CPSC 322: Introduction to Artificial Intelligence

Logics: Propositional and Propositional Definite Clause Logic: Syntax and Semantics

Textbook reference: [5.1,5.3]

Instructor: Varada Kolhatkar
University of British Columbia

Credit: These slides are adapted from the slides of the previous offerings of the course. Thanks to all instructors for creating and improving the teaching material and making it available!

Announcements

- Assignment 3 has been released.
  - Due date: **Nov 11, 11:59 PM**
- Final exam scheduled: **Dec 9 at 7:00pm**

Lecture outline

- Recap: Planning 👈
- Logic introduction
- Propositional logic: Syntax and semantics
- Propositional Definite Clause (PDC) Logic: Syntax
- Propositional Definite Clause (PDC) Logic: Semantics

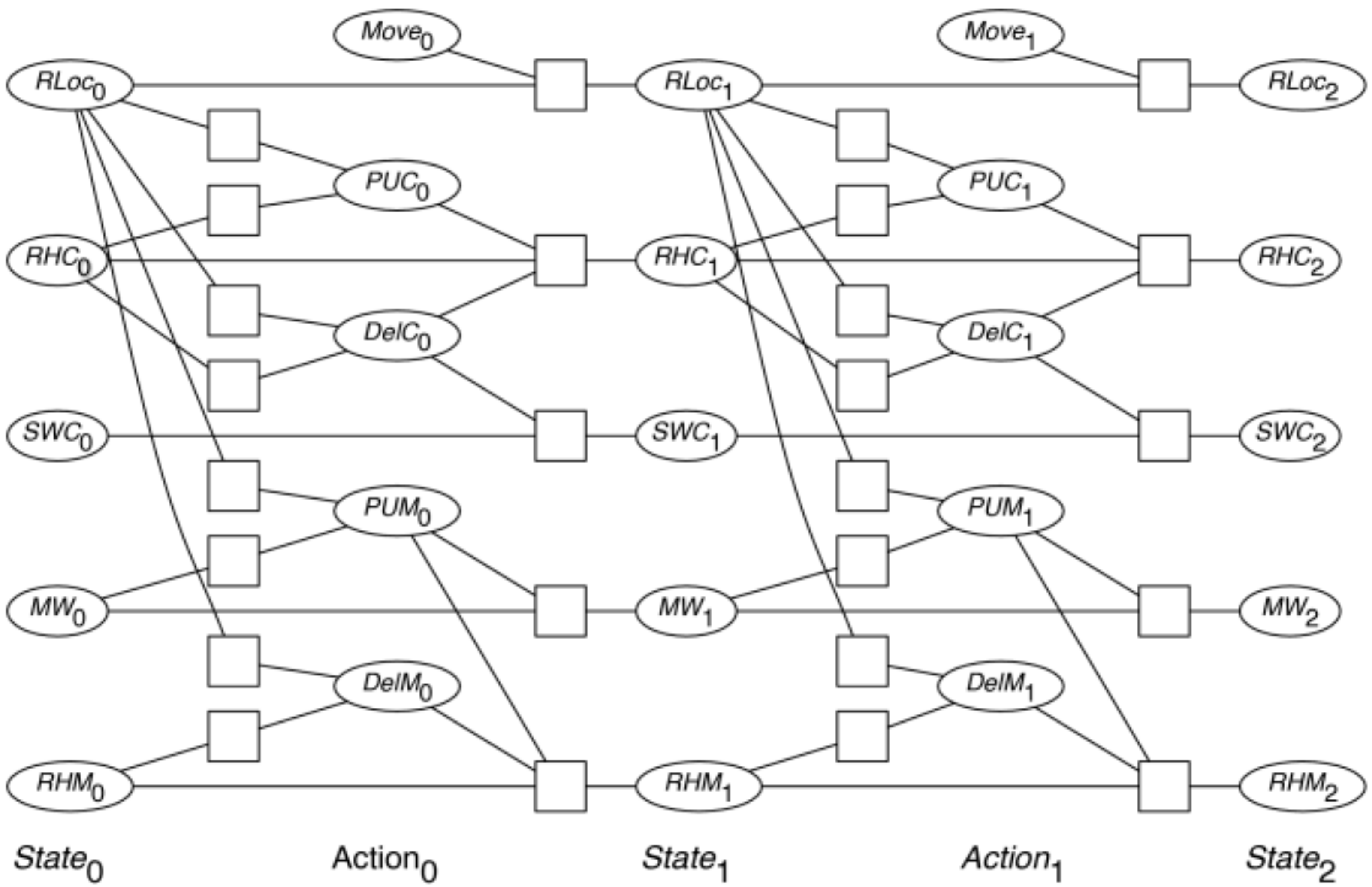
Recap: Planning (pair-share)

- The STRIPS representation for an action consists of what?
- What heuristic did we use to solve planning problem using forward search?
- What is meant by the *horizon* in a CSP planning problem?
- What are the precondition and effect constraints?

Solve planning as CSP: Pseudo code

```
solved = false
horizon = 0
while solved = false
     map STRIPS into CSP with horizon
     solve CSP → solution
     if solution then
         solved = T
     else
         horizon = horizon + 1
Return solution
```

Plan after solving the CSP



After you solve the CSP, what's the actual plan?

Lecture outline

- Recap: Planning
- Logic introduction 👈
- Propositional logic: Syntax and semantics
- Propositional Definite Clause (PDC) Logic: Syntax
- Propositional Definite Clause (PDC) Logic: Semantics

