Logic: Domain Modeling /Proofs + Top-Down Proofs

Computer Science cpsc322, Lecture 22

(Textbook Chpt 5.2)

Oct, 30, 2013

Department of Computer Science
Undergraduate Events
More details @ https://www.cs.ubc.ca/students/undergrad/life/upcoming-events

Mastering LinkedIn Workshop
Date: Mon., Oct 28
Time: 5:00 pm
Location: Wesbrook 100

Resume Drop-in Editing
Date: Tues., Oct 29
Time: 12:30 – 3:30 pm
Location: ICCS 253

Graduate Recruitment Panel
Date: Wed., Oct 30
Time: 12:30 – 1:30 pm
Location: X836, ICICS/CS

CSSS Meet the Profs Luncheon
Date: Thurs., Oct 31
Time: 12:30 – 2 pm
Location: X836, ICICS/CS

E-Portfolio Info Session & Talk by Eric Diep, Co-Founder, A Thinking Ape
Date: Tues., Nov 5
Time: 5:15 – 6:45 pm
Location: DMP 110

Programming Interview Resources Drop-in Clinic
Date: Wed., Nov 6
Time: 12:30 – 2 pm
Location: ICCS 202

Speed Mentoring Dinner
Date: Wed., Nov 6
Time: 5:45 – 7:15 pm
Location: ICCS X860

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 …

$KB \vdash_X G \Rightarrow KB \models G$

- A proof procedure X is complete….

$KB \models G \Rightarrow KB \vdash_X G$

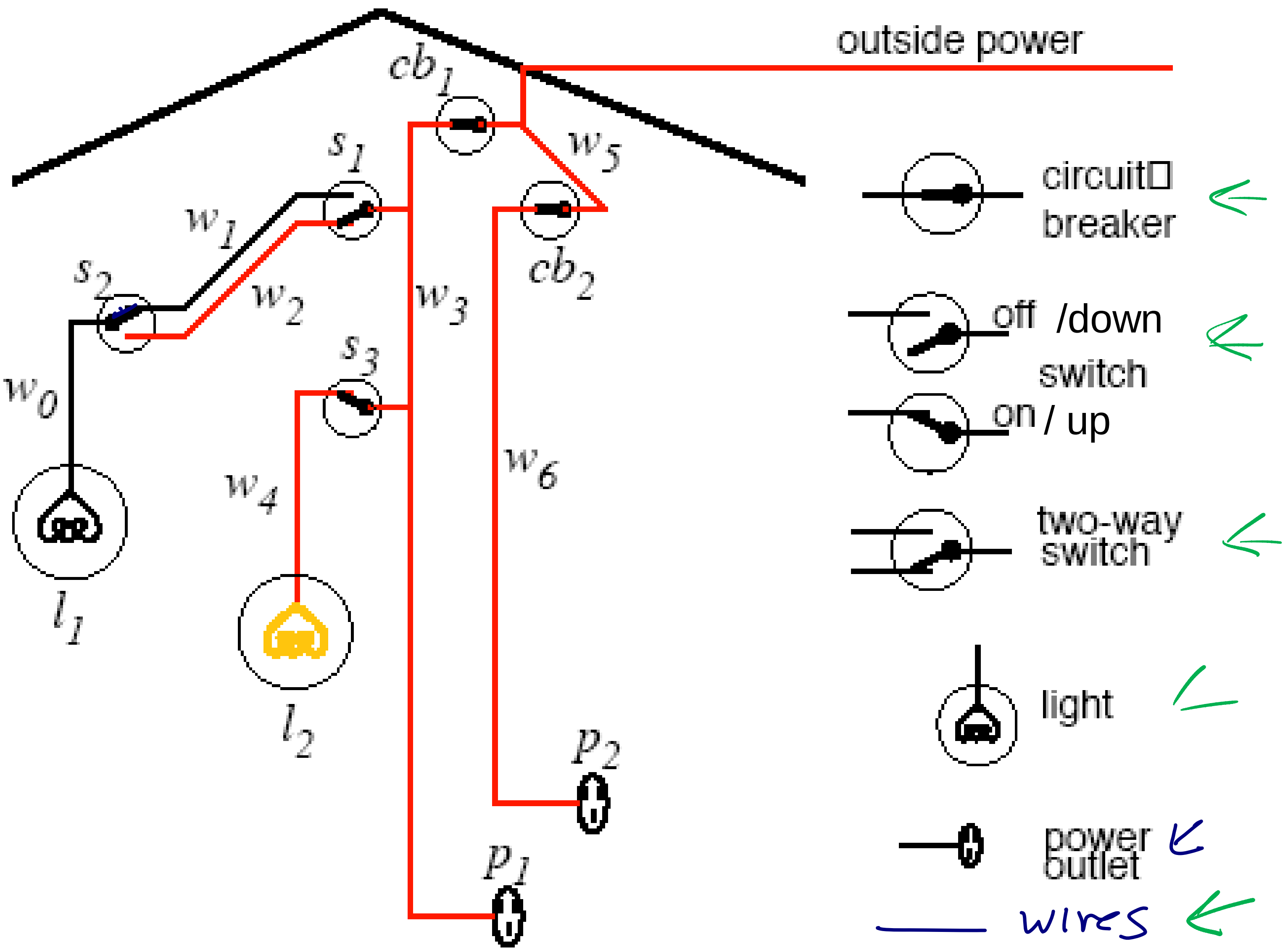
- BottomUp for PDCL is ← sound & complete

- We proved this in general even for domains represented by **thousands of propositions** and corresponding **KB with millions of definite clauses** !

