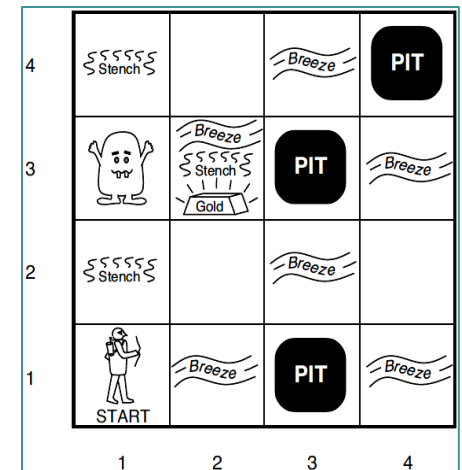


First-Order Logic (FOL)


Why FOL?

- Example #1: Try to write down all the sentences (rules in KB) that represent "a pit causes breeze in adjacent squares."
 - How many sentences do we need?
 - How many sentences do we need if the Wumpus world is 100x100 and there are 1000 pits?
- Example #2: How do you represent "Every student in NCTU is smart" in propositional logic?



Why FOL?



- Propositional logic is based on facts about the world, such as $P_{2,2}$. However, we can not have any connection between the two facts $P_{2,2}$ and $P_{3,2}$ even though they are very similar.
 - Propositional logic has very limited expressive power.
- FOL is concerned with objects. Facts represent "attributes of objects" and "relations among objects".
- Example: " x is a student", " x is in NYCU", and " x is smart" are all attributes of x , which can represent any object.
 - The sentence "Every student in NYCU is smart" can be written in this logical expression 

$$(x \text{ is a student}) \wedge (x \text{ is in NYCU}) \Rightarrow (x \text{ is smart})$$

Representations in FOL

- Constant symbols (objects): *John, Mary, Wumpus*, etc.
- Predicate symbols (attributes/relations): These look like functions that give truth values.
 - Single argument: *Student(John), Red(Wumpus)*, etc.
 - Multiple arguments: $\geq(2,1)$, *Classmate(John,Mary)*, *Product(3,5,15)*, *Inside(John, ED117)*, etc.
- Function symbols (functions): These give objects defined according to their relations to other objects: *Mother(Mary)*, *Sqrt(100)*, *LeftLeg(John)*, *Sum(3,5)*, etc.

Models in FOL

A model in FOL includes the following:

- **Domain**: A non-empty set of objects.
- **Relations** among the objects: A relation is defined by the set of all the **tuples** of objects that are related.
 - Example: Let the relation *Classmate* include the tuples $\langle John, Mary \rangle$ and $\langle Mary, John \rangle$.
 - ◆ Given the relation, the predicates *Classmate*(*John*,*Mary*) and *Classmate*(*Mary*,*John*) are true.
 - Unary relation: All the objects that satisfy a certain attribute. For example, the relation *Student* includes $\langle John \rangle$ and $\langle Mary \rangle$, but not $\langle John's\ dog \rangle$.
 - ◆ Given the relation, the predicates *Student*(*John*) and *Student*(*Mary*) are true, but the predicate *Student*(*John's dog*) is false.

Models in FOL

A model in FOL includes the following:

- When a relation can be made into a **function**: In this relation, an object can only be related to only one other particular object.
 - Example: Let the relation *Head* include the tuples $\langle \text{John}, \text{John's head} \rangle$ and $\langle \text{Mary}, \text{Mary's head} \rangle$.
 - ◆ Given the relation, we can have a function *Head*, where *Head(John)* is John's head and *Head(Mary)* is Mary's head.
 - Example: The relation *Classmate* can not become a function.
- **Interpretation**: This links the symbols with the actual objects and relations in the model.

Models in FOL

More about interpretation:

- There are multiple possible interpretations for linking the objects/relations and symbols. The truth value of a sentence depends on the interpretation.
- Example:
 - *Classmate(John,Mary)* is **true** if *John* refers to John the student and *Mary* refers to Mary the student.
 - *Classmate(John,Mary)* is **false** if *John* refers to John the student and *Mary* refers to John's cat.
- The actual object referred to by a term is called the term's **referent**.