A rough CPSC 322 overview

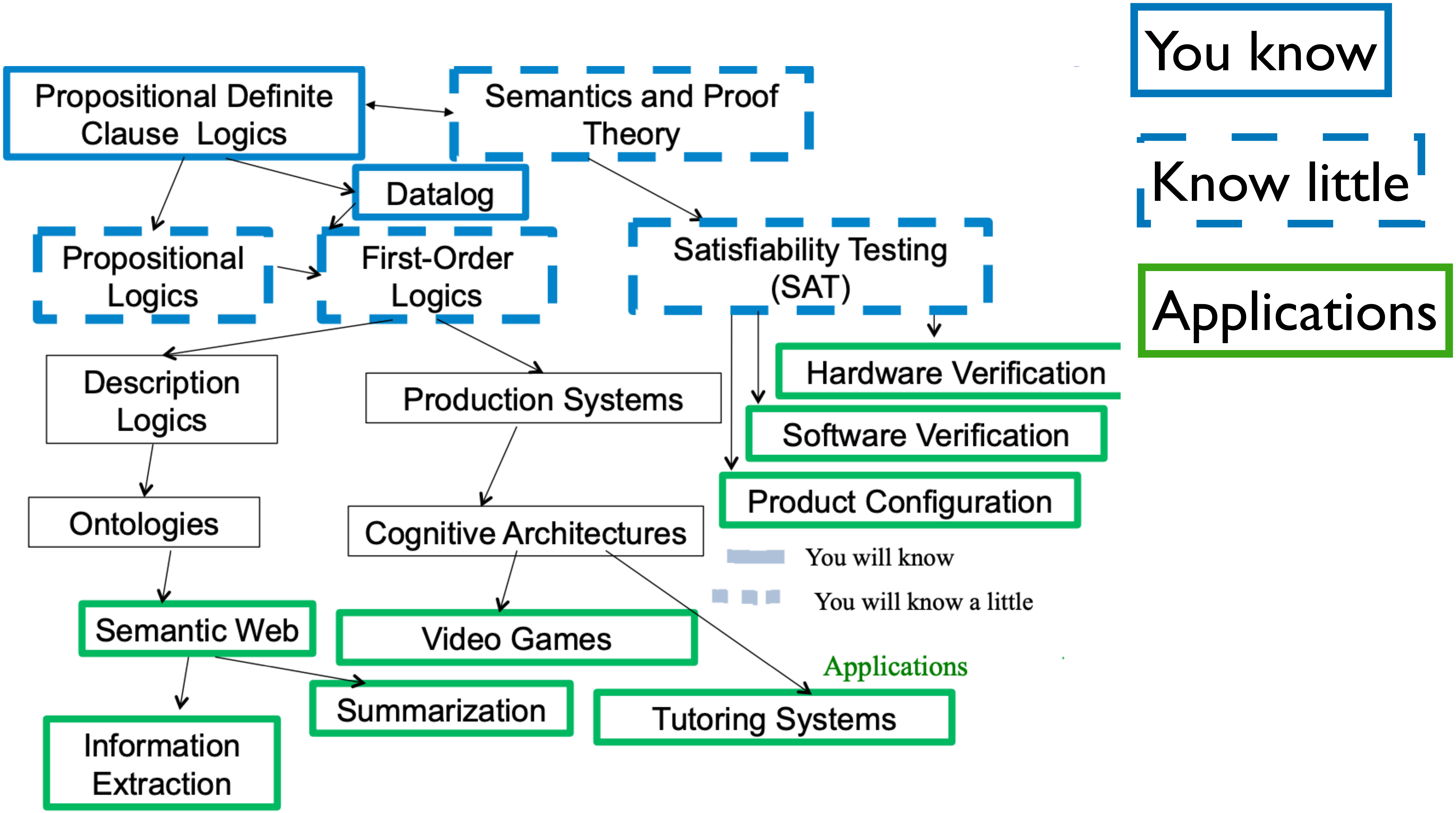
Representation and reasoning	Problem	Environment: Deterministic	Environment: Stochastic
Static	Constraint satisfaction	Variables + constraints Arc consistency Search	
Static	Query	Logics Search	Belief networks Variable elimination
Sequential	Planning	<u>STRIPS</u> Search	Decision networks Variable elimination Markov decision processes Value iteration

Today: Learning outcomes

From this lecture, students are expected to be able to:

- Verify whether a logical statement belongs to the language of full propositional logics.
- Verify whether a logical statement belongs to the language of propositional definite clauses
- Verify whether an interpretation is a model of a propositional definite clause logic knowledge base

Logics in AI



You know
Know little
Applications
Propositional Definite Clause Logics
Semantics and Proof Theory
Datalog
Propositional Logics
First-Order Logics
Satisfiability Testing (SAT)
Hardware Verification
Software Verification
Product Configuration
Description Logics
Production Systems
Ontologies
Cognitive Architectures
You will know
You will know a little
Semantic Web
Video Games
Applications
Summarization
Tutoring Systems
Information Extraction

Common sense knowledge

If the **baby** does not thrive on raw **milk**, boil **it**.

it = ?

Common sense knowledge

If the **baby** does not thrive on raw **milk**, boil **it**.

it = ?

common-sense knowledge

The boiling action is more common with milk.

Why propositional logic?

- We'll mostly focus on propositional logic as this is the starting point for more complex ones ….
- Natural to express knowledge about the world
  - What is true (boolean variables)
  - How it works (logical formulas)
- Well understood formal properties
- Boolean nature can be exploited for efficiency

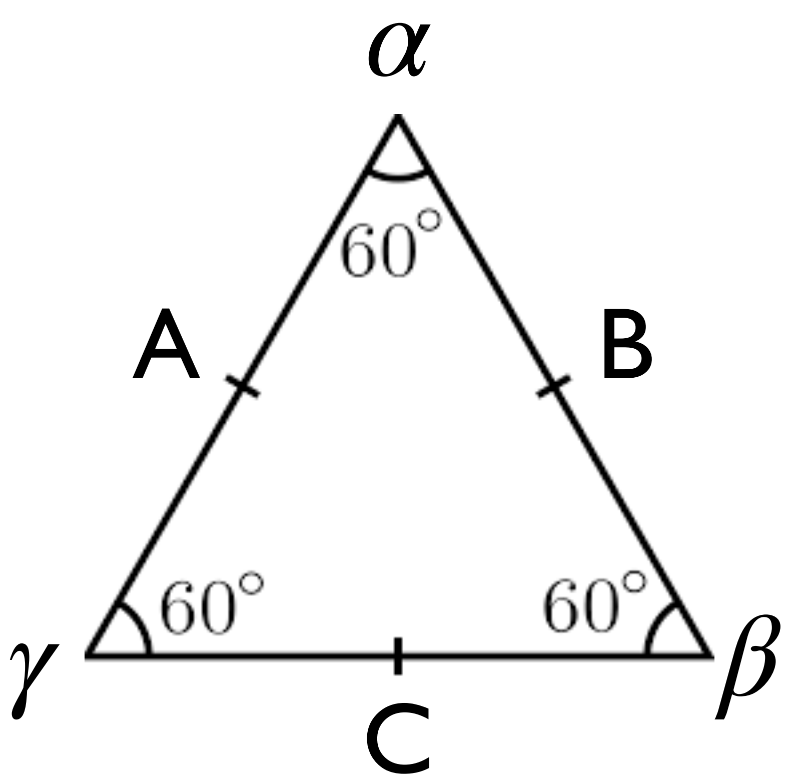
What you already know about logic

Some logical operators from **programming**:

```
If ((amount > 0) and (amount < 1000)) || !(age < 30)
```

