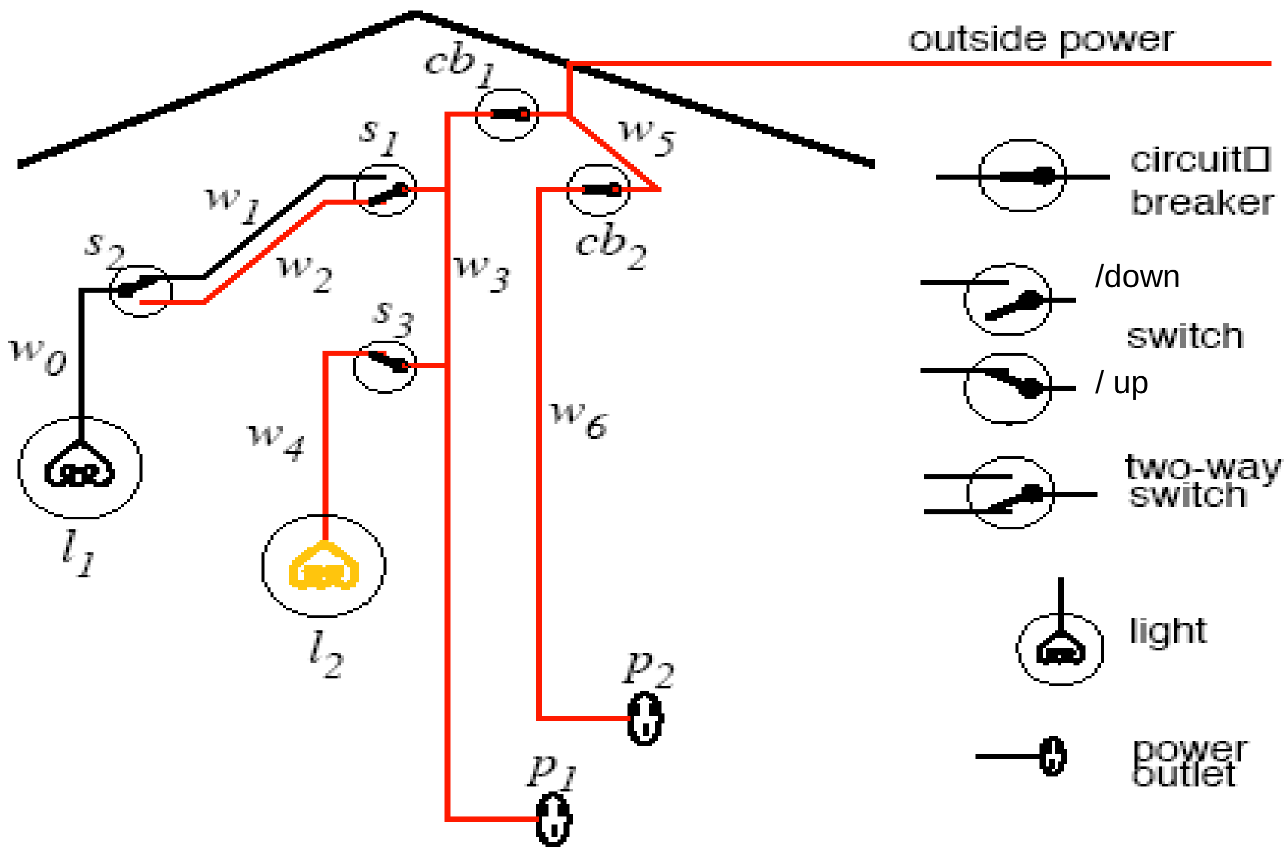
$live\_w_2 \leftarrow live\_w_3 \wedge down\_s_1.$

$live\_l_2 \leftarrow live\_w_4.$

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

$live\_p_1 \leftarrow live\_w_3.$

More on how the domain works….



