Lecture 15

Logic Intro and PDCL

Lecture Overview

- Intro to Logic
- Propositional Definite Clauses:
- Syntax
- Semantics
- Proof procedures (time permitting)

Where Are We?

Environment

Representation

Problem Type

Deterministic Arc Stochastic

ReasoningTechnique

Consistency

Constraint Satisfaction ***VarsConstraints+*** Search

Static ***Belief Nets***

Query ***Logics***

Variable

Search

Elimination

Sequential ***STRIPS*** ***Decision Nets*** Variable

Planning Search Elimination

First Part of ***Markov Processes*** Value the Course

Iteration

Where Are We?

Environment ***Representation***

Problem Type

Deterministic Arc Stochastic

Reasoning Technique

Consistency

Constraint Satisfaction ***Vars Constraints+***

Search

Static

Belief Nets

Query

Logics

Variable

Search

Elimination

Sequential

STRIPS

Decision Nets Variable

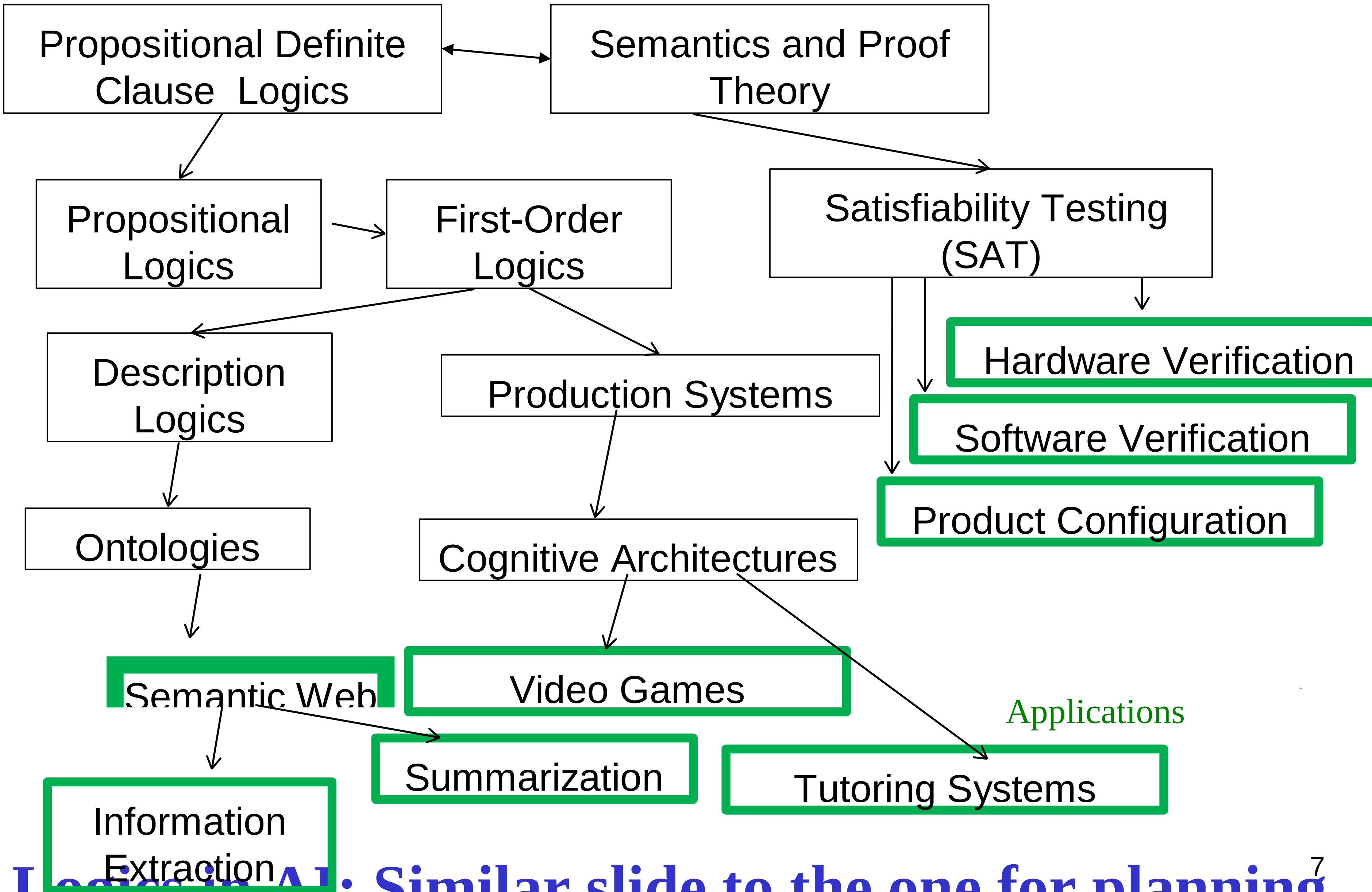
Planning

Search

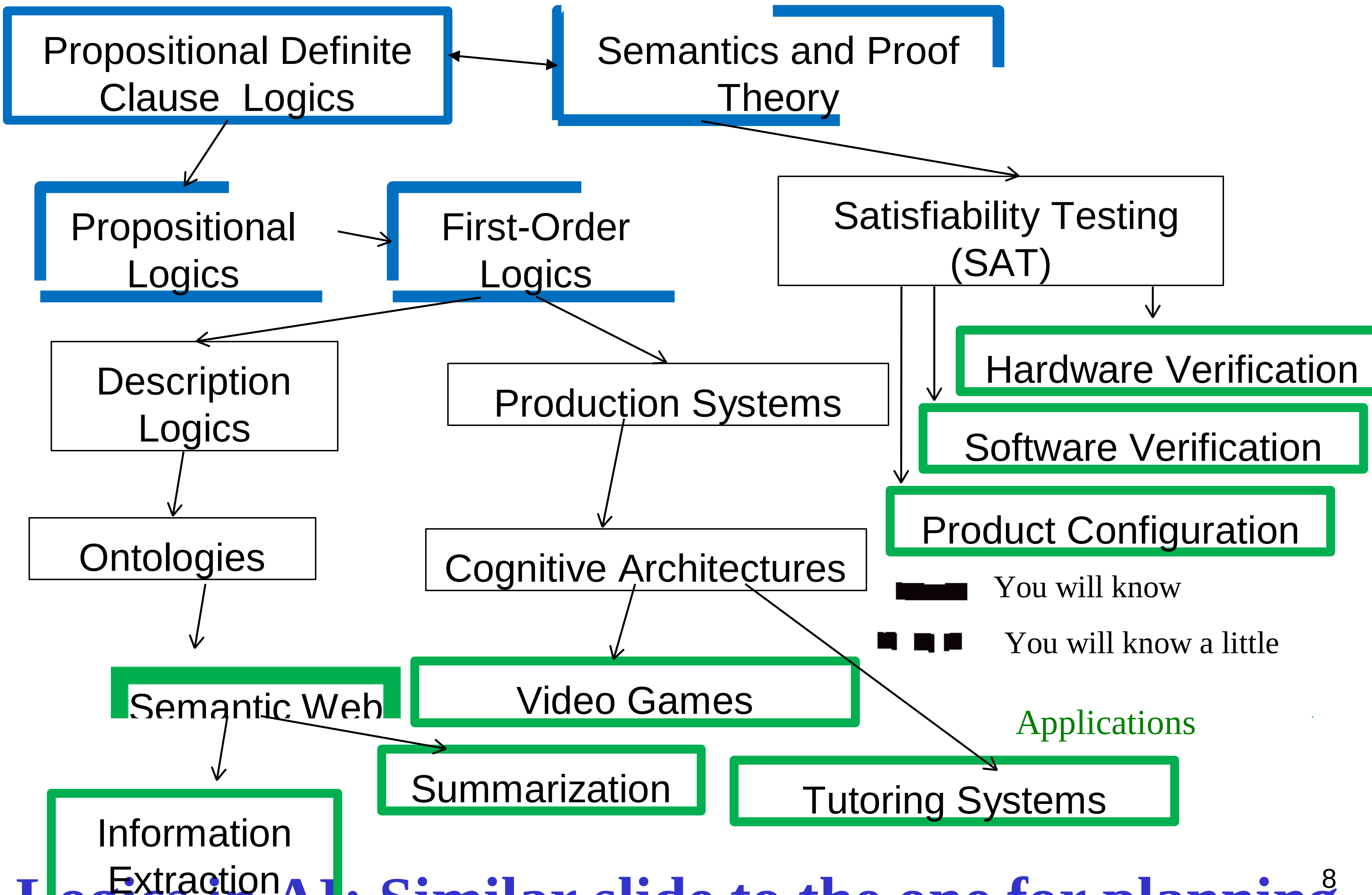
Elimination

Back to static problems, but with *Markov Processes* richer representation

Value Iteration



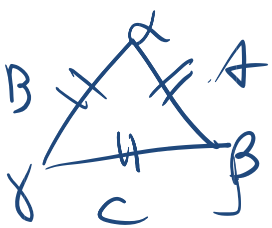
Logics in AI: Similar slide to the one for planning



Logics in AI: Similar slide to the one for planning

What you already know about logic...

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



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

- From programming: Some logical operators

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

  You know what they mean in a “procedural” way

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….

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”

    Coined “Artificial Intelligence”. Dartmouth W’shop (1956)

Why Logics?

