Example

Let \( I \) be a model of \( K \). Prove that \( I \) makes \( \text{person}(sally) \) true.

Can also prove this by checking all interpretations.

What does \( I \models K \models \text{person}(sally) \) mean?

Prove \( I \models K \models \text{person}(sally) \) means \( \text{person}(sally) \rightarrow \text{female}(X) \)

\( \text{female}(sally) \)

\( \text{KB} \models \)

Review

An interpretation maps any clause to either true or false.

It is a complete mapping.

If every model of \( K \) makes \( g \) true
then \( I \models K \models g \) for every clause in \( K \).

A model \( I \) of \( K \) is an interpretation.

An interpretation maps any clause to either true or false.

Overview

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

• Semantics
  ⇒ Queries

• Proof Procedures
  • Bottom-up Ground Proof Procedure
  • Top-down Ground Proof Procedure

More on Variables in Clauses (pg. 42)

• Say \( \text{parent}(X,Y) \leftarrow \text{father}(X,Y) \) is in \( \text{KB} \)
  - Implicit universal quantifiers around it
  - Anytime that \( \text{father}(X,Y) \) is true, so must \( \text{parent}(X,Y) \).

• Say \( \text{grandfather}(X,Y) \leftarrow \text{father}(X,Z) \land \text{parent}(Z,Y) \) is in \( \text{KB} \)
  - This clause is true for all \( X, Y, Z \)
  - \( \forall X \ Y \ Z (\text{grandfather}(X,Y) \leftarrow \text{father}(X,Z) \land \text{parent}(Z,Y)) \).

A Semantic Proof

\( \llbracket \phi \rrbracket^I = \{ \llbracket \phi \rrbracket \}_{I} \)

\( \llbracket \phi \rrbracket \in \text{KB} \)

(1) \( \llbracket \phi \rrbracket \in \text{KB} \) & \( \llbracket \phi \rrbracket \leq \{ \llbracket \phi \rrbracket \}_{I} \)

\( \llbracket \phi \rrbracket \in \text{KB} \) & \( \llbracket \phi \rrbracket \leq \{ \llbracket \phi \rrbracket \}_{I} \)
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)

?parent(X,ted)

- Who is Ted's parent?

- Could transform this to yes ← parent(X,ted)

- But, lets capture the variables in the body:

  yes(X) ← parent(X,ted)

• An answer is either
- instance of 'yes' that is a logical consequence of KB:
  yes(william)
- or no if no instance is a logical consequence of KB.

Ground Queries

• A query is a way to ask if a body is a logical consequence of the knowledge base:

  ?b\textsubscript{1} ∧ ... ∧ b\textsubscript{m}

• 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

+ We do not distinguish between it being false in all models or just some
+ Cannot tell if query is false in the intended interpretation

• Can do query-answering by:
- Transform query ?b\textsubscript{1} ∧ ... ∧ b\textsubscript{m} into yes ← b\textsubscript{1} ∧ ... ∧ b\textsubscript{m}
- Add (temporarily) yes ← b\textsubscript{1} ∧ ... ∧ b\textsubscript{m} to KB
- Check if yes is a logical consequence of 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 ← b\textsubscript{1} ∧ ... ∧ b\textsubscript{m}
- Transform query ?b\textsubscript{1} ∧ ... ∧ b\textsubscript{m} into yes ← b\textsubscript{1} ∧ ... ∧ b\textsubscript{m}

- Can do query-answering by:

  Cannot tell if body is a logical consequence of KB

  Cannot answer (no instances) but the answer

  A query is a way to ask if a body is a logical consequence of the KB

  Knowledge base: ?b\textsubscript{1} ∧ ... ∧ b\textsubscript{m}
Proof Procedures

- Soundness: if KB ⊢ g, then KB |= g
- Completeness: if KB |= g, then KB ⊢ g
- Semantic proof: |=, logically follows, logically entails, models
- Syntactic proof: ⊢, derives

Proof Procedures:

- Top-Down
- Bottom-Up
- Forward-Chaining
- Backward-Chaining

Properties of Proof Procedure:

- Proof procedure can produce counterexamples
- Proof procedure can be nondeterministic
- Proof procedure can be nonconstructive

Semantics Is Not Enough

- We have KB
- But, don't yet have a mechanical way of checking if YF |= b
- Can extract this so far can use facts with variables as well
- Can't extract this so far can't use facts with variables
- Can't check the KB
- Can, instead check the user's intended interpretation
- Checking KB interpretations is easy, expensive
- If YF |= b then YF |= YF |
- If YF |= YF it's true in all models of YF
- We know when conclusions are valid to make
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?

