Discussion 9

Shirley Chen. Sajad Darabi

Propositional Logic (Boolean Logic)

- Syntax
  - Propositional symbols (**atomic sentences**): A, B, C
  - Logical connectives *:* $\neg \wedge \vee \rightarrow \leftrightarrow$

- It is common to use standard lower-case roman letters to denote propositions
  - $p, q, r, \ldots$

First-order Logic

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

First-Ordre Logic

- Objects (terms)
  - Constant: Mary
  - Variable: x, y, z
- Predicates:
  - properties (unary) (True/False)
    - UCLA_student(Mary)
      - Can be True or False
  - relations (n-ary)
    - Loves(Richard, Dog_of_Richard)
    - Brother(Richard, John)
      - True or False

First-Ordre Logic

- Sentence
  - Atomic sentence (one predicate)
    - Owns(John, Car1)
    - Sold(John, Car1, Tom)
  - Complex sentence
    - Owns(John, Car1) $\land$ Owns(John, Car2)
    - Sold(John, Car1, Tom) $\Longrightarrow \neg$ Owns(John, Car1)

Models for FOL

- Each model links the vocabulary of the logical sentences to elements of the possible world, so that the truth of any sentence can be determined.

- Domain of a model (Must be non-empty)
  - The set of objects or **domain elements** it contains

(Will get into details later)

First-Ordre Logic

Express properties of QUANTIFIER entire collections of objects, instead of enumerating the objects by name.

- Quantifiers
  - Universal quantification $\forall$ (For all)
    - $\forall x \ \mathrm{King}(x) \Rightarrow \mathrm{Person}(x)$
    - Naturally uses $\Rightarrow$
  - Existential quantification $\exists$ (There exists)
    - $\exists x \ \mathrm{King}(x) \ \land \mathrm{OlderThan30}(x)$
    - Naturally uses $\land$

Exercise

Are they equivalent? What do they mean?

- $\forall x\ King(x) \Rightarrow Person(x)$
- $\forall x\ King(x) \land Person(x)$

- $\exists x\ King(x) \land OlderThan30(x)$
- $\exists x\ King(x) \Rightarrow OlderThan30(x)$

- Given:

○ Three persons:

Richard (King, 50 years old)

John (Richard's brother, 20 years old)

Elizabeth (Richard's mother)

○ A dog:

Gigi (Richard's dog)

First-Ordre Logic

Nesting quantifiers

- Same type quantifiers: order doesn't matter
  - $\forall x \forall y\ (\mathrm{Paren}(x, y) \land \mathrm{Male}(y) \Rightarrow \mathrm{Son}(y, x))$
  - $\exists\ x \exists y\ (\mathrm{Loves}(x, y) \land \mathrm{Loves}(y, x))$
    - $\exists\ x, y\ (\mathrm{Loves}(x, y) \land \mathrm{Loves}(y, x))$

- Mixed quantifiers: order does matter
  - $\forall\ x \exists y\ (\mathrm{Loves}(x, y))$
    - Everybody has someone they love.
  - $\exists y \forall\ x(\mathrm{Loves}(x, y))$
    - There is someone who is loved by everyone.
  - $\forall\ y \exists x\ (\mathrm{Loves}(x, y))$
    - Everybody has someone who loves them.
  - $\exists x \forall\ y(\mathrm{Loves}(x, y))$
    - There is someone who loves everyone.

More about quantifiers

- Variable scope
  - The **scope** of a variable is the sentence to which the quantifier syntactically applies.
    - $\forall \mathbf{x}\ \mathrm{King}(x) \Rightarrow \mathrm{Person}(x)$
    - $\forall \mathbf{x}\ \mathrm{King}(x) \lor (\exists\ \mathbf{x}\ \mathrm{Brother}(x, \mathrm{Richard}))$
      - The variable belongs to the **innermost** quantifier that mentions it. Then it will not be subject to any other quantification.
      - Equivalent sentence: $\exists z\ \mathrm{Brother}(z, \mathrm{Richard})$
      - Cause confusion. Not recommended.