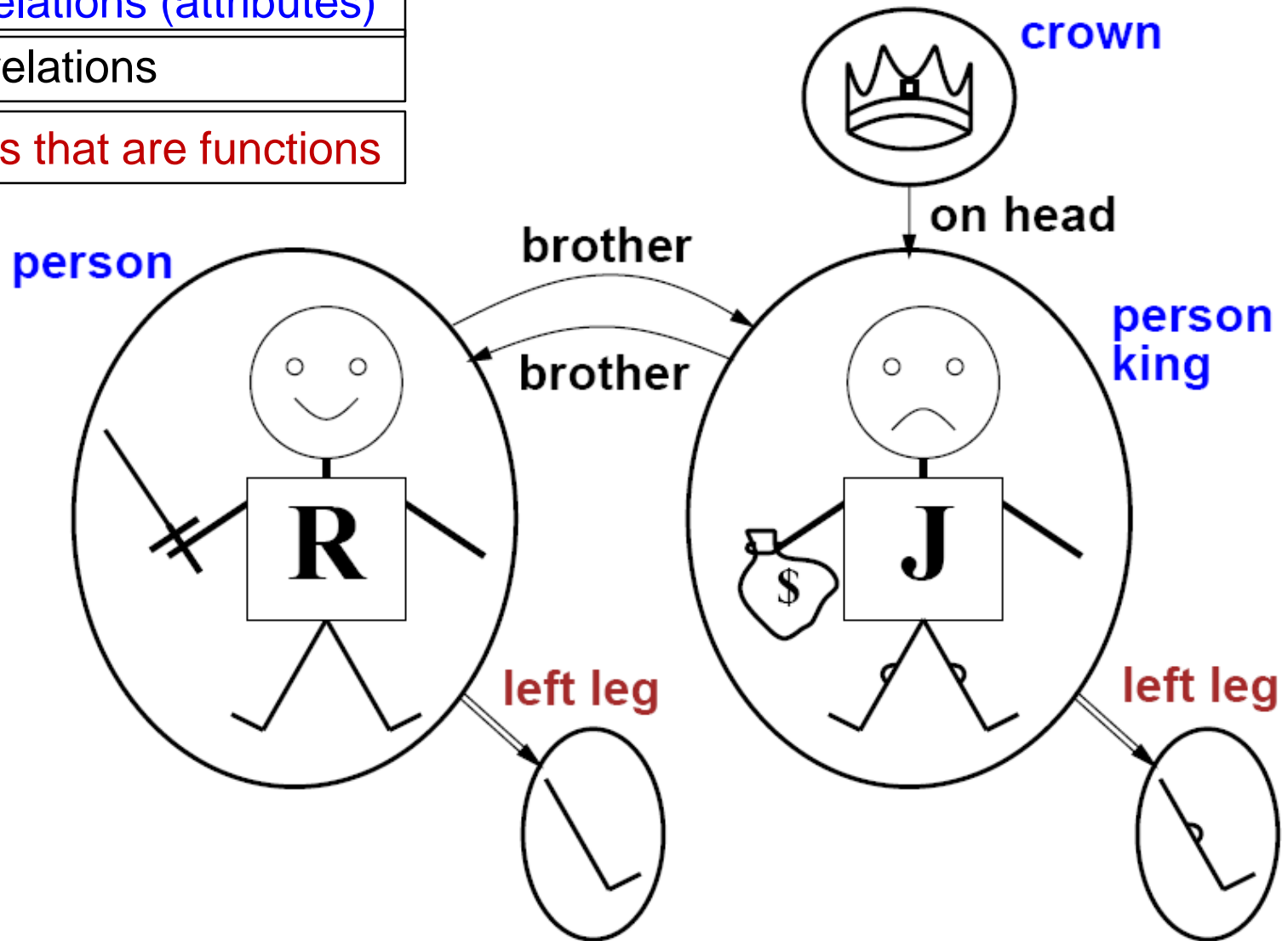
Models in FOL

An example model in the textbook:

unary relations (attributes)

binary relations

relations that are functions



Syntax of FOL

Sentence \rightarrow *AtomicSentence* | *ComplexSentence*
AtomicSentence \rightarrow *Predicate* | *Predicate(Term,...)* | *Term=Term*
ComplexSentence \rightarrow (*Sentence*) | [*Sentence*] | \neg *Sentence*
| *Sentence* \wedge *Sentence* | *Sentence* \vee *Sentence*
| *Sentence* \Rightarrow *Sentence* | *Sentence* \Leftrightarrow *Sentence*
| *Quantifier Variable,... Sentence*

Each sentence has
a truth value.

Term \rightarrow *Function(Term,...)* | *Constant* | *Variable*
Quantifier \rightarrow \forall | \exists
Constant \rightarrow *A* / *X*₁ | *John* | ...
Variable \rightarrow *a* | *x* | *s* | ...
Predicate \rightarrow *True* | *False* | *After* | *Loves* | *Raining* | ...
Function \rightarrow *Mother* / *LeftLeg* | ...