Bottom-up Ground Proof Procedure

- For now, only consider ground facts and ground rules
- Bottom-up or forward chaining procedure: starts from KB and works towards query
- Forward chaining rule
  - If \( h \leftarrow b_1 \land \ldots \land b_m \) 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)

For now, only consider ground facts and ground rules

Bottom-up Ground Proof Procedure

Top-down Ground Proof Procedure = Bottom-up Ground Proof Procedure

Overview
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 which order C was generated.

- Define I such that for atom h:
  - I(h) is true if h ∈ C.
  - Otherwise, I(h) is false.

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

- I is an interpretation, but is it also a model of KB?

Is it Sound?

- Does everything in C logically follow from KB?

- Proof by contradiction: assume KB ⊢ g but KB ̸|= g.

- g is the result of a finite number of derivations.

- Without loss of generality, assume g was derived by a cause:
  - g ← b₁ ∧ ... ∧ bₘ in KB, where the bᵢ’s have already been derived.

- b₁ ∧ ... ∧ bₘ logically follows from KB (from definition of ∧).

- g ← b₁ ∧ ... ∧ bₘ logically follows from KB since it is in KB.

- g logically follows from KB, since it is in KB.

- Using definition of ←, can show that g logically follows from KB.

- Example:

  a ← b ∧ c.
  b ← d ∧ e.
  b ← g ∧ e.
  c ← e.
  d.
  e.

- What is the consequence set?
Final Step in Completeness Proof

Let $g$ be atomic and $KB \models g$

- Need to make sure that $KB \vdash g$

- Since $KB \models g$, $g$ must be in every model of $KB$

- So, it is in the interpretation defined by the Consequence set

- Since $g$ is atomic and it is true in the interpretation, it must be in consequence set

- So $KB \vdash g$

Proof that Consequence Set is a Model

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

- Since $g \in KB$, $g$ 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 $I$

- So all $b_i$ must be true in $I$

- Hence $g$ is true in $I$

- Contradiction
Example

\[ a \lor b \rightarrow f \]
\[ c \]
\[ p \]
\[ e \rightarrow c \]
\[ q \rightarrow a \]
\[ q \rightarrow b \]
\[ q \rightarrow c \]

KB

Now for some definitions

- **Answer clause** is
  \[ \text{yes} \leftarrow a_1 \land \ldots \land a_m \]

- **Answer** is an answer clause with \( m = 0 \)

- **Derivation** of a query \( q_1 \land \ldots \land q_k \) from \( KB \) is a sequence of answer clauses \( \gamma_0, \gamma_1, \ldots, \gamma_n \) where
  \( \gamma_0 \) is the answer clause corresponding to the original query
  \( \gamma_i \) is obtained by resolving \( \gamma_{i-1} \) with a clause in \( KB \)
  \( \gamma_n \) is the answer

- **Nondeterminism** - In choosing which clause from \( KB \) to resolve with

- Can find all derivations by systematically considering all different choices (see Chapter 4)

Top-Down Ground Proof Procedure

- Alternative to bottom-up (forward-chaining)

- Top-down (backward-chaining)
  - Start with goal, work toward facts in \( KB \)

- **Definite Clause Resolution for Ground Case**

- Schema with goal \( \varphi \) and facts in \( KB \)

- **Top-down (backward-chaining)**
  - Alternative to bottom-up (forward-chaining)
Bottom-Up versus Top-Down

- Any top-down proof can be converted to a bottom-up proof.
- Any bottom-up proof can be converted to a top-down proof.
- So, top-down proof procedure is complete and sound.

- There are many other ways of doing proofs - e.g. Unit resolution - We will explore some of these later in the course.

- There are many other ways of doing proofs - e.g. Unit resolution.

- There are many other ways of doing proofs - e.g. Unit resolution.