  - Not well-formed
    - $\exists x\ P(y)$
      - All variables should be properly introduced!

  - **Ground expression**
    - No variables
    - King(Richard) ⇒ Person(Richard)

More about quantifiers

- ∀ and ∃

$$
\begin{aligned}
\forall x \;\; \neg P &\equiv \neg\exists x \;\; P \\
\neg\forall x \;\; P &\equiv \exists x \;\; \neg P \\
\forall x \;\; P &\equiv \neg\exists x \;\; \neg P \\
\exists x \;\; P &\equiv \neg\forall x \;\; \neg P
\end{aligned}
\qquad\qquad
\begin{aligned}
\neg(P \lor Q) &\equiv \neg P \land \neg Q \\
\neg(P \land Q) &\equiv \neg P \lor \neg Q \\
P \land Q &\equiv \neg(\neg P \lor \neg Q) \\
P \lor Q &\equiv \neg(\neg P \land \neg Q)\,.
\end{aligned}
$$

$\forall x \; \mathrm{King}(x) \Rightarrow \mathrm{Person}(x)$

$\neg\exists x \;\; \neg\,(\mathrm{King}(x) \Rightarrow \mathrm{Person}(x))$

First-Ordre Logic

- Equality $=$ (identity relation)
  - $\mathrm{Mother}(\mathrm{Richard}) = \mathrm{Elizabeth}$

  - $\exists\ \mathrm{x}, y\ \mathrm{Brother}(\mathrm{x}, \mathrm{Richard}) \wedge \mathrm{Bother}(\mathrm{y}, \mathrm{Richard}) \wedge (x \neq y)$
    - Richard has (at least) two brothers

Exercise

Are they equivalent? What do they mean?

- $\exists$ x, y Brother(x, Richard) $\wedge$ Bother(y, Richard)
- $\exists$ x, y Brother(x, Richard) $\wedge$ Bother(y, Richard) $\wedge (x \neq y)$

Consider the following cases:

1) Richard only has one brother John
2) Richard has two brothers: John and Tom

Exercise

Translate into FOL:

Everyone has exactly one mother.

- $\mathrm{Mother}(y, x)$ means y is the mother of x

Exercise

Translate into FOL:

Everyone has <u>exactly one</u> mother.

- $\mathrm{Mother}(\mathrm{y}, \mathrm{x})$ means y is the mother of x

- $\forall \mathrm{x}\ \exists y\ \mathrm{Mother}(\mathrm{y}, \mathrm{x})$?
  - Everyone has (at least one) mother.

- $\forall \mathrm{x}\ \exists \mathrm{y}\ \mathrm{Mother}(\mathrm{y}, \mathrm{x}) \wedge (\forall \mathbf{z}\ \mathbf{Mother}(\mathbf{z}, \mathbf{x}) \Rightarrow \mathbf{y} = \mathbf{z})$

Exercise – Translating English to FOL

- Every gardener likes sunshine

- You can fool some people all the time.
- You can fool all the people some of the time.

- There is a barber in town who shaves all men in town who do not shave themselves.
- There is a barber in town who shaves only and all men in town who do not shave themselves.

Grounding

- When is $\forall x\ P$ True?
  - If P is true for every object x.
  - If P is true in all possible *extended interpretations*.

- Given a model, how to determine if $\forall x\ P$ is true?

Example - Grounding

Given the following model:

- Three persons

Richard (King, 50 years old)

John (Richard's brother, 20 years old)

Elizabeth (Richard's mother)

- A dog:

Gigi (Richard's dog)

Determine if this is true:
$\forall x \, \mathrm{King}(x) \Rightarrow \mathrm{Person}(x)$

Example - Grounding

Determine if this is true:
$\forall x\ King(x) \Rightarrow Person(x)$

What do we do?

1. Extend the interpretation:

$x \rightarrow$ Richard

$x \rightarrow$ John

$x \rightarrow$ Elizabeth

$x \rightarrow$ Gigi

Example - Grounding

What do we do?