Operator Precedence: $\neg, =, \wedge, \vee, \Rightarrow, \Leftrightarrow$

Universal Quantification

■ $\forall x P$

- True iff P is true for x being every possible object in the model.

■ Example: Everyone in NYCU is smart.

- $\forall x AtNYCU(x) \Rightarrow Smart(x)$
- Any problem with this? Consider the domain of the model.

■ Example: Every student in NYCU is smart.

- $\forall x AtNYCU(x) \wedge Student(x) \Rightarrow Smart(x)$

■ What is the meaning of the following?

- $\forall x AtNYCU(x) \wedge Student(x) \wedge Smart(x)$

Existential Quantification

■ $\exists x P$

- True iff P is true for x being some possible object in the model.

■ Example: Someone in NTHU is smart.

- $\exists x AtNTHU(x) \wedge Smart(x)$

■ What is the problem with the following?

- $\exists x AtNTHU(x) \Rightarrow Smart(x)$
- Normally we do not use implication as the main connective with existential quantification.

Quantifier Duality

De Morgan's rules:

$$\blacksquare \exists x \neg P \equiv \neg \forall x P$$

$$\blacksquare \exists x P \equiv \neg \forall x \neg P$$

$$\blacksquare \forall x \neg P \equiv \neg \exists x P$$

$$\blacksquare \forall x P \equiv \neg \exists x \neg P$$

Examples:

$$\blacksquare \forall x \text{ Likes}(x, \text{IceCream}) \equiv \neg \exists x \neg \text{ Likes}(x, \text{IceCream})$$

$$\blacksquare \exists x \text{ Likes}(x, \text{Broccoli}) \equiv \neg \forall x \neg \text{ Likes}(x, \text{Broccoli})$$

Nesting Quantifiers

Try to link the sentences with their meanings:

■ $\forall x \forall y \text{ Loves}(x,y)$

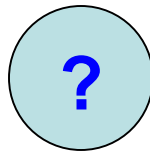
■ $\exists x \exists y \text{ Loves}(x,y)$

■ $\exists x \forall y \text{ Loves}(x,y)$

■ $\forall x \exists y \text{ Loves}(x,y)$

■ $\forall y \exists x \text{ Loves}(x,y)$

■ $\exists y \forall x \text{ Loves}(x,y)$



Everyone loves someone.

Someone loves everyone.

Everyone is loved by someone.

Someone is loved by everyone.

Someone is loved by someone.

Everyone loves everyone.

Notes:

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

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

■ $\exists x \forall y$ is not the same as $\forall y \exists x$

Fun with FOL Sentences

Brothers are siblings

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

"Sibling" is symmetric

$$\forall x \forall y \textit{Sibling}(x, y) \Leftrightarrow \textit{Sibling}(y, x)$$

One's mother is one's female parent

$$\forall x \forall y \textit{Mother}(x, y) \Leftrightarrow \textit{Female}(x) \wedge \textit{Parent}(x, y)$$

A first cousin is a child of a parent's sibling

$$\forall x \forall y [\textit{FirstCousin}(x, y) \Leftrightarrow \\ \exists p, q \textit{Parent}(p, x) \wedge \textit{Parent}(q, y) \wedge \textit{Sibling}(p, q)]$$

Equality in FOL

- The equality symbol ($=$) means that two terms refer to the same object.
- Example: "John has at least two brothers"
 - $\exists x \exists y \text{ Brother}(x, \text{John}) \wedge \text{Brother}(y, \text{John}) \wedge \neg(x=y)$
 - Not sufficient: $\exists x \exists y \text{ Brother}(x, \text{John}) \wedge \text{Brother}(y, \text{John})$
- Example: definition of (full) sibling

$$\forall x \forall y \text{ FullSibling}(x, y) \quad \Leftrightarrow$$

$$\neg(x=y) \wedge$$

$$[\exists m, f \quad \neg(m=f) \wedge \text{Parent}(m, x) \wedge \text{Parent}(m, y) \wedge \text{Parent}(f, x) \wedge \text{Parent}(f, y)]$$

Definitions, Axioms, Theorems

- We have used sentences in the following form to "define" new predicates:
 - $\forall x, \dots P(x, \dots) \Leftrightarrow \dots$
 - This way we can enrich the representation from a basic set of predicates.
- Example: From a basic set of *Child*, *Female*, and *Spouse*, we can define other relations in the kinship domain: *Parent*, *Male*, etc.
- Axioms: Sentences in the KB that are not entailed by other sentences. Axioms include definitions.
- Theorems: Useful facts that are entailed by the KB.
 - Example: $\forall x \forall y \text{ Sibling}(x, y) \Leftrightarrow \text{Sibling}(y, x)$

