CSC 442: ARTIFICIAL INTELLIGENCE

LEC 12: FIRST-ORDER LOGIC

LAST TIME

- Resolution proof applied to “three roads” problem.
-

THE LABYRINTH (OF LIARS)

- The guardian of the gold street: “This road will bring you straight to the center. Moreover, if the stones take you to the center, then also the marble takes you to the center.”

$$\neg(G \wedge (S \Rightarrow M))$$

- The guardian of the marble street: “Neither the gold nor the stones will take you to the center.”

$$\neg(\neg G \wedge \neg S)$$

- The guardian of the stone street: “Follow the gold and you’ll reach the center, follow the marble and you will be lost.”

$$\neg(G \wedge \neg M)$$

THE LABYRINTH (TO CNF)

$$\neg(G \wedge (S \Rightarrow M))$$

$$\neg(G \wedge (\neg S \vee M))$$

$$\neg G \vee \neg(\neg S \vee M)$$

$$\neg G \vee (S \wedge \neg M)$$

$$(\neg G \vee S) \wedge (\neg G \vee \neg M)$$

$$[\neg G, S] \qquad [\neg G, \neg M]$$

$$\neg(\neg G \wedge \neg S)$$

$$G \vee S$$

$$[G, S]$$

$$\neg(G \wedge \neg M)$$

$$\neg G \vee M$$

$$[\neg G, M]$$

- Let's prove the stone path is the good one: S

1 $[\neg G, S]$

2 $[\neg G, \neg M]$

3 $[G, S]$

4 $[\neg G, M]$

5 $[\neg S]$ ← Negated goal!

6 $[S]$ *1,3*

7 $[]$ *5,6*

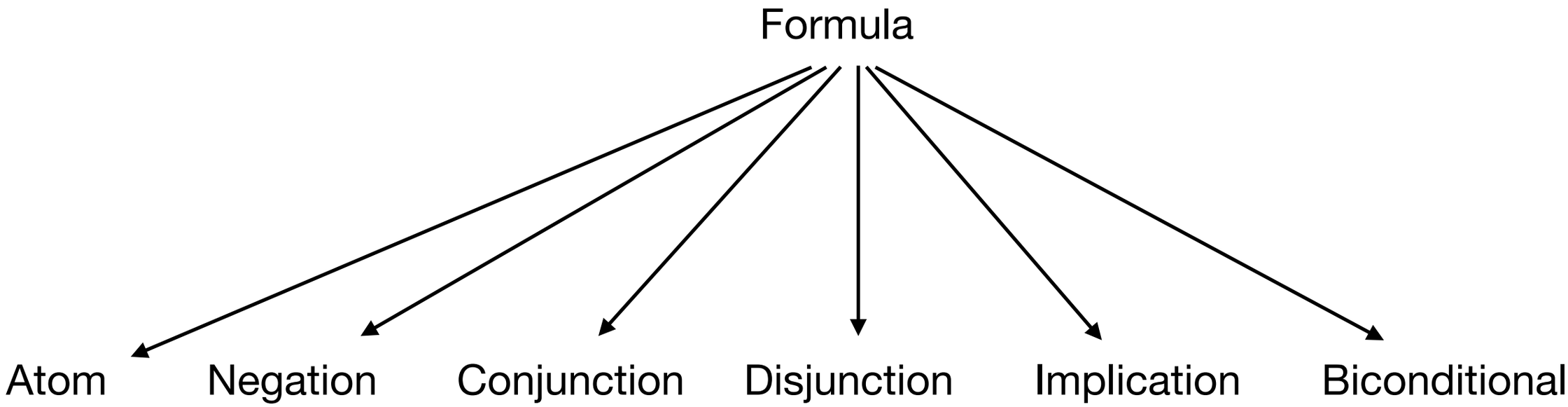
Therefore

$$\Delta \wedge \neg S$$

is unsatisfiable.

