Artificial Intelligence

Hai Thi Tuyet Nguyen

Outline

CHAPTER 1: INTRODUCTION (CHAPTER 1)

CHAPTER 2: INTELLIGENT AGENTS (CHAPTER 2)

CHAPTER 3: SOLVING PROBLEMS BY SEARCHING (CHAPTER 3)

CHAPTER 4: INFORMED SEARCH (CHAPTER 3)

CHAPTER 5: LOGICAL AGENT (CHAPTER 7)

CHAPTER 6: FIRST-ORDER LOGIC (CHAPTER 8, 9)

CHAPTER 7: QUANTIFYING UNCERTAINTY(CHAPTER 13)

CHAPTER 8: PROBABILISTIC REASONING (CHAPTER 14)

CHAPTER 9: LEARNING FROM EXAMPLES (CHAPTER 18)

CHAPTER 6: FIRST-ORDER LOGIC

6.1 Representation Revisited

6.1 Propositional logic

- Propositional logic is a ***declarative*** language because its semantics is based on a truth relation between sentences and possible worlds.
- Propositional logic is ***compositional***:

  $B_{1,1} \wedge P_{1,2}$ is derived from meaning of $B_{1,1}$ and of $P_{1,2}$
- Propositional logic has very limited expressive power to concisely describe an environment with many objects

  E.g., cannot say “pits cause breezes in adjacent squares”

  except by writing one sentence for each square

6.1 First-order logic

First-order logic (like natural language) assumes the world contains

- Objects: people, houses, numbers, theories, Ronald McDonald, colors, baseball games, wars, centuries
- Relations:
    - unary relations or properties such as red, round, bogus, prime, multistoried . . .,
    - n-ary relations such as brother of, bigger than, inside, part of, has color, occurred after, owns, comes between, . . .
- Functions: relations in which there is only one "value" for a given "input.": father of, best friend, third inning of, one more than, beginning of

6.1 First-order logic

Examples:

- “Squares neighboring the wumpus are smelly.”
  Objects: wumpus, squares
  Property: smelly
  Relation: neighboring

6.2 Syntax And Semantics Of First-Order Logic

6.2 Syntax And Semantics Of First-Order Logic

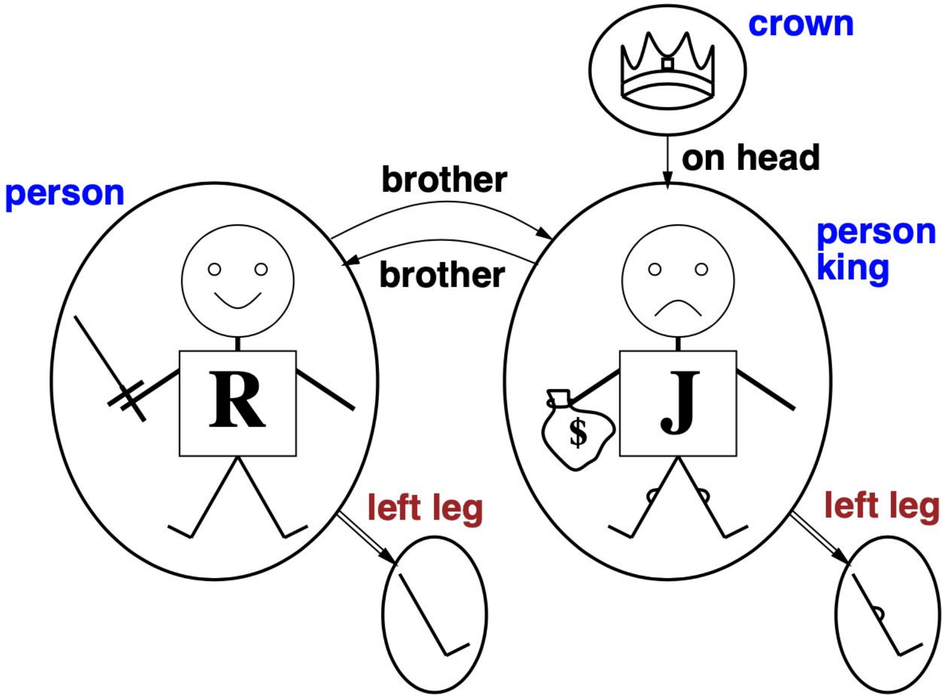
- **Model** contains at least objects and relations among them

- Basic **symbols**:
  - constant symbols → objects
  - predicate symbols → relations
  - function symbols → functional relations

- Each model includes an **interpretation** that specifies exactly which objects, relations and functions are referred to by the constant, predicate, and function symbols.

