Logic: Intro & Propositional Definite Clause Logic

Alan Mackworth

UBC CS 322 - Logic 1
February 27, 2013

P & M Textbook §5.1

Lecture Overview

- Recap: CSP planning
- Intro to Logic
- Propositional Definite Clause Logic: Syntax
- Propositional Definite Clause Logic: Semantics

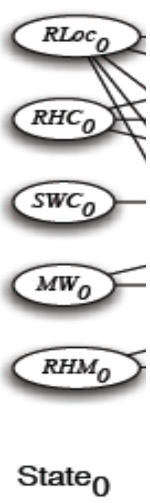
What is the difference between CSP and Planning?

- CSP: static
  - Find a single total variable assignment that satisfies all constraints
- Planning: sequential
  - Find a sequence of actions to get from start to goal
    - CSPs don't even have the concept of actions
  - Some similarities to CSP:
    - Use of variables/values
    - Can solve planning as CSP.
      But the CSP corresponding to a planning instance can be very large
      - Make CSP variable for each STRIPS variable at each time step
      - Make CSP variable for each STRIPS action at each time step

CSP Planning: Solving the problem

Map STRIPS Representation into CSP for horizon 0,1, 2, 3, …

Solve CSP for horizon h = 0, 1, 2, 3, … until solution found at the lowest possible horizon



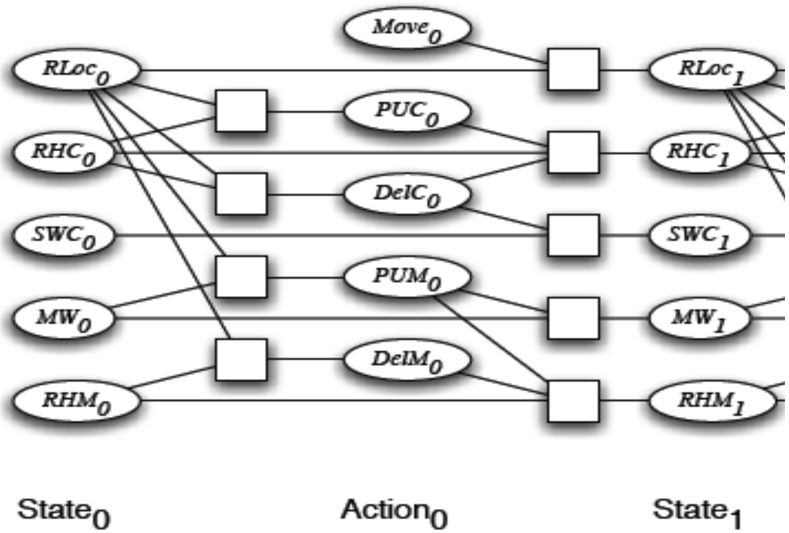
h = 0
Is there a solution for this horizon?

If yes, DONE!
If no, continue …

CSP Planning: Solving the problem

Map STRIPS Representation into CSP for horizon 0,1, 2, 3, …

Solve CSP for horizon h=0, 1, 2, 3, … until solution found at the lowest possible horizon



h = 1
Is there a solution for this horizon?

