Example

Let $I$ be a model of $\mathcal{L}$, prove that $I$ makes $\text{person}(\text{sally})$ true.

Let's do proof by contradiction.

Could prove this by checking all interpretations.

What does $\text{KB} \models \text{person}(\text{sally})$ mean?

- Prove $\text{KB} \models \text{person}(\text{sally})$.

Let $I$ be a model of $\text{KB}$, prove that $I$ makes $\text{person}(\text{sally})$ true.

Means that if interpretation $I$ models $\text{KB}$ then it models $\text{person}(\text{sally})$.

- Means that for every clause in $\text{KB}$, if $I$ makes the clause true then $I$ also makes $\text{person}(\text{sally})$ true.

Could prove this by checking all interpretations.

- Let's do proof instead.

- Let $I$ be a model of $\text{KB}$, prove that $I$ makes $\text{person}(\text{sally})$ true.

Review

- An interpretation maps any clause to either true or false.

- It is a complete mapping.

- A model of $\text{KB}$ is an interpretation $I$ such that $I$ maps every clause in $\text{KB}$ to true.

- $\text{KB} \models \phi$ if every model of $\text{KB}$ makes $\phi$ true.

Overview

- Queries
- Proof Procedures
- Bottom-up Ground Proof Procedure
- Top-down Ground Proof Procedure
- Questions
- Semantics
Overview

Semantics

Queries

Proof Procedures

• Bottom-up Ground Proof Procedure

• Top-down Ground Proof Procedure

More on Variables in Clauses (pp. 42)

More on Variables in Clauses (pp. 42)

More on Variables in Clauses (pp. 42)

A Semantic Proof

Since \( \phi(sally) \) is true under \( I \),

<table>
<thead>
<tr>
<th>( \phi(sally) )</th>
<th>( s = (\chi) )</th>
<th>$m(s) \in \pi(\chi)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>( m(s) )</td>
<td>( s = (\chi) )</td>
<td>$m(s) \in \pi(\chi)$</td>
</tr>
</tbody>
</table>

Consider variable assignment \( \delta \) where

<table>
<thead>
<tr>
<th>( \delta )</th>
<th>( m(s) )</th>
<th>$m(s) \in \pi(\chi)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>( s )</td>
<td>( m(s) )</td>
<td>$m(s) \in \pi(\chi)$</td>
</tr>
</tbody>
</table>

Let \( \{ \nu \phi \}$ be a model of \( KB \)
Overview

• Semantics
• Queries ⇒ Proof Procedures
  • Bottom-up Ground Proof Procedure
  • Top-down Ground Proof Procedure

Queries with Variables

You might not only want to check if something is true or false, but what value makes it true.

KB:
father(william,ted)

Example:
parent(X,Y) ← father(X,Y)

Example:
parent(X,Y) ← father(X,Y)

Who is Ted's parent?

KB:

Then when value makes it true?

You might not only want to check if something is true or false.

Ground Queries

• A query is a way to ask if a body is a logical consequence of the KB: ?b1 ∧ ... ∧ bm

• Ground query (no variables) has the answer:
  - "yes" if the body is a logical consequence of the KB
  - "no" if the body is not a logical consequence of the KB

• Can do query-answering by:
  - Transform query b1 ∧ ... ∧ bm into yes ← b1 ∧ ... ∧ bm
  - Add (temporarily) yes ← b1 ∧ ... ∧ bm to the KB
  - Check if yes is a logical consequence of the KB

This lets us view queries as just finding consequences from a KB.

• Check if yes is a logical consequence of KB:
  - Add (temporarily) yes ← b1 ∧ ... ∧ bm to KB
  - Do the usual resolution

• Can do query-answering by:
  - Check if yes is a logical consequence of the KB:
  - If yes is not a logical consequence, query fails.
  - If yes is a logical consequence, query succeeds.