6.2 Models for FOL: Example

- 5 objects:
  - Richard the Lionheart;
  - the evil King John;
  - the left legs of Richard and John;
  - a crown
- Relations:
  - binary relations:
    - "brother" and "on head"
  - unary relations, or properties
    - "person", "king", "crown"
  - functions: only one "value" for a given "input."
    - "left leg"



6.2 Syntax of FOL: Basic elements

- Constants: KingJohn, 2, UCB,...
- Variables: x, y, a, b,...
- Predicates (Relations): Brother, >,...
- Functions: LeftLeg, Mother
- Connectives: $\neg \wedge \vee \Rightarrow \Leftrightarrow$
- Quantifiers: $\forall$, $\exists$

6.2 Term & Atomic sentences

- Term: a logical expression that refers to an object.
  constant
  variable
  function(term1, ..., termn)
  E.g., > (Length(LeftLegOf(Richard)), Length(LeftLegOf(KingJohn)))

- Atomic sentence: (or atom for short) is formed from a predicate symbol optionally followed by a parenthesized list of terms
  Predicate | Predicate (Term , . . .) | Term = Term
  E.g., Brother(KingJohn, RichardTheLionheart)

6.2 Complex sentences

- Complex sentences are made from atomic sentences using connectives

  $\neg S,\ S_1 \wedge S_2,\ S_1 \vee S_2,\ S_1 \Rightarrow S_2,\ S_1 \Leftrightarrow S_2$

E.g.

$\neg Brother\ (LeftLeg\ (Richard\ ),\ John\ )$

$Brother\ (Richard\ ,\ John\ )\ \wedge\ Brother\ (John\ ,\ Richard\ )$

$King\ (Richard\ )\ \vee\ King\ (John\ )$

$\neg King(Richard) \Rightarrow King(John)$

6.2 Universal quantification

$\forall \langle variables \rangle \langle sentence \rangle$

$\forall x P$ is true in a model m iff the sentence P is true with the variable x being each possible object in the model

Everyone at Berkeley is smart: $\forall x\ At(x, Berkeley) \Rightarrow Smart(x)$

Equivalent to the conjunction of instantiations of P

$(At(KingJohn,Berkeley) \Rightarrow Smart(KingJohn))$
$\land\ (At(Richard,Berkeley) \Rightarrow Smart(Richard))$
$\land\ (At(Berkeley,Berkeley) \Rightarrow Smart(Berkeley))$
$\land\ \ldots$

6.2 Existential quantification

$\exists \langle variables \rangle \langle sentence \rangle$

$\exists x\ P$ is true in a model m iff the sentence P is true with the variable x being some possible object in the model

Someone at Stanford is smart: $\exists x\ At(x, Stanford) \wedge Smart(x)$

Equivalent to the disjunction of instantiations of P

$(At(KingJohn, Stanford) \wedge Smart(KingJohn))$
$\vee\ (At(Richard, Stanford) \wedge Smart(Richard))$
$\vee\ (At(Stanford, Stanford) \wedge Smart(Stanford))$
$\vee\ \ldots$

6.2 Properties of quantifiers

$\forall x\ \forall y$ is the same as $\forall y\ \forall x$

$\exists x\ \exists y$ is the same as $\exists y\ \exists x$

Quantifier duality: each can be expressed using the other

$\forall x\ Likes(x, IceCream) \quad \neg\exists x \neg Likes(x, IceCream)$

$\exists x\ Likes(x, Broccoli) \quad \neg\forall x\ \neg Likes(x, Broccoli)$

6.2 Equality

$term_1 = term_2$ is true if and only if $term_1$, $term_2$ refer to the same object

E.g., Father(John)=Henry

The syntax of first-order logic

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

6.3 Using First-Order Logic

6.3 Using First-Order Logic

Assertions and queries in first-order logic

Sentences are added to a knowledge base using TELL, (called as assertions)

For example, we can assert that John is a king, Richard is a person, and all kings are persons:

TELL(KB, King(John)) .

