Fundamentals of Artificial Intelligence

First Order Logic
(Predicate Logic)

Pros and Cons of Propositional Logic

- Propositional logic is declarative: pieces of syntax correspond to facts
- Propositional logic allows partial/disjunctive/negated information (unlike most data structures and databases)
- Propositional logic is compositional: meaning of $P \wedge Q$ is derived from meaning of P and of Q
- Meaning in propositional logic is context-independent (unlike natural language, where meaning depends on context)

- Propositional logic has very limited expressive power (unlike natural language)
  - E.g., cannot say 'pits cause breezes in adjacent squares' except by writing one sentence for each square

Limitations of Propositional Logic

- In propositional logic, we are not able represent the following sentences accurately.
    - “All men are mortal.”
    - “Socrates is a man.”
- And, we are not able deduct that “Socrates is mortal” from their representations.
- We need more expressive power, and the answer is the first order predicate logic.
- In predicate logic, we represent those sentences as:
    - $\forall x(Man(x) \rightarrow Mortal(x))$
    - Man(Socrates)
    - Mortal(Socrates)
- And, we are able to deduct the third one from first two.

Extending Propositional Logic into Predicate Logic

- Instead of propositions, Predicate Logic has predicates that take a predefined number ($\geq$0) of arguments (parameters).
- The arguments are terms intended to denote objects in some universe (not true/false statements).
- Terms may contains symbols denoting variables, constants, and functions.
- Formulas may include quantification over variables: "for all" x, "there exists" x.
- Logical connectives of propositional logic are still available in predicate logic.

Predicate Logic (First Order Logic)

- Whereas ***propositional logic*** assumes world contains **facts**, ***first-order logic*** (like natural language) assumes the world contains
- **Objects**: people, houses, numbers, theories, Ronald McDonald, colors, baseball games, wars, centuries, …
- **Relations**: red, round, bogus, prime, multistoried, brother of, bigger than, inside, part of, has color, occurred after, owns, comes between, …
- **Functions**: father of, best friend, third inning of, one more than, end of, …

Predicate Logic Syntax: Terms

- **Terms** are intended to denote **objects**, and terms are defined in BNF as follows:

$$\mathbf{t ::= x \mid c \mid f(t, \ldots, t)}$$

where x is a variable symbol, c is a constant (a function symbol with arity=0) and f is a function symbol with arity n > 0.

Definition: Term

- A **variable symbol** is a term.
- A **constant symbol** (a nullary function symbol) is a term.
- If $t_1, t_2, \ldots, t_n$ are terms and f is a function symbol with arity n > 0, then $\mathbf{f(t_1, t_2, \ldots, t_n)}$ is a term.
- Nothing else is a term.

Predicate Logic Syntax
Terms - Examples

- If x,y,z are variable symbols, a, b, c are constant symbols and f(with arity 1), g(with arity 2) are function symbols, then followings are tems.
  - x y z
  - a b c
  - f(a) f(x) f(f(x)) f(f(a)) f(f(f(a)))
  - g(a,b) g(x,y) g(a,z) g(f(a),g(b,y))

- But if P (with arity 1) is predicate symbol, the followings are NOT terms.
  - P(x) P(a) f(P(a))

Predicate Logic Syntax
Sentences (Formulas)

- A wff of predicate logic can be defined using following BNF rules.

$$\varphi ::= P(t_1, \ldots , t_n) \mid t_1 = t_2 \mid$$

Atomic Sentences

$$(\neg\varphi) \mid (\varphi \wedge \varphi) \mid (\varphi \vee \varphi) \mid (\varphi \Rightarrow \varphi) \mid (\varphi \Leftrightarrow \varphi) \mid$$

$$(\forall x\varphi) \mid (\exists x\varphi)$$

where P is a predicate symbol of arity $n \geq 0$, t_i are terms and x is a variable symbol.

Predicate Logic Syntax
Sentences (Formulas): Examples

- If P is a predicate symbol with arity 1, and Q is a predicate symbol with arity 2.
  - $P(x)$ $P(a)$ $P(f(a))$ $P(f(x))$
  - $Q(x,a)$ $Q(x,y)$ $Q(f(x),f(a))$
  - $\forall x P(x)$ $\exists x P(x)$
  - $\forall x Q(x,a)$ $\exists x Q(b,x)$ $\forall x Q(x,y)$ $\exists x Q(y,x)$
  - $\forall x \forall y Q(x,y)$ $\exists x \exists y Q(x,y)$ $\forall x \exists y Q(x,y)$
  - $\forall x (P(x) \Rightarrow \exists y Q(x,y))$

Free and Bound Variables

- Quantifiers bind variable occurrences within a sub-formula.
- Occurrences of a variable are bound by the most recent quantifier of that variable within that sub-formula.
- If the occurrence of a variable is not bound, it is said to be free.
- The same variable symbol may be bound by different quantifiers, when it occurs in different sub-formulas
    - renaming them improves clarity.
- A sentence (a closed formula) is a formula with no free variables.