Ground Queries (no variables) gives the answer:

A query is a way to ask if a body is a logical consequence of the KB.
Two Types of Proof Procedures

- **Top-Down**: Forward-Chaining
  - KB ⊢ g
  - g can be derived from KB with the proof procedure

- **Bottom-Up**: Backward-Chaining
  - Query
  - KB ⊨ g

**Properties of Proof Procedure**

- **Soundness**: if KB ⊨ g then KB ⊢ g
- **Completeness**: if KB ⊢ g then KB ⊨ g

**Terminology**

- **Semantic Proof**: |=, logically follows, logically entails, models
- **Syntactic Proof**: ⊢, derives

**Semantics Is Not Enough**

- We have KB - We know what conclusions are valid to make
- KB |= g
  - g is true in all models of KB
- Can extend this so we can make conclusions against KB as well
- Can extend this so we can ask questions with variables as well
- Query

**Proof Procedures**

- Computer can only access the KB
- Can't just check the user's intended interpretation
- Checking all interpretations is very expensive
- But, don't yet have a mechanical way of checking if KB |= g
- Prove a mechanically derivable demonstration that a formula
Overview

- Semantics
- Queries
- Proof Procedures
  ⇒ Bottom-up Ground Proof Procedure
- Top-down Ground Proof Procedure

Bottom-up Ground Proof Procedure

- For now, only consider ground facts and ground rules
  - no variables
- Bottom-up or forward chaining procedure:
  starts from $KB$ and works towards query
- Forward chaining rule
  - If $h ← b_1 ∧ ... ∧ b_m$ is a clause in the $KB$
  - and each $b_i$ has been derived
  - then $h$ can be derived
- Forward chaining rule also works if $h$ is a fact in $KB$ ($m = 0$)
  - Lets you derive $h$
- Call the set of derivables the consequence set $(C)$

Non-deterministic Specification

- Haven’t specified the exact order that things should be done in
  - What order should we pick clauses from $KB$ to try?
Is it Complete?

• Does C have every ground atom that logically follows from KB?

We need to prove something about consequence sets.

Let $C$ be the final consequent set generated by the algorithm
- Will stop because finite number of constants and predicate symbols
- Will stop with same $C$, no matter what order $C$ was generated

Define $I$ such that for atom $h$

$$I(h) = \begin{cases} 
\text{true} & \text{if } h \in C \\
\text{false} & \text{otherwise}
\end{cases}$$

$I$ is an interpretation because it defines a subset of ground atoms as being true and the rest as false.

Is it also a model of KB?

• i.e. for every $g \in \text{KB}$, is $I(g)$ true?

We need to prove something about consequence sets.

Proof by contradiction: assume $KB \vdash g$ but $KB \not\models g$

- g is the result of a finite number of derivations
- Without loss of generality, assume $g$ is first one in derivation such that $KB \not\models g$

Now $g$ was derived by a cause $g \leftarrow \bar{b}_1 \land \ldots \land \bar{b}_m$ in KB where the $\bar{b}_i$'s have already been derived.

$a \not\models \bar{b}_i$ by cause $a$.

- a is the result of a finite number of derivations.
- Therefore, $\bar{b}_i \not\models \bar{b}_i$ by definition of $\leftarrow$

$a \not\models \bar{b}_1 \land \ldots \land \bar{b}_m$ by definition of $\land$.

Using definition of $\leftarrow$, can show that $g$ logically follows from $KB$.

Contradiction.

Example:

$$a \leftarrow \bar{b} \land \bar{c}.$$ 

$$b \leftarrow \bar{d} \land \bar{e}.$$ 

$$b \leftarrow \bar{g} \land \bar{e}.$$ 

$$c \leftarrow \bar{e}.$$ 

$$
\text{Does C have every ground atom that logically follows from KB?}
$$
Overview

• Semantics
• Queries
• Proof Procedures

⇒

Bottom-up Ground Proof Procedure

Top-down Ground Proof Procedure

Final Step in Completeness Proof

Let \( g \) be atomic and \( \text{KB} \models g \)

- Need to make sure that \( \text{KB} \vdash g \)

- Since \( \text{KB} \models g \), \( g \) must be in every model of \( \text{KB} \)

- So, \( g \) is atomic and it is true in the interpretation.