2. Compute the propositional grounding

$$\mathrm{King}(\mathrm{Richard}) \Rightarrow \mathrm{Person}(\mathrm{Richard})$$
$$\mathrm{King}(\mathrm{John}) \Rightarrow \mathrm{Person}(\mathrm{John})$$
$$\mathrm{King}(\mathrm{Elizabeth}) \Rightarrow \mathrm{Person}(\mathrm{Elizabeth})$$
$$\mathrm{King}(\mathrm{Gigi}) \Rightarrow \mathrm{Person}(\mathrm{Gigi})$$

Example - Grounding

What do we do?

2. Compute the propositional grounding

$$
\begin{aligned}
&(\quad King(Richard) \Rightarrow Person(Richard) \quad) \wedge \\
&(\quad King(John) \Rightarrow Person(John) \quad) \wedge \\
&(\quad King(Elizabeth) \Rightarrow Person(Elizabeth) \quad) \wedge \\
&(\quad King(Gigi) \Rightarrow Person(Gigi) \quad)
\end{aligned}
$$

Example - Grounding

What do we do?

3. See if the new sentence is True

$$
\begin{array}{lll}
(\quad King(Richard) \Rightarrow Person(Richard) &) \land & \text{True} \\
(\quad King(John) \Rightarrow Person(John) &) \land & \text{True} \\
(\quad King(Elizabeth) \Rightarrow Person(Elizabeth) &) \land & \text{True} \\
(\quad King(Gigi) \Rightarrow Person(Gigi) &) & \text{True}
\end{array}
$$

This sentence is True!

Example - Grounding

3. See if its' True

$$
\begin{array}{lll}
(& King(Richard) \Rightarrow Person(Richard) &) \land \quad \text{True} \\
(& King(John) \Rightarrow Person(John) &) \land \quad \text{True} \\
(& King(Elizabeth) \Rightarrow Person(Elizabeth) &) \land \quad \text{True} \\
(& King(Gigi) \Rightarrow Person(Gigi) &) \qquad \text{True}
\end{array}
$$

This sentence is True!

Exercise - Grounding

Determine if this is true in the given model:
$\exists x \; \mathrm{King}(x) \;\; \land \mathrm{OlderThan30}(x)$

Exercise - Grounding

Given:

- FOL sentence:
  - $\forall \mathrm{x}, \mathrm{y}\ (\mathrm{Friend}(\mathrm{x}, \mathrm{y}) \land \mathrm{LovesBBQ}(\mathrm{x})) \Longrightarrow \mathrm{LovesBBQ}(\mathrm{y})$
- A finite domain {Alice, Bob} for variable x and y

Compute the propositional grounding for the FOL sentence with the given domain.

Entailment

- What can we know from a knowledge base?

Consider the following knowledge base:

==================================

$\forall x \ \mathrm{King}(x) \Rightarrow \mathrm{Person}(x)$

$\forall x, y \ \mathrm{Person}(x) \wedge \mathrm{Brother}(x, y) \Rightarrow \mathrm{Person}(y)$

$\forall x, y \ \mathrm{Brother}(x, y) \Rightarrow \mathrm{Brother}(y, x)$

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

$\mathrm{Brother}(\mathrm{John}, \mathrm{Richard})$

==================================

Entailment

- What can we infer from a knowledge base?

Consider the following knowledge base:

====================================

$\forall x\ King(x) \Rightarrow Person(x)$

$\forall x, y\ Person(x) \wedge Brother(x, y) \Rightarrow Person(y)$

$\forall x, y\ Brother(x, y) \Rightarrow Brother(y, x)$

General Knowledge

$King(Richard)$

$Brother(John, Richard)$

specific problem

====================================

Entailment

KB:

=======================================

$\forall x\ King(x) \Rightarrow Person(x)$

$\forall x, y\ Person(x) \wedge Brother(x, y) \Rightarrow Person(y)$

$\forall x, y\ Brother(x, y) \Rightarrow Brother(y, x)$

$King(Richard)$

$Brother(John, Richard)$

=======================================

We want to know if the following statements are true:

- $Person(Richard)$
- $Person(John)$
- $\forall x\ Person(x) \wedge Brother(y, x) \Rightarrow Person(y)$

Entailment

- Entailment
  - $\alpha \models \beta$ iff every model that satisfies α also satisfies β
    - That is, in every model where α is true, β is true
    - $M(\alpha) \subseteq M(\beta)$
  - What does this mean?