Examples

- $\exists x P(x,y)$
- $\exists x(P(x,y) \wedge \forall y Q(x,y))$
- $\exists x(P(x,y) \wedge \forall x Q(x,y))$

Semantics of Predicate Logic - Models

- The meanings of **constant**, **function** and **predicate** symbols come from models.

A **model** M consists of:

- Each model identifies a universe **U**, a ***non-empty set of objects*** to which terms are intended to be mapped.
- Each constant symbol **c** is mapped to an object $\mathbf{c^M}$ in U.
- Each function symbol **f** is mapped to a function $\mathbf{f^M}$ from U^n to U,
  where n is the arity of f. $f^M: U^n \rightarrow U$
- Each predicate symbol P is mapped to a n-ary relation $\mathbf{P^M}$ (a subset of n-tuples U^n)
  where n is the arity of P.

Model - Examples

- Let **c** be a constant symbol, **f** be a function symbol (with arity 1), **P** be a predicate symbol (with arity 2).
- And, let M be a model as follows
  - The universe of M = {1,2,3}
  - $c^M = 1$
  - f^M is the function {(1,2),(2,3),(3,1)}
  - P^M is the relation{(1,1),(1,2)}
- Term Meanings:
  - f(c) maps to $f^M(c^M) = f^M(1) = 2$
  - f(f(c)) maps to $f^M(f^M(c^M)) = f^M(f^M(1)) = f^M(2) = 3$
- Atomic Formula Meanings:
  - P(a,f(c)) maps to True since $(1,2) \in P^M$
  - P(f(c),f(c)) maps to False since $(2,2) \notin P^M$

Semantics of Predicate Logic
Environments

- Meanings of free variables come from **environments** (**state** or **look-up table**).

- An **environment** is a function maps variables to objects of the universe of a model.

- Examples:
  - {(x,1),(y,2),(z,1)} is an environment ℓ.
    - a function from variables {x,y,z} to the universe of our model M (={1,2,3})
  - ℓ(x) is 1 since ℓ maps x to 1

Semantics of Predicate Logic
Satisfaction Relation

- **Truth values of formulas** of predicate logic are evaluated wrt **a model M and an environment ℓ.**
  - **Truth values of closed formulas** of predicate logic are evaluated wrt **a model M.**
- Given a model M with the universe U and given an environment ℓ, we define the satisfaction relation $M \vDash_\ell \varphi$ for each logical formula φ by structural induction on φ.

Satisfaction Relation:

- If φ is of the form $P(t_1,\dots t_n)$, then we map the terms $t_1,\dots,t_n$ to the objects $a_1,\dots,a_n$ in set U by using the model M and the environment ℓ.

  Now $M \vDash_\ell P(t_1,\dots t_n)$ holds iff $(a_1,\dots, a_n)$ is in the relation P^M.
- $M \vDash_\ell \forall x\psi$ holds iff $M \vDash_{\ell[x\to a]} \psi$ holds for all $a \in U$.
- $M \vDash_\ell \exists x\psi$ holds iff $M \vDash_{\ell[x\to a]} \psi$ holds for some $a \in U$.
- $M \vDash_\ell \neg\psi$ holds iff it is not the case that $M \vDash_\ell \psi$ holds.
- $M \vDash_\ell \psi_1 \vee \psi_2$ holds iff $M \vDash_\ell \psi_1$ or $M \vDash_\ell \psi_2$ holds.
- $M \vDash_\ell \psi_1 \wedge \psi_2$ holds iff $M \vDash_\ell \psi_1$ and $M \vDash_\ell \psi_2$ hold.
- $M \vDash_\ell \psi_1 \to \psi_2$ holds iff $M \vDash_\ell \psi_2$ holds whenever $M \vDash_\ell \psi_1$ holds.

Satisfaction Relation - Example

- Let M be a model as follows
  - The universe of M = {1,2} $c^M = 1$ P^M is the relation{(1,1),(1,2)}
- Let e be an environment {x→1, y→2}
- The following satisfaction relations hold or not?
- $M \vDash_e P(x,y)$
- $M \vDash_e P(y,x)$
- $M \vDash_e P(x,c)$
- $M \vDash_e P(y,c)$

Satisfaction Relation - Example

- Let M be a model as follows
  - The universe of M = {1,2} $c^M = 1$ P^M is the relation{(1,1),(1,2)}
- Let e be an environment {x→1, y→2}
- The following satisfaction relations hold or not?
- $M \vDash_e P(x,y)$ YES $(1,2) \in P^M$
- $M \vDash_e P(y,x)$ NO $(2,1) \notin P^M$
- $M \vDash_e P(x,c)$ YES $(1,1) \in P^M$
- $M \vDash_e P(y,c)$ NO $(2,1) \notin P^M$

Satisfaction Relation - Example

- Let M be a model as follows
  - The universe of M = {1,2} $c^M = 1$ P^M is the relation{(1,1),(1,2)}
- Let e be an environment {x→1, y→2}
- The following satisfaction relations hold or not?
- $M \vDash_e \forall y P(x,y)$
- $M \vDash_e \forall x P(x,y)$
- $M \vDash \forall x \forall y P(x,y)$
- $M \vDash \forall x \exists y P(x,y)$
- $M \vDash \exists x \forall y P(x,y)$