So, $\Delta \models S$

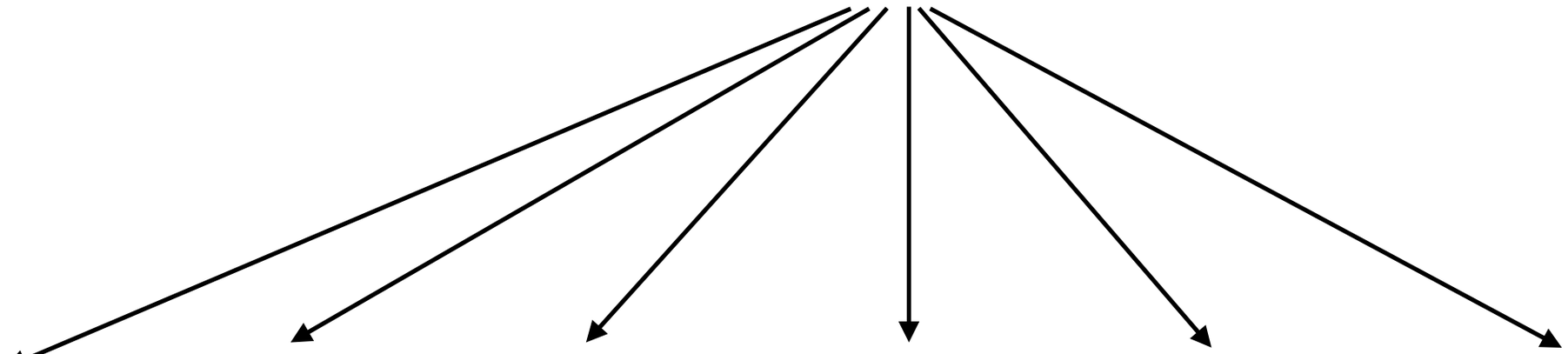
Classes for Propositional Theorem Proving



