Lecture 15
Logic Intro and PDCL
Lecture Overview

Intro to Logic

• Propositional Definite Clause:
  • Syntax
  • Semantics
  • Proof procedures (time permitting)
Where Are We?

Problem Type
- Static
  - Constraint Satisfaction
- Sequential
  - Planning

Environment
- Deterministic
  - Arc Consistency
  - Vars + Constraints
  - Search
- Stochastic
  - Belief Nets
    - Variable Elimination
  - Decision Nets
    - Variable Elimination
  - Markov Processes
    - Value Iteration

Logics
- STRIPS
  - Search

First Part of the Course
Where Are We?

**Problem Type**
- Static
  - Constraint Satisfaction
  - Query
- Sequential
  - Planning

**Environment**
- Deterministic
- Stochastic

**Representation**
- Reasoning Technique

**Reasoning Technique**
- Arc Consistency
- Search

**Static**
- Logics
  - Search
- STRIPS
  - Search

**Stochastic**
- Belief Nets
  - Variable Elimination
- Decision Nets
  - Variable Elimination
- Markov Processes
  - Value Iteration

Back to static problems, but with richer representation.
Logics in AI: Similar slide to the one for planning

- Propositional Definite Clause Logics
- Semantics and Proof Theory
- Propositional Logics
- First-Order Logics
- Satisfiability Testing (SAT)
- Description Logics
- Production Systems
- Hardware Verification
- Ontologies
- Cognitive Architectures
- Software Verification
- Video Games
- Product Configuration
- Summarization
- Tutoring Systems
- Information Extraction
- Applications
Logics in AI: Similar slide to the one for planning

- Propositional Definite Clause Logics
- Semantics and Proof Theory
- First-Order Logics
- Satisfiability Testing (SAT)
- Hardware Verification
- Software Verification
- Product Configuration
- You will know
- You will know a little
- Applications

- Description Logics
- Production Systems
- Cognitive Architectures
- Ontologies
- Semantic Web
- Video Games
- Summarization
- Tutoring Systems
- Information Extraction
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 those

\[ \forall(x)\text{triangle}(x) \rightarrow [A = B = C \iff \alpha = \beta = \gamma] \]

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....
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
    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)

  \[
  \forall(z) \text{student}(z) \& \text{registered}(z, c1) \rightarrow \text{will \_ pass}(z, c1)
  \]

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

\textit{live}_w_4? \textit{lit}_l_2?
Example: Electrical Circuit

- Outside power
- Circuit breaker
- Switch (on/off)
- Two-way switch
- Light
- Power outlet

Diagram showing electrical components with labels for circuits, switches, and power connections.
light_l1.
light_l2.
ok_l1.
ok_l2.
ok_cb1.
ok_cb2.
live_outside.

\[
\begin{align*}
\text{live}_l1 & \leftarrow \text{live}_w0. \\
\text{live}_w0 & \leftarrow \text{live}_w1 \land \text{up}_s2. \\
\text{live}_w0 & \leftarrow \text{live}_w2 \land \text{down}_s2. \\
\text{live}_w1 & \leftarrow \text{live}_w3 \land \text{up}_s1. \\
\text{live}_w2 & \leftarrow \text{live}_w3 \land \text{down}_s1. \\
\text{live}_l2 & \leftarrow \text{live}_w4. \\
\text{live}_w4 & \leftarrow \text{live}_w3 \land \text{up}_s3. \\
\text{live}_p1 & \leftarrow \text{live}_w3. \\
\text{live}_w3 & \leftarrow \text{live}_w5 \land \text{ok}_cb1. \\
\text{live}_p2 & \leftarrow \text{live}_w6. \\
\text{live}_w6 & \leftarrow \text{live}_w5 \land \text{ok}_cb2. \\
\text{live}_w5 & \leftarrow \text{live}_\text{outside}. \\
\text{lit}_l1 & \leftarrow \text{light}_l1 \land \text{live}_l1 \land \text{ok}_l1. \\
\text{lit}_l2 & \leftarrow \text{light}_l2 \land \text{live}_l2 \land \text{ok}_l2.
\end{align*}
\]
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
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 Clause:
- 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
Propositional Definite Clauses: Syntax

Definition (atom)
An **atom** is a symbol starting with a lower case letter

Examples: $p_1$; $\text{live}_1$

Definition (body)
A **body** is an atom or is of the form $b_1 \land b_2$ where $b_1$ and $b_2$ are bodies.