Satisfaction Relation - Example

- Let M be a model as follows
  - The universe of M = {1,2} $c^M = 1$ P^M is the relation{(1,1),(1,2)}
- Let e be an environment {x→1, y→2}
- The following satisfaction relations hold or not?
- $M \vDash_e \forall y P(x,y)$ YES (1,1) and (1,2) ∈ P^M
- $M \vDash_e \forall x P(x,y)$ NO (2,2)∉P^M
- $M \vDash \forall x \forall y P(x,y)$ NO (2,1) and (2,2) ∉ P^M
- $M \vDash \forall x \exists y P(x,y)$ NO (2,1) or (2,2) ∉ P^M
- $M \vDash \exists x \forall y P(x,y)$ YES (1,1) and (1,2) ∈ P^M

Reasoning Over All Models

- **Validity:**
  φ is valid if $M \models_{\ell} \varphi$ for all M, ℓ.

- **Unsatisfiable:**
  φ is unsatisfiable if $M \not\models_{\ell} \varphi$ for all M, ℓ.

- **Satisfiability:**
  φ is satisfiable if there exists M, ℓ such that $M \models_{\ell} \varphi$.

- **Entailment:**
  $\psi \models \varphi$ if $M \models_{\ell} \varphi$ whenever $M \models_{\ell} \psi$ holds for every M, ℓ.

Reasoning Over All Models - Examples

- Decide the following formulas are **valid**, **satisfiable** (but not valid) or **unsatisfiable**.
  - $\forall x(P(x) \rightarrow P(x))$
  - $\forall x(P(x) \lor Q(x))$
  - $\forall x(P(x)) \land \exists y(!P(y))$

- Decide whether following logical consequence relations hold or not.
  - $\forall x(P(x)) \land \forall y(Q(y)) \vDash \forall x(P(x) \land Q(x))$
  - $\forall x(P(x) \lor Q(x)) \vDash \forall x(P(x)) \lor \forall y(Q(y))$

Reasoning Over All Models - Examples

- Decide the following formulas are **valid**, **satisfiable** (but not valid) or **unsatisfiable**.
  - $\forall x(P(x)\rightarrow P(x))$ Valid
  - $\forall x(P(x)\vee Q(x))$ Satisfiable
  - $\forall x(P(x)) \wedge \exists y(!P(y))$ Unsatisfiable

- Decide whether following logical consequence relations hold or not.
  - $\forall x(P(x))\wedge\forall y(Q(y)) \vDash \forall x(P(x)\wedge Q(x))$ Holds
  - $\forall x(P(x)\vee Q(x)) \vDash \forall x(P(x))\vee\forall y(Q(y))$ Does NOT hold

Fun with Sentences: Quantifiers

- Everybody loves somebody $\forall x\ \exists y\ Loves(x, y)$
- There is someone who is loved by everyone $\exists y\ \forall x\ Loves(x, y)$
- Quantifier duality: each can be expressed using the other

  $\forall x\ Likes(x, IceCream)$ is equivalent to $\neg\exists x\ \neg Likes(x, IceCream)$

  $\exists x\ Likes(x, Broccoli)$ is equivalent to $\neg\forall x\ \neg Likes(x, Broccoli)$

  $\neg\forall x\ Likes(x, IceCream)$ is equivalent to $\exists x\ \neg Likes(x, IceCream)$

  $\neg\exists x\ Likes(x, Broccoli)$ is equivalent to $\forall x\ \neg Likes(x, Broccoli)$

- One's mother is one's female parent $\forall m \forall c\ Mother(m,c) \Leftrightarrow Female(m) \land Parent(m, c)$

Interacting with FOL KBs

Suppose a wumpus-world agent is using an FOL KB
and perceives a smell and a breeze (but no glitter) at $t = 5$:

$Tell(KB, Percept([Smell, Breeze, None], 5))$
$Ask(KB, \exists\, a \;\; Action(a, 5))$

I.e., does KB entail any particular actions at $t = 5$?

Answer: $Yes,\ \{a/Shoot\}$ $\leftarrow$ substitution (binding list)

Given a sentence S and a substitution σ,
$S\sigma$ denotes the result of plugging σ into S; e.g.,
$S = Smarter(x, y)$
$\sigma = \{x/Hillary, y/Bill\}$
$S\sigma = Smarter(Hillary, Bill)$

$Ask(KB, S)$ returns some/all σ such that $KB \models S\sigma$

Inference in Predicate Logic
Proofs

Proofs

- How can we produce a proof for query **Evil(x)** from the following KB?

  $\forall$x King(x) $\wedge$ Greedy(x) $\Rightarrow$ Evil(x)

  King(John)

  Greedy(John)

- Substitution {x/John} solves the query Evil(x)
  - To use the rule that greedy kings are evil, find some x such that x is a king and x is greedy, and then infer that this x is evil.
  - More generally, if there is some substitution θ that makes each of the conjuncts of the premise of the implication identical to sentences already in the knowledge base, then we can assert the conclusion of the implication, after applying θ.