Logic is the language of **mathematics**. It is used to define formal structures (e.g., sets, graphs) and to prove statements about them. For example:

$$\forall x, triangle(x) \implies A = B = C \iff \alpha = \beta = \gamma$$



Logic: a framework for representation and reasoning

We use logic as a Representation and Reasoning System that can be used to formalize a domain and to reason about it

- When we represent a domain about which we have only partial (but certain) information, we need to represent objects, properties, sets, groups, actions, events, time, space.

- All these can be represented as **objects** or **relationships between objects**

- **Logic** is the language to express knowledge about the world this way

Example

"Natural" to express knowledge about the world
(more natural than a "flat" set of variables & constraints)

"Every student who works diligently passes the course."

student(s) $\wedge$ registered(s, c) $\wedge$ course_name(c,322)
$\wedge$ works_diligently(s) $\Longrightarrow$ passes(s, c)

student(sam), registered(sam, c_1), course_name(c_1,322),
works_diligently(sam)

Query: passes(sam, c_1)?

Why logics?

Compact representation

- Compared to, e.g., a CSP with a variable for each student
- It is easy to **incrementally add knowledge**
- It is easy to **check and debug knowledge**
- Provides language for **asking complex queries**
- Well understood **formal properties**

Logic: a general framework for reasoning

- Let's think about how to represent a world about which we have only partial (but certain) information
- Our tool: **propositional logic**
- General problem:
  - tell the computer how the world works
  - tell the computer some facts about the world
  - ask a yes/no question about whether other facts must be true

Representation and reasoning system (RRS)

Definition (RRS):
A Representation and Reasoning System (RRS) consists of:

- **syntax**: specifies the symbols used, and how they can be combined to form legal sentences
- **semantics**: specifies the meaning of the symbols
- **reasoning theory or proof procedure**: a (possibly nondeterministic) specification of how an answer can be produced.

Representation and reasoning system (RRS)

Definition (RRS):
A Representation and Reasoning System (RRS) consists of:

- **syntax**: specifies the symbols used, and how they can be combined to form legal sentences
- **semantics**: specifies the meaning of the symbols
- **reasoning theory or proof procedure**: a (possibly nondeterministic) specification of how an answer can be produced.

We have seen several representations and reasoning procedures:

- State space graph + search
- CSP + search/arc consistency
- STRIPS + search/arc consistency

Lecture outline

- Recap: Planning
- Logic introduction
- Propositional logic: Syntax and semantics 👈
- Propositional Definite Clause (PDC) Logic: Syntax
- Propositional Definite Clause (PDC) Logic: Semantics

Propositional logic: syntax

Examples: It is sunny.

Definition (proposition):
A **proposition** is a sentence, written in a language, that has a truth value (i.e., it is true or false) in a world. It can be atomic or compound.

Examples: sunny, p_1, $live\_l_1$

Definition (atom):
An **atomic proposition** or **an atom** is a symbol. We use the convention that atomic propositions consist of letters, digits and the underscore (_) and start with a **lower-case** letter.

Propositional logic: syntax

Definition (formula): Examples: $(sunny \wedge happy) \vee (rain \wedge sad)$

A **proposition** or **logical formula** is either an atomic proposition or a compound proposition of the form shown below, where p and q are propositions and $\neg$, $\wedge$, $\vee$, $\rightarrow$, $\leftarrow$, $\leftrightarrow$ are **logical connectives.**

$\neg p$	**"not p"**	**negation of p**
$p \wedge q$	"p and q"	**conjunction** of p and q
$p \vee q$	"p or q"	**disjunction** of p and q
$p \rightarrow q$	"p implies q"	**implication** of q from p
$p \leftarrow q$	"p if q"	**implication** of p from q
$p \leftrightarrow q$	"p if and only if q"	**equivalence** of p and q

In other words …

Propositions are also called **sentences**.

- If S is a sentence, $\neg S$ is also a sentence (negation)
- If S_1 and S_2 are sentences, $S_1 \wedge S_2$ is a sentence (conjunction)
- If S_1 and S_2 are sentences, $S_1 \vee S_2$ is a sentence (disjunction)
- If S_1 and S_2 are sentences, $S_1 \rightarrow S_2$ is a sentence (implication)
- If S_1 and S_2 are sentences, $S_1 \leftrightarrow S_2$ is a sentence (biconditional)

ASIDE: A BNF grammar of sentences in propositional logic

$$
\begin{array}{rcl}
Sentence & \rightarrow & AtomicSentence \mid ComplexSentence \\
AtomicSentence & \rightarrow & True \mid False \mid P \mid Q \mid R \mid \ldots \\
ComplexSentence & \rightarrow & (\, Sentence \,) \mid [\, Sentence \,] \\
& \mid & \neg\, Sentence \\
& \mid & Sentence \land Sentence \\
& \mid & Sentence \lor Sentence \\
& \mid & Sentence \;\Rightarrow\; Sentence \\
& \mid & Sentence \;\Leftrightarrow\; Sentence \\
\\
\textsc{Operator Precedence} & : & \neg, \land, \lor, \Rightarrow, \Leftrightarrow
\end{array}
$$

Source: Russel and Nerving book

Propositional logic: semantics

Do any of these statements mean anything? Syntax doesn't answer this question!

Semantics allows you to relate the symbols in the logic to the domain you're trying to model.

Propositional logic: semantics

Definition (Interpretation):
An interpretation I assigns a truth value to each atom.

We can use the interpretation to determine the truth value of formulas. Note that truth values are only defined with respect to interpretations; propositions may have different truth values in different interpretations.