TELL(KB, Person(Richard)) .

TELL(KB, $\forall x King(x) \Rightarrow Person(x)$) .

We can ask questions of the knowledge base using ASK (queries or goals).

ASK(KB, King(John)) returns true

ASKVARS(KB,Person(x)) returns two answers: {x/John} and {x/Richard }

6.3 Inference rules for quantifiers

- **Universal Instantiation rule:**
  SUBST(θ, α) denote the result of applying the substitution θ to the sentence α
  Rule: substituting a ground term (a term without variables) for the variable

$$\frac{\forall v \quad \alpha}{\text{SUBST}(\{v/g\}, \alpha)}$$

  for any variable v, ground term g

  E.g., $\forall$ x King(x) $\land$ Greedy(x) $\Rightarrow$ Evil(x) yields

  King(John) $\land$ Greedy(John) $\Rightarrow$ Evil(John)
  King(Richard) $\land$ Greedy(Richard) $\Rightarrow$ Evil(Richard)
  King(Father(John)) $\land$ Greedy(Father(John)) $\Rightarrow$ Evil(Father(John))

6.3 Inference rules for quantifiers

- **Existential instantiation (EI)**

  For any sentence α, variable v, and constant symbol k that does not appear elsewhere in the knowledge base:

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

E.g., ∃x Crown(x) ∧ OnHead(x, John) yields

Crown(C1) ∧ OnHead(C1, John)

provided C1 is a new constant symbol

6.3 Inference rules for quantifiers

- Universal Instantiation (UI) rule can be applied several times to add new sentences; the new KB is logically equivalent to the old.
- Existential instantiation (EI) rule can be applied once to replace the existential sentence; the new KB is not equivalent to the old, but is satisfiable iff the old KB was satisfiable.

6.3 Reduction to propositional inference

Suppose the KB contains just the following:

$\forall x\ King(x) \land Greedy(x) \Rightarrow Evil(x)$
King(John)
Greedy(John)
Brother(Richard, John)

Instantiating the universal sentence in all possible ways, we have

{x/John} and {x/Richard}

$King(John) \land Greedy(John) \Rightarrow Evil(John)$
$King(Richard) \land Greedy(Richard) \Rightarrow Evil(Richard)$

King(John)
Greedy(John)
Brother(Richard, John)

The new KB is propositionalized: proposition symbols are

King(John), Greedy(John), Evil(John), King(Richard), etc.

6.3 Reduction to propositional inference

- Claim: every FOL KB can be propositionalized so as to preserve entailment
- Idea: propositionalize KB and query, apply propositional resolution, obtain result
- Problem: with function symbols, there are infinitely many ground terms,
  e.g., Father(Father(Father(John))
- Theorem: Herbrand (1930).
  If a sentence α is entailed by an FOL KB, it is entailed by a finite subset of the propositional KB
- Theorem: Turing (1936), Church (1936), entailment in FOL is semidecidable

6.3 Generalized Modus Ponens (GMP)

- For atomic sentences p_i, p'_i, and q, where there is a substitution θ, such that $\text{SUBST}(\theta, p_i) = \text{SUBST}(\theta, p'_i)$, for all i

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

KB:

King(John)
Greedy(y)
King(x) ⋀ Greedy(x) ⇒ Evil(x)

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)	

6.3 Unification

- We can get the inference if we can find a substitution θ such that King(x) and Greedy(x) match King(John) and Greedy(y)

UNIFY(p, q) = θ where SUBST(θ, p) = SUBST(θ, q) .

