
Overview

⇒ Semantics

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

Review

- An interpretation maps any clause to either true or false
 - It is a complete mapping
- A model I of KB is an interpretation that maps every clause in KB to true
- $KB \models g$ iff every model of KB makes g true

Example

- KB :
 - $hasdetector(foyer)$
 - $room(X) \leftarrow hasdetector(X)$
- Prove $KB \models room(foyer)$
- What does $KB \models room(foyer)$ mean?
 - Means that if interpretation I models KB then it models $room(foyer)$
 - Could prove this by checking all interpretations
- Let's do proof by contraction instead
 - Assume that I is a model of KB but not a model of $room(foyer)$

Proof by Contraction: A Semantic Proof

- Assume $I = \{D, \phi, \pi\}$ is a model of $KB = \begin{matrix} hasdetector(foyer) \\ room(X) \leftarrow hasdetector(X) \end{matrix}$
 - So $\pi(hasdetector)(\phi(foyer))$ is true
 - Say $\phi(foyer) = \langle F \rangle$, so $\langle F \rangle \in \pi(hasdetector)$ (1)
- Assume $room(foyer)$ is not true under I
 - So, $\langle F \rangle \notin \pi(room)$ (2)
- Let ρ be a variable assignment in which $\rho(X) = F$
 - $room(X) \leftarrow hasdetector(X)$ must be true under I_ρ
 - So if $hasdetector(X)$ is true for I_ρ then $room(X)$ must be true for I_ρ (3)
 - $\rho(X) = F$ and $\langle F \rangle \in \pi(hasdetector)$ so $hasdetector(X)$ is true for I_ρ (4)
 - So $room(X)$ must be true for I_ρ (from (3) and (4))
 - $\rho(X) = F$ so $\langle F \rangle \in \pi(room)$ (5)
- Contradiction from (2) and (5)

More on Variables in Clauses (pg. 42)

- Consider if $\text{parent}(X, Y) \leftarrow \text{father}(X, Y)$ is in KB
 - Implicit universal quantifiers around it
 - Anytime that $\text{father}(X, Y)$ true, so must $\text{parent}(X, Y)$
- If $\text{grandfather}(X, Y) \leftarrow \text{father}(X, Z) \wedge \text{parent}(Z, Y)$ is in KB
 - This clause is true for all X, Y, Z
 - $\forall X Y Z$ ($\text{grandfather}(X, Y) \leftarrow \text{father}(X, Z) \wedge \text{parent}(Z, Y)$).
 - Z only appears in the body
- How does Z work here (variable just in the body)?
 - For an X and Y , if we find any Z that makes body true, head must be true
 - Now it seems that Z is just existentially quantified
 - + We just need to find one Z , not make sure it is true for all Z
 - $\forall XY$ ($\text{grandfather}(X, Y) \leftarrow (\exists Z \text{father}(X, Z) \wedge \text{parent}(Z, Y))$).

Overview

- Semantics
- \Rightarrow Queries
- Proof Procedures
- Bottom-up Ground Proof Procedure
- Top-down Ground Proof Procedure

Ground Queries

- A query is a way to ask if a body is a logical consequence of the knowledge base: $? b_1 \wedge \dots \wedge b_m$
- 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 $b_1 \wedge \dots \wedge b_m$ into $yes \leftarrow b_1 \wedge \dots \wedge b_m$
 - Add (temporarily) $yes \leftarrow b_1 \wedge \dots \wedge b_m$ to KB
 - Check if yes is a logical consequence of KB
 - This lets us view queries as just finding consequences from a KB

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)

parent(X,Y) ← father(X,Y)

- Example: *?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

Overview

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

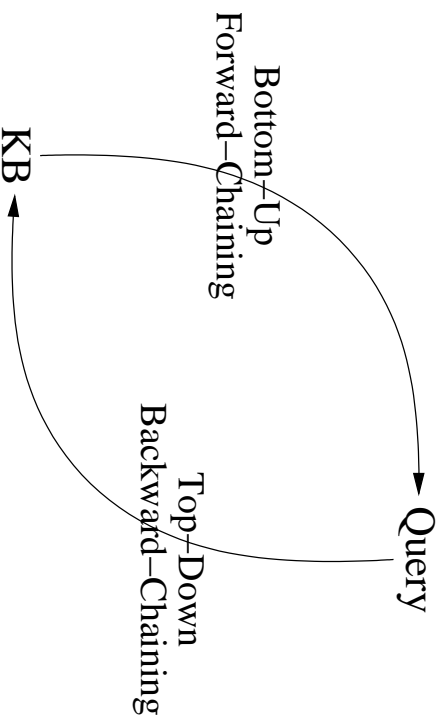
Semantics Is Not Enough

- We have **KB**
 - We know what conclusions are valid to make
 - $KB \models g$ iff g is true in all models of KB
- Can extend this so user can ask queries with variables as well
- But, don't yet have a mechanical way of checking if $KB \models g$
 - Can't access user's intended model, just the KB
 - Model checking is very expensive