Exercises: Writing FOL Sentences

Starting from the basic set of three predicates *Male*, *Parent*, and *Spouse*, give definitions of these predicates:

■ *Female*: $\text{Female}(x) \Leftrightarrow \neg \text{Male}(x)$

■ *Child*:

■ *Father*:

■ *Sister*:

■ *Grandfather*:

■ *Uncle*: $\text{Uncle}(x,y) \Leftrightarrow \exists z \text{ Brother}(x,z) \wedge \text{Parent}(z,y)$

■ *Niece*:

Exercises: Writing FOL Sentences

- John has exactly two brothers.

$$\begin{aligned} \exists x,y \text{ Brother}(\text{John},x) \wedge \text{Brother}(\text{John},y) \wedge (x \neq y) \\ \wedge [\forall z \text{ Brother}(\text{John},z) \Rightarrow (x=z) \vee (y=z)] \end{aligned}$$

- Each of John's classmates has at least one sibling.

$$\forall x \text{ Classmate}(\text{John},x) \Rightarrow [\exists y \text{ Sibling}(y,x)]$$

- One of Mary's classmates is John's brother.

$$\exists x \text{ Classmate}(\text{Mary},x) \wedge \text{Brother}(\text{John},x)$$

Assertions and Queries for a FOL KB

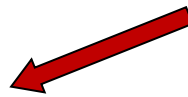
■ Assertions: Sentences added to the KB. Examples:

- $Tell(KB, King(John))$
- $Tell(KB, Person(Richard))$
- $Tell(KB, \forall x King(x) \Rightarrow Person(x))$

■ Queries (inference required here)

- $Ask(KB, King(John))$
- $Ask(KB, Person(John))$
- $Ask(KB, \exists x Person(x))$
- $AskVars(KB, Person(x))$

Which x in the domain makes the sentence $Person(x)$ true?



■ **Substitution** (binding list): The response to $AskVars$ that lists the possible variable assignments that make the query true.

- Example for the last query: $\{x/John\}$ and $\{x/Richard\}$

Inference in FOL

- Difference with propositional logic: variables
 - Symbols (sentences) in FOL that do not involve variables can be treated as propositional symbols.
Example: *Student(John)*, *Red(Wumpus)*, etc.
- How do we handle variables (and quantifiers)?
 - **Propositionalization**
 - **Unification**

Universal Instantiation (UI)

- Instantiation means to replace a variable in a sentence with a **ground term** (a term with no variables).
- Every instantiation of a universally quantified sentence is entailed by it:

$$\frac{\forall v \alpha}{\text{Subst}(\{v / g\}, \alpha)} \quad \begin{array}{l} \alpha: \text{sentence} \\ v: \text{variable} \\ g: \text{ground term} \end{array}$$

$\text{Subst}(\theta, \alpha)$ is the substituted version of α by θ .

Example: We can infer *Likes(John, IceCream)*

from $\forall x \text{ Likes}(x, \text{IceCream})$

with the substitution $\{x/\text{John}\}$

Universal Instantiation (UI)

- Universal instantiation can be applied multiple times to add new sentences to the KB.
 - There are infinitely many possible new sentences if the KB contains functions (see example below).
- Example:

The sentence $\forall x \text{ King}(x) \wedge \text{Greedy}(x) \Rightarrow \text{Evil}(x)$ entails

$\text{King}(\text{John}) \wedge \text{Greedy}(\text{John}) \Rightarrow \text{Evil}(\text{John})$

$\text{King}(\text{Richard}) \wedge \text{Greedy}(\text{Richard}) \Rightarrow \text{Evil}(\text{Richard})$

$\text{King}(\text{Father}(\text{John})) \wedge \text{Greedy}(\text{Father}(\text{John})) \Rightarrow \text{Evil}(\text{Father}(\text{John}))$

etc.