Example – King Richard and his family

- ➢ α: (Our KB, all sentences connected by $\wedge$)

$$
\begin{array}{lcl}
(& \forall x\ \mathrm{King}(x) \Rightarrow \mathrm{Person}(x) &)\wedge \\
(& \forall x,y\ \mathrm{Person}(x) \wedge \mathrm{Brother}(x,y) \Rightarrow \mathrm{Person}(y) &)\wedge \\
(& \forall x,y\ \mathrm{Brother}(x,y) \Rightarrow \mathrm{Brother}(y,x) &)\wedge \\
(& \mathrm{King}(\mathrm{Richard}) &)\wedge \\
(& \mathrm{Brother}(\mathrm{John},\mathrm{Richard}) &)
\end{array}
$$

- ➢ β: $\mathrm{Person}(\mathrm{John})$

Does $\alpha \vDash \beta$?

Example – King Richard and his family

- What is "a model that satisifies α" ?

A model is a possible world.

- Possible world 1 (model 1):

Both Richard and John are human. Richard is not a king. John is Richard's brother.

Is α true in this possible world?

Is β true in this possible world?

Example – King Richard and his family

- Possible world 2 (model 2):

Both Richard and John are human. Richard is the king. John is a person but not Richard's brother.

Is α true in this possible world?

Is β true in this possible world?

Example – King Richard and his family

- Possible world 3 (model 3):

Both Richard and John are human. Richard is the King and John is his brother. All other common sense rules apply.

This possible world (model) satisfied α.

$\forall \mathrm{x}\ \mathrm{King}(\mathrm{x}) \Rightarrow \mathrm{Person}(\mathrm{x})$	True
$\forall \mathrm{x}, \mathrm{y}\ \mathrm{Person}(\mathrm{x}) \land \mathrm{Brother}(\mathrm{x}, \mathrm{y}) \Rightarrow \mathrm{Person}(\mathrm{y})$	True
$\forall \mathrm{x}, \mathrm{y}\ \mathrm{Brother}(\mathrm{x}, \mathrm{y}) \Rightarrow \mathrm{Brother}(\mathrm{y}, \mathrm{x})$	True
$\mathrm{King}(\mathrm{Richard})$	True
$\mathrm{Brother}(\mathrm{John}, \mathrm{Richard})$	True

In this possible world, β is also true.

Example – King Richard and his family

Obviously (to us smart CS161 students), in this example, whenever α is true, β is also true.

$\forall x \text{ King}(x) \Rightarrow \text{Person}(x)$	True
$\forall x, y \text{ Person}(x) \land \text{Brother}(x, y) \Rightarrow \text{Person}(y)$	True
$\forall x, y \text{ Brother}(x, y) \Rightarrow \text{Brother}(y, x)$	True
$\text{King}(\text{Richard})$	True
$\text{Brother}(\text{John}, \text{Richard})$	True

$\text{Person}(\text{John})$	True

But how can a machine know that?

Entailment

- Given:
  - A knowledge base α
  - A desired sentence β
- We want to know if $\alpha \vDash \beta$
  - Of course, we can do *model checking*
  - But sometimes we don't want to enumerate all possible worlds (models)!