Methods

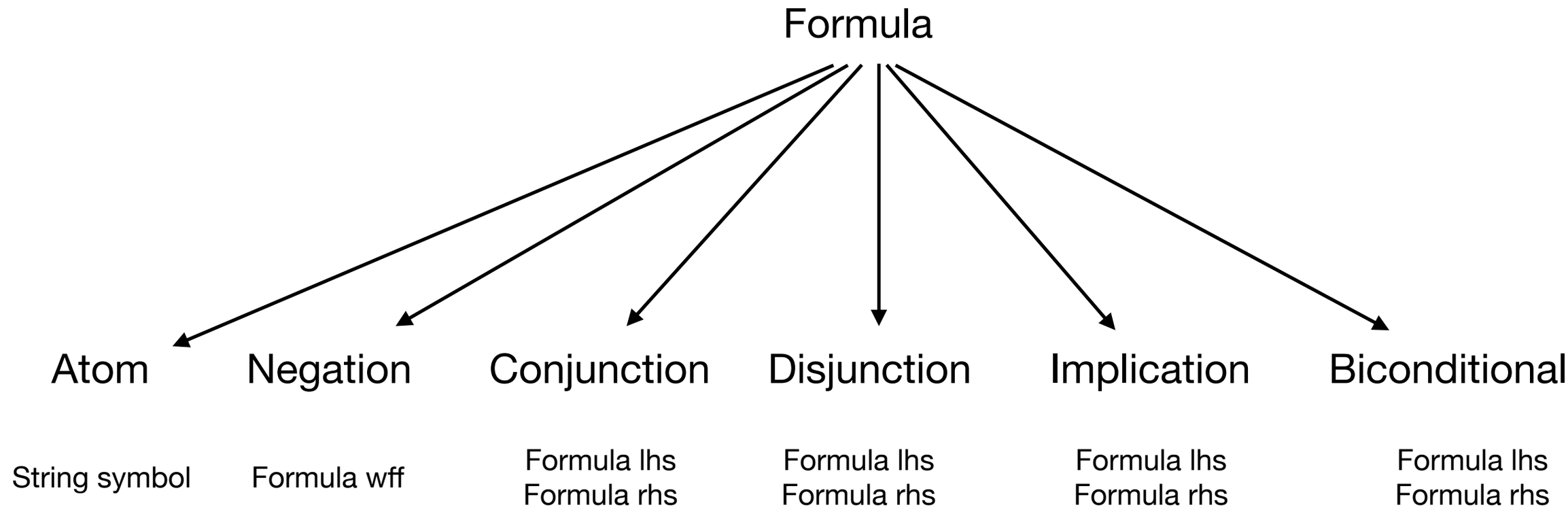
toStr models symbols **fromStr**

Formula

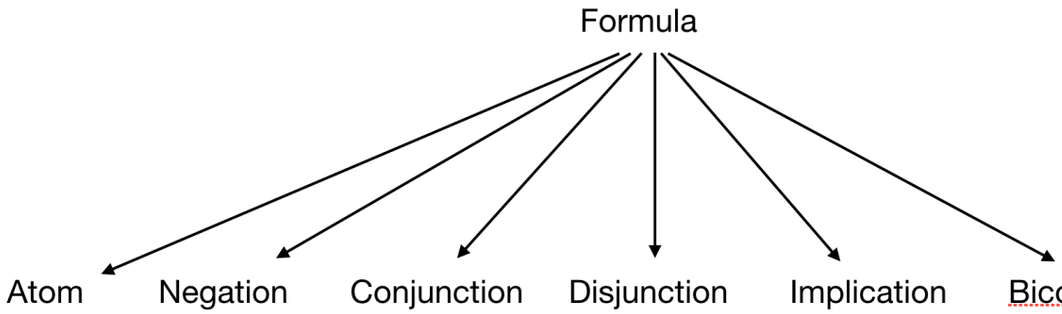


Atom	Negation	Conjunction	Disjunction	Implication	Biconditional
toStr	**toStr**	**toStr**	**toStr**	**toStr**	**toStr**
models	**models**	**models**	**models**	**models**	**models**
symbols	**symbols**	**symbols**	**symbols**	**symbols**	**symbols**
fromSeq	**fromSeq**	**fromSeq**	**fromSeq**	**fromSeq**	**fromSeq**

Instance Variables



Additional Classes



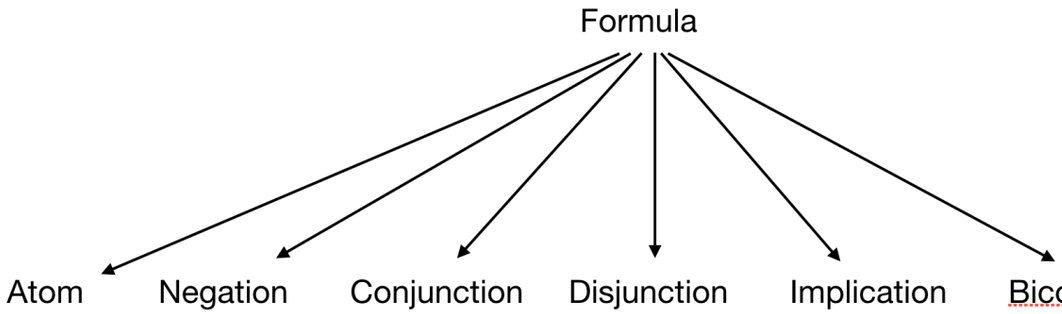
Driver

Methods: ask/tell

Class variable: knowledge base

Role: user interaction

Resolution



Driver

Methods: ask/tell

Class variable: knowledge base

Role: user interaction

Extend Formula with toCNF() method.

OOP Challenge: Negation.toCNF needs to know the type of child!

(Could have all implement moveNegationIn()?)

toCNF(moveNegationIn)

then just need to know if Atom or not…

FIRST ORDER LOGIC

FIRST ORDER LOGIC

Formulas

Terms

Functions

Relations

Quantifiers

Truth Conditions

Models = Domain + Interpretation

Validity Satisfiability

Entailment

LANGUAGE AND THOUGHT

- ``We cut nature up, organize it into concepts, and ascribe significances as we do, largely because we are parties to an agreement to organize it they way – an agreement that holds throughout our speech community and is codified in the patterns of our language.'' – Benjamin Whorf (1956)

LANGUAGE INFLUENCES THOUGHT

- Loftus and Palmer (1974) showed experimental subjects a movie of an auto accident.
- "How fast were the cars going when they **contacted** each other?" – 32mph
- "How fast were the cars going when they **smashed into** each other?" – 41mph

FIRST ORDER LOGIC

FIRST ORDER LO

FIRST ORDER LOGIC

- First order logic generalizes propositional logic by adding **objects**, **relations**, and **quantifiers**.
- First Order Logic (FOL) is a generalization of propositional logic which incorporates **quantifiers**, **terms**, **variables**, and **relations** (also called **predicates**).

FIRST ORDER LOGIC

- First order logic generalizes propositional logic by adding **objects**, **relations**, and **quantifiers**.
- Objects are usually called **terms**. An object can be any thing. Examples: the number 3, yourself, the city of Rochester, the projector we're using right now, my dog, ...
- Those objects all had names. We can also represent terms as **functions** from other terms. Examples: mother-of, square-of, next-prime, ...

FIRST ORDER LOGIC

- First order logic generalizes propositional logic by adding **objects**, **relations**, and **quantifiers**.
- **Relations** are sometimes called **predicates**. So first-order logic is also called predicate logic. It is also called the first-order predicate calculus (FOPC).
- Relations are properties which are true of a term, or of an (ordered) set of terms. Examples: Brothers, Green, Hungry, Student, Friends,...

FIRST ORDER LOGIC

- First order logic generalizes propositional logic by adding **objects**, **relations**, and **quantifiers**.