p	q	θ
$Knows(John, x)$	$Knows(John, Jane)$	$\{x/Jane\}$
$Knows(John, x)$	$Knows(y, OJ)$	$\{x/OJ, y/John\}$
$Knows(John, x)$	$Knows(y, Mother(y))$	$\{y/John, x/Mother(John)\}$
$Knows(John, x)$	$Knows(x, OJ)$	$fail$

Standardizing apart (renaming) eliminates overlap of variables, e.g., Knows(x_{17},OJ)

6.3 Forward chaining

- A definite clause:
  - atomic
  - a conjunction of positive literals => a single positive literal.
- First-order literals can include variables, in which case those variables are assumed to be universally quantified.
- Examples:

  $King(x) \land Greedy(x) \Rightarrow Evil(x)$.

  $King(John)$.

  $Greedy(y)$.

6.3 Forward chaining

Example:

The law says that it is a crime for an American to sell weapons to hostile nations. The country Nono, an enemy of America, has some missiles, and all of its missiles were sold it by Colonel West, who is American.

Prove that Colonel West is a criminal.

6.3 Example knowledge base contd.

Represent these facts as first-order definite clauses:

- It is a crime for an American to sell weapons to hostile nations:
  - $American(x) \wedge Weapon(y) \wedge Hostile(z) \wedge Sells(x, y, z) \Rightarrow Criminal(x)$ (9.3)
- Nono . . . has some missiles, i.e., $\exists\ x\ Owns(Nono, x) \wedge Missile(x)$: transform it into two definite clauses with M1 as a new constant
  - $Owns(Nono, M1)$ (9.4)
  - $Missile(M1)$ (9.5)
- . . . all of its missiles were sold to it by Colonel West
  - $\forall x\ Missile(x) \wedge Owns(Nono,x) \Rightarrow Sells(West,x,Nono)$ (9.6)
- Missiles are weapons:
  - $Missile(x) \Rightarrow Weapon(x)$ (9.7)
- An enemy of America counts as "hostile":
  - $Enemy(x,America) \Rightarrow Hostile(x)$ (9.8)
- West, who is American . . . :
  - $American(West)$ (9.9)
- The country Nono, an enemy of America . . .
  - $Enemy(Nono, America)$ (9.10)

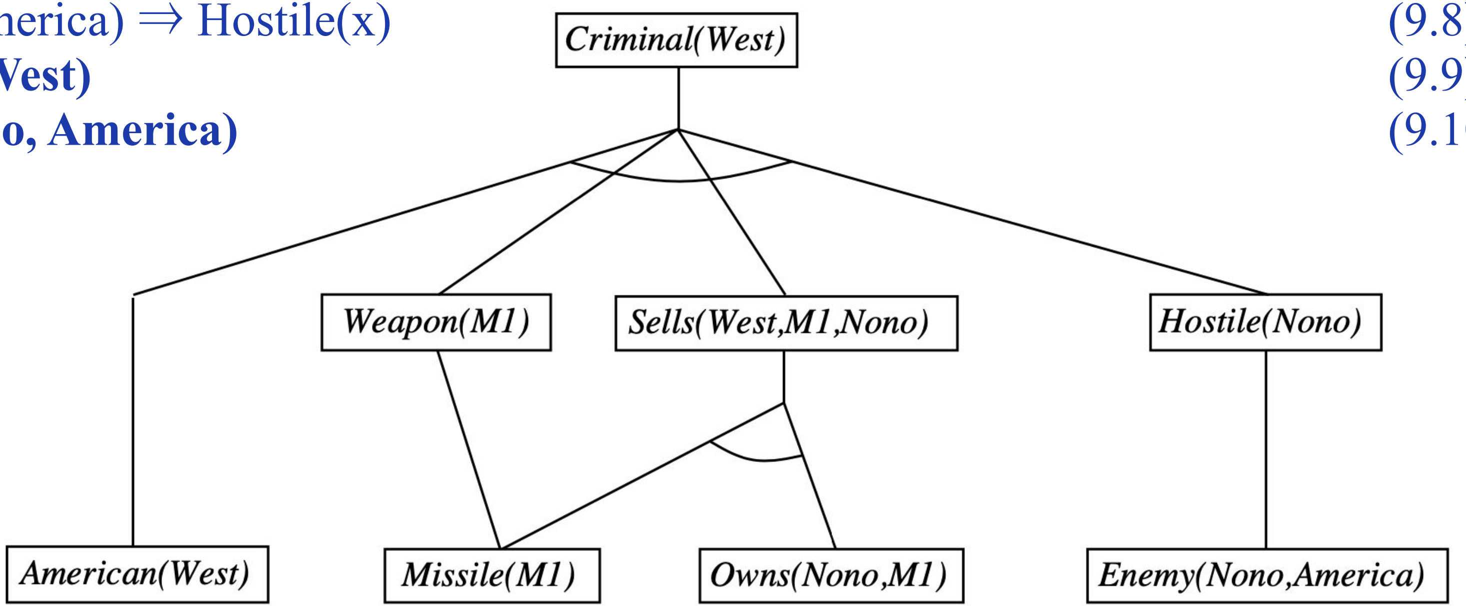
6.3 Forward chaining algorithm

```
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 ← { }
     for each rule in KB do
        (p_1 ∧ ... ∧ p_n ⇒ q) ← STANDARDIZE-VARIABLES(rule)
        for each θ such that SUBST(θ, p_1 ∧ ... ∧ p_n) = SUBST(θ, p'_1 ∧ ... ∧ p'_n)
                   for some p'_1, ..., p'_n in KB
           q' ← SUBST(θ, q)
           if q' does not unify with some sentence already in KB or new then
              add q' to new
              φ ← UNIFY(q', α)
              if φ is not fail then return φ
     add new to KB
  return false
```