- Theorem proving
  - Applying rules of inference

Evaluation of an inference algorithm

- Soundness
- Completeness

Entailment

To show that $K \vDash \alpha$, we show that $(KB \wedge \neg\alpha)$ is unsatisfiable

- By resolution
  - Sound
  - Complete

- Forward Chaining and Backward Chaining
  - Sound
  - Complete for <u>definite clauses</u>

Propositional Inference

- Modus Ponen: $$\frac{\alpha, \alpha \rightarrow \beta}{\therefore \beta}$$
    - Example: $\Delta = \{A, B, B \vee C, B \rightarrow D\}$
- Or introduction: $$\frac{\alpha, \beta}{\therefore \alpha \vee \beta}$$
- And introduction $$\frac{\alpha, \beta}{\therefore \alpha \wedge \beta}$$
- Resolution $$\frac{\alpha \vee \beta, \neg\beta \vee \delta}{\therefore \alpha \vee \delta}$$

Propositional Inference

function PL-RESOLUTION(KB, α) **returns** *true* or *false*
  inputs: KB, the knowledge base, a sentence in propositional logic
      α, the query, a sentence in propositional logic

  $clauses \leftarrow$ the set of clauses in the CNF representation of $KB \wedge \neg\alpha$
  $new \leftarrow \{\}$
  loop do
    for each pair of clauses C_i, C_j **in** *clauses* **do**
      $resolvents \leftarrow$ PL-RESOLVE(C_i, C_j)
      if *resolvents* contains the empty clause **then return** *true*
      $new \leftarrow new \cup resolvents$
    if $new \subseteq clauses$ **then return** *false*
    $clauses \leftarrow clauses \cup new$

Figure 7.12 A simple resolution algorithm for propositional logic. The function PL-RESOLVE returns the set of all possible clauses obtained by resolving its two inputs.

Example – Proof by resolution

- α:

$$
\begin{aligned}
&(\quad \mathrm{Person}(\mathrm{Richard}) \wedge \mathrm{Brother}(\mathrm{John}, \mathrm{Richard}) \longrightarrow \mathrm{Person}(\mathrm{John}) \quad) \wedge \\
&(\quad \mathrm{Person}(\mathrm{Richard}) \quad) \wedge \\
&(\quad \mathrm{Brother}(\mathrm{John}, \mathrm{Richard}) \quad)
\end{aligned}
$$

- β: $\mathrm{Person}(\mathrm{John})$

(The above sentences are shown in FOL syntax. For propositional logic, represent the clauses by A, B, C, ...)

Show that $(\alpha \wedge \neg\beta)$ is unsatisfiable.

(Unsatisfiable when *resolvents* contain the empty clause)

Example – Proof by resolution

➢ α:

$$\begin{array}{lcl} (& \text{Person(Richard)} \land \text{Brother(John, Richard)} \longrightarrow \text{Person(John)} &) \\ (& \text{Person(Richard)} &)\land \\ (& \text{Brother(John, Richard)} &) \end{array}$$

➢ β: Happy(John)

Is this $(\alpha \land \neg\beta)$ unsatisfiable?

PL-resolution

- Sound
- Complete

First-Order Inference

- Lifting
- Generalized Modus Ponens

Let's see how it works before getting back to terminologies.

First-Order Inference

- Universal Instantiation
- Existential Instantiation

- Propositionalization

- Generalized (lifted) Modus Ponens
- Unification

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

  - Most General Unifier (MGU)

Exercise – Propositionalization

- α:

∀x King(x) ⇒ Person(x)
King(Richard)
Brother(Richard, John)

- β: Person(Richard)

Domain of x: {Richard, John}

Exercise – Propositionalization

- α:

∀x King(x) ⇒ Person(x)
King(Richard)
Brother(Richard, John)

- β: Person(John)

Exercise – Generalized Modus Ponens

- α:

$\forall x\ King(x) \Rightarrow Person(x)$
$King(Richard)$

- β: $Person(Richard)$

Exercise - Resolution

- α:

$\forall x \, King(x) \Rightarrow Person(x)$
$\forall x, y \, Person(x) \wedge Brother(x, y) \Rightarrow Person(y)$
$King(Richard)$
$Brother(Richard, John)$

- β: $Person(John)$

Exercise - Resolution

- α:

$\forall x \text{ King}(x) \Rightarrow \text{Person}(x)$
$\forall x, y \text{ Person}(x) \wedge \text{Brother}(x, y) \Rightarrow \text{Person}(y)$
$\text{King}(\text{Richard})$
$\text{Brother}(\text{Richard}, \text{John})$

King(Edward)

- β: Person(John)

Entailment

To show that $K \vDash \alpha$, we show that $(KB \land \neg\alpha)$ is unsatisfiable

- By resolution
  - Sound
  - Complete
- Forward Chaining and Backward Chaining
  - Sound
  - Complete for definite clauses

Definite Clauses

- Horn Clauses (Definite clauses)
  - $p_1 \wedge p_2 \wedge p_3 \ldots \implies q$
  - Exactly one positive literal (q) in the CNF

- Deciding entailment can be done in **linear** time in the size of KB for Horn clauses

Forward Chaining

- Data-driven reasoning

Begins from known facts (positive literals).

Incrementally add conclusions to the set of known facts.

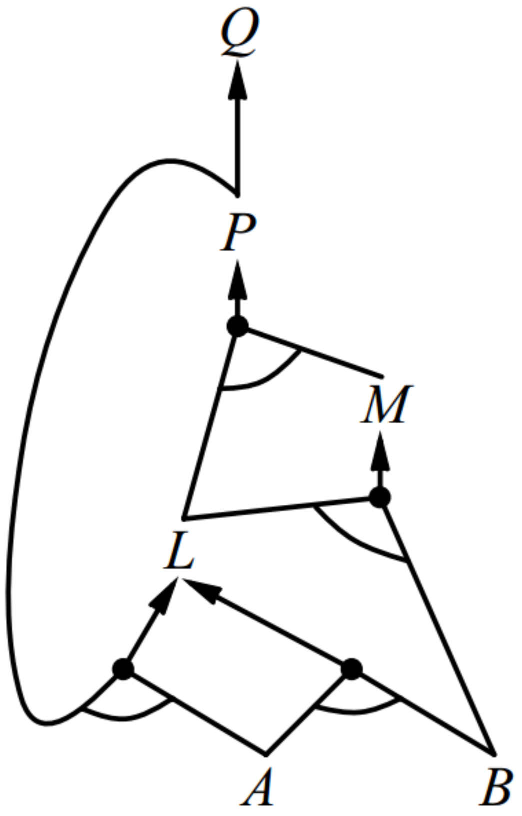
Every entailed atomic sentence will be derived

We show the propositional logic in this discussion and they can be easily extended to FOL. (See Chapter 9.3 and 9.4 in textbook)

AND-OR Graph

$$P \Rightarrow Q$$
$$L \wedge M \Rightarrow P$$
$$B \wedge L \Rightarrow M$$
$$A \wedge P \Rightarrow L$$
$$A \wedge B \Rightarrow L$$
$$A$$
$$B$$

(a)



(b)

Figure 7.16 (a) A set of Horn clauses. (b) The corresponding AND–OR graph.

Forward Chaining

```
function PL-FC-ENTAILS?(KB, q) returns true or false
  inputs: KB, the knowledge base, a set of propositional definite clauses
          q, the query, a proposition symbol
  count ← a table, where count[c] is the number of symbols in c's premise
  inferred ← a table, where inferred[s] is initially false for all symbols
  agenda ← a queue of symbols, initially symbols known to be true in KB

  while agenda is not empty do
     p ← POP(agenda)
     if p = q then return true
     if inferred[p] = false then
         inferred[p] ← true
         for each clause c in KB where p is in c.PREMISE do
             decrement count[c]
             if count[c] = 0 then add c.CONCLUSION to agenda
  return false
```