- "Natural" way to express knowledge about the world

  e.g. "Every 101 student will pass the course"
  Course (c1)
  Name-of (c1, 101)

  $$\forall (z) student(z) \& registered(z,c1) \rightarrow will\_pass(z,c1)$$

- 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

General problem: Query answering

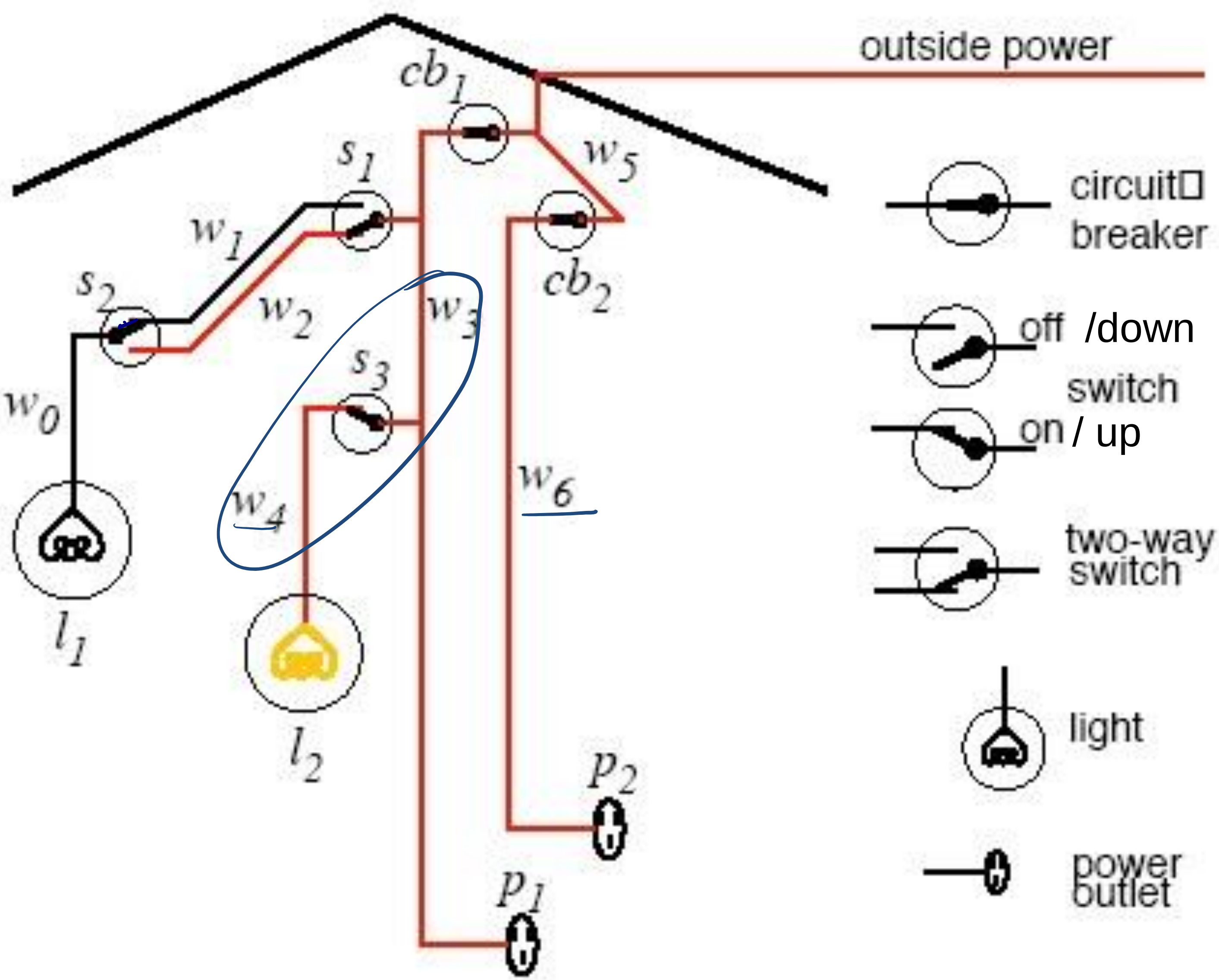
- tell the computer how the world works
- tell the computer some facts about the world
- ask a yes/no question about whether other facts must be true

Solving it with Logic

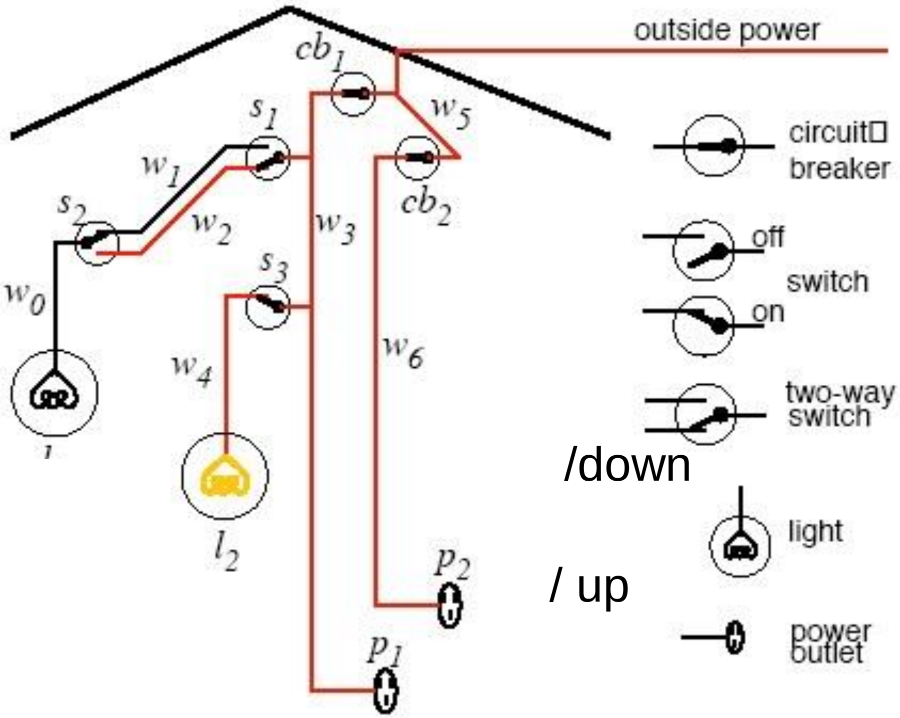
1. Begin with a task domain.
2. Distinguish those things you want to talk about (the ontology)