Examples: $p_1 \land p_2$; $\text{ok}_w_1 \land \text{live}_w_0$

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 $b$”.)

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

Definition (KB)
A **knowledge base (KB)** is a set of definite clauses
\begin{itemize}
  \item \textit{atoms}:
    \begin{itemize}
      \item light\(_{l1}\).
      \item light\(_{l2}\).
      \item ok\(_{l1}\).
      \item ok\(_{l2}\).
      \item ok\(_{cb1}\).
      \item ok\(_{cb2}\).
      \item live\textunderscore outside.
    \end{itemize}
  \item \textit{definite clauses, KB}:
    \begin{itemize}
      \item live\(_{l1}\) ← live\(_{w0}\).
      \item live\(_{w0}\) ← live\(_{w1}\) ∧ up\(_{s2}\).
      \item live\(_{w0}\) ← live\(_{w2}\) ∧ down\(_{s2}\).
      \item live\(_{w1}\) ← live\(_{w3}\) ∧ up\(_{s1}\).
      \item live\(_{w2}\) ← live\(_{w3}\) ∧ down\(_{s1}\).
      \item live\(_{l2}\) ← live\(_{w4}\).
      \item live\(_{w4}\) ← live\(_{w3}\) ∧ up\(_{s3}\).
      \item live\(_{p1}\) ← live\(_{w3}\).
      \item live\(_{w3}\) ← live\(_{w5}\) ∧ ok\(_{cb1}\).
      \item live\(_{p2}\) ← live\(_{w6}\).
      \item live\(_{w6}\) ← live\(_{w5}\) ∧ ok\(_{cb2}\).
      \item live\(_{w5}\) ← live\textunderscore outside.
      \item lit\(_{l1}\) ← light\(_{l1}\) ∧ live\(_{l1}\) ∧ ok\(_{l1}\).
      \item lit\(_{l2}\) ← light\(_{l2}\) ∧ live\(_{l2}\) ∧ ok\(_{l2}\).
    \end{itemize}
  \item \textit{rules}:
\end{itemize}
Which of the clauses below are legal PDCL clauses?

a) Sunny_today

b) sunny_today ∨ cloudy_today

c) high_pressure_system ← sunny_today

d) sunny_today ← high_pressure_system ∧ summer

e) sunny_today ← high_pressure-system ∧ ¬ winter

f) ai_is_fun ← f(time_spent, material_learned)

g) high_pressure_system ← sunny_today ∧ summer
PDC Syntax: more examples

Legal PDC clause  Not a legal PDC clause

a) Sunny_today  

b) sunny_today \lor cloudy_today  

c) high_pressure_system \Leftarrow sunny-today  

d) sunny_today \Leftarrow high_pressure_system \land summer  

e) sunny_today \Leftarrow high_pressure-system \land \neg winter  

f) ai_is_fun \Leftarrow f(time_spent, material_learned)  

g) high_pressure_system \Leftarrow sunny_today \land summer  

Do any of these statements mean anything? Syntax doesn't answer this question!
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**
Propositional Definite Clauses: Syntax

**Definition (atom)**
An **atom** is a symbol starting with a lower case letter

**Examples:**  
- $p_1$  
- `live_l1`

**Definition (body)**
A **body** is an atom or is of the form $b_1 \land b_2$ where $b_1$ and $b_2$ are bodies.

**Examples:**  
- $p_1 \land p_2$  
- `ok_w1 \land live_w0`

**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 $b$."

**Examples:**
- $p_1 \leftarrow p_2$  
- `live_w0 \leftarrow live_w1 \land up_s2`

**Definition (KB)**
A **knowledge base (KB)** is a set of definite clauses
light_l1.  
light_l2.  
ok_l1.  
ok_l2.  
ok_cb1.  
ok_cb2.  
live_outside.

atoms

definite clauses, KB

live_l1 ← live_w0.
live_w0 ← live_w1 ∧ up_s2.
live_w0 ← live_w2 ∧ down_s2.
live_w1 ← live_w3 ∧ up_s1.
live_w2 ← live_w3 ∧ down_s1.
live_l2 ← live_w4.
live_w4 ← live_w3 ∧ up_s3.
live_p1 ← live_w3.
live_w3 ← live_w5 ∧ ok_cb1.
live_p2 ← live_w6.
live_w6 ← live_w5 ∧ ok_cb2.
live_w5 ← live_outside.
lit_l1 ← light_l1 ∧ live_l1 ∧ ok_l1.
lit_l2 ← light_l2 ∧ live_l2 ∧ ok_l2.
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
Lecture Overview

• Intro to Logic
• Propositional Definite Clause:
  • Syntax
  ➡️ Semantics
  • Proof Procedures
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?