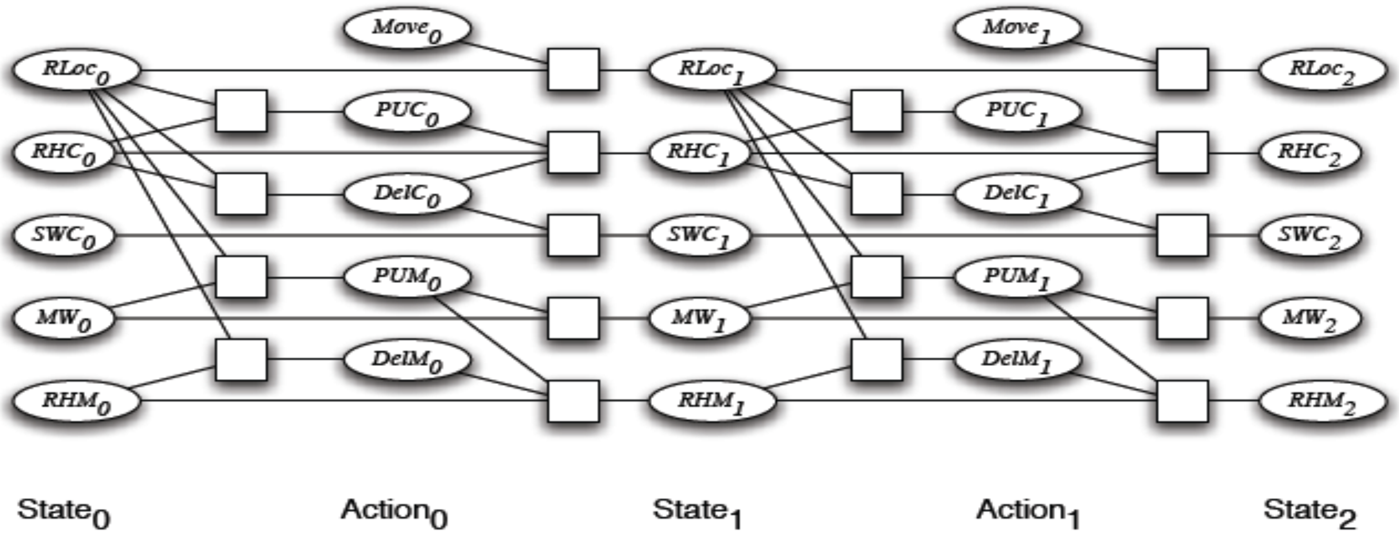
If yes, DONE!
If no, continue …

N.B. Notice the use of k-ary constraints, k= 1,2,3,4, .., so need GAC to solve CSP at each step, not just AC for binary constraints.

CSP Planning: Solving the problem

Map STRIPS Representation into CSP for horizon 0,1, 2, 3, …

Solve CSP for horizon h=0, 1, 2, 3, … until solution found at the lowest possible horizon



h = 2: Is there a solution for this horizon?

If yes, DONE!

If no….continue

Solving Planning as CSP: pseudo code

```
for horizon h=0,1,2,…
    map STRIPS into a CSP csp with horizon h
    solve that csp
    if solution to the csp exists then
        return solution
end for
```

Solve each of the CSPs based on systematic search

- Not SLS! SLS cannot determine that no solution exists!