3. Choose symbols in the computer to denote elements of your ontology
4. Tell the system knowledge about the domain

Example: Electrical Circuit


outside power
cb1
w5
s1
w1
s2
w2
w3
cb2
s3
w0
w4
w6
l1
l2
p2
p1
circuit
breaker
off /down
switch
on / up
two-way
switch
light
power
outlet

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



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

Logic: A general framework for reasoning

General problem: Query answering

- 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

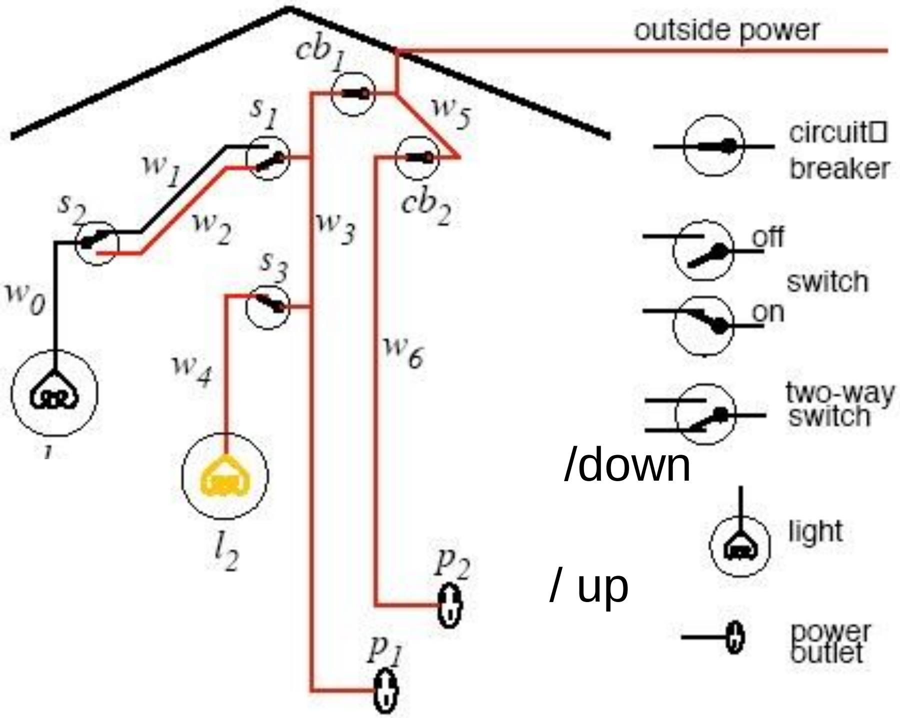
Solving it with Logic

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 elements of your ontology
4. Tell the system knowledge about the domain

5. Ask the system whether new statements about the domain are true or false

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

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



$live\_w_4$?
lit_l_2?

To Define a Logic We Need

- Syntax: specifies the symbols used, and how they can be combined to form legal sentences
- Knowledge base is a set of sentences in the language
- Semantics: specifies the meaning of symbols and sentences
- Reasoning theory or proof procedure: a specification of how an answer can be produced.

- Sound: only generates correct answers with respect to the semantics
- Complete: Guaranteed to find an answer if it exists 17

Propositional Definite Clauses

We will start with a simple logic

- Primitive elements are propositions: Boolean variables that can be {true, false}

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

We need to specify: syntax, semantics, proof procedure

Lecture Overview

- Intro to Logic
- Propositional Definite Clauses:
- Syntax

- Semantics
- Proof Procedures

To Define a Logic We Need

- Syntax: specifies the symbols used, and how they can be combined to form legal sentences
  - Knowledge base is a set of sentences in the language
- Semantics: specifies the meaning of symbols and sentences
- Reasoning theory or proof procedure: a specification of how an answer can be produced.

- Sound: only generates correct answers with respect to the semantics

- Complete: Guaranteed to find an answer if it exists 20

Propositional Definite Clauses: Syntax

Definition (atom)

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

Examples: p_1; $live\_l_1$

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)

A **definite clause** is

- an atom or
- a rule of the form ***h←b*** where *h* is an atom ("head") and *b* is a body. (Read this as "*h* if *b*".)