- We can also represent arbitrary terms as **variables**.
  Examples: x, y, z, …

- Which is helpful when we want to use **quantifiers**, which state something about a formula with respect to a set of objects. Examples: some, all, every, none, …

FIRST-ORDER LOGIC

- Terms
- Predicates/Relations
- Variables
- Quantifiers
- Formulas

- First Order Models (Domain + Interpretive Mapping)
- Entailment.. truth in all models.
- Truth
- Symbols (Constant, predicate, function)

FIRST-ORDER SYNTAX

- Like propositional logic, first-order logic defines a **language** with useful properties.

SENTENCES

- An atomic sentence (atom) is a predicate symbol followed by a list of terms.
  - Brother(Richard, John)
  - Married(Father(Richard), Mother(John))

- An atomic sentence is true in a given model if the relation referred to by the predicate symbol holds among the objects referred to by the arguments. (AIMA, p295)

SENTENCES

- As in propositional logic, **complex sentences** can be built out of other sentences and logical connectives (and, or, not, implies, biconditionals).

- Complex sentences built from connectives are true in a given model according to the similar conditions as their propositional counterparts.

SENTENCES

- First order logic also allows quantified sentences as a special kind of complex sentence. Examples:
  - All dogs are animals.
  - No robot is evil.
  - Some cat is friendly.
- Quantified sentences are true according to truth conditions specific to each quantifier. More on quantifiers in a moment.

A GRAMMAR FOR FIRST-ORDER LOGIC

$$
\begin{array}{rcl}
\mathit{Sentence} & \rightarrow & \mathit{AtomicSentence} \mid \mathit{ComplexSentence} \\
\mathit{AtomicSentence} & \rightarrow & \mathit{Predicate} \mid \mathit{Predicate}(\mathit{Term}, \ldots) \mid \mathit{Term} = \mathit{Term} \\
\mathit{ComplexSentence} & \rightarrow & (\ \mathit{Sentence}\ ) \mid [\ \mathit{Sentence}\ ] \\
 & \mid & \neg\ \mathit{Sentence} \\
 & \mid & \mathit{Sentence} \wedge \mathit{Sentence} \\
 & \mid & \mathit{Sentence} \vee \mathit{Sentence} \\
 & \mid & \mathit{Sentence} \Rightarrow \mathit{Sentence} \\
 & \mid & \mathit{Sentence} \Leftrightarrow \mathit{Sentence} \\
 & \mid & \mathit{Quantifier}\ \mathit{Variable}, \ldots\ \mathit{Sentence} \\
 & & \\
\mathit{Term} & \rightarrow & \mathit{Function}(\mathit{Term}, \ldots) \\
 & \mid & \mathit{Constant} \\
 & \mid & \mathit{Variable} \\
\mathit{Quantifier} & \rightarrow & \forall \mid \exists \\
\mathit{Constant} & \rightarrow & A \mid X_1 \mid \mathit{John} \mid \cdots \\
\mathit{Variable} & \rightarrow & a \mid x \mid s \mid \cdots \\
\mathit{Predicate} & \rightarrow & \mathit{True} \mid \mathit{False} \mid \mathit{After} \mid \mathit{Loves} \mid \mathit{Raining} \mid \cdots \\
\mathit{Function} & \rightarrow & \mathit{Mother} \mid \mathit{LeftLeg} \mid \cdots
\end{array}
$$

AIMA Fig 8.3

FIRST-ORDER MODELS

- A first-order model consists of a **domain** and an **interpretive mapping**.
- The domain specifies the set of objects. Examples: dogs, people, numbers, students, colors, cities, ...
- The interpretive mapping identifies the meaning of terms, the meaning of relations, and the truth of sentences.

$$\mathcal{I}_R \subseteq \mathcal{D}x\mathcal{D} \qquad \mathcal{I}_c \in \mathcal{D}$$

$$\mathcal{I}_P \subseteq \mathcal{D} \qquad [\mathcal{I}_f](x)\mathcal{D} \rightarrow \mathcal{D}$$