Proofs

- We will assume that our KB
  - **contains only definite clauses (exactly one positive literal) and**
  - **all variables are universally quantified.**

- We will use a single inference rule called as **Generalized Modus Ponens** *(A Variation of Resolution Rule for Horn Clauses)* in our proofs.

Generalized Modus Ponens

- **Generalized Modus Ponens (GMP) Inference Rule:**

$$\frac{p_1{}',\ p_2{}',\ \ldots,\ p_n{}',\ (p_1 \wedge p_2 \wedge \ldots \wedge p_n \Rightarrow q)}{q\theta} \qquad \text{where } p_i{}'\theta = p_i\theta \text{ for all } i$$

- There are n+1 premises to this GMP rule: the n atomic sentences $p_i{}'$ and the one implication.
- The conclusion is the result of applying the substitution θ to the consequent q.

Generalized Modus Ponens

- A proof for query **Evil(x)** from the following KB using GMP?

  ∀x King(x) ∧ Greedy(x) ⇒ Evil(x)

  King(John)

  Greedy(John)

$$\frac{p_1{}',\ p_2{}',\ \ldots,\ p_n{}',\ (p_1 \wedge p_2 \wedge \ldots \wedge p_n \Rightarrow q)}{q\theta}$$ where $p_i{}'\theta = p_i\theta$ for all i

$p_1{}'$ is $King(John)$ $\quad$ p_1 is $King(x)$

$p_2{}'$ is $Greedy(y)$ $\quad$ p_2 is $Greedy(x)$

θ is $\{x/John, y/John\}$ $\quad$ q is $Evil(x)$

$q\theta$ is $Evil(John)$

Unification

- Finding substitutions that make different logical expressions look identical is called **unification.**

- The **UNIFY algorithm** takes two sentences and returns a unifier for them if one exists:

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

Unification

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

$\theta = \{x/John, y/John\}$ works

$\text{UNIFY}(\alpha, \beta) = \theta$ if $\alpha\theta = \beta\theta$

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

Unification Algorithm

```
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
```

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

   if {var/val} ∈ θ then return UNIFY(val, x, θ)
   else if {x/val} ∈ θ then return UNIFY(var, val, θ)
   else if OCCUR-CHECK?(var, x) then return failure
   else return add {var/x} to θ
```

Forward Chaining Algorithm