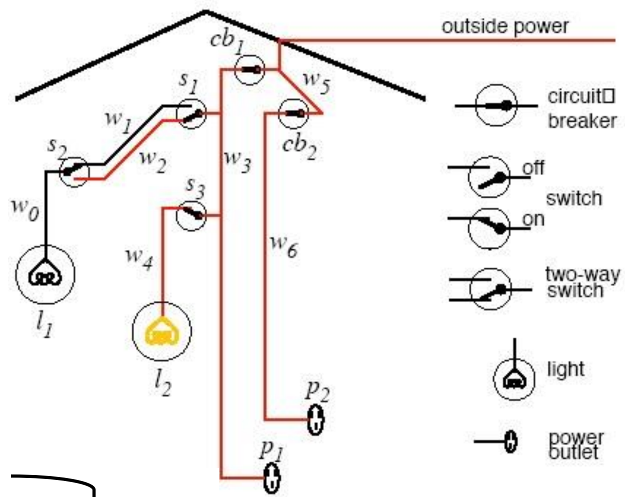
Examples: $p_1 \leftarrow p_2$; $live\_w_0 \leftarrow live\_w_1 \land up\_s_2$

Definition (KB)

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

PDCL Syntax: more examples

Definition (definite clause)

A **definite clause** is

- an atom or
- 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.')

How many of the clauses below are legal PDCL clauses?

a) $Sunny\_today$

b) $sunny\_today \vee cloudy\_today$

c) $vdjhsaekwrq$

d) $high\_pressure\_system \leftarrow sunny\text{-}today$

e) $sunny\_today \leftarrow high\_pressure\_system \wedge summer$

f) $sunny\_today \leftarrow high\_pressure\text{-}system \ \land \neg \ winter$

g) $ai\_is\_fun \leftarrow f(time\_spent, material\_learned)$

h) $summer \leftarrow sunny\_today \land high\_pressure\_system$

PDCL Syntax: more examples

Definition (definite clause)

A **definite clause** is

- an atom or
- 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.')

How many of the clauses below are legal PDCL clauses?

a) $Sunny\_today$

b) $sunny\_today \lor cloudy\_today$

c) *vdjhsaekwrq* A. 3

d) *high_pressure_system* ← *sunny-today* B. 4

e) *sunny_today* ← *high_pressure_system*∧ *summer*

C. 5

f) *sunny_today* ← *high_pressure*-*system* ∧¬ *winter*

g) *ai_is_fun* ← *f(time_spent, material_learned)* D. 6

h) *summer* ← *sunny_today* ∧*high_pressure_system*

PDC Syntax: more examples

Legal PDC clause | Not a legal PDC clause

a) *Sunny_today*

b) *sunny_today∨cloudy_today*

c) *vdjhsaekwrq*

d) *high_pressure_system ← sunny-today* B. 4

e) *sunny_today ← high_pressure_system∧ summer*

f) *sunny_today ← high_pressure-system ∧¬ winter*

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

h) *summer ← sunny_today ∧high_pressure_system*

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

To Define a Logic We Need

- Syntax: specifies the symbols used, and how they can be combined to form legal sentences
- Knowledge base is a set of sentences in the language
- Semantics: specifies the meaning of symbols and sentences
- Reasoning theory or proof procedure: a specification of how an answer can be produced.

Propositional Definite Clauses: Semantics

- Sound: only generates correct answers with respect to the semantics
- Complete: Guaranteed to find an answer if it exists 26

Lecture Overview

- Intro to Logic
- Propositional Definite Clauses:
- Syntax

- Semantics

- Proof Procedures
- Semantics allows one to relate the symbols in the logic to the domain to be modeled.

Definition (interpretation)

An interpretation I assigns a truth value to each atom.

- If our domain has 8 atoms, how many interpretations are there?

Propositional Definite Clauses: Semantics

A. $8+2$

B. $8*2$

C. 8^2

D. 2^8

Propositional Definite Clauses: Semantics

- Semantics allows one to relate the symbols in the logic to the domain to be modeled.

Definition (interpretation)

An interpretation I assigns a truth value to each atom.

- If our domain has 8 atoms, how many interpretations are there?
- 2 values for each atom, so 2^8 combinations

Propositional Definite Clauses: Semantics

- Similar to possible worlds in CSPs
- Semantics allows one to relate the symbols in the logic to the domain to be modeled.

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 $b_1 \wedge b_2$ 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.

PDC Semantics: Example

Truth values under different interpretations

F=false, T=true

a_1	a_2	$a_1 \wedge a_2$	h	b	$h \leftarrow b$

Propositional Definite Clauses: Semantics

I_1	F	F	F
I_2	F	T	F
I_3	T	F	F
I_4	T	T	T

I_1	F	F	T
I_2	F	T	F
I_3	T	F	T
I_4	T	T	T

PDC Semantics: Example

Truth values under different interpretations

F=false, T=true

	h	b	h ← b
I1	F	F	T
I2	F	T	F
I3	T	F	T
I4	T	T	T

h ← b is false only when b is true and h is false

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

$h \leftarrow a_1 \wedge a_2$ Body of the clause is $a_1 \wedge a_2$ Body is true only if both a_1 and a_2 are true in I

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 $b_1 \wedge b_2$** 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 $b_1 \wedge b_2$** 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.