Propositional logic: semantics

Definition (interpretation):
An interpretation I assigns a truth value to each atom.

If our domain has 10 atoms, how many interpretations are there?

A. 2^{10}

B. 10^2

C. 10×2

Propositional logic: semantics

iclicker.

Definition (interpretation):
An interpretation I assigns a truth value to each atom.

If our domain has 10 atoms, how many interpretations are there?

A. 2^{10} ✅

B. 10^2

C. 10×2

Propositional logic: semantics

Definition (interpretation):
An interpretation I assigns a truth value to each atom.

If our domain has 10 atoms, how many interpretations are there?

A. 2^{10}

B. 10^2

C. 10×2

Similar to possible worlds in CSP

Propositional logic: semantics

Whether a compound proposition is true in an interpretation is inferred using the truth table

p	q	$\neg p$	$p \wedge q$	$p \vee q$	$p \leftarrow q$	$p \rightarrow q$	$p \leftrightarrow q$
true	true	false	true	true	true	true	true
true	false	false	false	true	true	false	false
false	true	true	false	true	false	true	false
false	false	true	false	false	true	true	true

Interpretation example (pair-share)

Suppose we have three atoms: ai_is_powerful, sky_is_orange, happy. Suppose interpretation I_1 assigns true to ai_is_powerful, false to happy and false to sky_is_orange

Which of the following are **true** in I_1?

1. ai_is_powerful ∧ happy
2. ¬happy
3. happy ← ai_is_powerful
4. ¬happy ← sky_is_orange

I_1
ai_is_powerful = true
happy = false
sky_is_orange = false

Propositional logic in practice

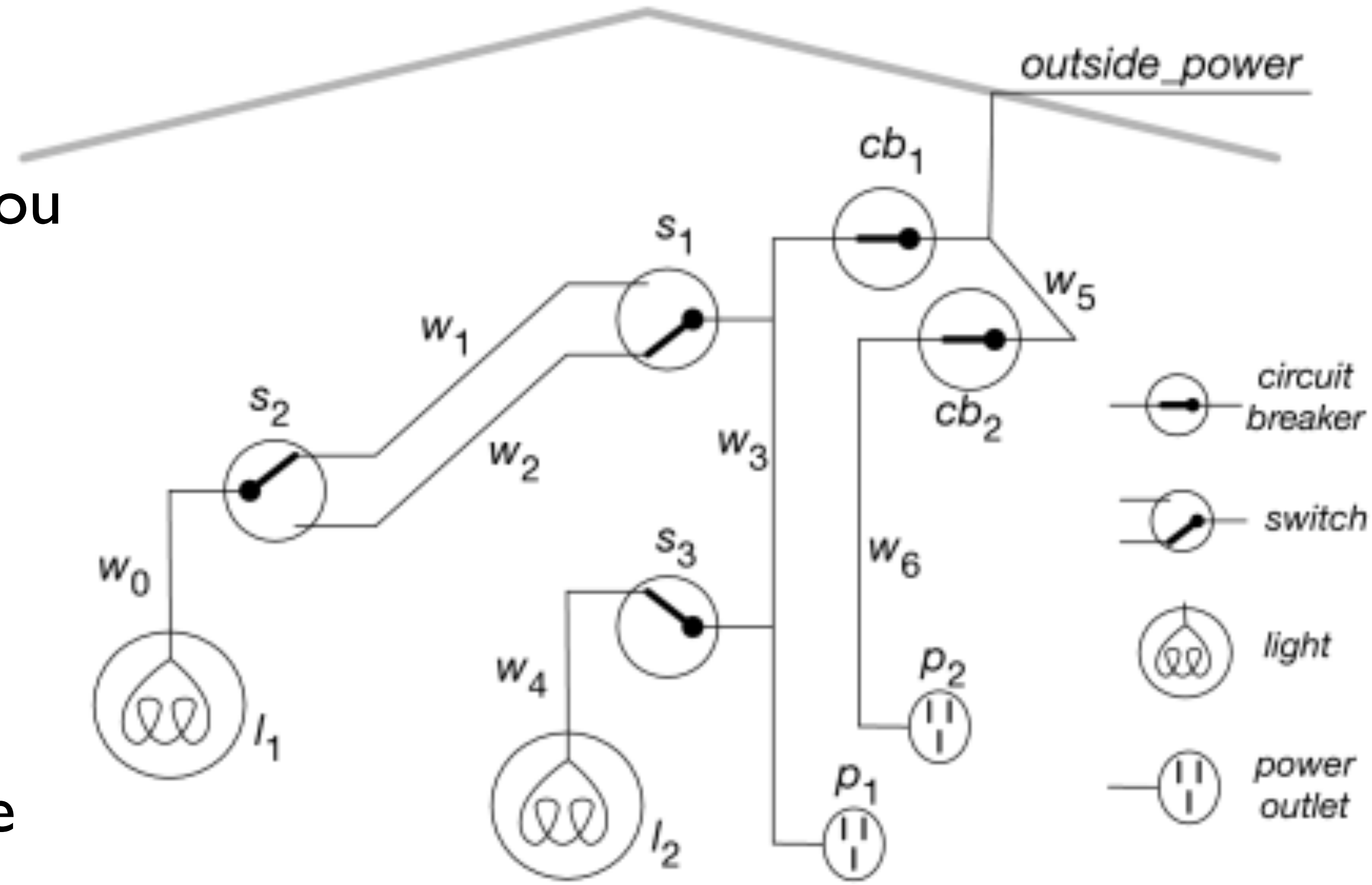
- Agent is told (perceives) some facts about the world (some propositions are true)
- Agent is told (already knows / learns) how the world works (logical formulas)
- Agent can answer **yes/no questions** about whether other **facts must be true**

Using logics to make inferences

1. Begin with a task domain.
2. Distinguish those things you want to talk about (the ontology)
3. Choose symbols in the computer to denote propositions
4. Tell the system knowledge about the domain
5. Ask the system whether new statements about the domain are true or false

Example: Electric environment

1. Begin with a task domain.
2. Distinguish those things you want to talk about (the ontology)
3. Choose symbols in the computer to denote propositions
4. Tell the system knowledge about the domain
5. Ask the system whether new statements about the domain are true or false



Lecture outline

- Recap: Planning
- Logic introduction
- Propositional logic: Syntax and semantics
- Propositional Definite Clause (PDC) Logic: Syntax 👈
- Propositional Definite Clause (PDC) Logic: Semantics

Propositional Definite Clause (PDC) Logic

Propositional Definite Clauses: our first **logical** representation and reasoning system. (Very simple!)