A. $8 + 2$
B. $8 \times 2$
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
  • Similar to possible worlds in CSPs
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.

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

<table>
<thead>
<tr>
<th></th>
<th>a₁</th>
<th>a₂</th>
<th>a₁ ∧ a₂</th>
</tr>
</thead>
<tbody>
<tr>
<td>I₁</td>
<td>F</td>
<td>F</td>
<td>F</td>
</tr>
<tr>
<td>I₂</td>
<td>F</td>
<td>T</td>
<td></td>
</tr>
<tr>
<td>I₃</td>
<td>T</td>
<td>F</td>
<td></td>
</tr>
<tr>
<td>I₄</td>
<td>T</td>
<td>T</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>h</th>
<th>b</th>
<th>h ← b</th>
</tr>
</thead>
<tbody>
<tr>
<td>I₁</td>
<td>F</td>
<td>F</td>
<td></td>
</tr>
<tr>
<td>I₂</td>
<td>F</td>
<td>T</td>
<td></td>
</tr>
<tr>
<td>I₃</td>
<td>T</td>
<td>F</td>
<td></td>
</tr>
<tr>
<td>I₄</td>
<td>T</td>
<td>T</td>
<td></td>
</tr>
</tbody>
</table>
### PDC Semantics: Example

**Truth values under different interpretations**

<table>
<thead>
<tr>
<th></th>
<th>h</th>
<th>b</th>
<th>h ← b</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>I₁</strong></td>
<td>F</td>
<td>F</td>
<td>T</td>
</tr>
<tr>
<td><strong>I₂</strong></td>
<td>F</td>
<td>T</td>
<td>F</td>
</tr>
<tr>
<td><strong>I₃</strong></td>
<td>T</td>
<td>F</td>
<td>T</td>
</tr>
<tr>
<td><strong>I₄</strong></td>
<td>T</td>
<td>T</td>
<td>T</td>
</tr>
</tbody>
</table>

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

<table>
<thead>
<tr>
<th></th>
<th>h</th>
<th>a₁</th>
<th>a₂</th>
<th>h ← a₁ ∧ a₂</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>I₁</strong></td>
<td>F</td>
<td>F</td>
<td>F</td>
<td>F</td>
</tr>
<tr>
<td><strong>I₂</strong></td>
<td>F</td>
<td>F</td>
<td>T</td>
<td>T</td>
</tr>
<tr>
<td><strong>I₃</strong></td>
<td>F</td>
<td>T</td>
<td>F</td>
<td>F</td>
</tr>
<tr>
<td><strong>I₄</strong></td>
<td>F</td>
<td>T</td>
<td>T</td>
<td>T</td>
</tr>
<tr>
<td><strong>I₅</strong></td>
<td>T</td>
<td>F</td>
<td>F</td>
<td>F</td>
</tr>
<tr>
<td><strong>I₆</strong></td>
<td>T</td>
<td>F</td>
<td>T</td>
<td>T</td>
</tr>
<tr>
<td><strong>I₇</strong></td>
<td>T</td>
<td>T</td>
<td>F</td>
<td>F</td>
</tr>
<tr>
<td><strong>I₈</strong></td>
<td>T</td>
<td>T</td>
<td>T</td>
<td>T</td>
</tr>
</tbody>
</table>
PDC Semantics: Example for truth values

Truth values under different interpretations

| F=\text{false}, \ T=\text{true} | h & a_1 & a_2 & h \leftarrow a_1 \land a_2 |
|-------------------------------|-------------|------------|-----------------|
| $I_1$ | F & F & F & F | T |
| $I_2$ | F & F & T & F |
| $I_3$ | F & T & F & T |
| $I_4$ | T & T & T & T |
| $I_5$ | T & F & F & T |
| $I_6$ | T & F & T & T |
| $I_7$ | T & T & F & T |
| $I_8$ | T & T & T & T |