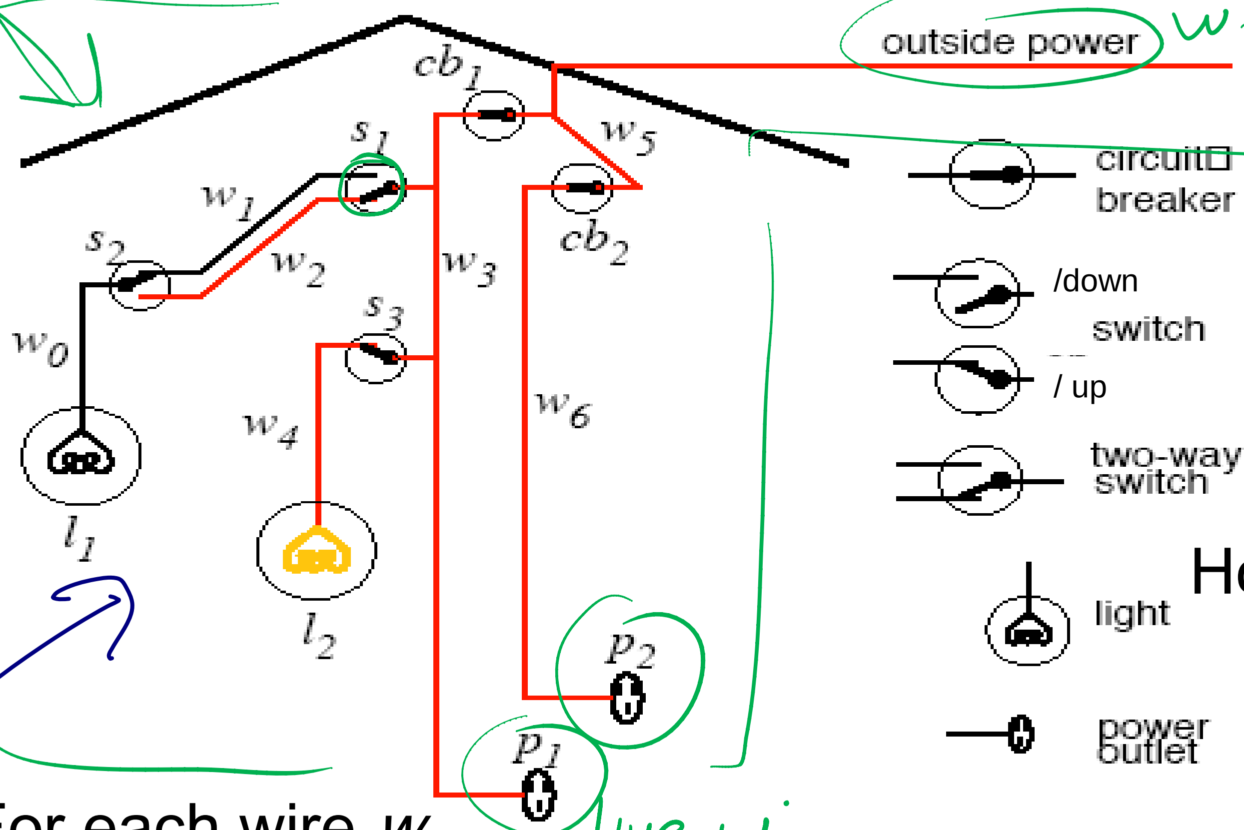
Lecture Overview

- 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



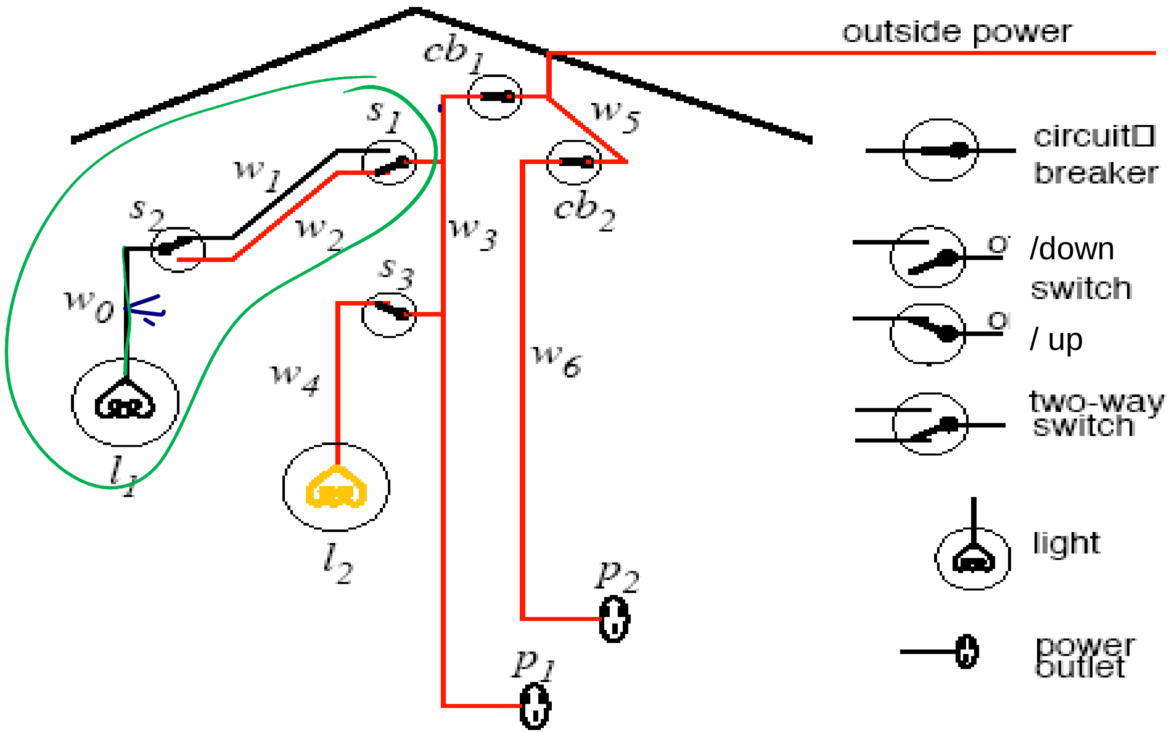
How many interpretations? 2^{19} $\sim 5 * 10^5$

- For each wire *w* live_wi
- For each circuit breaker *cb* ok_cbi
- For each switch *s* up_si, down_si
- For each light *ℓ* live_li
- For each outlet *p* live_p