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

$live\_p_2 \leftarrow live\_w_6.$

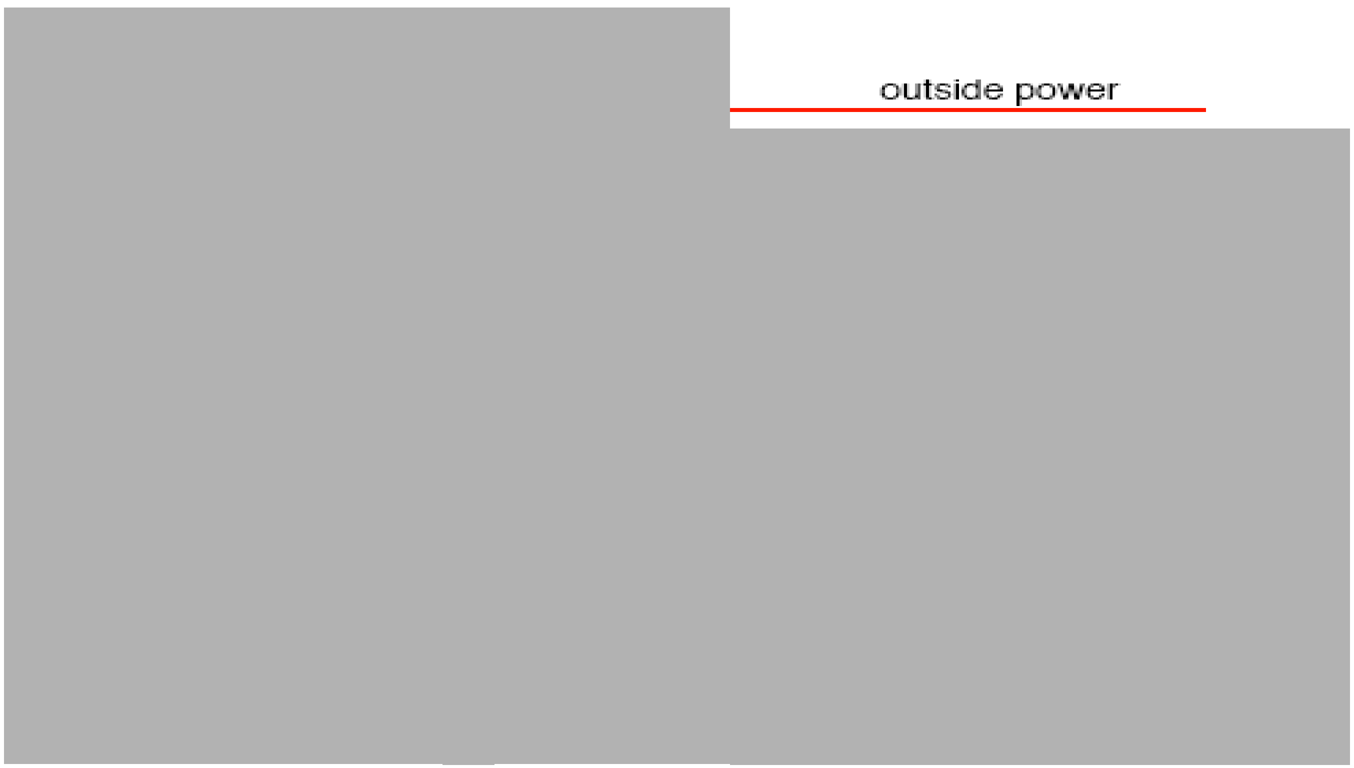
$live\_w_6 \leftarrow live\_w_5 \wedge ok\_cb_2.$

$live\_w_5 \leftarrow live\_outside.$

What else we may know about this domain?