```
function FOL-FC-ASK(KB, α) returns a substitution or false
  repeat until new is empty
     new ← { }
     for each sentence r in KB do
        ( p1 ∧ . . . ∧ pn ⇒ q) ← STANDARDIZE-APART(r)
        for each θ such that (p1 ∧ . . . ∧ pn)θ = (p'1 ∧ . . . ∧ p'n)θ
                      for some p'1, . . . , p'n in KB
           q' ← SUBST(θ, q)
           if q' is not a renaming of a sentence already in KB or new then do
               add q' to new
               φ ← UNIFY(q', α)
               if φ is not fail then return φ
     add new to KB
  return false
```

- On each iteration, it adds all the atomic sentences that can be inferred in one step from the implication sentences and the atomic sentences already in KB.
- STANDARDIZE-VARIABLES replaces all variables with new ones that have not been used before.

Generalized Modus Ponens: Example

Knowledge Base:

- 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 Col. West is a criminal.

Generalized Modus Ponens: Example

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

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

Nono ... has some missiles, i.e., $\exists\, x\ Owns(Nono, x) \wedge Missile(x)$:

$Owns(Nono, M_1)$ and $Missile(M_1)$

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

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

Missiles are weapons:

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

An enemy of America counts as "hostile":

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

West, who is American ...

$American(West)$

The country Nono, an enemy of America ...

$Enemy(Nono, America)$

Forward Chaining Proof

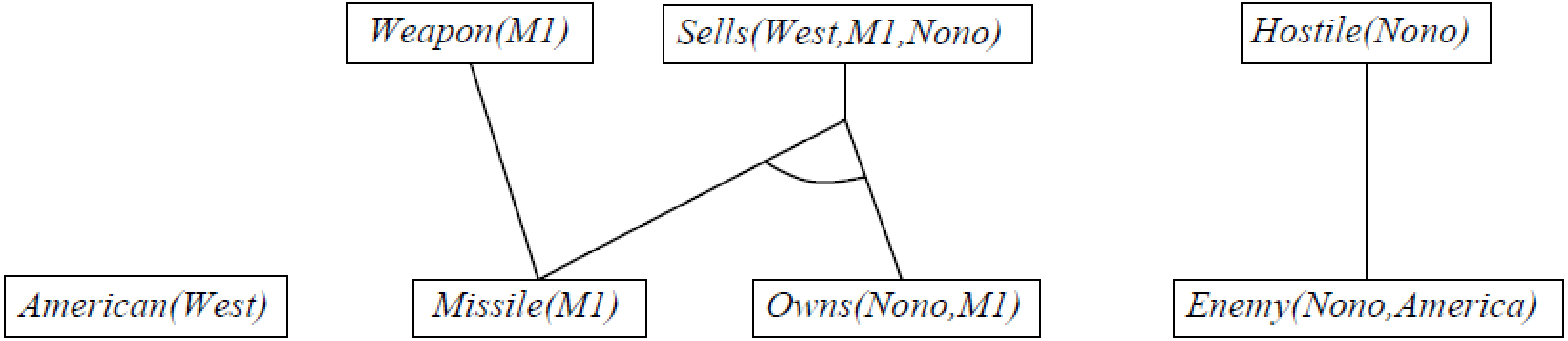
American(West)

Missile(M1)

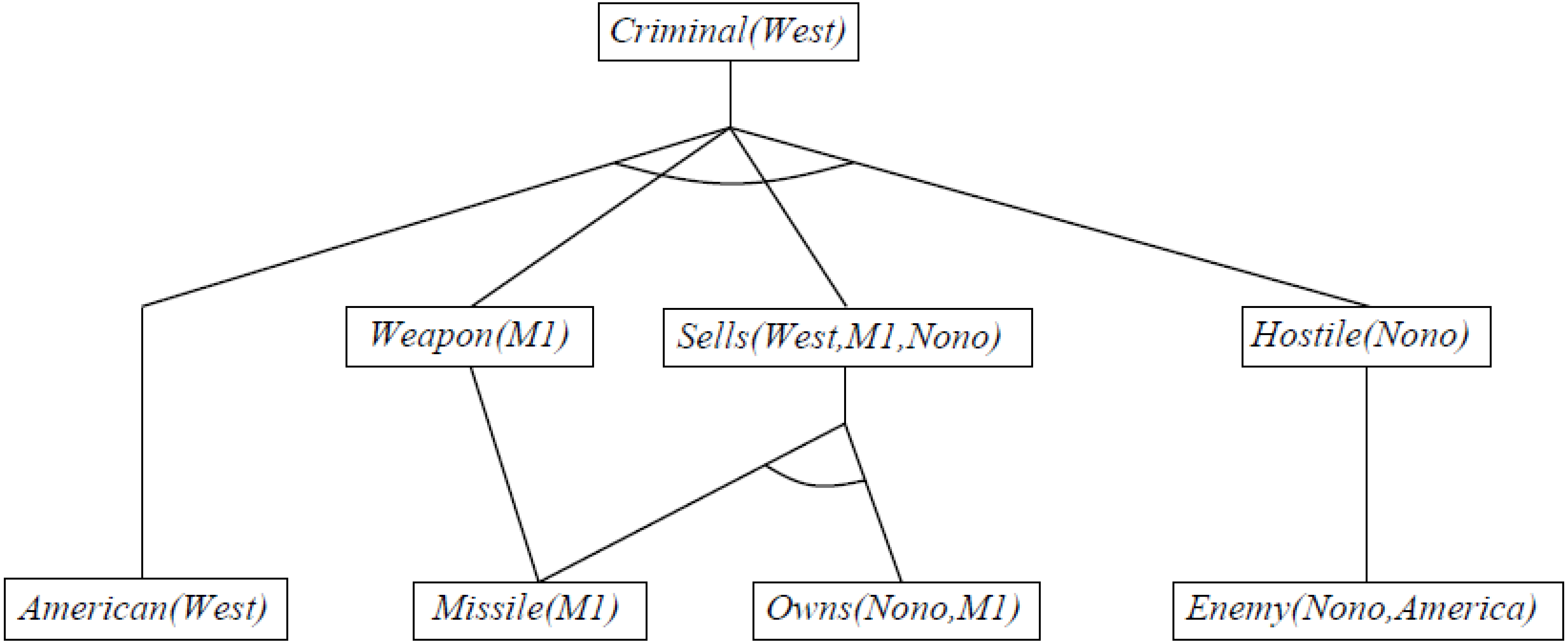
Owns(Nono,M1)

Enemy(Nono,America)

Forward Chaining Proof



Forward Chaining Proof



Backward Chaining Algorithm

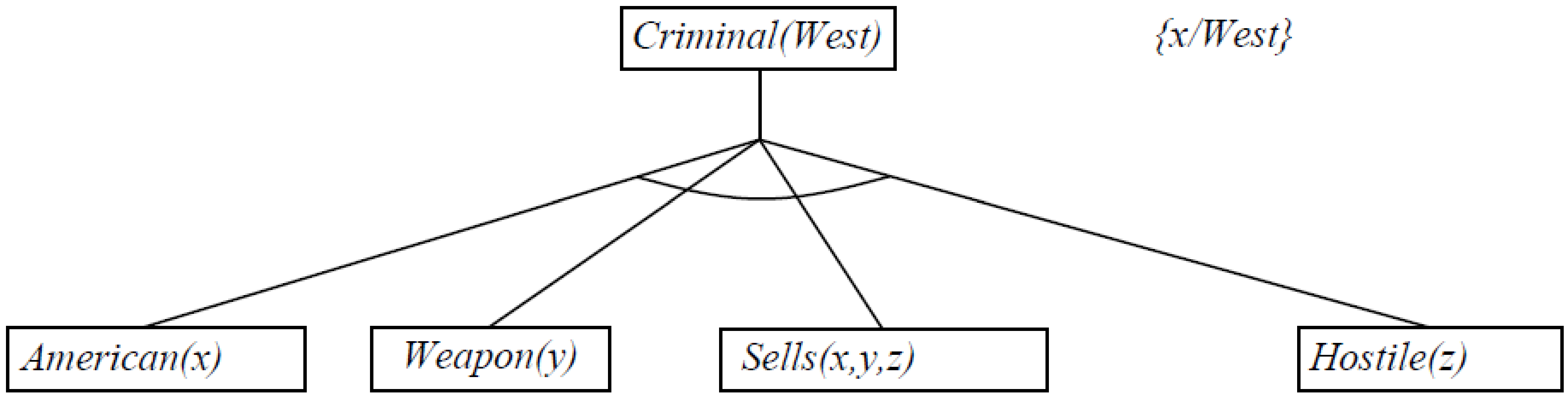
function FOL-BC-ASK(KB, $goals$, θ) **returns** a set of substitutions
  inputs: KB, a knowledge base
    $goals$, a list of conjuncts forming a query (θ already applied)
    θ, the current substitution, initially the empty substitution $\{\ \}$
  local variables: $answers$, a set of substitutions, initially empty