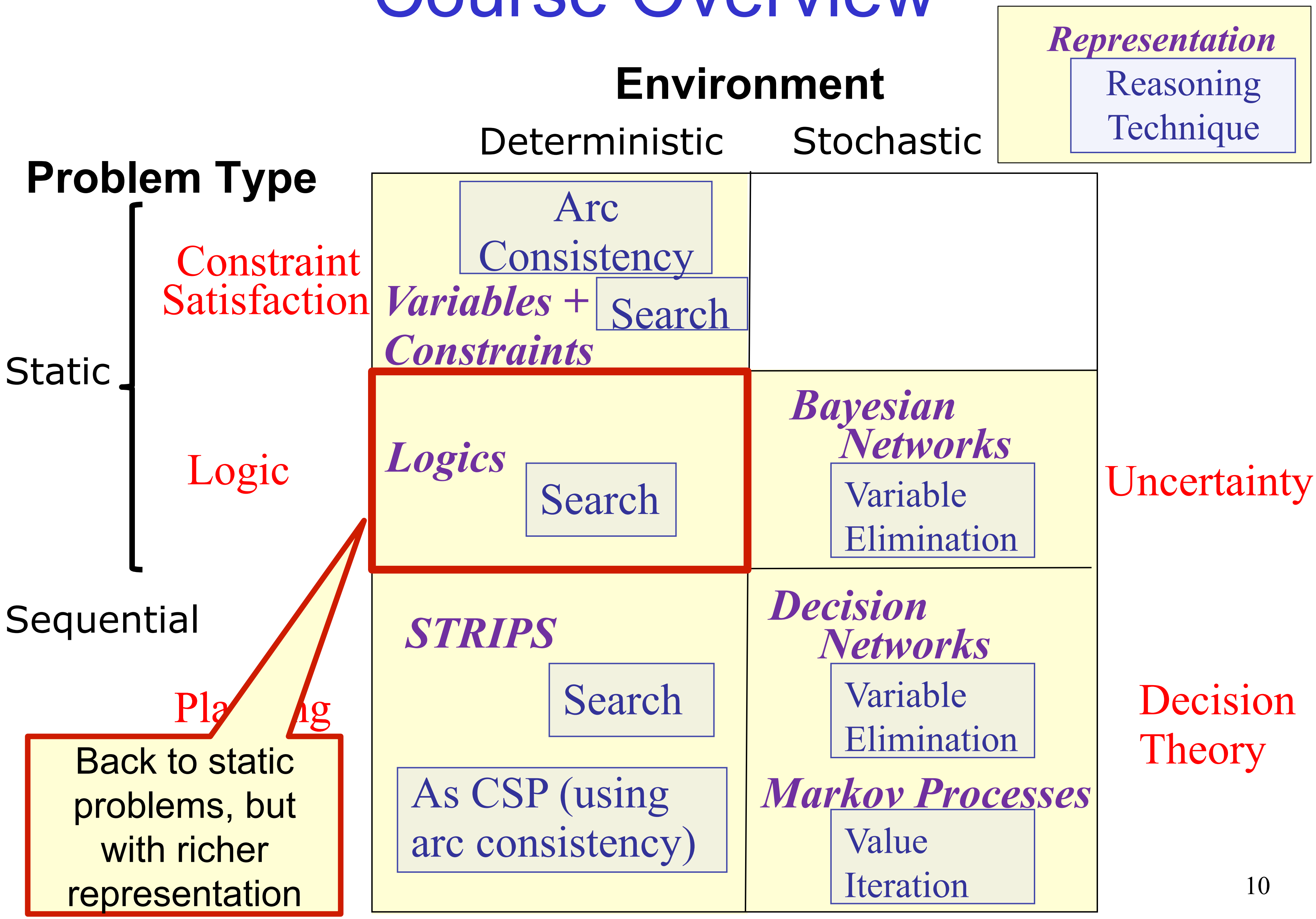
Learning Goals for Planning

- STRIPS
  - Represent a planning problem with the STRIPS representation
  - Explain the STRIPS assumption
- Forward planning
  - Solve a planning problem by search (forward planning). Specify states, successor function, goal test and solution.
  - Construct and justify a heuristic function for forward planning
- CSP planning
  - Translate a planning problem represented in STRIPS into a corresponding CSP problem (and vice versa)
  - Solve a planning problem with CSP by expanding the horizon