$h \leftarrow a_1 \land a_2$

Body of the clause is $a_1 \land a_2$

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* $b_1 \land b_2$ 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 $I$. 
### 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: Knowledge Base (KB)

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

<table>
<thead>
<tr>
<th>p</th>
<th>q</th>
<th>r</th>
<th>s</th>
</tr>
</thead>
<tbody>
<tr>
<td>I₁</td>
<td>false</td>
<td>true</td>
<td>true</td>
</tr>
</tbody>
</table>

A. KB₁

\[ p \]
\[ r \]
\[ s \leftarrow q \land p \]

B. KB₂

\[ r \]
\[ s \leftarrow p \]
\[ q \leftarrow p \land s \]

C. KB₃

\[ r \]
\[ q \]
\[ s \leftarrow q \]

Which of the three KB above is True in I₁
PDC Semantics: Knowledge Base (KB)

- A **knowledge base KB** is true in $I_1$ if and only if every clause in KB is true in $I_1$.

<table>
<thead>
<tr>
<th>p</th>
<th>q</th>
<th>r</th>
<th>s</th>
</tr>
</thead>
<tbody>
<tr>
<td>$I_1$</td>
<td>true</td>
<td>true</td>
<td>false</td>
</tr>
</tbody>
</table>

$KB_2$

$r$

$s \leftarrow p$

$q \leftarrow p \land s$

Which of the three KB above are True in $I_1$? $KB_2$
Definition (model)
A model of a knowledge base KB is an interpretation in which every clause in KB is true.

\[ \begin{align*}
   &p \leftarrow q \\
   &s \\
   &s \leftarrow r
\end{align*} \]

KB =

Which of the interpretations below are models of KB?

<table>
<thead>
<tr>
<th></th>
<th>p</th>
<th>q</th>
<th>r</th>
<th>s</th>
</tr>
</thead>
<tbody>
<tr>
<td>I₁</td>
<td>T</td>
<td>T</td>
<td>T</td>
<td>T</td>
</tr>
<tr>
<td>I₂</td>
<td>F</td>
<td>F</td>
<td>F</td>
<td>T</td>
</tr>
<tr>
<td>I₃</td>
<td>T</td>
<td>T</td>
<td>F</td>
<td>F</td>
</tr>
<tr>
<td>I₄</td>
<td>T</td>
<td>T</td>
<td>T</td>
<td>F</td>
</tr>
<tr>
<td>I₅</td>
<td>T</td>
<td>T</td>
<td>F</td>
<td>T</td>
</tr>
</tbody>
</table>

A. I₁
B. I₁, I₂
C. I₁, I₂, I₅
D. All of them
Definition (model)
A **model** of a knowledge base KB is an interpretation in which every clause in KB is true.

KB = \[
p \leftarrow q \\
q & \quad s \\
q & \quad s \leftarrow r
\]

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

<table>
<thead>
<tr>
<th></th>
<th>p</th>
<th>q</th>
<th>r</th>
<th>s</th>
<th>p ← q</th>
<th>s</th>
<th>s ← r</th>
<th>KB</th>
</tr>
</thead>
<tbody>
<tr>
<td>l₁</td>
<td>T</td>
<td>T</td>
<td>T</td>
<td>T</td>
<td>T</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>l₂</td>
<td>F</td>
<td>F</td>
<td>F</td>
<td>F</td>
<td></td>
<td></td>
<td></td>
<td>T</td>
</tr>
<tr>
<td>l₃</td>
<td>T</td>
<td>T</td>
<td>F</td>
<td>F</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>l₄</td>
<td>T</td>
<td>T</td>
<td>T</td>
<td>F</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>l₅</td>
<td>T</td>
<td>T</td>
<td>F</td>
<td>T</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### Definition (model)

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

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

Which of the interpretations below are models of $KB$?