- That some simple propositions are true

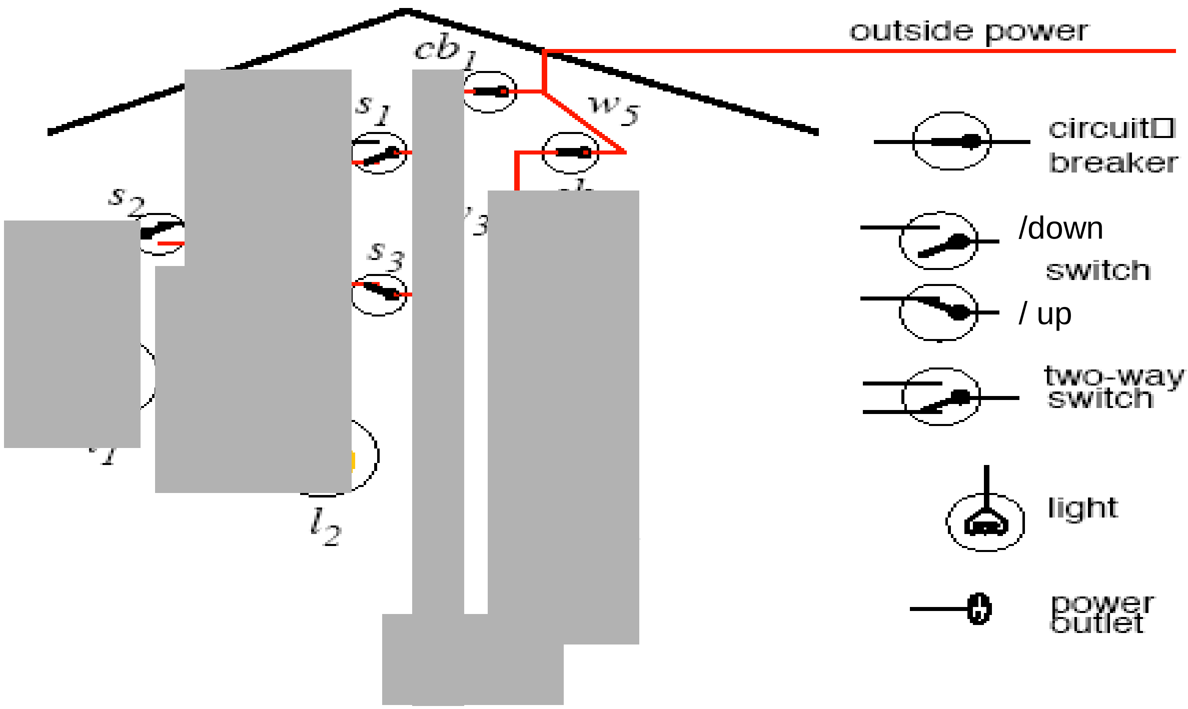
  live_outside.



What else we may know about this domain?

- That some additional simple propositions are true

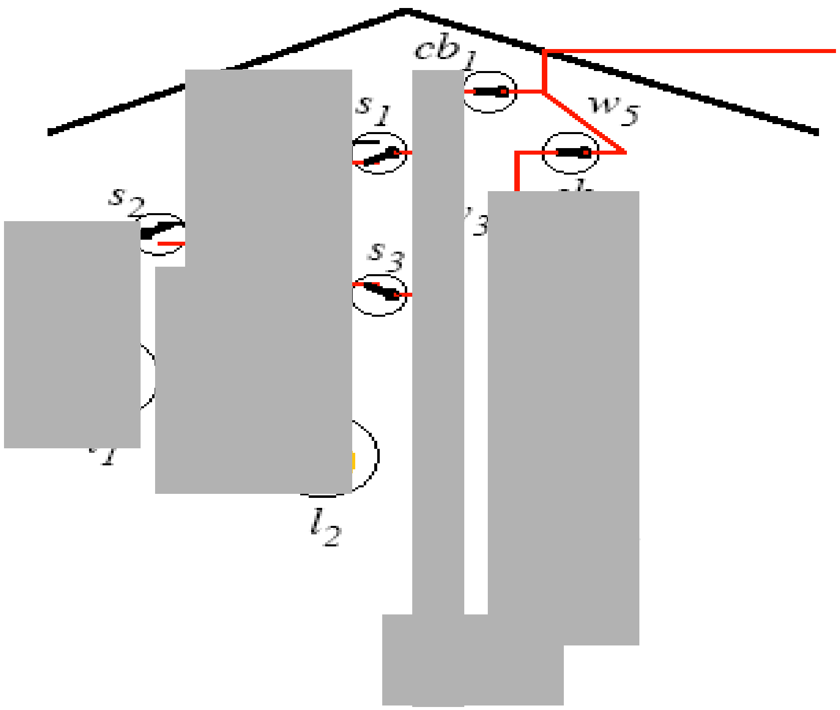
$down\_s_1$. up_s_2. up_s_3. ok_cb_1. ok_cb_2.
live_outside.



All our knowledge…..