ENUMERATING MODELS

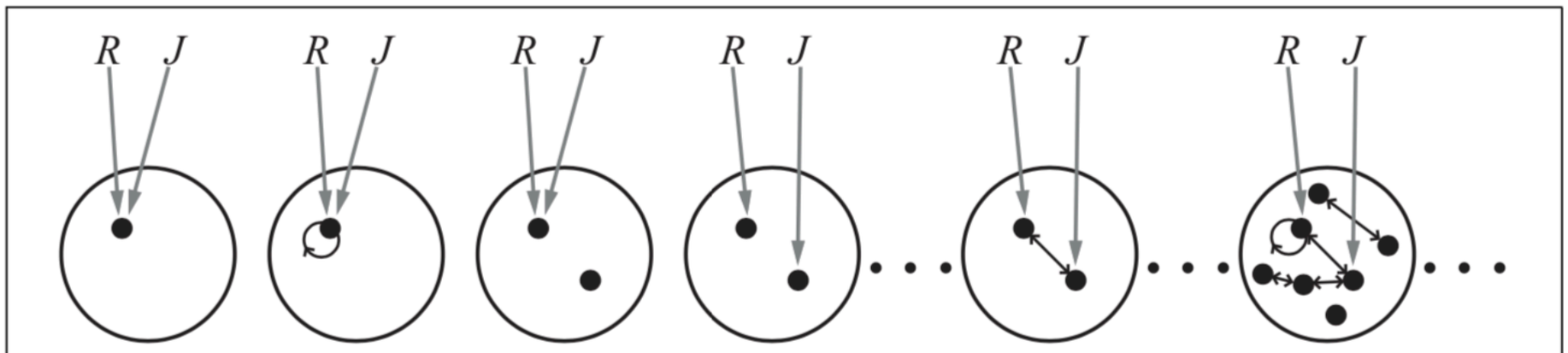


Figure 8.4 Some members of the set of all models for a language with two constant symbols, R and J, and one binary relation symbol. The interpretation of each constant symbol is shown by a gray arrow. Within each model, the related objects are connected by arrows.

AIMA Fig 8.4

QUANTIFIERS

- Quantifiers are words which specify a scope of objects which are under consideration.
- There are many quantifiers in English:
  - All, Some, None/No
  - Many, Most, Few
  - Rarely, Usually, Almost all,

QUANTIFIERS

- First-order logic supports two quantifiers:
  - The Universal Quantifier – All – $\forall$
  - The Existential Quantifier – Some – $\exists$

EXAMPLE QUANTIFIED SENTENCES

- All dogs are canines.

$$\forall x \ \mathrm{Dog}(x) \Rightarrow \mathrm{Canine}(x).$$

- Some dog is black.

$$\exists x \ \mathrm{Dog}(x) \wedge \mathrm{Black}(x).$$

- No dog can fly.

$$\neg \exists x \ \mathrm{Dog}(x) \wedge \mathrm{CanFly}(x).$$

$$\forall x \ \mathrm{Dog}(x) \Rightarrow \neg \mathrm{CanFly}(x).$$

STOPPED HERE WEDNESDAY

NESTING AND ITERATING QUANTIFIERS

- Everybody loves somebody.

$$\forall x \; \exists y \; \mathrm{Loves}(x, y).$$

- There is someone who is loved by everyone.

$$\exists y \; \forall x \; \mathrm{Loves}(x, y).$$

CONNECTION BETWEEN THE UNIVERSAL AND THE EXISTENTIAL

$$\neg \forall x \ \phi \equiv \exists x \ \neg \phi$$

$$\neg \exists x \ \phi \equiv \forall x \ \neg \phi$$

EQUALITY

- First-order logic also typically includes **term equality**.
- For example,

$$Father(John) = Henry$$

EQUALITY

- Equality can be used to assert uniqueness and identity. For example, “Richard’s brothers are John and Geoffrey” can be written as:

$$Brother(John, Richard) \wedge Brother(Geoffrey, Richard) \wedge John \neq Geoffrey$$
$$\wedge \forall x \;\; Brother(x, Richard) \Rightarrow (x = John \vee x = Geoffrey) \; .$$

EXAMPLES OF FIRST-ORDER LOGIC

- UR is a university in Rochester, NY.
- Sisterhood is a reflexive relationship.
- If a pet dog is treated well, then it is friendly.
- A natural number is prime if there does not exist any other natural number other than itself or one which evenly divides the number.

EXAMPLES OF FIRST-ORDER LOGIC

- UR is a university in Rochester, NY.

$$\text{LocatedIn}(\text{UR}, \ \text{RochesterNY})$$

EXAMPLES OF FIRST-ORDER LOGIC

- Sisterhood is a reflexive relationship.

$$\forall x \ \forall y \ \text{Sister}(x, y) \Leftrightarrow \text{Sister}(y, x).$$

EXAMPLES OF FIRST-ORDER LOGIC

- If a pet dog is treated well, then it is friendly.

$$\forall x(\mathrm{Dog}(x) \wedge \mathrm{TreatedWell}(x)) \Rightarrow \mathrm{Friendly}(x).$$

EXAMPLES OF FIRST-ORDER LOGIC

- A natural number is prime if there does not exist any other natural number other than itself or one which evenly divides the number.

$$\forall n\,[(NN(x) \wedge \neg(\exists k((k \neq n) \wedge (k \neq 1) \wedge \text{Divides}(n, k)))] \Rightarrow \text{Prime}(n).$$

"DATABASE SEMANTICS"

- First-order models can be difficult to specify thoroughly
- Three common assumptions are the **unique-names** assumption, the **closed-world** assumption, and the **domain closure** assumption.