Existential Instantiation (EI)

- A new (previously unused) constant symbol is used to represent an instance:

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

α : sentence
 v : variable
 k : new symbol

- This new constant is called a **Skolem constant**.
- Existential instantiation can be applied once to replace a sentence with existential quantification.
- Example:

The sentence $\exists x \text{ King}(x) \wedge \text{Greedy}(x)$
becomes $\text{King}(C_1) \wedge \text{Greedy}(C_1)$

The sentence $\exists x \text{ Crown}(x) \wedge \text{OnHead}(x, \text{John})$
becomes $\text{Crown}(W_2) \wedge \text{OnHead}(W_2, \text{John})$

Propositionalization

- Idea: Convert all the sentences in a FOL KB and the query to ground sentences (no variables).
- We can then apply inference rules in propositional logic to get our answers.
- Example: Consider the KB with the following sentences:

$\forall x \text{ King}(x) \wedge \text{Greedy}(x) \Rightarrow \text{Evil}(x)$

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

$\text{Greedy}(\text{John})$

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

Propositionalization

- For the universally quantified sentence, we need to generate corresponding ground sentences using all the possible ground terms:

King(John) \wedge Greedy(John) \Rightarrow Evil(John)

King(Richard) \wedge Greedy(Richard) \Rightarrow Evil(Richard)

- Problem: When there is a function symbol, there are an infinite number of ground terms. Example:

Father(John)

Father(Father(John))

Father(Father(Father(John)))

etc.

Propositionalization

- Herbrand's theorem (1930): If a sentence is entailed by an FOL KB, it is entailed by a finite subset of the propositional KB.
- Solution: We can iteratively increase the depth of nested function terms until the entailment is proved.
- Problem: If the sentence is not entailed, the process can go on forever.
- Inference in FOL via propositionalization is complete (all entailed sentences can be found).
- Entailment in FOL is semidecidable if the KB contains functions; non-entailment can not be proved.

Inefficiency of Propositionalization

- Propositionalization can generate many irrelevant sentences. Example: The goal is to query the KB about

Evil(John)

- However, the following is also generated, which is irrelevant:

King(Richard) \wedge Greedy(Richard) \Rightarrow Evil(Richard)

etc.

Unification

- This is the process of making two sentences identical through substitution.
- Example: *Greedy(John)* and *Greedy(x)* are unified by the substitution $\{x/John\}$.
- **Unifier**: $\text{Unify}(p, q) = \theta$ where $\text{Subst}(\theta, p) = \text{Subst}(\theta, q)$

p	q	θ
<i>Knows(John, x)</i>	<i>Knows(John, Jane)</i>	$\{x/Jane\}$
<i>Knows(John, x)</i>	<i>Knows(y, Jane)</i>	$\{x/Jane, y/John\}$
<i>Knows(John, x)</i>	<i>Knows(y, Mother(y))</i>	$\{x/Mother(John), y/John\}$
<i>Knows(John, x)</i>	<i>Knows(x, Jane)</i>	???

The last case requires "standardizing apart": Renaming variables to avoid name clashes.

Generalized Modus Ponens

- Example: Consider the sentence

$$\forall x \text{ King}(x) \wedge \text{Greedy}(x) \Rightarrow \text{Evil}(x)$$

$$\forall y \text{ Greedy}(y)$$

If we know $\text{King}(\text{John})$, then the substitution $\{x/\text{John}, y/\text{John}\}$ makes the premise of the implication true.

\Rightarrow The consequence of the implication ($\text{Evil}(\text{John})$) is true.

- **Generalized Modus Ponens:** If there is a substitution θ such that $\text{Subst}(\theta, p_i') = \text{Subst}(\theta, p_i)$ for all i

then we have
$$\frac{p_1', p_2', \dots, p_n', (p_1 \wedge p_2 \wedge \dots \wedge p_n \Rightarrow q)}{\text{Subst}(\theta, q)}$$

- ◆ Note: Variables with no quantification are assumed to be universally quantified.

Example KB

Given: 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 to it by Colonel West, who is American.

Prove that Colonel West is a criminal

Example KB in FOL Definite Clauses

... it is a crime for an American to sell weapons to hostile nations:

$R1: American(x) \wedge Weapon(y) \wedge Sells(x, y, z) \wedge Hostile(z) \Rightarrow Criminal(x)$

Nono ...has some missiles

$\exists x Owns(Nono, x) \wedge Missile(x)$

(EI) $\rightarrow R2: Owns(Nono, M_1)$ and $R3: Missile(M_1)$

... all of its missiles were sold to it by Colonel West

$R4: Owns(Nono, x) \wedge Missile(x) \Rightarrow Sells(West, x, Nono)$

West, who is American

$R5: American(West)$

The country Nono, an enemy of America

$R6: Enemy(Nono, America)$

Missiles are weapons:

$R7: Missile(x) \Rightarrow Weapon(x)$

An enemy of America counts as "hostile"

$R8: Enemy(x, America) \Rightarrow Hostile(x)$

Forward Chaining for FOL

- New sentences are iteratively added to the KB.
- Termination:
 - Success: A sentence unifies with the query (the unifier is returned).
 - Failure: No new sentence can be generated.
 - Can loop forever if the KB contains functions (FOL entailment with definite clauses is semidecidable).
- In each iteration, every KB rule is checked:
 - If GMP can be applied to the rule (i.e., there exists some substitution that makes the premises true), then add the consequence to the KB if it is new (i.e., it does not unify with any existing sentence).