$down\_s_1.$
$up\_s_2.$
$up\_s_3.$
$ok\_cb_1.$
$ok\_cb_2.$
$live\_outside.$

$live\_l_1 \leftarrow live\_w_0.$
$live\_w_0 \leftarrow live\_w_1 \wedge up\_s_2.$
$live\_w_0 \leftarrow live\_w_2 \wedge down\_s_2.$
$live\_w_1 \leftarrow live\_w_3 \wedge up\_s_1.$
$live\_w_2 \leftarrow live\_w_3 \wedge down\_s_1.$
$live\_l_2 \leftarrow live\_w_4.$
$live\_w_4 \leftarrow live\_w_3 \wedge up\_s_3.$
$live\_p_1 \leftarrow live\_w_3.$
$live\_w_3 \leftarrow live\_w_5 \wedge ok\_cb_1.$
$live\_p_2 \leftarrow live\_w_6.$
$live\_w_6 \leftarrow live\_w_5 \wedge ok\_cb_2.$
$live\_w_5 \leftarrow live\_outside.$



Lecture Overview

- Recap
- Using Logic to Model a Domain (Electrical System)
- Reasoning/Proofs (in the Electrical Domain)
- Top-Down Proof Procedure

What Semantics is telling us

- Our KB (all we know about this domain) is going to be true only in a **subset** of all possible interpretations

- What is logically entailed by our KB are all the propositions that are true in all those models

- This is what we should be able to derive given a sound and complete proof procedure

If we apply the bottom-up (BU) proof procedure

$down\_s_1.$
$up\_s_2.$
$up\_s_3.$
$ok\_cb_1.$
$ok\_cb_2.$
$live\_outside.$

$live\_l_1 \leftarrow live\_w_0$
$live\_w_0 \leftarrow live\_w_1 \wedge up\_s_2.$
$live\_w_0 \leftarrow live\_w_2 \wedge down\_s_2.$
$live\_w_1 \leftarrow live\_w_3 \wedge up\_s_1.$
$live\_w_2 \leftarrow live\_w_3 \wedge down\_s_1.$
$live\_l_2 \leftarrow live\_w_4.$
$live\_w_4 \leftarrow live\_w_3 \wedge up\_s_3.$
$live\_p_1 \leftarrow live\_w_3.$
$live\_w_3 \leftarrow live\_w_5 \wedge ok\_cb_1.$
$live\_p_2 \leftarrow live\_w_6.$
$live\_w_6 \leftarrow live\_w_5 \wedge ok\_cb_2.$
$live\_w_5 \leftarrow live\_outside.$

$live\_l_2$?
$live\_l_1$?

iclicker.

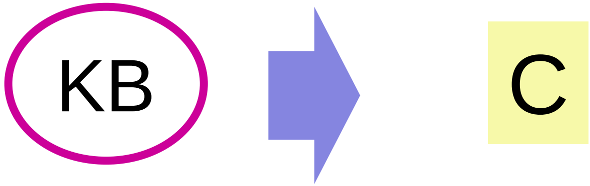
A. Both proved
B. Only live_l2 proved
C. Only live_l1 proved
D. Neither proved
E. The cake is a lie

Lecture Overview

- Recap
- Using Logic to Model a Domain (Electrical System)
- Reasoning/Proofs (in the Electrical Domain)
- Top-Down Proof Procedure

Bottom-up vs. Top-down

Bottom-up



G is proved if $G \subseteq C$

When does BU look at the query G ?

A. In every loop iteration
B. Never
C. Only at the end
D. Only at the beginning
E. Only if G has a video of a cute cat or puppy

Bottom-up vs. Top-down

- **Key Idea of top-down:** search backward from a query G to determine if it can be derived from *KB*.

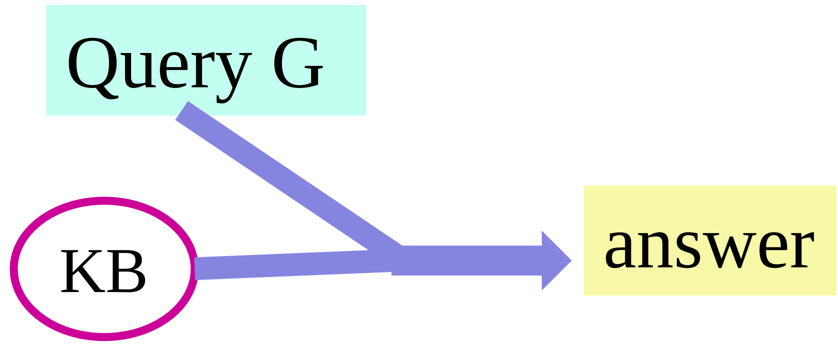
Bottom-up

G is proved if $G \subseteq C$

When does BU look at the query G?