Backward Chaining

- Goal-directed reasoning

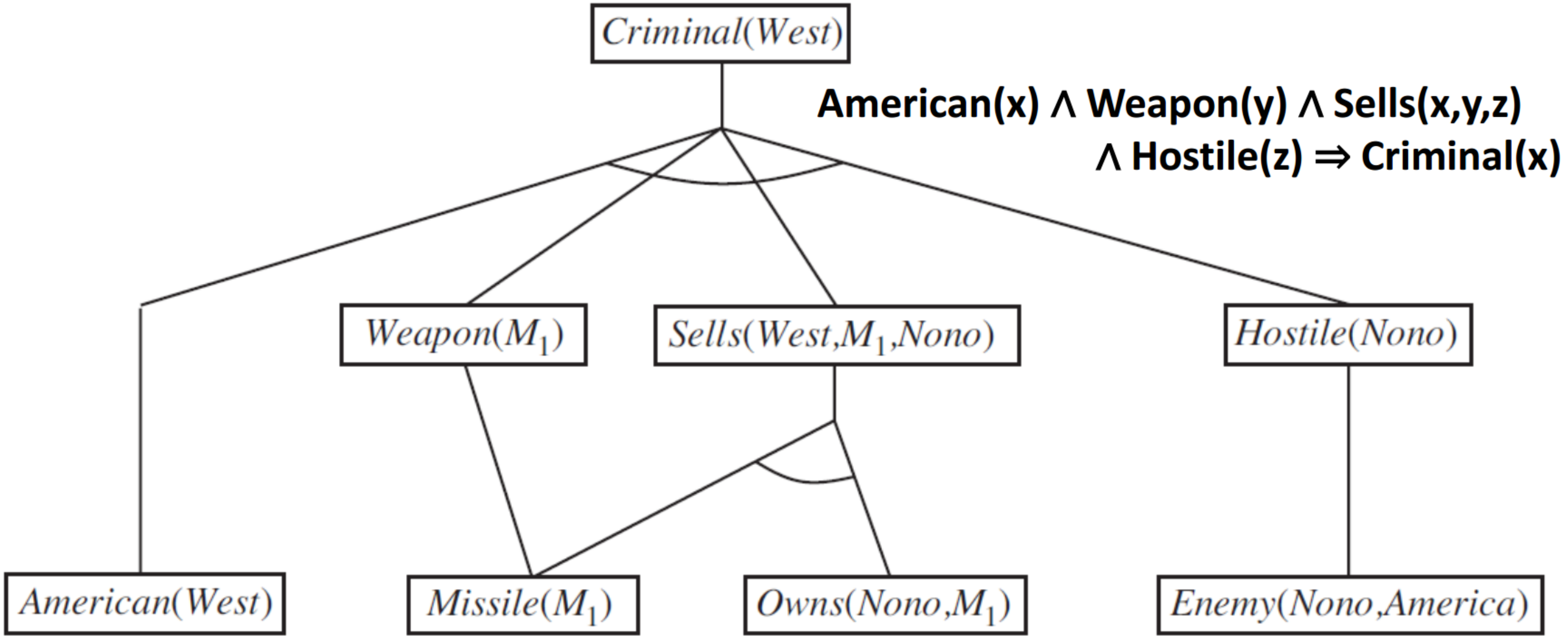
Often, the cost of backward chaining is *much less* than linear in the size of the knowledge base, because the process touches only relevant facts.

Forward Chaining

function FOL-FC-ASK(KB, α) **returns** a substitution or *false*
  inputs: KB, the knowledge base, a set of first-order definite clauses
    α, the query, an atomic sentence
  local variables: new, the new sentences inferred on each iteration

  repeat until new is empty
    $new \leftarrow \{\}$
    for each $rule$ in KB **do**
      $(p_1 \wedge \ldots \wedge p_n \Rightarrow q) \leftarrow$ STANDARDIZE-VARIABLES($rule$)
      for each θ such that SUBST($\theta, p_1 \wedge \ldots \wedge p_n$) = SUBST($\theta, p'_1 \wedge \ldots \wedge p'_n$)
          for some $p'_1, \ldots, p'_n$ in KB
        $q' \leftarrow$ SUBST(θ, q)
        if q' does not unify with some sentence already in KB or new **then**
          add q' to new
          $\phi \leftarrow$ UNIFY(q', α)
          if ϕ is not $fail$ **then return** ϕ
    add new to KB
  return *false*

Forward Chaining



Backward Chaining