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: Knowledge Base (KB)

r

$s \leftarrow p\ q$

$\leftarrow p \wedge s$

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

	p	q	r	s
I_1	false	true	true	false

A. KB_1 **B. KB_3** **C. KB_2**

p
r
$s \leftarrow q \wedge p$

$r\ q\ s$
$\leftarrow q$

Which of the three KB above is True in I_1

PDC Semantics: Knowledge Base (KB)

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

	p	q	r	s
I_1	false	true	true	false

C. KB_2

r

$s \leftarrow p\ q$

$\leftarrow p \wedge s$

Which of the three KB above is True in I_1

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.

$$KB = \left\{ \begin{array}{l} p \leftarrow q \\ s \\ s \leftarrow r \end{array} \right.$$

Which of the interpretations below are models of KB?

A. I_1

	p	q	r	s
I_1	T	T	T	T
I_2	F	F	F	T
I_3	T	T	F	F
I_4	T	T	T	F
I_5	T	T	F	T

B. I_1 , I_2

C. I_1, I_2, I_5

D. All of them

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 = { s

Which of the interpretations below are models of KB?
All interpretations where KB is true

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

$p \leftarrow q$

$s \leftarrow r$

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 = { s, $s \leftarrow r$, $p \leftarrow q$ }

All interpretations where KB is true: I_1, I_2, and I_5

C. I_1, I_2, I_5

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

Which of the interpretations below are models of KB?

OK but….

…. Who cares? Where are we going with this?

Remember what we want to do with Logic

1) Tell the system knowledge about a task domain.
   - This is your KB
   - which expresses true statements about the world

2) Ask the system whether new statements about the domain are true or false.
   - We want the system responses to be
   - Sound: only generates correct answers with respect to the semantics

- Complete: Guaranteed to find an answer if it exists

For Instance,…

1) Tell the system knowledge about a task domain.

$$KB = \left\{ \begin{array}{l} p \leftarrow q. \\ q. \\ r \leftarrow s. \end{array} \right.$$

2) Ask the system whether new statements about the domain are true or false

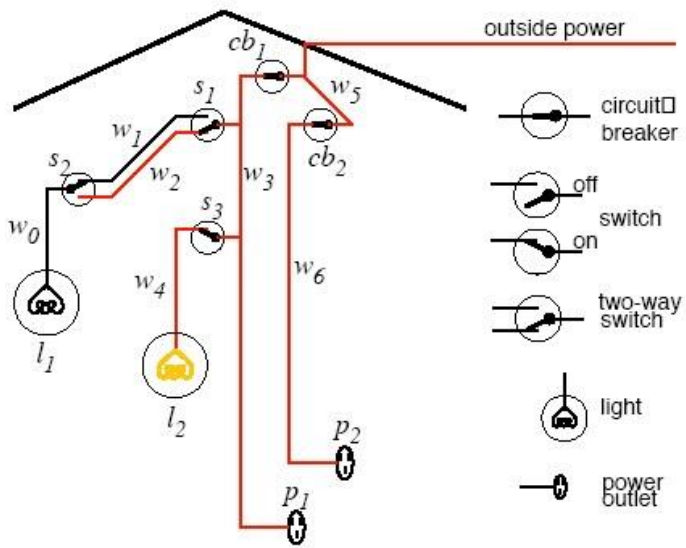
p? r?
s?

Or, More Interestingly

1) Tell the system knowledge about a task domain.

light_l$_1$.
light_l$_2$.
ok_l$_1$.
ok_l$_2$.
ok_cb$_1$.
ok_cb$_2$.
live_outsic

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



2) Ask the system whether new statements about the domain are true or false

- live_w$_4$?

lit_l_2?

To Obtain This We Need One More Definition

To Obtain This We Need One More Definition

Definition (logical consequence)

If KB is a set of clauses and G is a conjunction of atoms, G is a logical consequence of KB, written KB ⊧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 ⊧Gif there is no interpretation in which KB is true and G is false.
- when KB is TRUE, then G must be TRUE

- We want a reasoning procedure that can find all and only the logical consequences of a knowledge base

Example of Logic Entailment

KB=□□*p*←*q*∧*r*.

□*q*.

How many models are there?

KB=□□*p*←*q*∧*r*.

□*q*.

A. 1B. 2 C. 3   D. 4

E. 5

Ex Entailment

interpretations

r	q	p
T	T	T
T	T	F
T	F	T
T	F	F
F	T	T
F	T	F
F	F	T
F	F	F

How many models are there?

A. 1 B. 2 C. 3

D. 4

E. 5

Example of Logic Entailment

$$KB = \left\{ \begin{array}{l} p \leftarrow q \wedge r. \\ q. \end{array} \right.$$

Interpretations

r	q	p
T	T	T
T	T	F
T	F	T
T	F	F
F	T	T
F	T	F
F	F	T
F	F	F

Models

C. 3

Example of Logic Entailment

Interpretations

• We want a reasoning procedure that can find all and only the

Example of Logic Entailment

Interpretations

logical consequences of a knowledge base

Which atoms are logically entailed?