- 7
- 2
- 3 x 2
- 2
- 2

19

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



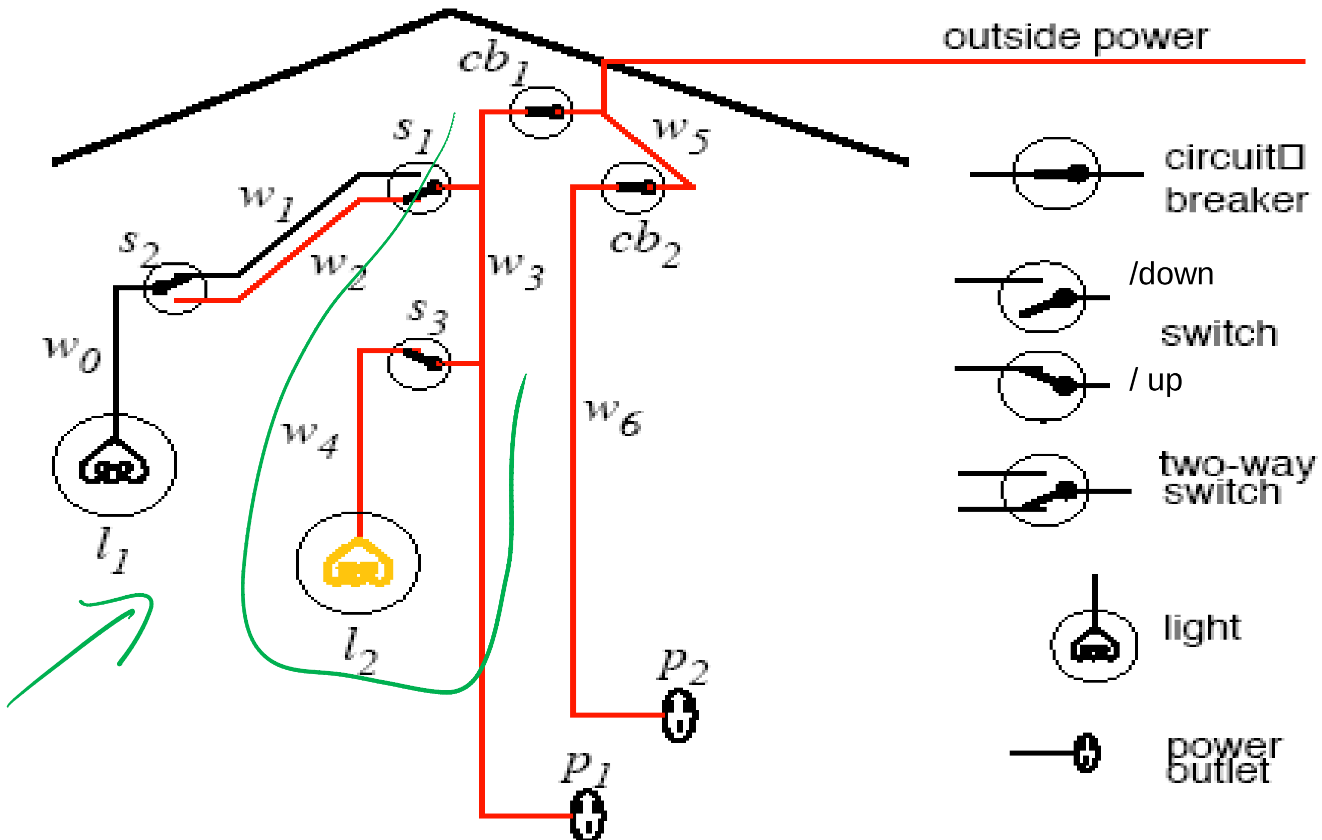
$live\_l_1 \leftarrow live\_w_0$

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

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

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