Only two kinds of statements:

- that a **proposition is true**
- that a **proposition is true if one or more other propositions are true**

Sublanguage of propositional logic that does not allow uncertainty or ambiguity.

PDC logic

Limited because we are only considering propositions. But still useful because

- **Adequate in many domains** (with some adjustments)
- Reasoning steps **easy to follow by humans**
- **Inference linear in size** of your set of statements
- Similar formalisms used in **cognitive architectures**

PDC logic: Syntax

Definition (atom):

Examples: $p_1, live\_l_1$

An **atomic proposition** or **an atom** is a symbol.

Definition (body):

Examples: $p_1, p_1 \land p_2, ok\_w_1 \land live\_w_0$

A **body** is of the form $a_1 \land \ldots \land a_m$, where $a_1, \ldots, a_m$ are atoms, $m \geq 0$.

Definition (definite clause):

A **definite clause** is of the form $h \leftarrow b$, where h is an atom ("head") and b is a body. (Read this as "h if b".)

Examples: $p_1, p_1 \leftarrow p_2, live\_w_0 \leftarrow live\_w_1 \land up\_s_2$

PDC logic: Syntax

Definition (definite clause):
A definite clause is of the form $h \leftarrow b$, where h is an atom ("head") and b is a body. (Read this as "h if b".)

Examples: $p_1, p_1 \leftarrow p_2, live\_w_0 \leftarrow live\_w_1 \wedge up\_s_2$

In definite clause $h \leftarrow b$ (i.e., $h \leftarrow a_1 \wedge \ldots \wedge a_m$),

Definition (rule):
If $m > 0$, the definite clause is called a **rule**.

Examples: $p_1 \leftarrow p_2, live\_w_0 \leftarrow live\_w_1 \wedge up\_s_2$

Definition (atomic clause):

Examples: p_1

If $m = 0$, the arrow can be omitted and the clause is an **atomic clause** or a **fact**, with an empty body.

Identify definite clauses (pair-share)

Which of the following are legal definite clauses?

1. $ai\_is\_powerful$

2. $\neg ai\_is\_powerful$

3. $ai\_is\_fun \leftarrow learn\_useful\_techniques \wedge \neg tooMuch\_work$

4. $sam\_is\_in\_room \wedge night\_time \leftarrow switch\_1\_is\_up$

5. $switch\_1\_is\_up \leftarrow sam\_is\_in\_room \wedge night\_time$

6. $happy \vee sad \vee \neg alive$

7. $srtsyj \leftarrow errt \wedge gffdgdgd$

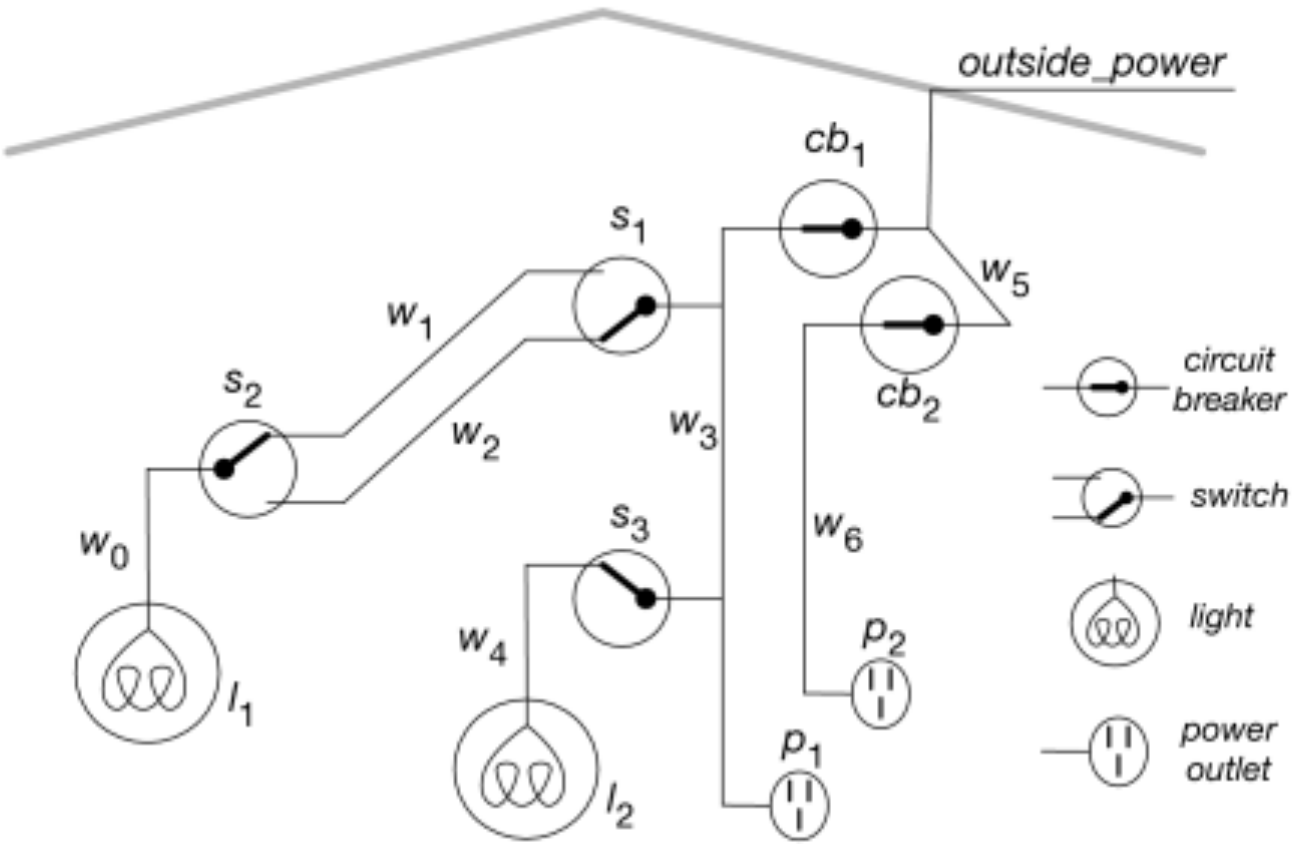
PDC logic semantics: Knowledge Base (KB)

Definition: A **knowledge base** (KB) is a set of definite clauses.