TRUTH AND INFERENCE

- An atomic sentence is true in a given model if the relation referred to by the predicate symbol holds among the objects referred to by the arguments.
- Complex sentences built from connectives are true in a given model according to the similar conditions as their propositional counterparts.

TRUTH CONDITIONS FOR QUANTIFIED SENTENCES

- A universally quantified sentence is true in a model (domain + interpretation) iff the sentence is true under **every possible substitution of domain element for the variables.**

- An existentially quantified sentence is true in a model iff the sentence is true for **at least one** substitution of domain element for the variables.

FIRST-ORDER ENTAILMENT

- If every model M (a domain and interpretive mapping) of a formula α is also a model of a formula β, then we say that α entails β and write:

$$\alpha \models \beta$$

FIRST-ORDER ENTAILMENT

- If **every model** M (a domain and interpretive mapping) of a formula α is also a model of a formula β, then we say that α entails β and write:

$$\alpha \models \beta$$

How do we identify when $\alpha \models \beta$?

- Can have infinite domains, so model checking not enough.
- Interesting result – Herbrand's theorem – even with an infinite domain, if a formula is entailed by a knowledge base, then it can be proved using a finite subset of the domain.
- Briefly: A formula is unsatisfiable if and only if there is a finite subset of ground clauses which are unsatisfiable as propositional logic.

- However, if $\alpha \not\models \beta$ then Herbrand's theorem does not apply, and there may not be a finite proof!
- So first-order entailment is only **semi-decidable**.

- Database semantics (domain closure esp.) can be enough to perform model checking for small enough domains.
- Let's consider **theorem proving** instead.

FIRST ORDER INFERENCE

- Inference Rules (a la *state-space search*)
  - Universal Instantiation
  - Existential Instantiation

FIRST ORDER INFERENCE

- Universal Instantiation – we can infer any sentence obtained by substituting a ground term for a variable.

$$\frac{\forall\ v\ \alpha}{\mathrm{Subst}(\{v/g\}, \alpha)} \quad \text{For} \quad \theta = \{v/g\}$$

- Example:
  "If every tool is on sale, then wrenches are on sale"

$$\forall x\ \mathrm{Tool}(x) \Rightarrow \mathrm{OnSale}(x).$$

$$\mathrm{Tool}(wrench) \Rightarrow \mathrm{OnSale}(wrench)$$

- Existential Instantiation – if k is a symbol that does not appear anywhere else in a knowledge base, then

$$\frac{\exists\ v\ \alpha}{\mathrm{Subst}(\{v/k\}, \alpha)}$$

- Example:
  "There is a black cat."

$$\exists\ x\ \mathrm{Black}(x) \wedge \mathrm{Cat}(x).$$

$$\mathrm{Black}(t_{723}) \wedge \mathrm{Cat}(t_{723})$$

The new term is called a **Skolem constant.**

- When an existential is embedded in a universal, then instead of a Skolem constant we need a **Skolem function**.
- Example:

"Every dog has a bone."

$$\forall\ x\ \mathrm{Dog}(x) \Rightarrow \exists\ y\ \mathrm{Bone}(y) \wedge \mathrm{Has}(x, y).$$

$$\forall\ x\ \mathrm{Dog}(x) \Rightarrow [\mathrm{Bone}(f(x)) \wedge \mathrm{Has}(x, f(x))].$$

f(x) is a Skolem function – "the bone belonging to x"

- Note: a **skolemized** knowledge base is not logically equivalent to the original knowledge base, but it is **inferentially equivalent** – it is satisfiable if and only if the original knowledge base is satisfiable.

PROPOSITIONALIZATION

- We can reduce a first-order knowledge base to a propositional knowledge base by aggressively applying universal and existential instantiation.
- This is called **propositionalization**.

PROPOSITIONALIZATION

$$\text{Number}(0)$$

$$\forall\ n\ \text{Number}(n) \Rightarrow \text{Number}(\text{Successor}(n)).$$

$$\text{Number}(0) \Rightarrow \text{Number}(\text{Successor}(0))$$

$$\text{Number}(\text{Successor}(0)) \Rightarrow \text{Number}(\text{Successor}(\text{Successor}(0)))$$

$$\text{Number}(\text{Successor}(\text{Successor}(0))) \Rightarrow \text{Number}(\text{Successor}(\text{Successor}(\text{Successor}(0))))$$

...