More on how the domain works….



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

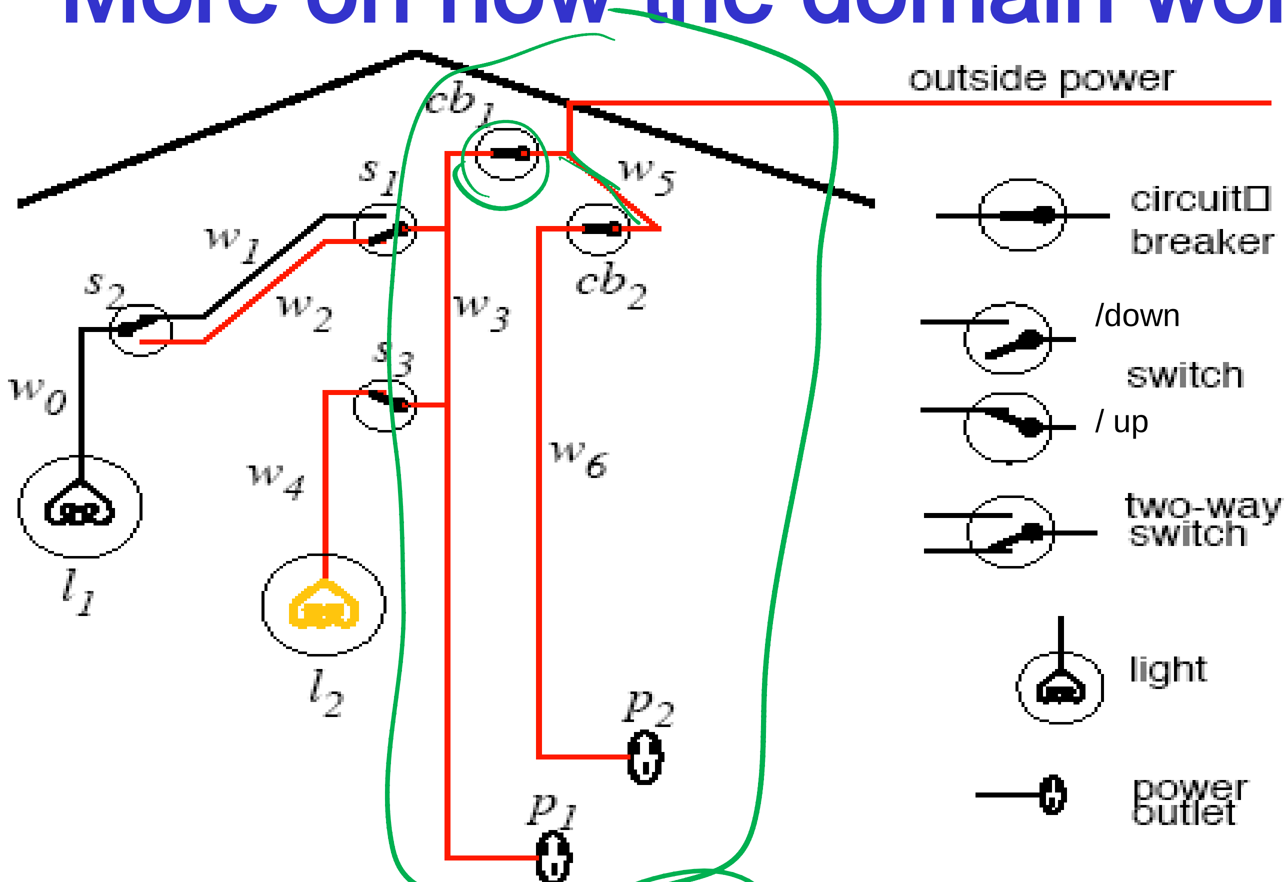
$live\_l_2 \leftarrow live\_w_4.$

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

$live\_p_1 \leftarrow live\_w_3.$

.

More on how the domain works….