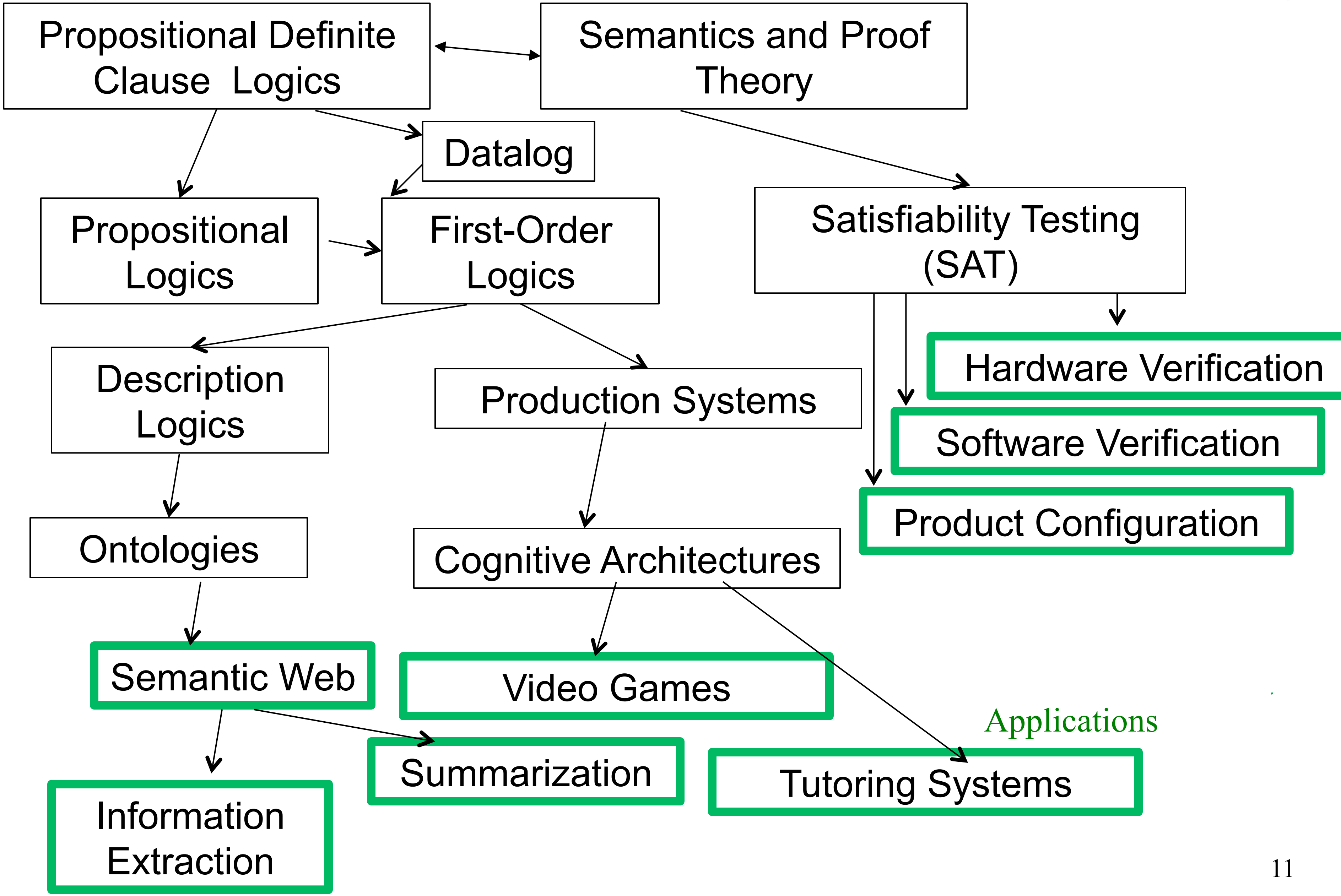
Lecture Overview

- Recap: CSP planning
- Intro to Logic
- Propositional Definite Clause Logic: Syntax
- Propositional Definite Clause Logic: Semantics