Example of Logic Entailment

$$KB = \left\{ \begin{array}{l} p \leftarrow q \wedge r. \\ q. \end{array} \right.$$

Interpretations

r	q	p
T	T	T
T	T	F
T	F	T
T	F	F
F	T	T
F	T	F
F	F	T
F	F	F

Models

C. 3

Which atoms are logically entailed?

Example: Logical Consequences

q

- We want a reasoning procedure that can find all and only the logical consequences of a knowledge base

Example: Logical Consequences

	p	q	r	s
I_1	true	true	true	true
I_2	true	true	true	false
I_3	true	true	false	false
I_4	true	true	false	true
I_5	false	true	true	true
I_6	false	true	true	false
I_7	false	true	false	false
I_8	false	true	false	true
I_9	true	false	true	true
I_{10}	true	false	true	false
I_{11}	true	false	false	false

Example: Logical Consequences

I_{12}	true false false false
.....	....

$p \leftarrow q.\ KB =$

$q.$

$r \leftarrow s.$

D. KB ⊨r, KB ⊨s,

Example: Logical Consequences

Which of the following is true?

A. KB $\models q$ and KB $\models r$

B. KB $\models q$, and KB $\models s$

C. KB $\models q$, and KB $\models p$

E. None of the above

Example: Logical Consequences

	p	q	r	s
I_1	true	true	true	true
I_2	true	true	true	false
I_3	true	true	false	false
I_4	true	true	false	true
I_5	false	true	true	true
I_6	false	true	true	false
I_7	false	true	false	false
I_8	false	true	false	true
I_9	true	false	true	true

Example: Logical Consequences

I_{10}	true	false	true	false
I_{11}	true	false	false	false
I_{12}	true	false	false	false
…..	……	…..	……	…….

$KB =$ □ □$p \leftarrow q.$ □ $q.$ □ □$r \leftarrow s.$

Which of the following is true?

- $KB \models q$,
- $KB \models p$,
- $KB \models s$,

Example: Logical Consequences

- KB $\models$ r

Example: Logical Consequences

	p	q	r	s
I_1	true	true	true	true
I_2	true	true	true	false
I_3	true	true	false	false
I_4	true	true	false	true
I_5	false	true	true	true
I_6	false	true	true	false
I_7	false	true	false	false
I_8	false	true	false	true
I_9	true	false	true	true

I_{10}	true	false	true	false
I_{11}	true	false	false	false
I_{12}	true	false	false	false
…..	……	…..	……	…….

$$KB = \left\{ \begin{array}{l} p \leftarrow q. \\ q. \\ r \leftarrow s. \end{array} \right.$$

models

Which of the following is true?

- KB ⊧ q,
- KB ⊧ p,
- KB ⊧ s,

Example: Logical Consequences

- KB $\models$ r

	p	q	r	s
I_1	true	true	true	true
I_2	true	true	true	false
I_3	true	true	false	false
I_4	true	true	false	true
I_5	false	true	true	true
I_6	false	true	true	false
I_7	false	true	false	false
I_8	false	true	false	true
I_9	true	false	true	true
I_{10}	true	false	true	false

Example: Logical Consequences

I_{11}	true	false	false	false
I_{12}	true	false	false	false
…..	……	…..	……	…….

models

$KB=$ □ □□□ $r\leftarrow q.\ s.$ □ $p\leftarrow q.$

Which of the following is true?

- KB ⊧q, T
- KB ⊧p, T
- KB ⊧s, F

•KB ⊨r F

C. KB ⊨q, and KB ⊨p

User's View of Semantics

- Choose a task domain: intended interpretation.
- For each proposition you want to represent, associate a proposition symbol in the language.
- Tell the system clauses that are true in the intended interpretation: axiomatize the domain.
- Ask questions about the intended interpretation.

Example: Logical Consequences

- If KB |= g , then g must be true in the intended interpretation.

Computer's View of Semantics

- The computer doesn't have access to the intended interpretation.
- All it knows is the knowledge base.
- The computer can determine if a formula is a logical consequence of KB.
- If KB |= g then g must be true in the intended interpretation.
- Otherwise, there is a model of KB in which g is false.

Computer's View of Semantics

This could be the intended interpretation.

The computer wouldn't know!