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

$live\_p_2 \leftarrow live\_w_6.$

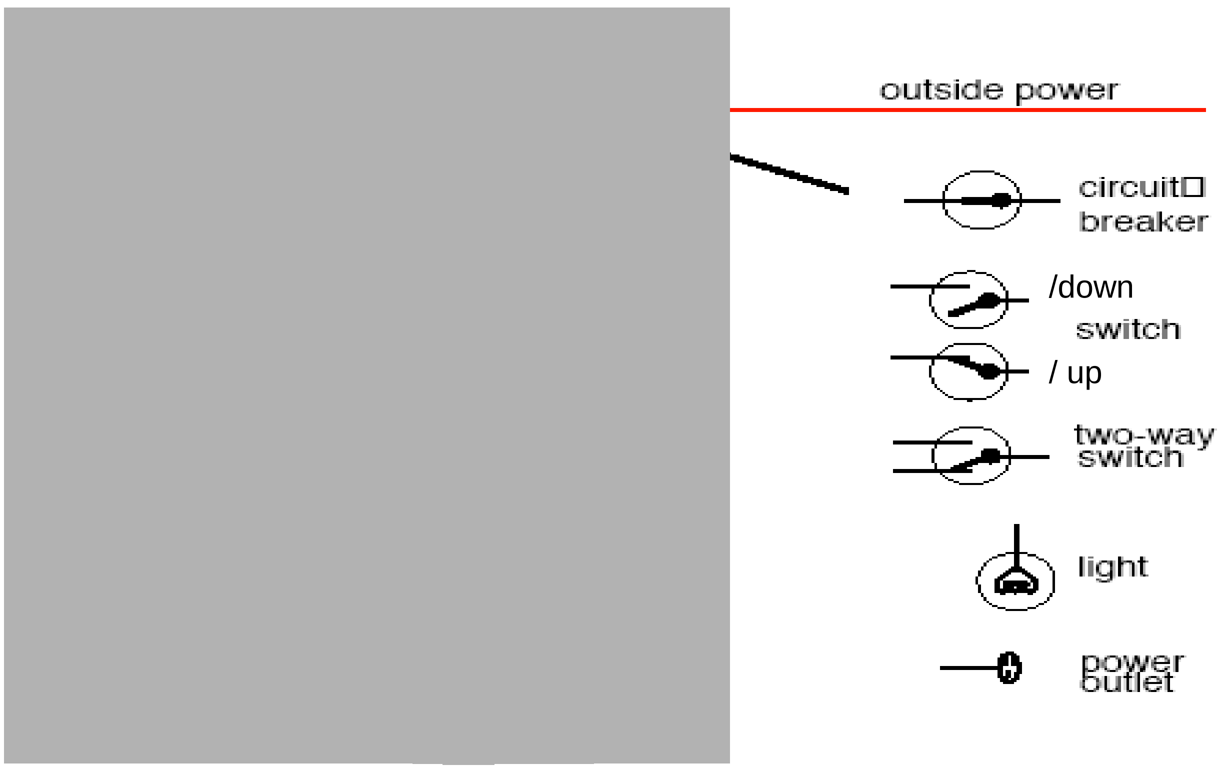
$live\_w_6 \leftarrow live\_w_5 \wedge ok\_cb_2.$

$live\_w_5 \leftarrow live\_outside.$

What else we may know about this domain?

- That some simple propositions are true

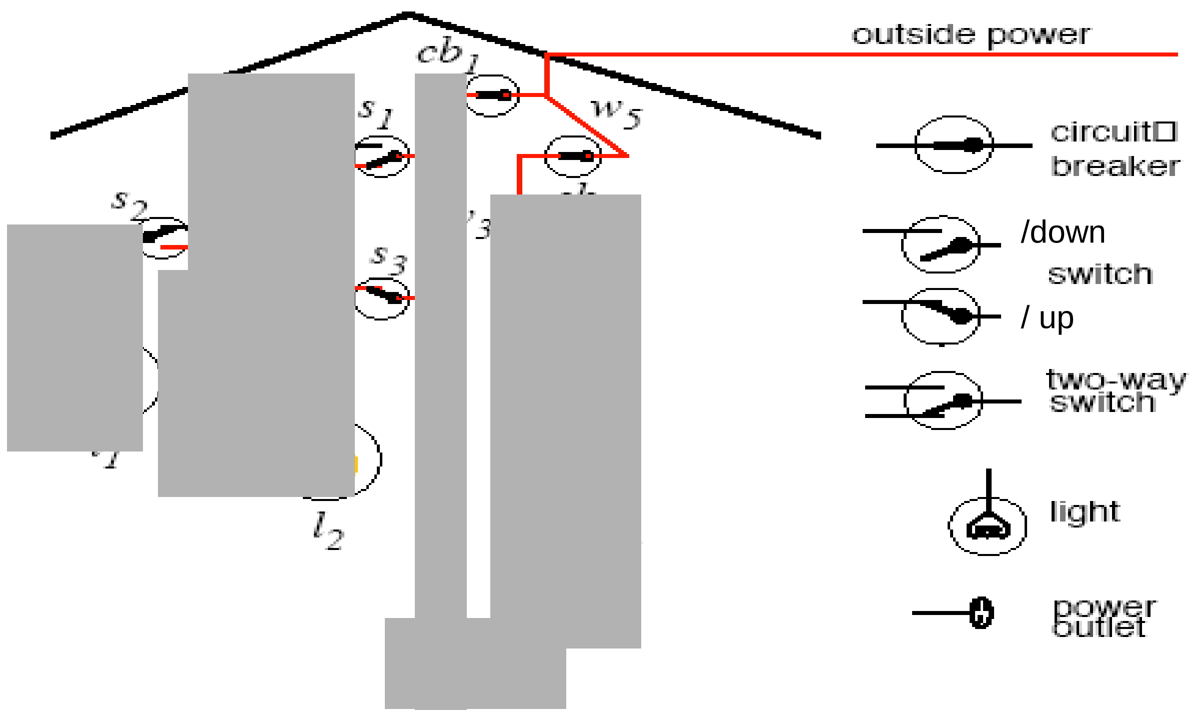
live_outside.