6.3 Forward chaining proof

American(x) $\wedge$ Weapon(y) $\wedge$ Hostile(z) $\wedge$ Sells(x, y, z) $\Rightarrow$ Criminal(x) (9.3)
Owns(Nono, M1) (9.4)
Missile(M1) (9.5)
$\forall$x Missile(x) $\wedge$ Owns(Nono,x) $\Rightarrow$ Sells(West,x,Nono) (9.6)
Missile(x) $\Rightarrow$ Weapon(x) (9.7)
Enemy(x,America) $\Rightarrow$ Hostile(x) (9.8)
American(West) (9.9)
Enemy(Nono, America) (9.10)



6.3 Backward chaining algorithm

function FOL-BC-ASK(KB, $query$) **returns** a generator of substitutions
  return FOL-BC-OR(KB, $query$, { })

---

generator FOL-BC-OR(KB, $goal$, θ) **yields** a substitution
  for each rule ($lhs \Rightarrow rhs$) in FETCH-RULES-FOR-GOAL(KB, $goal$) **do**
    (lhs, rhs) $\leftarrow$ STANDARDIZE-VARIABLES((lhs, rhs))
    for each θ' **in** FOL-BC-AND(KB, lhs, UNIFY(rhs, $goal$, θ)) **do**
      yield θ'

---

generator FOL-BC-AND(KB, $goals$, θ) **yields** a substitution
  if $\theta = failure$ **then return**
  else if LENGTH($goals$) = 0 **then yield** θ
  else do
    $first, rest \leftarrow$ FIRST($goals$), REST($goals$)
    for each θ' **in** FOL-BC-OR(KB, SUBST(θ, $first$), θ) **do**
      for each θ'' **in** FOL-BC-AND(KB, $rest$, θ') **do**
        yield θ''

6.3 Backward chaining example

$American(x) \land Weapon(y) \land Hostile(z) \land Sells(x, y, z) \Rightarrow Criminal(x)$ (9.3)

$\mathbf{Owns(Nono, M1)}$ (9.4)

$\mathbf{Missile(M1)}$ (9.5)