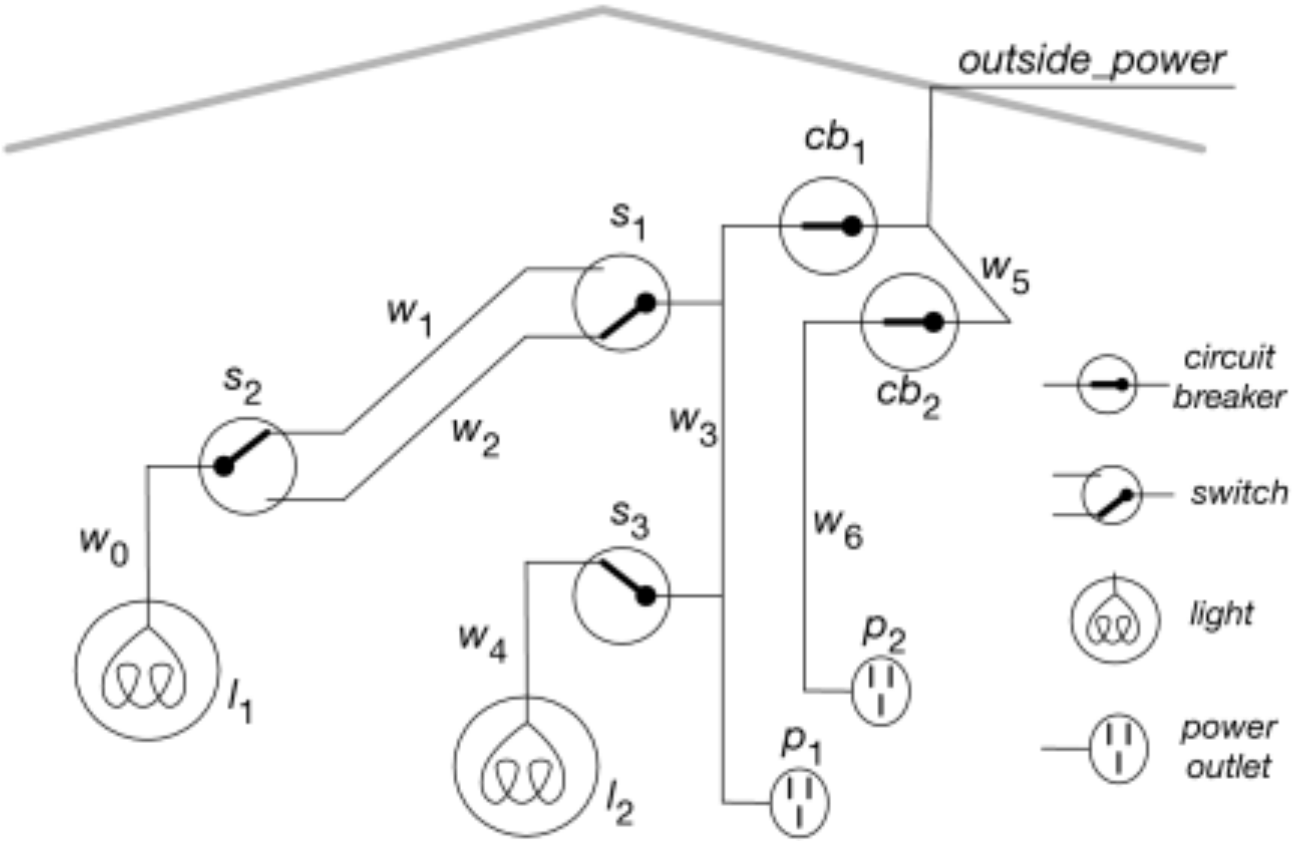
<u>Example:</u>

$$\{\, p_2, p_3, p_4, p_1 \leftarrow p_2 \wedge p_3 \wedge p_4, live\_l_1 \}$$

Example: Electric environment

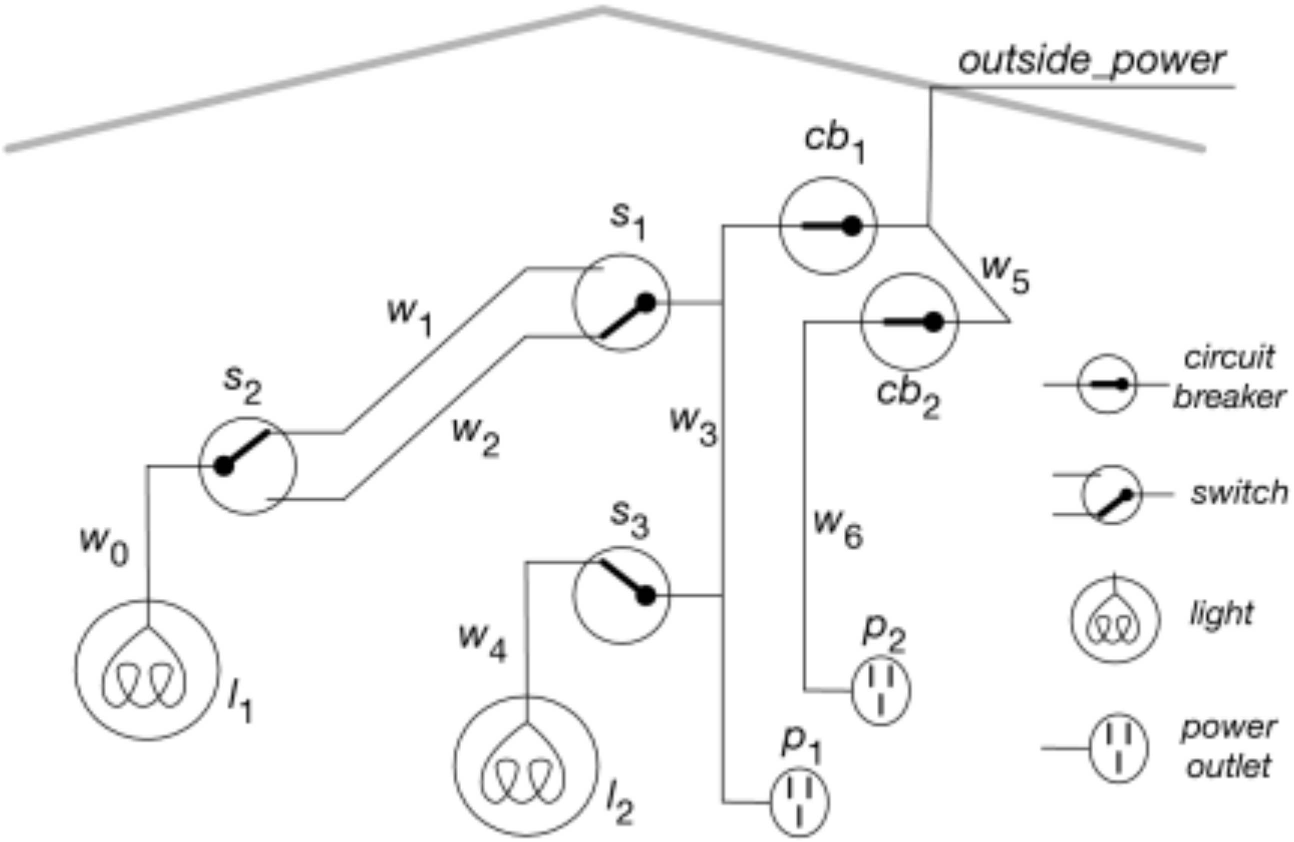


Example: Electric environment



$$\left.\begin{array}{l} light\_l_1. \\ light\_l_2. \\ ok\_l_1. \\ ok\_l_2. \\ ok\_cb_1. \\ ok\_cb_2. \\ live\_outside. \end{array}\right\} \text{atoms}$$

Example: Electric environment



atoms

$light\_l_1.$

$light\_l_2.$

$ok\_l_1.$

$ok\_l_2.$

$ok\_cb_1.$

$ok\_cb_2.$

$live\_outside.$

rules