Forward Chaining Example

Iteration #1:

■ $\text{Unify}(\text{R2} \wedge \text{R3}, \text{premise}(\text{R4})) =$

→ *R9: Sells(West, M_1 , Nono)*

■ $\text{Unify}(\text{R3}, \text{premise}(\text{R7})) =$

→ *R10: Weapon(M_1)*

■ $\text{Unify}(\text{R6}, \text{premise}(\text{R8})) =$

→ *R11: Hostile(Nono)*

Iteration #2:

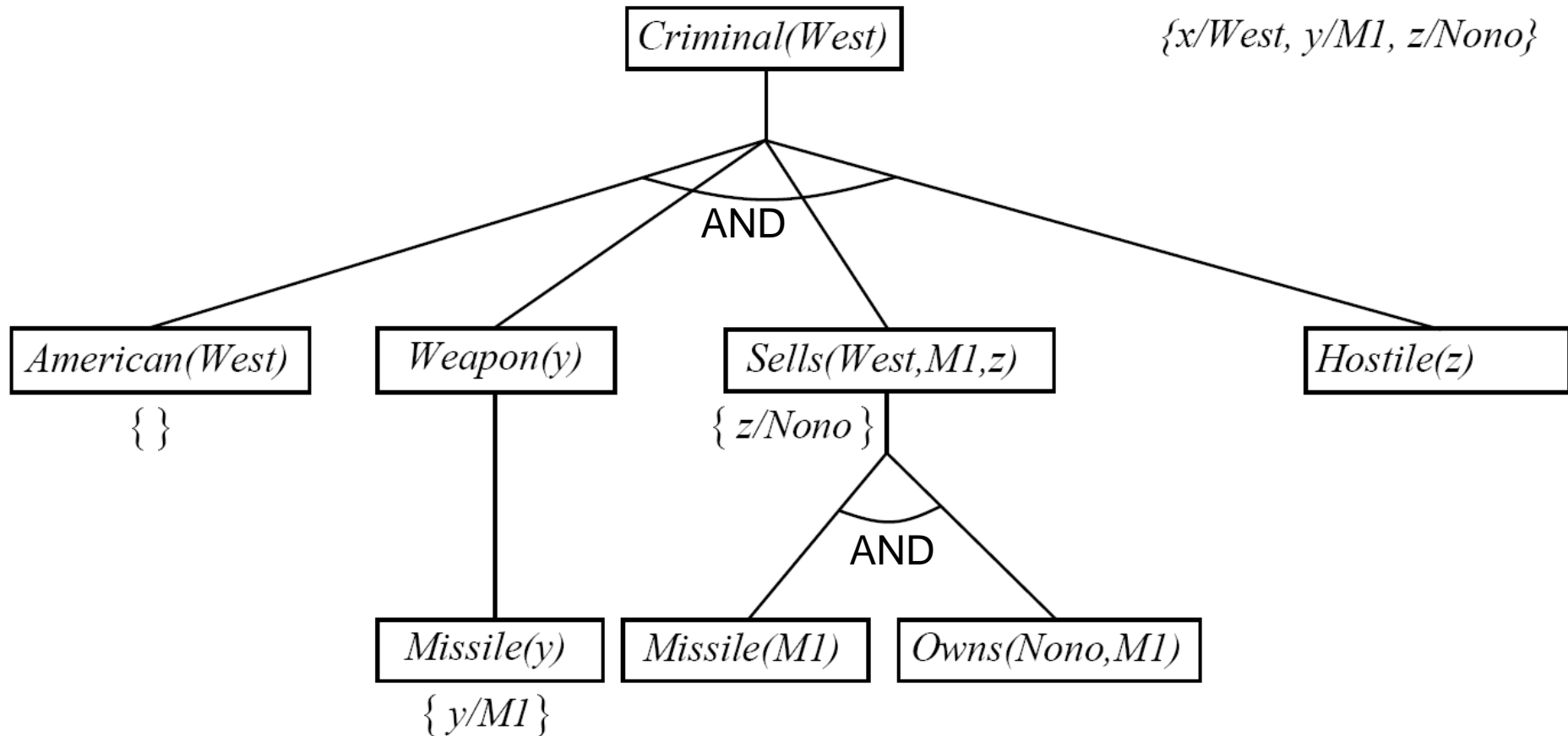
■ $\text{Unify}(\text{R5} \wedge \text{R10} \wedge \text{R9} \wedge \text{R11}, \text{premise}(\text{R1})) =$

→ *R12: Criminal(West)*

Backward Chaining for FOL

- Depth-first search starting from the goal.
 - OR-step: Find all the rules whose consequences unify with the goal.
 - AND-step: For such a (potentially useful) rule, try to prove each conjunct of its premise as a sub-goal.
- Each line of search terminates if it unifies with a known fact.
- The required substitution is kept tracked of during the search.
- Difficulties: repeated states, incompleteness.