What else we may know about this domain?

- That some additional simple propositions are true

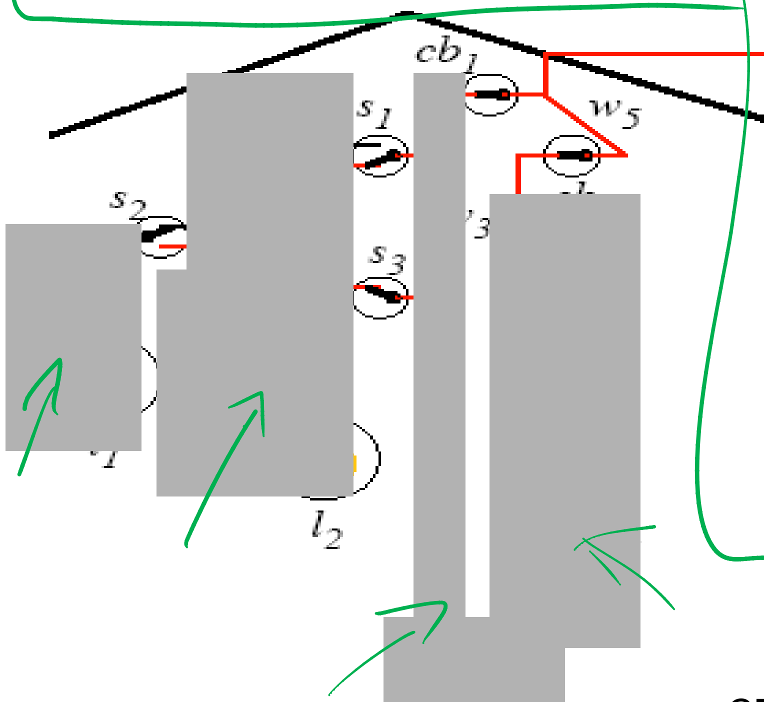
down_s$_1$. *up_s*$_2$. *up_s*$_3$. *ok_cb*$_1$. *ok_cb*$_2$. *live_outside*.



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 2^{19} __________ interpretations
- What is logically entailed by our KB are all the propositions that are true in all those ~~interpretations~~ 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

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

C

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

BU generates C

all the atoms added to C are in green

$live\_l_2$? ✓ → $live\_l_2 \in C \Rightarrow KB \vdash_{BU} live\_l_2 \Rightarrow KB \models live\_l_2$

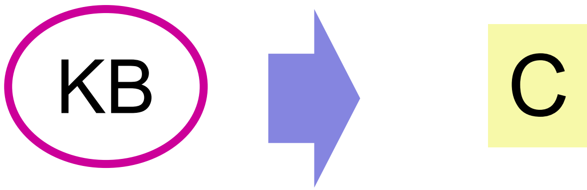
$live\_l_1$ ✗ which is not the case for $live\_l_1$

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 $G \subseteq C$

iclicker.

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

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*.

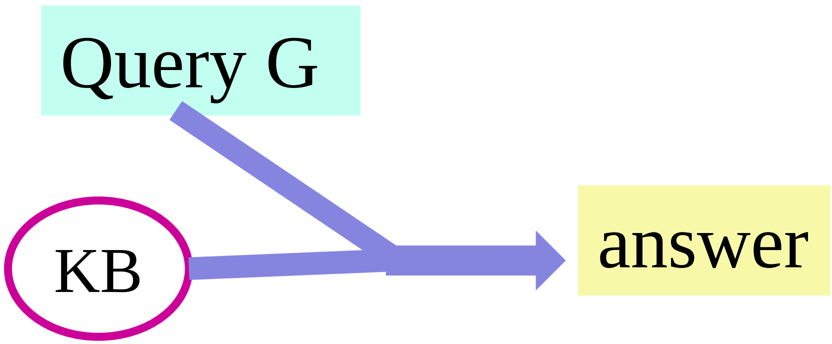
Bottom-up

G is proved if $G \subseteq C$

When does BU look at the query G?