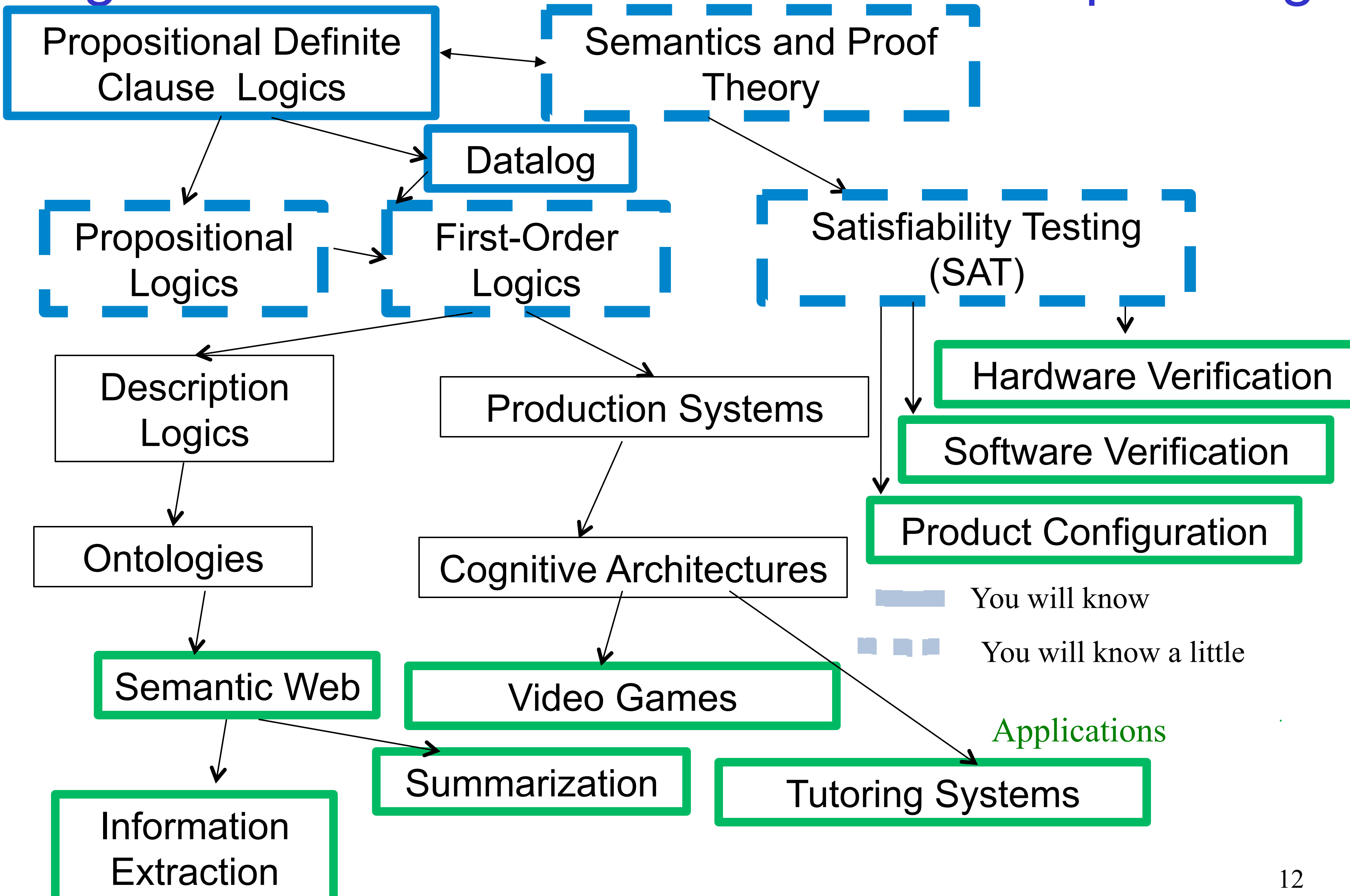
Course Overview



Logics in AI: Similar slide to the one for planning



Logics in AI: Similar slide to the one for planning



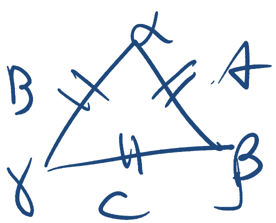
What you already know about logic...

- **From programming: Some logical operators**

- `If ((amount > 0) && (amount < 1000)) || !(age < 30)`
- `...`

You know what they mean in a "procedural" way

Logic is the language of mathematics. To define formal structures (e.g., sets, graphs) and to prove statements about them



$$\forall(x)\, triangle(x) \longrightarrow [A = B = C \longleftrightarrow \alpha = \beta = \gamma]$$

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

Logic: a framework for representation & reasoning

- 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
  - Relationships between objects
- Logic is the language to express knowledge about the world this way
- http://en.wikipedia.org/wiki/John_McCarthy (1927 - 2011)

  Logic and AI, The Advice Taker, LISP, situation calculus,…

  Coined "Artificial Intelligence". Dartmouth W'shop (1956)

Why Logics?

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

- *E.g. "Every 322 student who works hard passes the course"*
  - *student(s) ∧ registered(s, c) ∧ course_name(c, 322) ∧ works_hard(s) ⇒ passes(s,c)*

  - *student(sam)*
  - *registered(sam, c1)*
  - *course_name(c1, 322)*

  - *Query: passes(sam, c1) ?*

- 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:

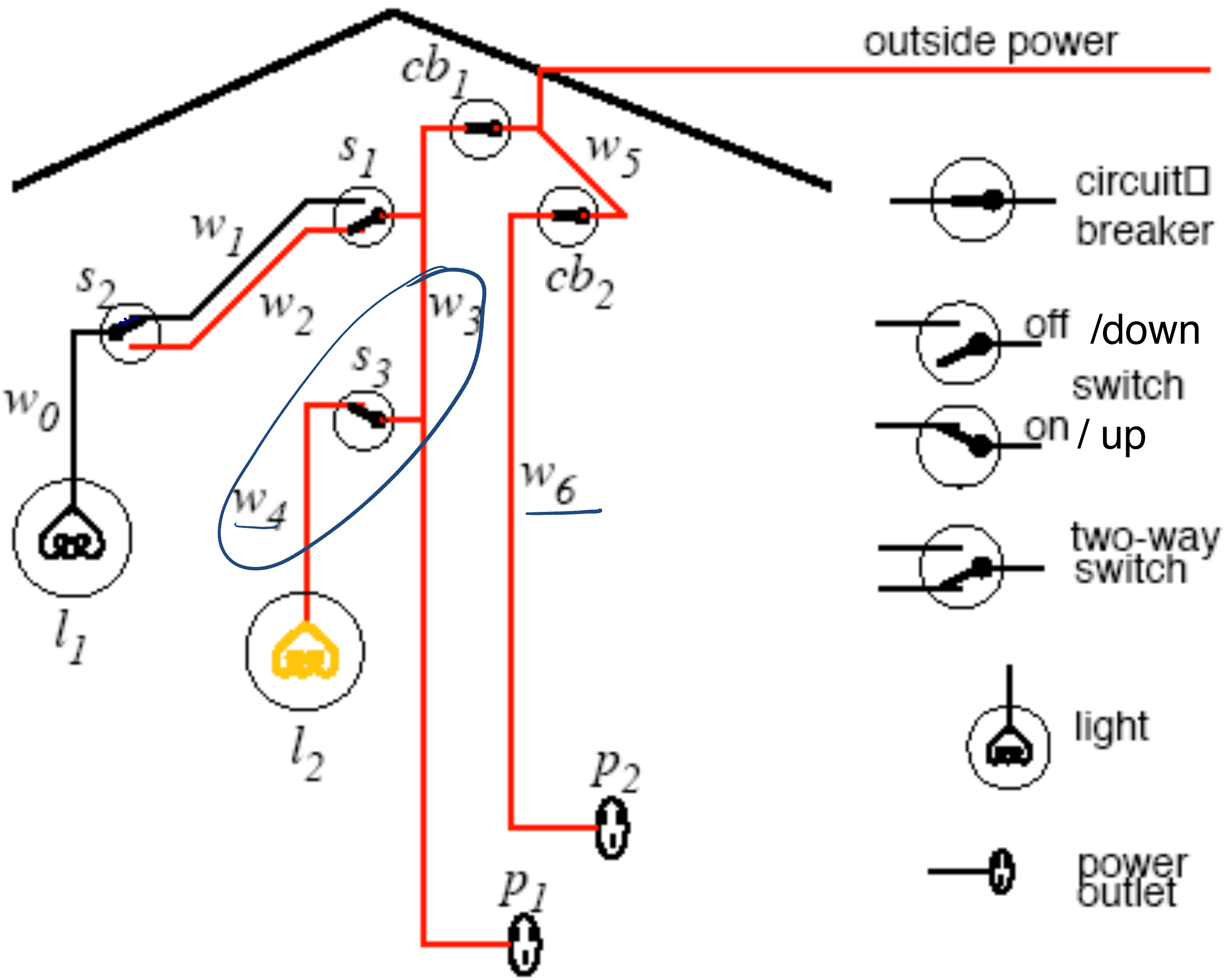
- syntax: specifies the symbols used, and how they can be combined to form legal sentences
- semantics: specifies the meaning of the symbols
- reasoning theory or proof procedure: a (possibly nondeterministic) specification of how an answer can be produced.

- We have seen several representations and reasoning procedures:
  - State space graph + search
  - CSP + search/arc consistency
  - STRIPS + search/arc consistency