Backward Chaining Example



FOL Inference with Resolution

$$\frac{l_1 \vee \dots \vee l_k, \quad m_1 \vee \dots \vee m_n}{\text{Subst}(\theta, \quad l_1 \vee \dots \vee l_{i-1} \vee l_{i+1} \vee \dots \vee l_k \vee m_1 \vee \dots \vee m_{j-1} \vee m_{j+1} \vee \dots \vee m_n)}$$

if θ unifies l_i and $\neg m_j$.

Example:
$$\frac{\neg Rich(x) \vee Unhappy(x), \quad Rich(Ken)}{Unhappy(Ken)}$$

with $\theta = \{x/Ken\}$

Convert FOL Sentences to CNF

- Example sentence:

$$\forall x [\forall y \textit{Animal}(y) \Rightarrow \textit{Loves}(x,y)] \Rightarrow [\exists y \textit{Loves}(y,x)]$$

- Eliminate biconditionals and implications

$$\forall x [\forall y \neg \textit{Animal}(y) \vee \textit{Loves}(x,y)] \Rightarrow [\exists y \textit{Loves}(y,x)]$$

$$\forall x [\neg \forall y \neg \textit{Animal}(y) \vee \textit{Loves}(x,y)] \vee [\exists y \textit{Loves}(y,x)]$$

- Move negations inward (de Morgan's)

$$\forall x [\exists y \neg(\neg \textit{Animal}(y) \vee \textit{Loves}(x,y))] \vee [\exists y \textit{Loves}(y,x)]$$

$$\forall x [\exists y \textit{Animal}(y) \wedge \neg \textit{Loves}(x,y)] \vee [\exists y \textit{Loves}(y,x)]$$

- Standardize variables (different for each quantifier)

$$\forall x [\exists y \textit{Animal}(y) \wedge \neg \textit{Loves}(x,y)] \vee [\exists \textcolor{red}{z} \textit{Loves}(\textcolor{red}{z},x)]$$

Convert FOL Sentences to CNF

- **Skolemize**: (more general than existential instantiation)
replace each existentially quantified variable with a **Skolem function** of all the enclosing universally quantified variables

$$\forall x [Animal(F(x)) \wedge \neg Loves(x, F(x))] \vee Loves(G(x), x)$$

(Use regular EI for existentially quantified variables that are not enclosed by universal quantification.)

- Drop universal quantifiers

$$[Animal(F(x)) \wedge \neg Loves(x, F(x))] \vee Loves(G(x), x)$$

- Distribute \wedge over \vee

$$[Animal(F(x)) \vee Loves(G(x), x)] \wedge [\neg Loves(x, F(x)) \vee Loves(G(x), x)]$$

Resolution Example (Criminal KB)

KB in CNF:

R1: $\neg American(x) \vee \neg Weapon(y) \vee \neg Sells(x, y, z) \vee \neg Hostile(z) \vee Criminal(x)$