- At the end

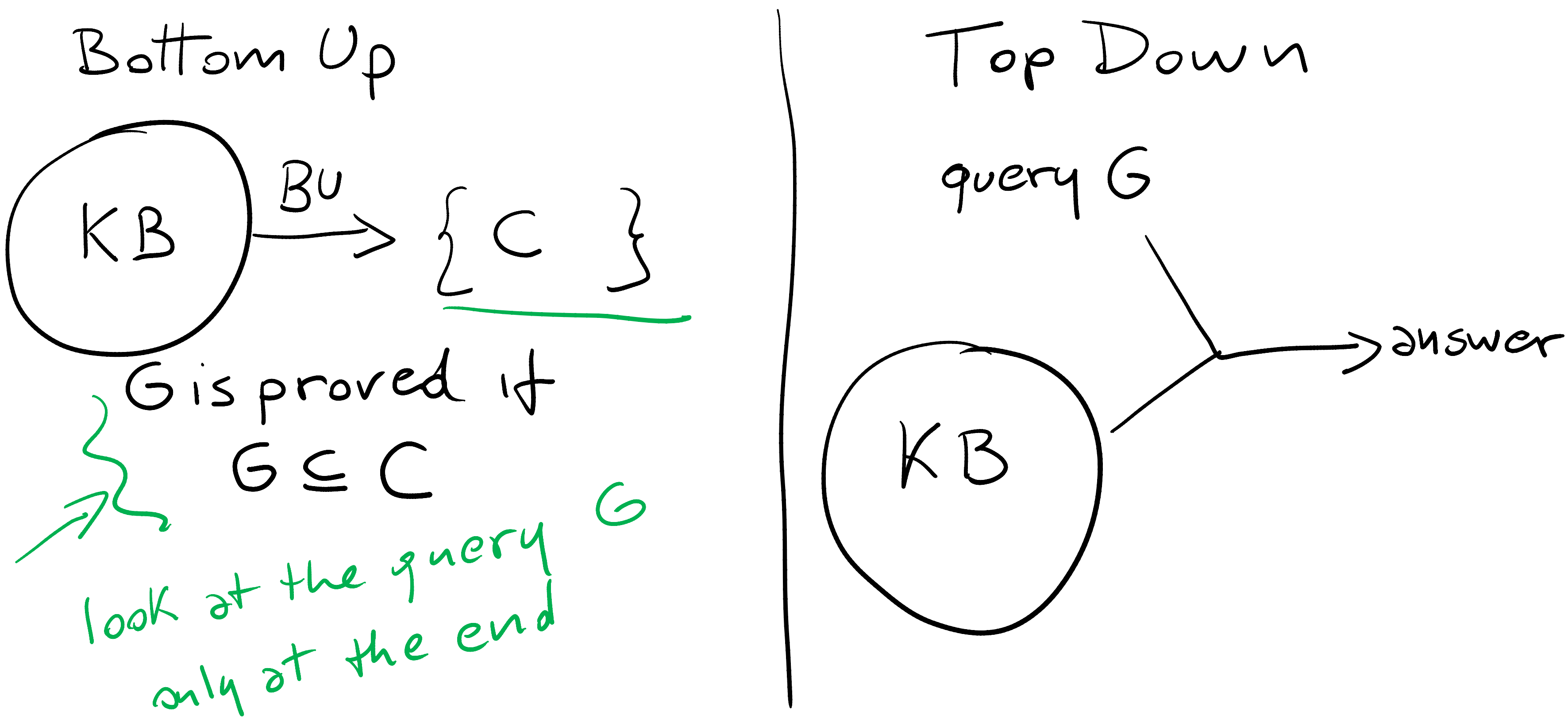
Top-down



TD performs a backward search starting at G

Top-down Ground Proof Procedure

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



Top-down Proof Procedure: Basic elements

Notation: An answer clause is of the form:

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

G

Express query as an answer clause (e.g., query $a_1 \wedge a_2 \wedge \ldots \wedge a_m$)

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

Rule of inference (called SLD Resolution)

Given an answer clause of the form:

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

KB

and the clause:

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

You can generate the answer clause

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

Rule of inference: Examples

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

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

KB clause

$yes \leftarrow b \land c.$ $\quad b \leftarrow k \land f.$ $\quad \Rightarrow yes \leftarrow k \land f \land c$

KB

$yes \leftarrow e \land f.$ $\quad e \leftarrow$ $\quad \Rightarrow yes \leftarrow f$

(successful) Derivations

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

  G

- 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 answer. $yes \leftarrow .$

- An unsuccessful derivation…..