- Otherwise, there is a model of KB in which g is false. This could be the intended interpretation.
  The computer wouldn't know

	p	q	r	s
I_1	true	true	true	true
I_2	true	true	true	false

$$KB = \left\{ p \leftarrow q.\ \ q.\ \ r \leftarrow s. \right\}$$

Computer’s View of Semantics

- Otherwise, there is a model of KB in which g is false. This could be the intended interpretation. The computer wouldn't know

	p	q	r	s
I_1	true	true	true	true
I_2	true	true	true	false

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

Computer's View of Semantics

I_1 and I_2 above are both models for KB, each
could be the
intended interpretation. The computer cannot know, thus it cannot say anything about the truth value of s

Learning Goals for Logic Up To Here

- PDCL syntax & semantics
  - Verify whether a logical statement belongs to the language of propositional definite clauses
  - Verify whether an interpretation is a model of a PDCL KB.
    - Verify when a conjunction of atoms is a logical consequence of a knowledge base

Next: Proof Procedures (5.2.2)

Lecture Overview

- Intro to Logic
- Propositional Definite Clauses:
- Syntax
- Semantics

Proof Procedures

To Define a Logic We Need

- Syntax: specifies the symbols used, and how they can be combined to form legal sentences
- Knowledge base is a set of sentences in the language

Reasoning theory or proof procedure: a specification of how an answer can be produced (sound and complete)

- Semantics: specifies the meaning of symbols and sentences
-

- Bottom-up and Top-Down Proof Procedure for Finding Logical Consequence

Proof Procedures

- A proof procedure is a mechanically derivable demonstration that a formula logically follows from a knowledge base.

- Given a proof procedure P, KB $\vdash_P$ g means g can be derived from knowledge base KB with the proof procedure.
- If I tell you I have a proof procedure for PDCL • What do I need to show you in order for you to trust my procedure?

That is sound and complete

Soundness and Completeness

Definition (soundness)

A proof procedure P is sound if KB $\vdash_P$ g implies KB $\models$ g.

sound: every atom derived by P follows logically from KB (i.e. is true in every model)

- Soundness of proof procedure P: need to prove that

If g can be derived by the procedure (KB $\vdash_P$ g) then g is true in all models of KB (KB $\models$ g)

Definition (completeness)

A proof procedure P is complete if KB $\models$ g implies KB $\vdash_P$ g.

complete: every atom that logically follows from KB is derived by P

- Completeness of proof procedure P: need to prove that

  If g is true in all models of KB ($KB \models g$) then
  g is derived by the procedure ($KB \vdash_P g$)

Simple Proof Procedure

Simple proof procedure S

- Enumerate all interpretations
- For each interpretation I, check whether it is a model of KB
  - ✓i.e., check whether all clauses in KB are true in I
- KB $\vdash_S$ g if g holds in all such models

Simple proof procedure S

- Enumerate all interpretations
- For each interpretation I, check whether it is a model of KB
  - ✓i.e., check whether all clauses in KB are true in I
- KB $\vdash_S$ g if g holds in all such models

Simple Proof Procedure

Simple proof procedure S

- Enumerate all interpretations
- For each interpretation I, check whether it is a model of KB
  - ✓i.e., check whether all clauses in KB are true in I
- KB $\vdash_S$ g if g holds in all such models

problem with this approach? problem with this approach?

- If there are n propositions in the KB, must check all the interpretations!

problem with this approach?

Simple Proof Procedure

- If there are n propositions in the KB, must check all the 2^n interpretations!

Goal of proof theory

- find sound and complete proof procedures that allow us to prove that a logical formula follows from a KB avoiding to do the above

Bottom-up proof procedure

- One rule of derivation, a generalized form of modus ponens:
- If "$h \leftarrow \mathbf{b_1} \wedge \ldots \wedge \mathbf{b_m}$" is a clause in the knowledge base, and

  each $\mathbf{b_i}$ has been derived,

  then h can be derived.

- This rule also covers the case when $m = 0$.

Bottom-up (BU) proof procedure

C :={};
repeat
select clause "$h \leftarrow b_1 \wedge \ldots \wedge b_m$" in KB such that $b_i \in C$ for all i, and $h \notin C$;
C:= C ∪{ h } until no more clauses can be selected.

$KB \vdash_{BU} G$ if $G \subseteq C$ at the end of this procedure

The C at the end of BU procedure is a fixed point:

- Further applications of the rule of derivation will not change C!

Bottom-up proof procedure: example

C := {};

repeat

select clause $h \leftarrow b_1 \wedge \ldots \wedge b_m$ in KB such that $b_i \in C$ for all i, and $h \notin C$;

C := C ∪ {h} until no more clauses can be selected.

$a \leftarrow b \wedge c$

{}

$a \leftarrow e \wedge f$

$b \leftarrow f \wedge k$

$c \leftarrow e$ $d \leftarrow$

k

e.

$f \leftarrow j \wedge$

$e\ f \leftarrow c$

$j \leftarrow c$

Bottom-up proof procedure: example

C := {};

repeat

select clause $h \leftarrow b_1 \wedge \ldots \wedge b_m$ in KB such

that $b_i \in C$ for all i, and $h \notin C$;

C := C ∪ {h} until no more
clauses can be selected.

$a \leftarrow b \wedge c$

{}

$a \leftarrow e \wedge f$

$b \leftarrow f \wedge k$

$c \leftarrow e$

$d \leftarrow k$

$e.$

$f \leftarrow j \wedge e$

$f \leftarrow c$

$j \leftarrow c$

$\{e\}$

$\{c,e\}$

$\{c,e,f\}$

$\{c,e,f,j\}$

$\{a,c,e,f,j\}$

Done.