- Since \( \text{KB} \models g = 0 \), \( g \) must be in every model of \( \text{KB} \)

- So \( g \) must be in consequence set

- Since \( g \) is atomic and it is true in the interpretation,

- \( g \) must be in consequence set

- So \( \text{KB} \vdash g \)

Proof that Consequence Set is a Model

- Proof by Contradiction: Let \( g \in \text{KB} \) but where \( I(g) \) is false

- Since \( g \in \text{KB} \), \( g \) must have the form \( h \leftarrow b_1 \land \ldots \land b_m \)

- So \( h \leftarrow b_1 \land \ldots \land b_m \) is false in \( I \)

- Remember, definition of \( \leftarrow \) comes from Datalog, not \( I \)

- So \( h \) must be false in \( I \) and \( b_1 \land \ldots \land b_m \) must be true in \( I \)

- Remember, definition of \( \land \) comes from Datalog, not \( I \)

- So all \( b_i \) in \( C \) and \( h \leftarrow b_1 \land \ldots \land b_m \) is in \( \text{KB} \)

- Hence \( g \) is true in \( I \)

- So \( \text{KB} \vdash g \)

- Proof by Completeness: Let \( g \in \text{KB} \) where \( I(g) \) is false

- Since \( \text{KB} \vdash g \)

- Since \( g \in \text{KB} \), \( g \) must have the form \( h \leftarrow b_1 \land \ldots \land b_m \)

- So \( h \leftarrow b_1 \land \ldots \land b_m \) is false in \( I \)

- Remember, definition of \( \leftarrow \) comes from Datalog, not \( I \)

- So \( h \) must be false in \( I \) and \( b_1 \land \ldots \land b_m \) must be true in \( I \)

- Remember, definition of \( \land \) comes from Datalog, not \( I \)

- So all \( b_i \) in \( C \) and \( h \leftarrow b_1 \land \ldots \land b_m \) is in \( \text{KB} \)

- Hence \( g \) is true in \( I \)
Example

\[ a \leftarrow b \land c. \]
\[ b \leftarrow d \land e. \]
\[ b \leftarrow g \land e. \]
\[ c \leftarrow e. \]
\[ d. \]
\[ e. \]
\[ f \leftarrow a \land g. \]

Now for some definitions

- **Answer clause** is \( \text{yes} \leftarrow a_1 \land \ldots \land a_m \).
- **Answer** is answer clause with \( m = 0 \).
- **Derivation of a query \( \exists q_1 \land \ldots \land q_k \) from \( \text{KB} \) is a sequence of answer clauses \( \gamma_0, \gamma_1, \ldots, \gamma_n \) - \( \gamma_0 \) is the answer clause corresponding to the original query, \( \gamma_i \) is obtained by resolving \( \gamma_{i-1} \) with a clause in \( \text{KB} \), and \( \gamma_n \) is the answer.
- **Nondeterminism** - In choosing which clause from \( \text{KB} \) to resolve with.
- **Top-down Ground Proof Procedure**
  - Alternative to bottom-up (forward-chaining).
  - Top-down (backward-chaining).
  - Start with goal, work toward facts in \( \text{KB} \).
  - Definite Clause Resolution for Ground Case.
  - Top-down (backward-chaining).
  - Alternative to bottom-up (forward-chaining).
Bottom-up versus Top-down

- Both top-down and bottom-up are sufficient for datalog.
- We will explore some of these later in the course.
- E. tamekido.

There are many other ways of doing proofs:

- There are many other ways of doing proofs.
- There are many other ways of doing proofs.

So, top-down proof procedure is complete and sound:

- Any top-down proof can be converted to a bottom-up proof.
- Any bottom-up proof can be converted to a top-down proof.

Bottom-up versus Top-down