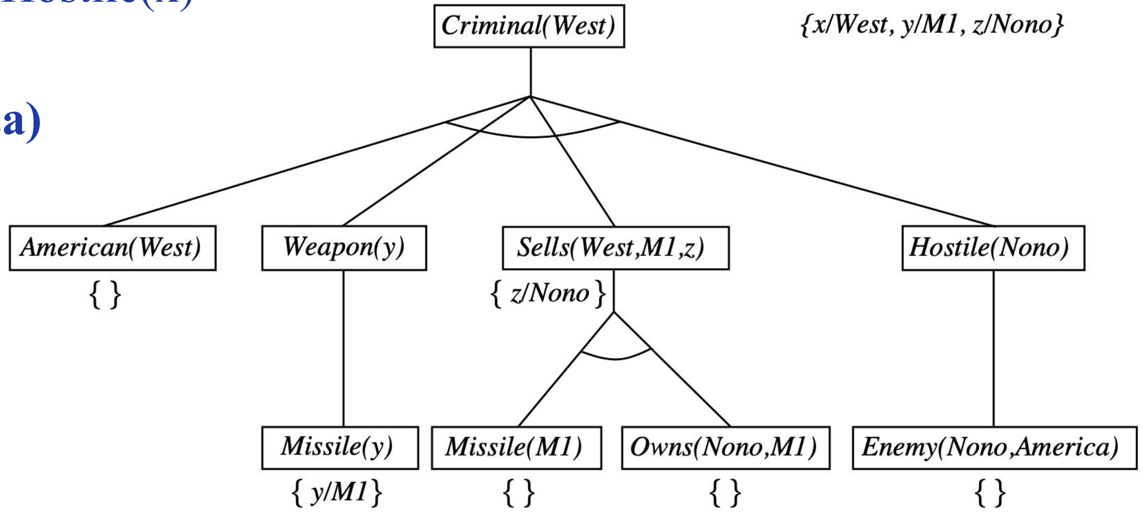
$\forall x\ Missile(x) \land Owns(Nono,x) \Rightarrow Sells(West,x,Nono)$ (9.6)

$Missile(x) \Rightarrow Weapon(x)$ (9.7)

$Enemy(x,America) \Rightarrow Hostile(x)$ (9.8)

$\mathbf{American(West)}$ (9.9)

$\mathbf{Enemy(Nono, America)}$ (9.10)



6.3 Resolution

Full first-order version:

$$\frac{\ell_1 \vee \cdots \vee \ell_k, \qquad m_1 \vee \cdots \vee m_n}{(\ell_1 \vee \cdots \vee \ell_{i-1} \vee \ell_{i+1} \vee \cdots \vee \ell_k \vee m_1 \vee \cdots \vee m_{j-1} \vee m_{j+1} \vee \cdots \vee m_n)\theta}$$

where Unify(l_i, $\neg m_j$) = θ.

For example,

$$\frac{\begin{array}{l}\neg Rich(x) \vee Unhappy(x) \\ Rich(Ken)\end{array}}{Unhappy(Ken)}$$

with θ = {x/Ken}

Apply resolution steps to CNF(KB ∧ ¬α); complete for FOL

6.3 Conversion to CNF

1. Eliminate biconditionals and implications

$S_1 => S_2 \equiv \neg S_1 \vee S_2$

$S_1 <=> S_2 \equiv (S_1 => S_2) \wedge (S_2 => S_1)$

2. Move $\neg$ inwards:

$\neg\forall x,p \equiv \exists x \neg p, \neg\exists x,p \equiv \forall x \neg p$