Using a Representation and Reasoning System

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: Electrical Circuit



$light\_l_1.$

$light\_l_2.$

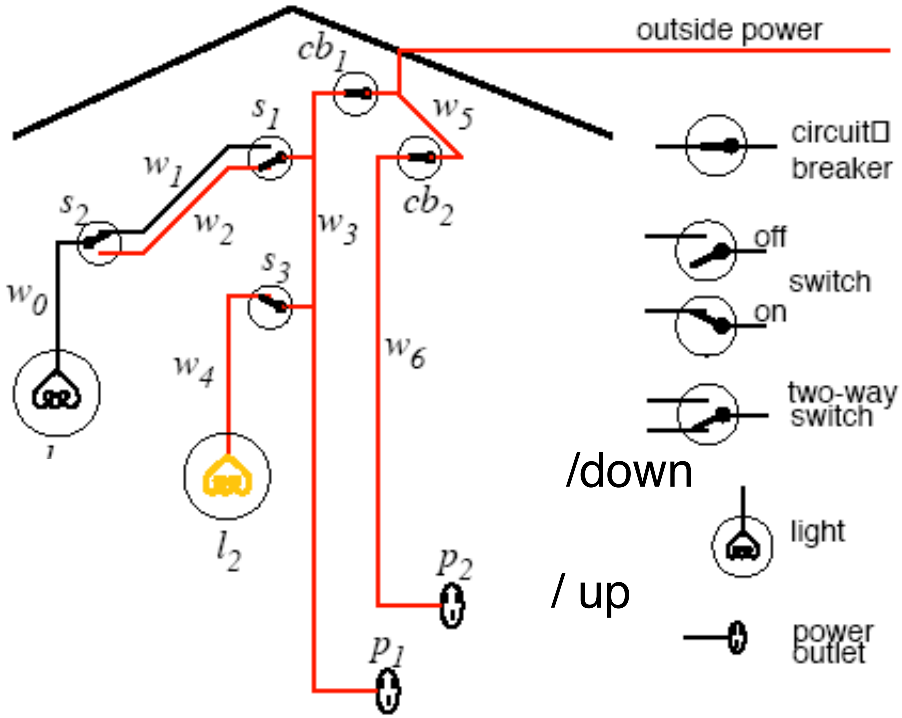
$ok\_l_1.$

$ok\_l_2.$

$ok\_cb_1.$

$ok\_cb_2.$

$live\_outside.$



$live\_l_1 \leftarrow live\_w_0.$

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

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

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

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

$live\_l_2 \leftarrow live\_w_4.$

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

$live\_p_1 \leftarrow live\_w_3.$

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

$live\_p_2 \leftarrow live\_w_6.$

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

$live\_w_5 \leftarrow live\_outside.$

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

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

Propositional Definite Clauses

- A simple representation and reasoning system
- Two kinds of statements:
  - that a proposition is true
  - that a proposition is true if one or more other propositions are true
- Why only propositions?
  - We can exploit the Boolean nature for efficient reasoning
  - Starting point for more complex logics
- To define this RRS, we'll need to specify:
  - syntax
  - semantics
  - proof procedure

Lecture Overview

- Recap: CSP planning
- Intro to Logic
- Propositional Definite Clause (PDC) Logic: Syntax
- Propositional Definite Clause (PDC) Logic: Semantics

Propositional Definite Clauses: Syntax

Definition (atom)

Examples: p_1, $live\_l_1$

An **atom** is a symbol starting with a lower case letter

Definition (body)

A **body** is an atom or is of the form $b_1 \wedge b_2$ where b_1 and b_2 are bodies.

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

Definition (definite clause)

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

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

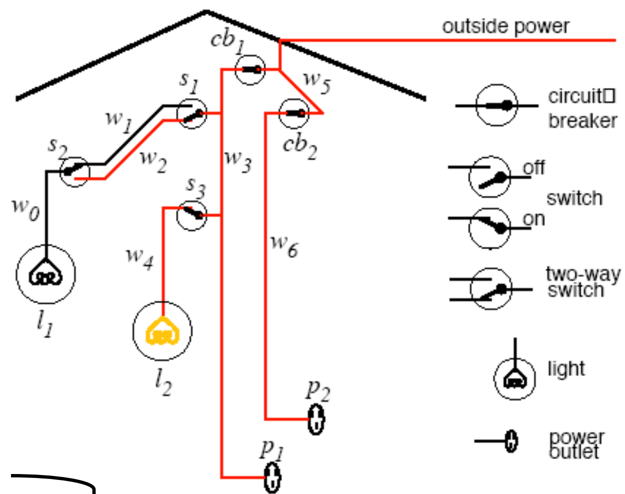
Definition (KB)

Example: $\{p_1 \leftarrow p_2, live\_w_0 \leftarrow live\_w_1 \wedge up\_s_2\}$

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

$light\_l_1.$
$light\_l_2.$
$ok\_l_1.$
$ok\_l_2.$
$ok\_cb_1.$
$ok\_cb_2.$
$live\_outside.$

atoms



definite clauses, KB

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

rules

PDC Syntax: more examples

Definition (definite clause)

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

Legal PDC clause | Not a legal PDC clause

a) *ai_is_fun*

b) *ai_is_fun* ∨ *ai_is_boring*

c) *ai_is_fun* ← *learn_useful_techniques*

d) *ai_is_fun* ← *learn_useful_techniques* ∧ *notTooMuch_work*

e) *ai_is_fun* ← *learn_useful_techniques* ∧ ¬ *TooMuch_work*

f) *ai_is_fun* ← *f(time_spent, material_learned)*

g) *srtsyj* ← *errt* ∧ *gffdgdgd*

PDC Syntax: more examples

Legal PDC clause

Not a legal PDC clause

a) *ai_is_fun*

b) *ai_is_fun* $\vee$ *ai_is_boring*

c) *ai_is_fun* $\leftarrow$ *learn_useful_techniques*

d) *ai_is_fun* $\leftarrow$ *learn_useful_techniques* $\wedge$ *notTooMuch_work*

e) *ai_is_fun* $\leftarrow$ *learn_useful_techniques* $\wedge$ $\neg$ *TooMuch_work*

f) *ai_is_fun* $\leftarrow$ *f(time_spent, material_learned)*

g) *srtsyj* $\leftarrow$ *errt* $\wedge$ *gffdgdgd*

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

Lecture Overview

- Recap: CSP planning
- Intro to Logic
- Propositional Definite Clause (PDC) Logic: Syntax
- Propositional Definite Clause (PDC) Logic: Semantics

Propositional Definite Clauses: Semantics

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

Definition (interpretation)

An **interpretation I** assigns a truth value to each atom.