Proof Procedures

- *Proof*: a mechanically derivable demonstration that a formula logically follows from a KB
- *Proof procedure*: an algorithm that constructs proofs
 - $KB \vdash g$ means g can be *derived* from KB with the proof procedure
- Proof procedure can be nondeterministic
 - So as to simplify the specification
 - Still need to specify an actual implementation
- Properties of Proof Procedure
 - Soundness*: if $KB \vdash g$ then $KB \models g$
 - Completeness*: if $KB \models g$ then $KB \vdash g$
- Terminology:
 - semantic proof: \models , logically follows, logically entails, models
 - syntactic proof: \vdash , derives

Two Types of Proof Procedures



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
- Bottom-up or forward chaining procedure:
starts from KB and works towards query
- Forward chaining rule
 - If $h \leftarrow b_1 \wedge \dots \wedge b_m$ is a clause in the KB
 - and each b_i has been derived
 - then h can be derived
- 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?

Example

- $a \leftarrow b \wedge c.$
 $b \leftarrow d \wedge e.$
 $b \leftarrow g \wedge e.$
 $c \leftarrow e.$
 $d.$
- What is the consequence set?

Is it Sound?

- Does everything in C logically follow from KB ?
- **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 $g \leftarrow b_1 \wedge \dots \wedge b_m$ in KB where the b_i 's have already been derived
 - Since g was first bad one, all b_i 's logically follow from KB
 - So $b_1 \wedge \dots \wedge b_m$ logically follows from KB (from definition of \wedge)
 - $g \leftarrow b_1 \wedge \dots \wedge b_m$ logically follows from KB since it is in the KB
 - Using definition of \leftarrow , can show that g must logically follow from KB
- Contradiction

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
 - $I(h)$ is true if $h \in C$
 - 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 ?
 - i.e. for every $g \in KB$, is $I(g)$ true?

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 $h \leftarrow b_1 \wedge \dots \wedge b_m$
 - So $h \leftarrow b_1 \wedge \dots \wedge b_m$ is false in I
 - + Remember, definition of \leftarrow comes from Datalog, not I
 - So h must be false in I and $b_1 \wedge \dots \wedge b_m$ must be true in I
 - If $b_1 \wedge \dots \wedge b_m$ is true in I , each individually must be true in I
 - + Remember, definition of \wedge comes from Datalog, not I
 - So, all of the b_i must be in C (due to how we defined I)
 - Since all b_i in C and $h \leftarrow b_1 \wedge \dots \wedge b_m$ is in KB
bottom up algorithm must have applied this rule and hence $h \in C$
 - Hence h is true in I
- Contradiction

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 $KB \vdash g$

Overview

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

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

$$\frac{\frac{yes \leftarrow a_1 \wedge \dots \wedge a_m}{a_i \leftarrow b_1 \wedge \dots \wedge b_p}}{yes \leftarrow a_1 \wedge \dots \wedge a_{i-1} \wedge b_1 \wedge \dots \wedge b_p \wedge a_{i+1} \wedge \dots \wedge a_m}}$$

Now for some definitions

- *Answer clause is* $yes \leftarrow a_1 \wedge \dots \wedge a_m$
- *Answer is answer clause with* $m = 0$
- *Derivation of a query* $?q_1 \wedge \dots \wedge q_k$ from KB is a sequence of answer clauses $\gamma_0, \gamma_1, \dots, \gamma_n$
 - γ_0 is the answer clause corresponding to the original query
 - γ_i is obtained by resolving γ_{i-1} with a clause in KB
 - γ_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)

Example

- KB
 - $a \leftarrow b \wedge c.$
 - $b \leftarrow d \wedge e.$
 - $b \leftarrow g \wedge e.$
 - $c \leftarrow e.$
 - $d.$
 - $e.$
 - $f \leftarrow a \wedge g.$
 - $?a.$

Bottom-Up versus Top-Down

KB	Top-Down	KB Rule	Bottom-Up
$a \leftarrow b \wedge c$	$yes \leftarrow a$	$a \leftarrow b \wedge c$	$a \in C$
$b \leftarrow d \wedge e$	$yes \leftarrow b \wedge c$	$b \leftarrow d \wedge e$	$C = \{e, c, d, b, a\}$
$b \leftarrow g \wedge e$	$yes \leftarrow d \wedge e \wedge c$	$b \leftarrow d \wedge e$	$C = \{e, c, d, b\}$
$c \leftarrow e$	$yes \leftarrow e \wedge c$	d	$C = \{e, c, d\}$
d	$yes \leftarrow c$	c	$C = \{e, c\}$
e	$yes \leftarrow e$	$c \leftarrow e$	$C = \{e, c\}$
$f \leftarrow a \wedge g$	$yes \leftarrow$	e	$C = \{e\}$
$?a$			

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
 - But top-down and bottom-up are sufficient for datalog