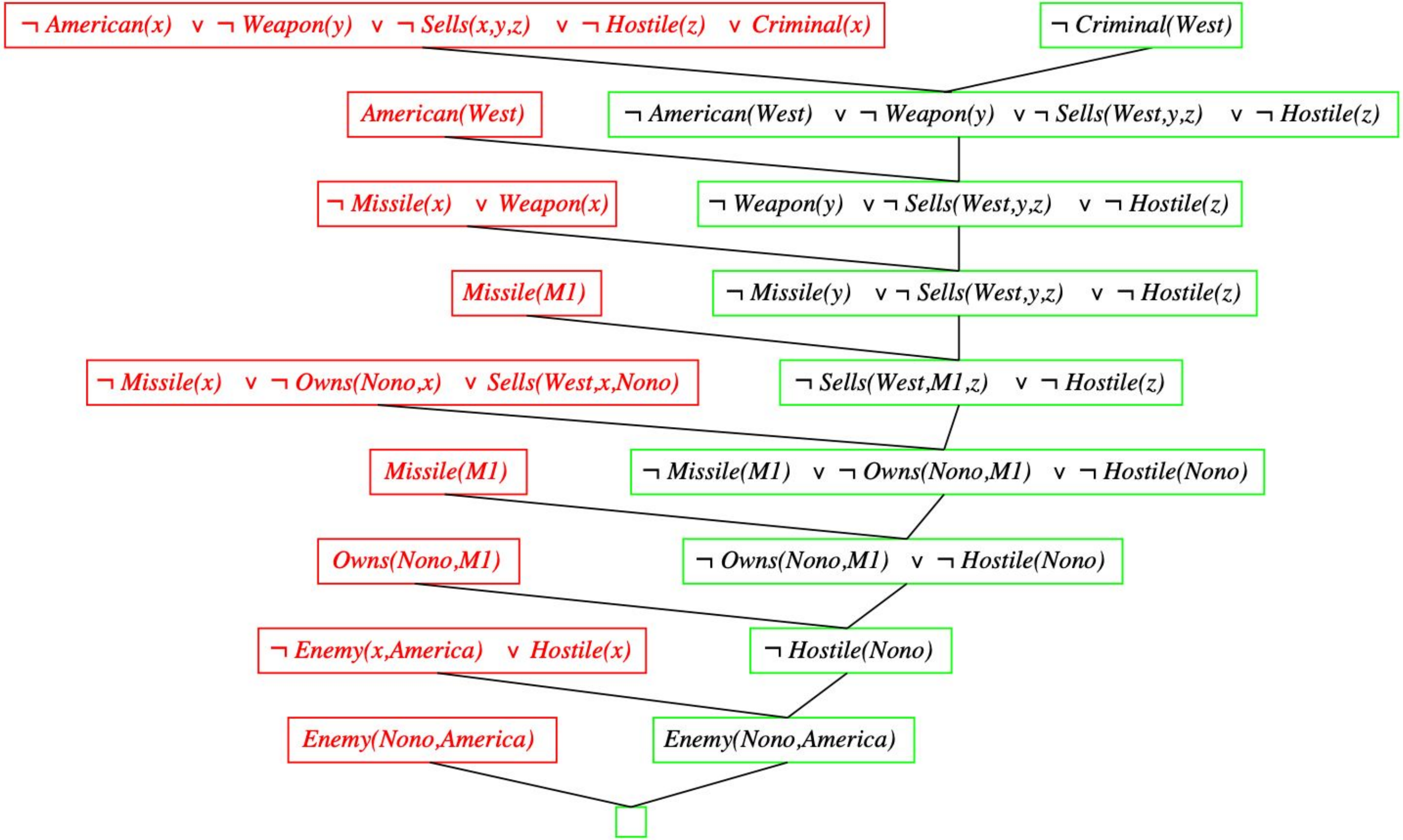
3. Standardize variables: each quantifier should use a different one

4. Skolemize: each existential variable is replaced by a Skolem function of the enclosing universally quantified variables

5. Drop universal quantifiers

6. Distribute $\wedge$ over $\vee$

6.3 Resolution proof: definite clauses



Examples

Consider a vocabulary with the following symbols:

> Occupation (p, o): Predicate. Person p has occupation o.
> Customer (p1, p2): Predicate. Person p1 is a customer of person p2.
> Boss (p1, p2): Predicate. Person p1 is a boss of person p2.
> Doctor, Surgeon, Lawyer, Actor: Constants denoting occupations.
> Emily, Joe: Constants denoting people.

Use these symbols to write the following assertions in first-order logic:

a. Emily is either a surgeon or a lawyer.
b. Joe is an actor, but he also holds another job.
c. All surgeons are doctors.
d. Emily has a boss who is a lawyer.
e. There exists a lawyer all of whose customers are doctors.
f. Every surgeon has a lawyer.