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

<table>
<thead>
<tr>
<th></th>
<th>$p$</th>
<th>$q$</th>
<th>$r$</th>
<th>$s$</th>
<th>$p \leftarrow q$</th>
<th>$s$</th>
<th>$s \leftarrow r$</th>
<th>$KB$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$I_1$</td>
<td>T</td>
<td>T</td>
<td>T</td>
<td>T</td>
<td>T</td>
<td>T</td>
<td>T</td>
<td>T</td>
</tr>
<tr>
<td>$I_2$</td>
<td>F</td>
<td>F</td>
<td>F</td>
<td>T</td>
<td>T</td>
<td>T</td>
<td>T</td>
<td>T</td>
</tr>
<tr>
<td>$I_3$</td>
<td>T</td>
<td>T</td>
<td>F</td>
<td>F</td>
<td>T</td>
<td>F</td>
<td>T</td>
<td>F</td>
</tr>
<tr>
<td>$I_4$</td>
<td>T</td>
<td>T</td>
<td>T</td>
<td>T</td>
<td>T</td>
<td>F</td>
<td>T</td>
<td>F</td>
</tr>
<tr>
<td>$I_5$</td>
<td>T</td>
<td>T</td>
<td>F</td>
<td>T</td>
<td>T</td>
<td>T</td>
<td>T</td>
<td>T</td>
</tr>
</tbody>
</table>
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.

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

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.

   - `live_l1.
   - live_l2.
   - ok_l1.
   - ok_l2.
   - ok_cb1.
   - ok_cb2.
   - live_outside.

   ```
   live_l1 ← live_w0.
   live_w0 ← live_w1 ∧ up_s2.
   live_w0 ← live_w2 ∧ down_s2.
   live_w1 ← live_w3 ∧ up_s1.
   live_w2 ← live_w3 ∧ down_s1.
   live_l2 ← live_w4.
   live_w4 ← live_w3 ∧ up_s3.
   live_p1 ← live_w3.
   live_w3 ← live_w5 ∧ ok_cb1.
   live_p2 ← live_w6.
   live_w6 ← live_w5 ∧ ok_cb2.
   live_w5 ← live_outside.
   lit_l1 ← light_l1 ∧ live_l1 ∧ ok_l1.
   lit_l2 ← light_l2 ∧ live_l2 ∧ ok_l2.
   ```

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

   - `live_w4?`
   - `lit_l2?`
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 $KB \models G$, 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, $KB \not\models G$ if 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

\[ KB = \begin{cases} 
  p & \leftarrow q \land r. \\
  q. 
\end{cases} \]

How many models are there?

A. 1   B. 2   C. 3   D. 4   E. 5
Example of Logic Entailment

Interpretations

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>$r$</td>
<td>$q$</td>
<td>$p$</td>
</tr>
<tr>
<td>T</td>
<td>T</td>
<td>T</td>
</tr>
<tr>
<td>T</td>
<td>T</td>
<td>F</td>
</tr>
<tr>
<td>T</td>
<td>F</td>
<td>T</td>
</tr>
<tr>
<td>T</td>
<td>F</td>
<td>F</td>
</tr>
<tr>
<td>F</td>
<td>T</td>
<td>T</td>
</tr>
<tr>
<td>F</td>
<td>T</td>
<td>F</td>
</tr>
<tr>
<td>F</td>
<td>F</td>
<td>T</td>
</tr>
<tr>
<td>F</td>
<td>F</td>
<td>F</td>
</tr>
</tbody>
</table>

How many models are there?

A. 1  B. 2  C. 3  D. 4  E. 5
Example of Logic Entailment

\[ KB = \begin{cases} 
  p \leftarrow q \land r. \\
  q. 
\end{cases} \]

Models

Interpretations

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>T</td>
<td>T</td>
<td>T</td>
</tr>
<tr>
<td>T</td>
<td>T</td>
<td>F</td>
</tr>
<tr>
<td>T</td>
<td>F</td>
<td>T</td>
</tr>
<tr>
<td>T</td>
<td>F</td>
<td>F</td>
</tr>
<tr>
<td>F</td>
<td>T</td>
<td>T</td>
</tr>
<tr>
<td>F</td>
<td>T</td>
<td>F</td>
</tr>
<tr>
<td>F</td>
<td>F</td>
<td>T</td>
</tr>
<tr>
<td>F</td>
<td>F</td>
<td>F</td>
</tr>
</tbody>
</table>

Which atoms are logically entailed?