- At the end

Top-down



TD performs a backward **search** starting at G

Top-down Proof Procedure: Elements

Notation: An answer clause is of the form:

$yes \leftarrow a_1 \land a_2 \land \ldots \land a_m$

Express query as an answer clause (e.g., if query

$= a_1 \land a_2 \land \ldots \land a_m$)

$$yes \leftarrow a_1 \land a_2 \land \ldots \land a_m$$

Rule of inference (called SLD Resolution)

Given an answer clause of the form:

$$yes \leftarrow a_1 \land a_2 \land \ldots \land a_m$$

and the KB clause:

$$a_i \leftarrow b_1 \land b_2 \land \ldots \land b_p$$

You can generate the answer clause

$yes \leftarrow a_1 \land \ldots \land a_{i-1} \land b_1 \land b_2 \land \ldots \land b_p \land a_{i+1} \land \ldots \land a_m$

Rule of inference: Examples

Rule of inference (called SLD Resolution)

Given an answer clause of the form:

$$yes \leftarrow a_1 \wedge a_2 \wedge \ldots \wedge a_m$$

and the KB clause:

$$a_i \leftarrow b_1 \wedge b_2 \wedge \ldots \wedge b_p$$

You can generate the answer clause

$$yes \leftarrow a_1 \wedge \ldots \wedge a_{i-1} \wedge b_1 \wedge b_2 \wedge \ldots \wedge b_p \wedge a_{i+1} \wedge \ldots \wedge a_m$$

answer clause	KB clause	resulting inference
yes ← b ^ c	b ← k ^ f	yes ← k ^ f ^ c
yes ← e ^ f	e	yes ← f

(successful) Derivations

- An answer is an answer clause with $m = 0$. That is, it is the "empty" answer clause $yes \leftarrow$.

- A (successful) derivation of query "?$q_1 \wedge \ldots \wedge q_k$" from *KB* is a sequence of answer clauses $\gamma_0, \gamma_1, \ldots, \gamma_n$

  such that

  - γ_0 is the answer clause $yes \leftarrow q_1 \wedge \ldots \wedge q_k$
  - γ_i is obtained by resolving γ_{i-1} with a clause in *KB*, and
  - γ_n is an "empty" answer.

- An unsuccessful derivation…..

Example: derivations

$a \leftarrow e \wedge f.$	$a \leftarrow b \wedge c.$	$b \leftarrow k \wedge f.$
$c \leftarrow e.$	$d \leftarrow k.$	$e.$
$f \leftarrow j \wedge e.$	$f \leftarrow c.$	$j \leftarrow c.$

Query: ?a `(two ways)`

$yes \leftarrow a.$

$yes \leftarrow a.$

Example: derivations

$k \leftarrow e.$	$a \leftarrow b \wedge c.$	$b \leftarrow k \wedge f.$
$c \leftarrow e.$	$d \leftarrow k.$	$e.$
$f \leftarrow j \wedge e.$	$f \leftarrow c.$	$j \leftarrow c.$

Query: $b \wedge e$

A. Provable by Top-Down
B. Not provable by Top-Down
C. It depends
D. 42?
E. We will never forgive you

R&R systems we'll cover in this course

		Environment	
Problem		**Deterministic**	**Stochastic**
Static	Constraint Satisfaction	*Variables + Constraints* Search Arc Consistency Local Search	
	Query	*Logics* Search	*Bayesian (Belief) Networks* Variable Elimination
Sequential	Planning	*STRIPS* Search	*Decision Networks* Variable Elimination

Representation
Reasoning Technique

Search for Specific R&R systems

Constraint Satisfaction (Problems):

- State: assignments of values to a subset of the variables
- Successor function: assign values to a “free” variable
- Goal test: set of constraints
- Solution: possible world that satisfies the constraints
- Heuristic function: *none (all solutions at the same distance from start)*

Planning :

- State possible world
- Successor function states resulting from valid actions
- Goal test assignment to subset of vars
- Solution sequence of actions
- Heuristic function empty-delete-list (solve simplified problem)

Logical Inference

- State answer clause
- Successor function states resulting from substituting one atom with all the clauses of which it is the head
- Goal test empty answer clause
- Solution start state
- Heuristic function ….. *(next time)*