- If our domain has 5 atoms, how many interpretations are there?

$5+2$ $5*2$ 5^2 2^5

Propositional Definite Clauses: Semantics

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

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

- If our domain has 5 atoms, how many interpretations are there?
  - 2 values for each atom, so 2^5 combinations
  - Similar to possible worlds in CSPs

Propositional Definite Clauses: Semantics

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

Definition (interpretation)

An **interpretation I** assigns a truth value to each atom.

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

Definition (truth values of statements)

- A **body $b_1 \land b_2$** is true in I if and only if b_1 is true in I and b_2 is true in I.
- A **rule $h \leftarrow b$** is false in I if and only if b is true in I and h is false in I.

PDC Semantics: Example

Truth values under different interpretations

F=false, T=true

	a_1	a_2	$a_1 \wedge a_2$
I_1	F	F	F
I_2	F	T	F
I_3	T	F	F
I_4	T	T	T

	h	b	$h \leftarrow b$			
I_1	F	F	F	T	T	T
I_2	F	T	F	F	F	T
I_3	T	F	T	F	T	F
I_4	T	T	T	T	T	T

PDC Semantics: Example

Truth values under different interpretations

F=false, T=true

	h	b	$h \leftarrow b$
I_1	F	F	T
I_2	F	T	F
I_3	T	F	T
I_4	T	T	T

$h \leftarrow b$ is only false
if b is true and h is false

	h	a_1	a_2	$h \leftarrow a_1 \wedge a_2$			
I_1	F	F	F	F	T	T	T
I_2	F	F	T	T	T	T	T
I_3	F	T	F	T	T	T	F
I_4	F	T	T	T	F	F	T
I_5	T	F	F	F	T	T	T
I_6	T	F	T	F	T	T	T
I_7	T	T	F	T	T	T	F
I_8	T	T	T	T	T	F	T

PDC Semantics: Example for truth values

Truth values under different interpretations

F=false, T=true

	h	b	$h \leftarrow b$
I_1	F	F	T
I_2	F	T	F
I_3	T	F	T
I_4	T	T	T

$h \leftarrow a_1 \land a_2$

Body of the clause: $a_1 \land a_2$

Body is only true if both a_1 and a_2 are true in I

	h	a_1	a_2	$h \leftarrow a_1 \land a_2$
I_1	F	F	F	T
I_2	F	F	T	T
I_3	F	T	F	T
I_4	F	T	T	F
I_5	T	F	F	T
I_6	T	F	T	T
I_7	T	T	F	T
I_8	T	T	T	T

Propositional Definite Clauses: Semantics

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

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

We can use the interpretation to determine the truth value of clauses and knowledge bases:

> **Definition (truth values of statements)**
>
> - A **body $b_1 \wedge b_2$** is true in I if and only if b_1 is true in I and b_2 is true in I.
> - A **rule $h \leftarrow b$** is false in I if and only if b is true in I and h is false in I.
> - A **knowledge base KB** is true in I if and only if every clause in KB is true in I.

Propositional Definite Clauses: Semantics

Definition (interpretation)

An **interpretation I** assigns a truth value to each atom.

Definition (truth values of statements)

- A **body $b_1 \land b_2$** is true in I if and only if b_1 is true in I and b_2 is true in I.
- A **rule $h \leftarrow b$** is false in I if and only if b is true in I and h is false in I.
- A **knowledge base KB** is true in I if and only if every clause in KB is true in I.

Definition (model)

A **model** of a knowledge base KB is an interpretation in which KB is true.

Similar to CSPs: a model of a set of clauses is an interpretation that makes all of the clauses true

PDC Semantics: Example for models

Definition (model)
A **model** of a knowledge base KB is an interpretation in which every clause in KB is true.

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

Which of the interpretations below are models of KB?

I_1 , I_3 | I_1, I_3, I_4 | All of them | I_3

	p	q	r	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	F	T	F	T

PDC Semantics: Example for models

Definition (model)
A **model** of a knowledge base KB is an interpretation in which every clause in KB is true.

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

Which of the interpretations below are models of KB?

I_1 , I_3 | I_1, I_3, I_4 | All of them | I_3

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

PDC Semantics: Example for models

Definition (model)
A **model** of a knowledge base KB is an interpretation in which every clause in KB is true.

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

Which of the interpretations below are models of KB?
All interpretations where KB is true: I_1, I_3, and I_4

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

Next class

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