$live\_l_1 \leftarrow live\_w_0.$

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

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

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

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

$live\_l_2 \leftarrow live\_w_4.$

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

$live\_p_1 \leftarrow live\_w_3.$

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

$live\_p_2 \leftarrow live\_w_6.$

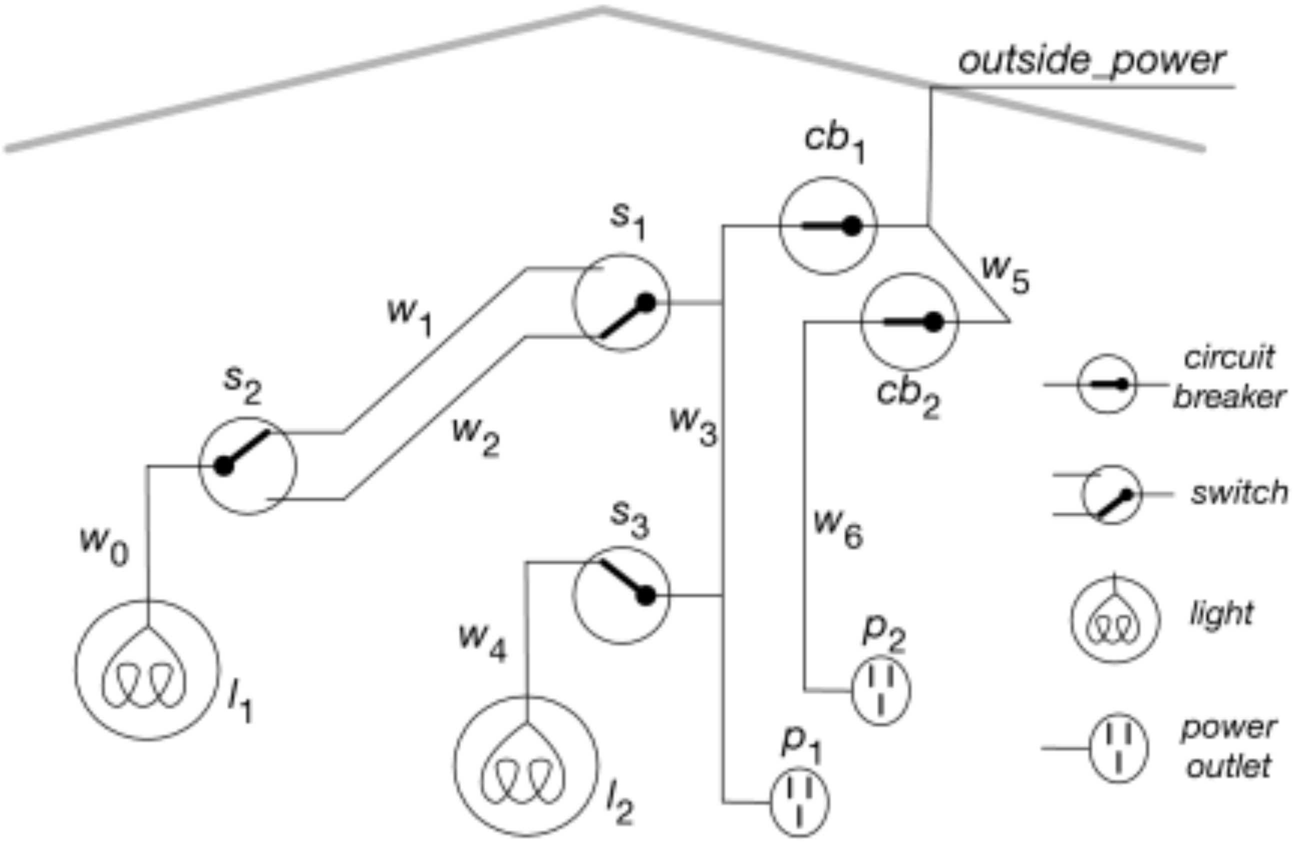
$live\_w_6 \leftarrow live\_w_5 \wedge ok\_cb_2.$