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

\[ KB = \begin{cases} p \leftarrow q \land r. \\ q. \end{cases} \]

Interpretations

<table>
<thead>
<tr>
<th>r</th>
<th>q</th>
<th>p</th>
</tr>
</thead>
<tbody>
<tr>
<td>T</td>
<td>T</td>
<td>T</td>
</tr>
<tr>
<td>T</td>
<td>T</td>
<td>F</td>
</tr>
<tr>
<td>T</td>
<td>F</td>
<td>T</td>
</tr>
<tr>
<td>T</td>
<td>F</td>
<td>F</td>
</tr>
<tr>
<td>F</td>
<td>T</td>
<td>T</td>
</tr>
<tr>
<td>F</td>
<td>T</td>
<td>F</td>
</tr>
<tr>
<td>F</td>
<td>F</td>
<td>T</td>
</tr>
<tr>
<td>F</td>
<td>F</td>
<td>F</td>
</tr>
</tbody>
</table>

Models

Which atoms are logically entailed?

q

- We want a reasoning procedure that can find all and only the logical consequences of a knowledge base.
Example: Logical Consequences

<table>
<thead>
<tr>
<th></th>
<th>p</th>
<th>q</th>
<th>r</th>
<th>s</th>
</tr>
</thead>
<tbody>
<tr>
<td>I₁</td>
<td>true</td>
<td>true</td>
<td>true</td>
<td>true</td>
</tr>
<tr>
<td>I₂</td>
<td>true</td>
<td>true</td>
<td>true</td>
<td>false</td>
</tr>
<tr>
<td>I₃</td>
<td>true</td>
<td>true</td>
<td>false</td>
<td>false</td>
</tr>
<tr>
<td>I₄</td>
<td>true</td>
<td>true</td>
<td>false</td>
<td>true</td>
</tr>
<tr>
<td>I₅</td>
<td>false</td>
<td>true</td>
<td>true</td>
<td>true</td>
</tr>
<tr>
<td>I₆</td>
<td>false</td>
<td>true</td>
<td>true</td>
<td>false</td>
</tr>
<tr>
<td>I₇</td>
<td>false</td>
<td>true</td>
<td>false</td>
<td>false</td>
</tr>
<tr>
<td>I₈</td>
<td>false</td>
<td>true</td>
<td>false</td>
<td>true</td>
</tr>
<tr>
<td>I₉</td>
<td>true</td>
<td>false</td>
<td>true</td>
<td>true</td>
</tr>
<tr>
<td>I₁₀</td>
<td>true</td>
<td>false</td>
<td>true</td>
<td>false</td>
</tr>
<tr>
<td>I₁₁</td>
<td>true</td>
<td>false</td>
<td>false</td>
<td>false</td>
</tr>
<tr>
<td>I₁₂</td>
<td>true</td>
<td>false</td>
<td>false</td>
<td>false</td>
</tr>
</tbody>
</table>

\( KB = \{ p \leftarrow q, \quad q, \quad r \leftarrow s \} \)

Which of the following is true?

- \( KB \not\models q, \)
- \( KB \not\models p, \)
- \( KB \not\models s, \)
- \( KB \not\models r \)
### Example: Logical Consequences

<table>
<thead>
<tr>
<th>I</th>
<th>p</th>
<th>q</th>
<th>r</th>
<th>s</th>
</tr>
</thead>
<tbody>
<tr>
<td>I₁</td>
<td>true</td>
<td>true</td>
<td>true</td>
<td>true</td>
</tr>
<tr>
<td>I₂</td>
<td>true</td>
<td>true</td>
<td>true</td>
<td>false</td>
</tr>
<tr>
<td>I₃</td>
<td>true</td>
<td>true</td>
<td>false</td>
<td>false</td>
</tr>
<tr>
<td>I₄</td>
<td>true</td>
<td>true</td>
<td>false</td>
<td>true</td>
</tr>
<tr>
<td>I₅</td>
<td>false</td>
<td>true</td>
<td>true</td>
<td>true</td>
</tr>
<tr>
<td>I₆</td>
<td>false</td>
<td>true</td>
<td>true</td>
<td>false</td>
</tr>
<tr>
<td>I₇</td>
<td>false</td>
<td>true</td>
<td>false</td>
<td>false</td>
</tr>
<tr>
<td>I₈</td>
<td>false</td>
<td>true</td>
<td>false</td>
<td>true</td>
</tr>
<tr>
<td>I₉</td>
<td>true</td>
<td>false</td>
<td>true</td>
<td>true</td>
</tr>
<tr>
<td>I₁₀</td>
<td>true</td>
<td>false</td>
<td>true</td>
<td>false</td>
</tr>
<tr>
<td>I₁₁</td>
<td>true</td>
<td>false</td>
<td>false</td>
<td>false</td>
</tr>
<tr>
<td>I₁₂</td>
<td>true</td>
<td>false</td>
<td>false</td>
<td>false</td>
</tr>
</tbody>
</table>

\[ KB = \{ p \leftarrow q, \ q, \ r \leftarrow s. \} \]

Which of the following is true?