HERBRAND'S THEOREM

- Jaques Herbrand (1930) proved that if a sentence is entailed by a first-order knowledge base, then there is a proof involving a **finite subset** of the propositionalized knowledge base.

- So we can try generating all terms at depth 0, depth 1, ..., and so on, until we find a proof!

HERBRAND'S THEOREM

- ... unless the sentence is not entailed, in which case we may keep successively trying deeper and deeper searches, never quite knowing if we need "just one more step".

- Entailment in first-order logic is **semi-decidable.** Algorithms exist which say "yes" to every entailed sentence, but no algorithm exists which also says "no" to every non entailed sentence.

FIRST ORDER INFERENCE

- Option 1 – propositionalization + something like iterative deepening
- Option 2 – ``Lifting'' inference to work directly in the first-order theory.

GENERALIZED MODUS PONENS — DEFINITION

Given atomic sentences p_i, p'_i, and q,

where there is a substitution θ such that

$$\text{Subst}(\theta, p'_i) = \text{Subst}(\theta, p_i) \text{ for all } i, \text{ then}$$

$$\frac{p'_1, p'_2, \ldots, p'_n, (p_1 \wedge p_2 \wedge \ldots \wedge p_n \Rightarrow q)}{\text{Subst}(\theta, q)}$$

GENERALIZED MODUS PONENS — EXAMPLE

$$\forall\, x\ \mathrm{King}(x) \wedge \mathrm{Greedy}(x) \Rightarrow \mathrm{Evil}(x).$$

$$\mathrm{King}(\mathrm{John})$$

$$\forall y\ \mathrm{Greedy}(y).$$

$$\frac{p_1', p_2', \ldots, p_n', (p_1 \wedge p_2 \wedge \ldots \wedge p_n \Rightarrow q)}{\mathrm{Subst}(\theta, q)}$$

p_1' is $King(John)$	p_1 is $King(x)$
p_2' is $Greedy(y)$	p_2 is $Greedy(x)$
θ is $\{x/John, y/John\}$	q is $Evil(x)$

SUBST(θ, q) is $Evil(John)$.

UNIFICATION

- We therefore need an algorithm for determining, for any pair of formulas p and p', whether there exists some set of substitutions θ such that:

$$\text{Subst}(\theta, p) = \text{Subst}(\theta, p')$$

UNIFICATION

- Basic premise,
- Standardizing Apart,
- Most general unifier,
- Occurs-check

UNIFICATION — BASIC PREMISE

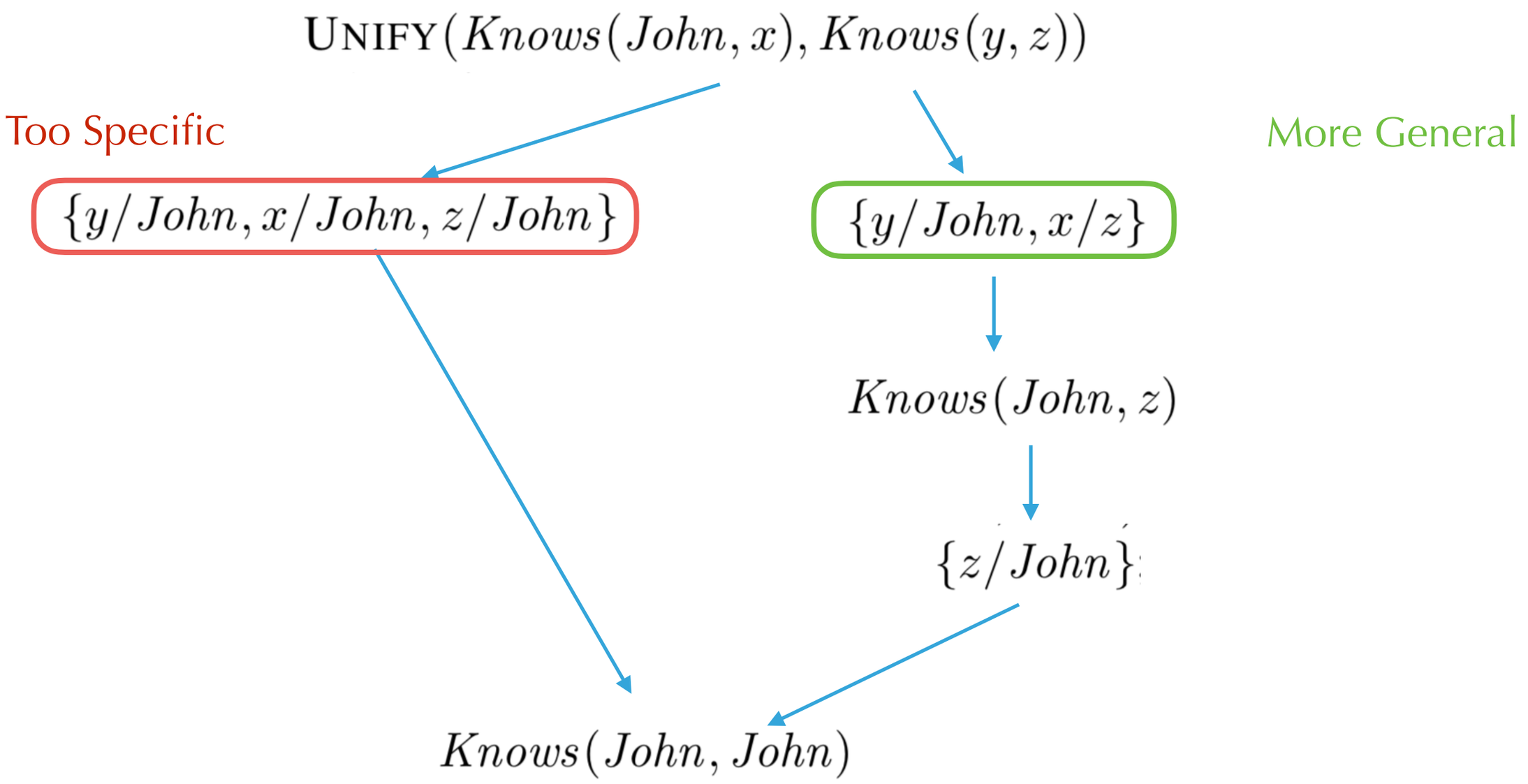
$$\textsc{Unify}(p, q) = \theta \text{ where } \textsc{Subst}(\theta, p) = \textsc{Subst}(\theta, q) \ .$$

$\textsc{Unify}(Knows(John, x),\ Knows(John, Jane)) = \{x/Jane\}$
$\textsc{Unify}(Knows(John, x),\ Knows(y, Bill)) = \{x/Bill, y/John\}$
$\textsc{Unify}(Knows(John, x),\ Knows(y, Mother(y))) = \{y/John, x/Mother(John)\}$
$\textsc{Unify}(Knows(John, x),\ Knows(x, Elizabeth)) = fail \ .$

UNIFICATION — STANDARDIZING APART

$$\textsc{Unify}(Knows(John, x),\ Knows(x_{17}, Elizabeth)) = \{x/Elizabeth, x_{17}/John\}\ .$$

UNIFICATION — MOST GENERAL UNIFIER



function UNIFY(x, y, θ) **returns** a substitution to make x and y identical
  inputs: x, a variable, constant, list, or compound expression
    y, a variable, constant, list, or compound expression
    θ, the substitution built up so far (optional, defaults to empty)

  if θ = failure **then return** failure
  else if $x = y$ **then return** θ
  else if VARIABLE?(x) **then return** UNIFY-VAR(x, y, θ)
  else if VARIABLE?(y) **then return** UNIFY-VAR(y, x, θ)
  else if COMPOUND?(x) **and** COMPOUND?(y) **then**
    return UNIFY(x.ARGS, y.ARGS, UNIFY(x.OP, y.OP, θ))
  else if LIST?(x) **and** LIST?(y) **then**
    return UNIFY(x.REST, y.REST, UNIFY(x.FIRST, y.FIRST, θ))
  else return failure

AIMA Fig 9.1, pp 328

function UNIFY-VAR(var, x, θ) **returns** a substitution

if $\{var/val\} \in \theta$ **then return** UNIFY(val, x, θ)
else if $\{x/val\} \in \theta$ **then return** UNIFY(var, val, θ)
else if OCCUR-CHECK?(var, x) **then return** failure
else return add $\{var/x\}$ to θ

AIMA Fig 9.1, pp 328

FIRST ORDER RESOLUTION

- Convert conjunction of KB and negated query to CNF.
- Apply resolution until the empty clause is derived, or until no more new resolvents can be found.

FIRST ORDER RESOLUTION

- Convert to CNF –
  - Eliminate implications.
  - Move negation inwards.
  - Standardize variables.
  - Skolemize.
  - Drop universal quantifiers.
  - Distribute disjunction over conjunction.