$live\_w_5 \leftarrow live\_outside.$

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

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

Example: Electric environment



$$
\left.\begin{array}{l}
light\_l_1. \\
light\_l_2. \\
ok\_l_1. \\
ok\_l_2. \\
ok\_cb_1. \\
ok\_cb_2. \\
live\_outside.
\end{array}\right\} \text{atoms}
\quad + \quad
\left.\begin{array}{l}
live\_l_1 \leftarrow live\_w_0. \\
live\_w_0 \leftarrow live\_w_1 \wedge up\_s_2. \\
live\_w_0 \leftarrow live\_w_2 \wedge down\_s_2. \\
live\_w_1 \leftarrow live\_w_3 \wedge up\_s_1. \\
live\_w_2 \leftarrow live\_w_3 \wedge down\_s_1. \\
live\_l_2 \leftarrow live\_w_4. \\
live\_w_4 \leftarrow live\_w_3 \wedge up\_s_3. \\
live\_p_1 \leftarrow live\_w_3. \\
live\_w_3 \leftarrow live\_w_5 \wedge ok\_cb_1. \\
live\_p_2 \leftarrow live\_w_6. \\
live\_w_6 \leftarrow live\_w_5 \wedge ok\_cb_2. \\
live\_w_5 \leftarrow live\_outside. \\
lit\_l_1 \leftarrow light\_l_1 \wedge live\_l_1 \wedge ok\_l_1. \\
lit\_l_2 \leftarrow light\_l_2 \wedge live\_l_2 \wedge ok\_l_2.
\end{array}\right\} \text{rules}
\quad = \quad \text{KB: Set of definite clauses}
$$

PDC logic: Semantics

Definition (interpretation):
An interpretation I assigns a truth value to each atom.

We can use the interpretation to determine the truth value of clauses.

PDC logic: Semantics

Definition (truth values of statements):
A body $b_1 \ldots b_m$ is true in interpretation I if and only if $\forall b_i, 0 \leq i \leq m, b_i$ is true in I.

	p	q	r	s	$p \wedge r$	$p \wedge r \wedge s$
I_1	T	T	T	T		
I_2	F	F	F	F		
I_3	T	T	F	F		
I_4	T	T	T	F		
I_5	T	T	F	T		

PDC logic: Semantics

Definition (truth values of statements):
A rule $h \leftarrow b$ is false in I if and only if b is true in I and h is false in I.

	p	q	r	s	$p \leftarrow r$	$s \leftarrow q \wedge r$
I_1	T	T	T	T		
I_2	F	F	F	F		
I_3	T	T	F	F		
I_4	T	T	T	F		
I_5	T	T	F	T		

In other words: "if b **is true** I am claiming that h **must be true**, otherwise I am not making any claim".

PDC logic semantics: Knowledge Base (KB)

iclicker.

Definition: A **knowledge base KB** is true in I if and only if every clause in KB is true in I.

Which of the following KBs below are true in *K*?

	p	q	r	s
K	T	T	F	F

A.

p
r
$s \leftarrow p \wedge q$

B.

p
r
$s \leftarrow p$

C.

p
$q \leftarrow r \wedge s$

PDC logic semantics: Knowledge Base (KB)

iclicker

Definition: A **knowledge base KB** is true in I if and only if every clause in KB is true in I.

	p	q	r	s
K	T	T	F	F

Which of the following KBs below are true in K?

A.

p
r
$s \leftarrow p \wedge q$

B.

p
r
$s \leftarrow p$

C.

p
$q \leftarrow r \wedge s$

✅

PDC logic semantics: Models

Definition: A **model** of a set of clauses (KB) is an interpretation in which all the clauses are true.

Which interpretations are models?

$$KB = \begin{cases} p \leftarrow q. \\ q. \\ r \leftarrow s. \end{cases}$$

	p	q	r	s	model of KB?
I_1	T	T	T	T	
I_2	F	F	F	F	
I_3	T	T	F	F	
I_4	T	T	T	F	
I_5	T	T	F	T	

PDC logic semantics: Models

Definition: A **model** of a set of clauses (KB) is an interpretation in which all the clauses are true.

Which interpretations are models?

$$KB = \begin{cases} p \leftarrow q. \\ q. \\ r \leftarrow s. \end{cases}$$

	p	q	r	s	$p \leftarrow q$	$r \leftarrow s$	KB
I_1	T	T	T	T			
I_2	F	F	F	F			
I_3	T	T	F	F			
I_4	T	T	T	F			
I_5	T	T	F	T			

PDC logic semantics: Entailment

Given a knowledge base KB, can we **infer** implicit information (i.e., new clauses)?

Definition: If KB is a knowledge base and g is a proposition, g is a **logical consequence** of KB, written as $KB \models g$, if g is true in every model of KB.

We also say that g **logically follows** from KB, or that KB **entails** g. In other words, $KB \models g$ if there is no interpretation in which KB is true and g is false.

PDC semantics: Entailment

Definition: If KB is a knowledge base and g is a proposition, g is a **logical consequence** of KB, written as $KB \models g$, if g is true in every model of KB.

$$KB = \begin{cases} p \leftarrow q. \\ \quad q. \\ r \leftarrow s. \end{cases}$$

Given the knowledge base KB, which of the following are true?

A. $KB \models p$

B. $KB \models q$

C. $KB \models r$

D. $KB \models s$

PDC semantics: Entailment

iclicker

Definition: If KB is a knowledge base and g is a proposition, g is a **logical consequence** of KB, written as $KB \models g$, if g is true in every model of KB.

$$KB = \begin{cases} p \leftarrow q. \\ q. \\ r \leftarrow s. \end{cases}$$

Given the knowledge base KB, which of the following are true?

A. $KB \models p$ ✅

B. $KB \models q$ ✅

C. $KB \models r$

D. $KB \models s$

Revisit: Learning outcomes

From this lecture, students are expected to be able to:

- Verify whether a logical statement belongs to the language of full propositional logics.
- Verify whether a logical statement belongs to the language of propositional definite clauses
- Verify whether an interpretation is a model of a propositional definite clause logic knowledge base

Coming up

We'll start using all these definitions for automated proofs!

<u>5.3.2 Proofs</u>