- \( KB \vdash q, \)
- \( KB \vdash p, \)
- \( KB \vdash s, \)
- \( KB \vdash r \)
Example: Logical Consequences

<table>
<thead>
<tr>
<th></th>
<th>p</th>
<th>q</th>
<th>r</th>
<th>s</th>
</tr>
</thead>
<tbody>
<tr>
<td>I₁</td>
<td>true</td>
<td>true</td>
<td>true</td>
<td>true</td>
</tr>
<tr>
<td>I₂</td>
<td>true</td>
<td>true</td>
<td>true</td>
<td>false</td>
</tr>
<tr>
<td>I₃</td>
<td>true</td>
<td>true</td>
<td>false</td>
<td>false</td>
</tr>
<tr>
<td>I₄</td>
<td>true</td>
<td>true</td>
<td>false</td>
<td>true</td>
</tr>
<tr>
<td>I₅</td>
<td>false</td>
<td>true</td>
<td>true</td>
<td>true</td>
</tr>
<tr>
<td>I₆</td>
<td>false</td>
<td>true</td>
<td>true</td>
<td>false</td>
</tr>
<tr>
<td>I₇</td>
<td>false</td>
<td>true</td>
<td>false</td>
<td>false</td>
</tr>
<tr>
<td>I₈</td>
<td>false</td>
<td>true</td>
<td>false</td>
<td>true</td>
</tr>
<tr>
<td>I₉</td>
<td>true</td>
<td>false</td>
<td>true</td>
<td>true</td>
</tr>
<tr>
<td>I₁₀</td>
<td>true</td>
<td>false</td>
<td>true</td>
<td>false</td>
</tr>
<tr>
<td>I₁₁</td>
<td>true</td>
<td>false</td>
<td>false</td>
<td>false</td>
</tr>
<tr>
<td>I₁₂</td>
<td>true</td>
<td>false</td>
<td>false</td>
<td>false</td>
</tr>
</tbody>
</table>

Which of the following is true?

- $\text{KB} \models q$,  
- $\text{KB} \models p$,  
- $\text{KB} \not\models s$,  
- $\text{KB} \not\models r$

$ KB = \begin{cases} p \leftarrow q. \\ q. \\ r \leftarrow s. \end{cases}$
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.
  • If $KB \models g$, 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 $KB \models g$ then $g$ must be true in the intended interpretation.

  • Otherwise, there is a model of $KB$ in which $g$ is false. This could be the intended interpretation.

The computer wouldn't know!
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

<table>
<thead>
<tr>
<th></th>
<th>$p$</th>
<th>$q$</th>
<th>$r$</th>
<th>$s$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$I_1$</td>
<td>true</td>
<td>true</td>
<td>true</td>
<td>true</td>
</tr>
<tr>
<td>$I_2$</td>
<td>true</td>
<td>true</td>
<td>true</td>
<td>false</td>
</tr>
</tbody>
</table>

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

<table>
<thead>
<tr>
<th></th>
<th>$p$</th>
<th>$q$</th>
<th>$r$</th>
<th>$s$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$I_1$</td>
<td>true</td>
<td>true</td>
<td>true</td>
<td>true</td>
</tr>
<tr>
<td>$I_2$</td>
<td>true</td>
<td>true</td>
<td>true</td>
<td>false</td>
</tr>
</tbody>
</table>

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

$I_1$ 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 Clause:
  - 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 and complete)
  - **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, $\text{KB } \vdash_{P} g$ 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 $KB \not\models g$.

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 $KB$ ($KB \not\models g$)

Definition (completeness)
A proof procedure $P$ is **complete** if $KB \not\models g$ 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 $KB$ ($KB \not\models g$) then $g$ is derived by the procedure ($KB \vdash_P g$)