R2: $Owns(Nono, M_1)$

R3: $Missile(M_1)$

R4: $\neg Owns(Nono, x) \vee \neg Missile(x) \vee Sells(West, x, Nono)$

R5: $American(West)$

R6: $Enemy(Nono, America)$

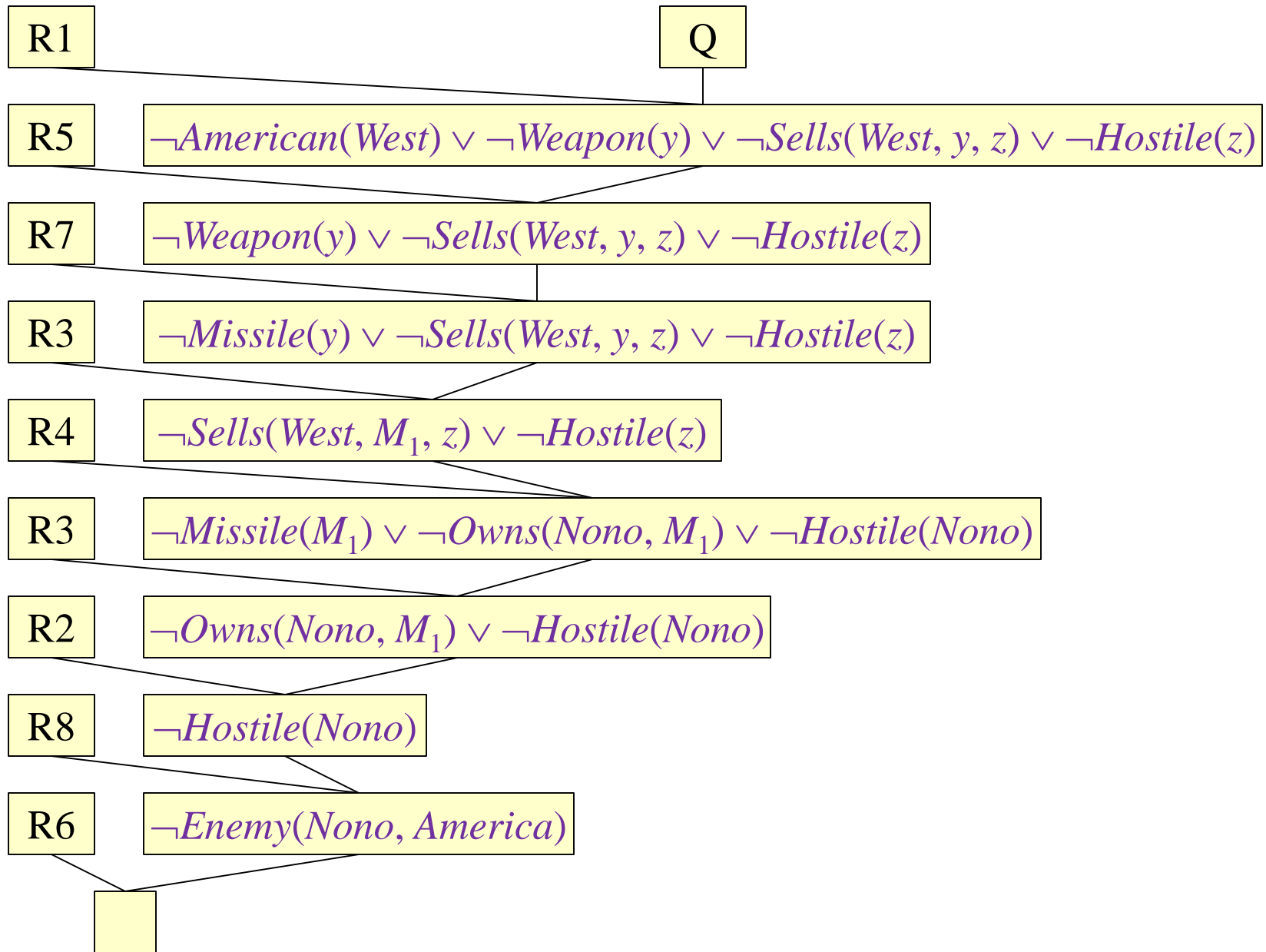
R7: $\neg Missile(x) \vee Weapon(x)$

R8: $\neg Enemy(x, America) \vee Hostile(x)$

Query: Q: $Criminal(West)$

Goal: To prove that $(KB \wedge \neg Q)$ is invalid (always false)

Resolution Example (Criminal KB)



What Next?

- Second-order logic: "Quantification over predicates (sets of objects)" and "Predicates defined on predicates", and some more.
- Many other different types of logics.
- Automated theorem proving:
 - Based on FOL (mostly).
 - Can have human-in-the-loop.
 - Many available systems today.
 - Have found proofs for some open mathematical problems.
 - Hardware and software verifications are the most practically important applications.

Logical Programming

- A snapshot of **Prolog**:

```
mother_child(trude, sally).  
father_child(tom, sally).  
father_child(tom, erica).  
father_child(mike, tom).  
sibling(X, Y) :- parent_child(Z, X), parent_child(Z, Y).  
parent_child(X, Y) :- father_child(X, Y).  
parent_child(X, Y) :- mother_child(X, Y).  
  
?- sibling(sally, erica).  
Yes
```

- The use of Prolog in real-world applications has been quite limited. Problems include limits of the language itself, efficiency, and portability.
- Its usefulness has been demonstrated in some specialized fields. An example is that Prolog is part of IBM Watson machine that won the *Jeopardy!* game in 2011.