  $yes \leftarrow a \wedge b$

Example: derivations

$a \leftarrow e \wedge f.$ $\quad a \leftarrow b \wedge c.$ $\quad b \leftarrow k \wedge f.$

$c \leftarrow e.$ $\quad d \leftarrow k.$ $\quad e.$

$f \leftarrow j \wedge e.$ $\quad f \leftarrow c.$ $\quad j \leftarrow c.$

KB

Query: a `(two ways)`

$yes \leftarrow a.$

$\quad$ " $\leftarrow b \wedge c$

$\quad$ " $\leftarrow k \wedge f \wedge c$

K cannot be eliminated so will Fail

$yes \leftarrow a.$

$\quad$ " $\leftarrow e \wedge f$

$\quad$ " $\leftarrow f$

$\quad$ " $\leftarrow c$

$\quad$ " $\leftarrow e$

$\quad yes \leftarrow$ . . . .

Example: derivations

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

$c \leftarrow e.$ $d \leftarrow k.$ $e.$

$f \leftarrow j \wedge e.$ $f \leftarrow c.$ $j \leftarrow c.$

KB

Query: $b \wedge e$

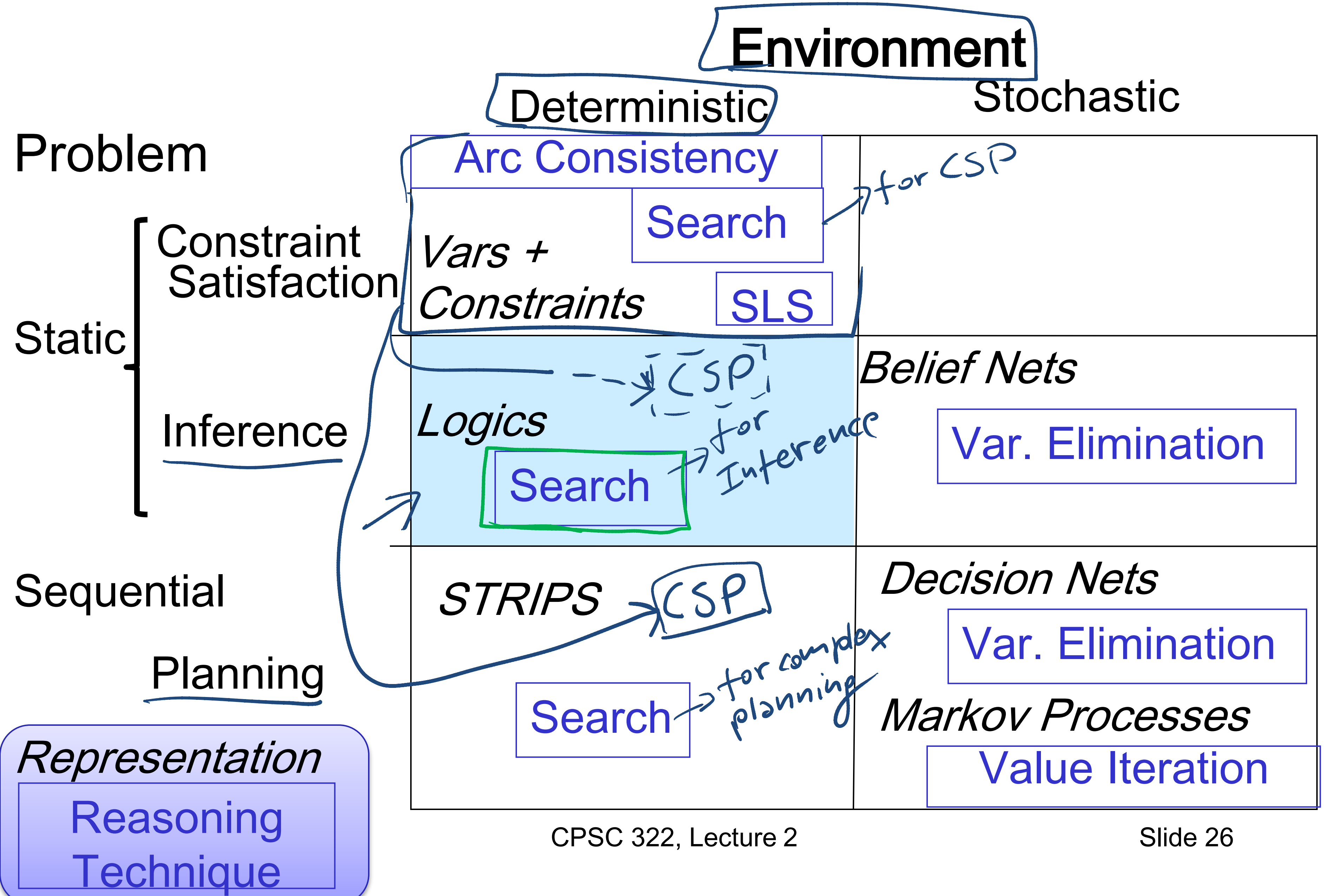
iclicker.

A. Provable by TD

B. It depends

C. Not Provable by TD

Course Big Picture



Standard Search vs. 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)*

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