  if $goals$ is empty **then return** $\{\theta\}$
  $q' \leftarrow$ SUBST(θ, FIRST($goals$))
  for each sentence r **in** KB
    where STANDARDIZE-APART(r) = ($p_1 \land \ldots \land p_n \Rightarrow q$)
    and $\theta' \leftarrow$ UNIFY(q, q') succeeds
   $new\_goals \leftarrow [p_1, \ldots, p_n | \text{REST}(goals)]$
   $answers \leftarrow$ FOL-BC-ASK(KB, new_goals, COMPOSE(θ', θ)) $\cup$ $answers$
  return $answers$

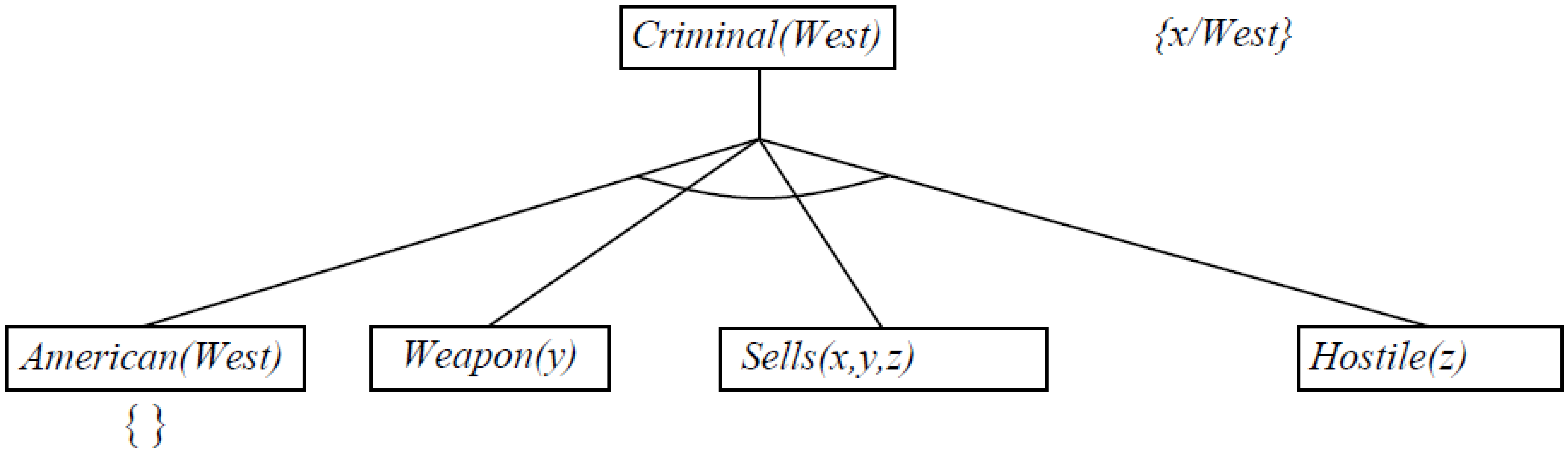
Backward Chaining Algorithm: Example

$Criminal(West)$

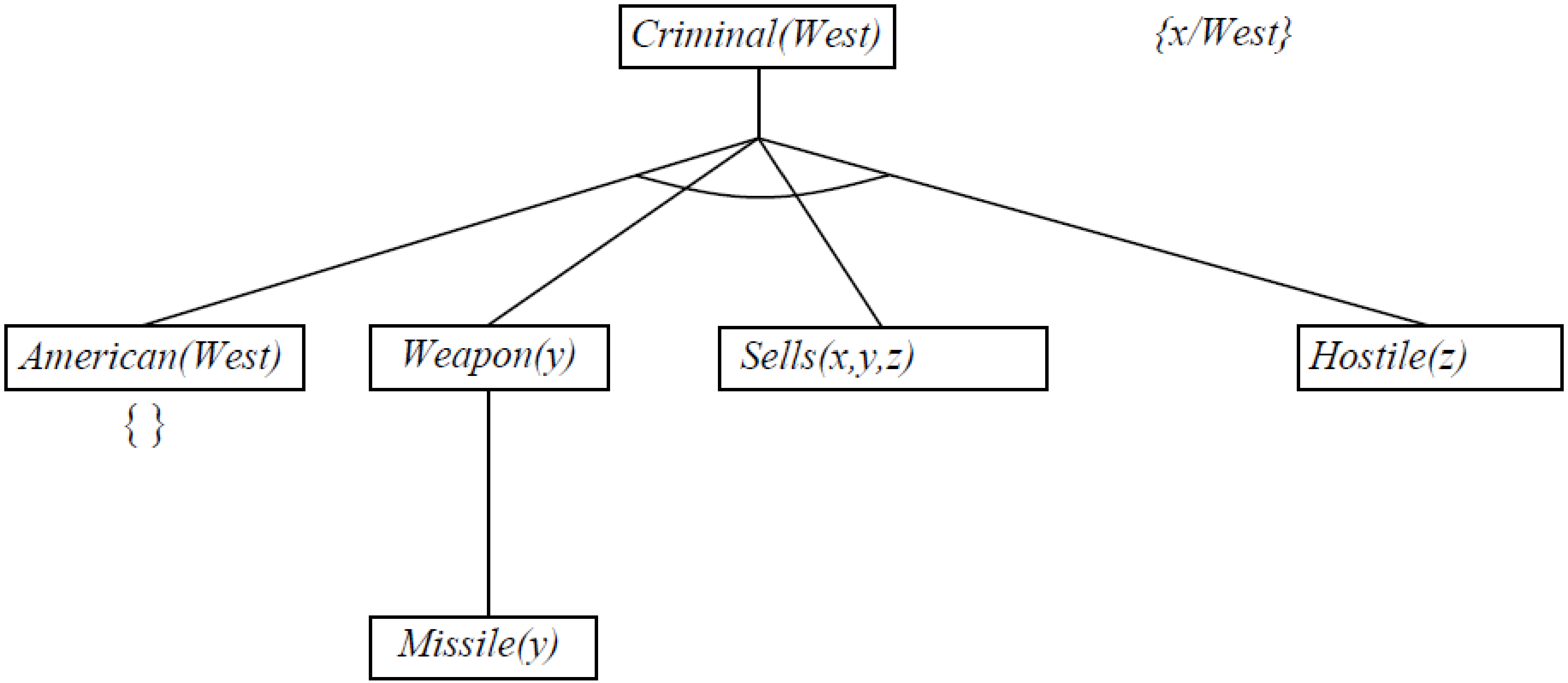
Backward Chaining Algorithm: Example



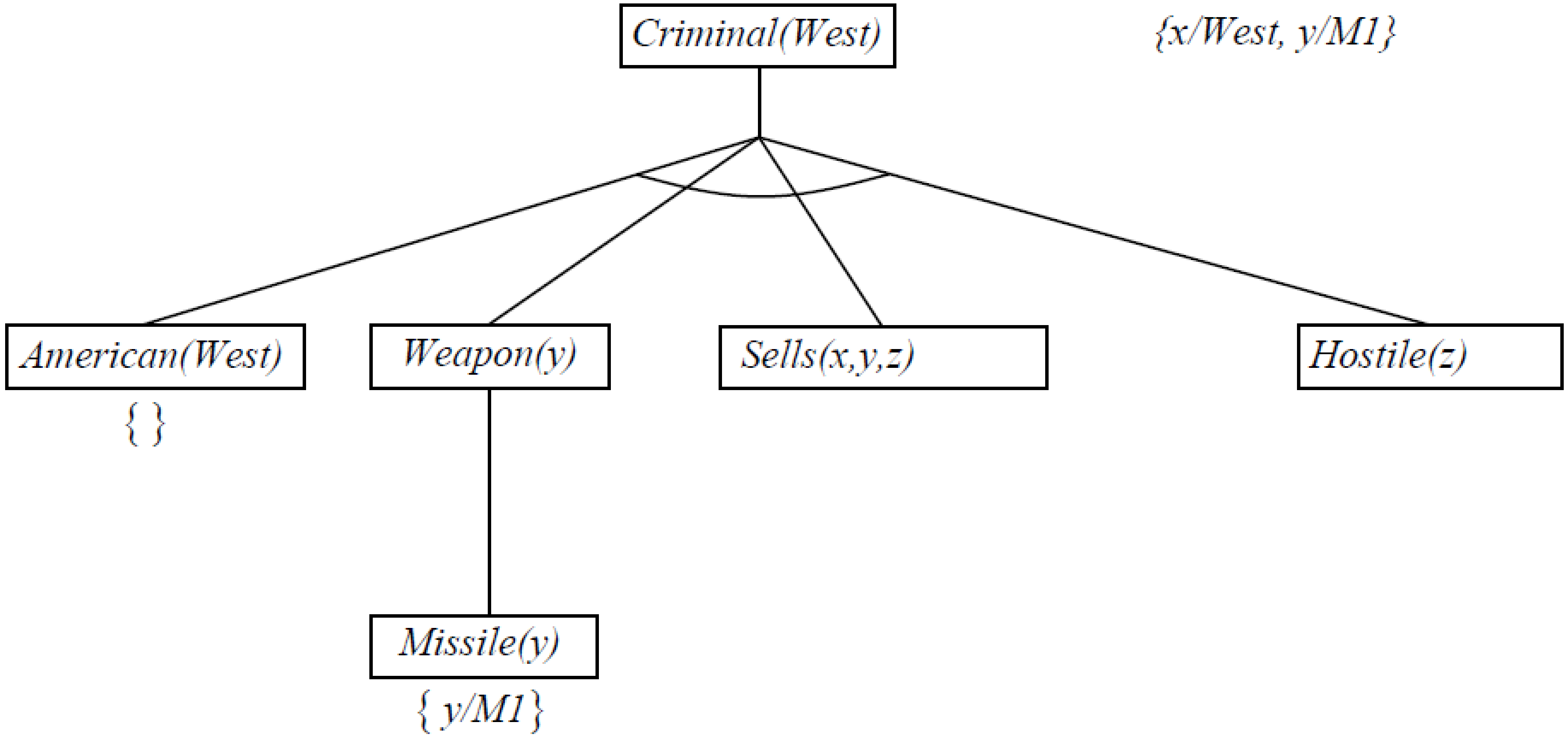
Backward Chaining Algorithm: Example



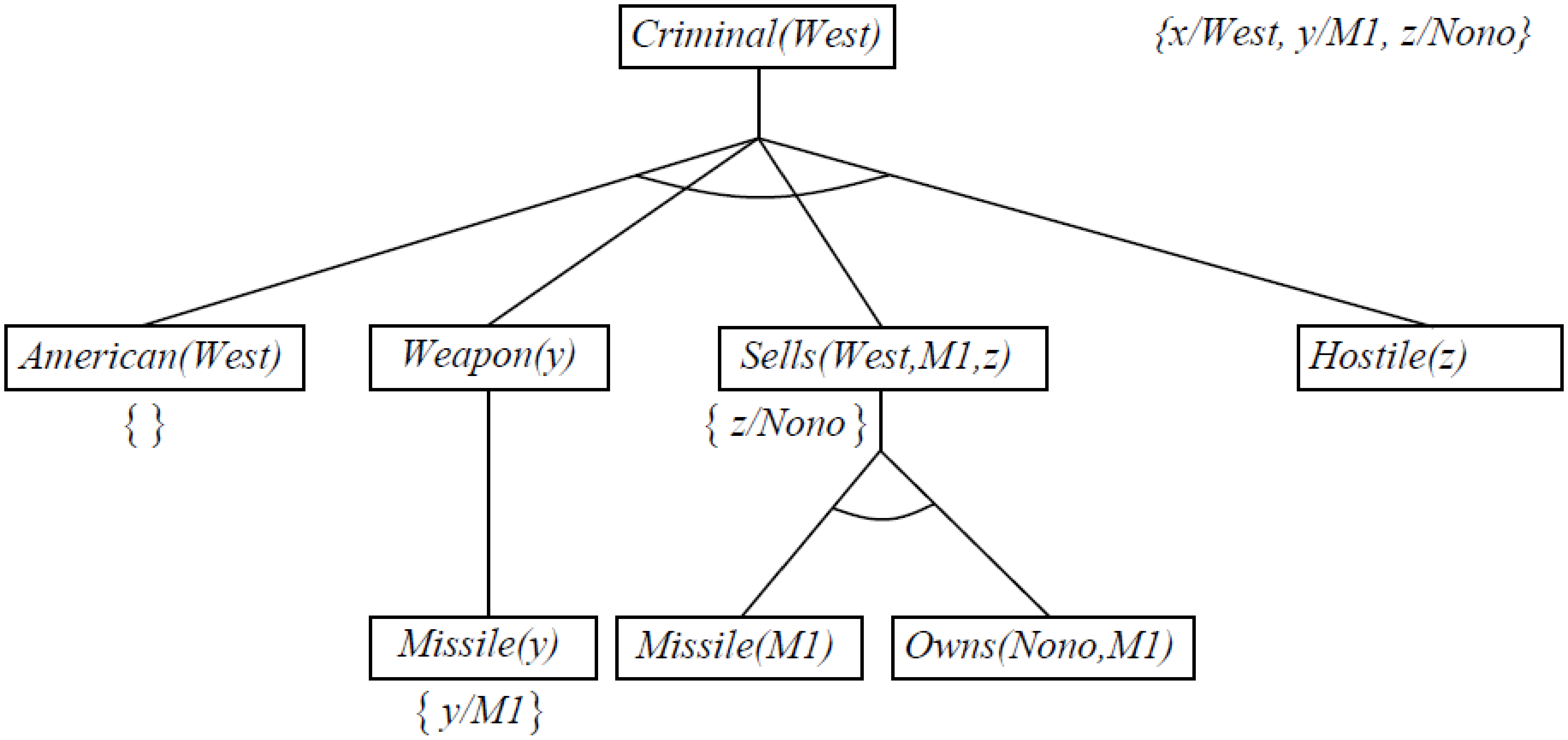
Backward Chaining Algorithm: Example



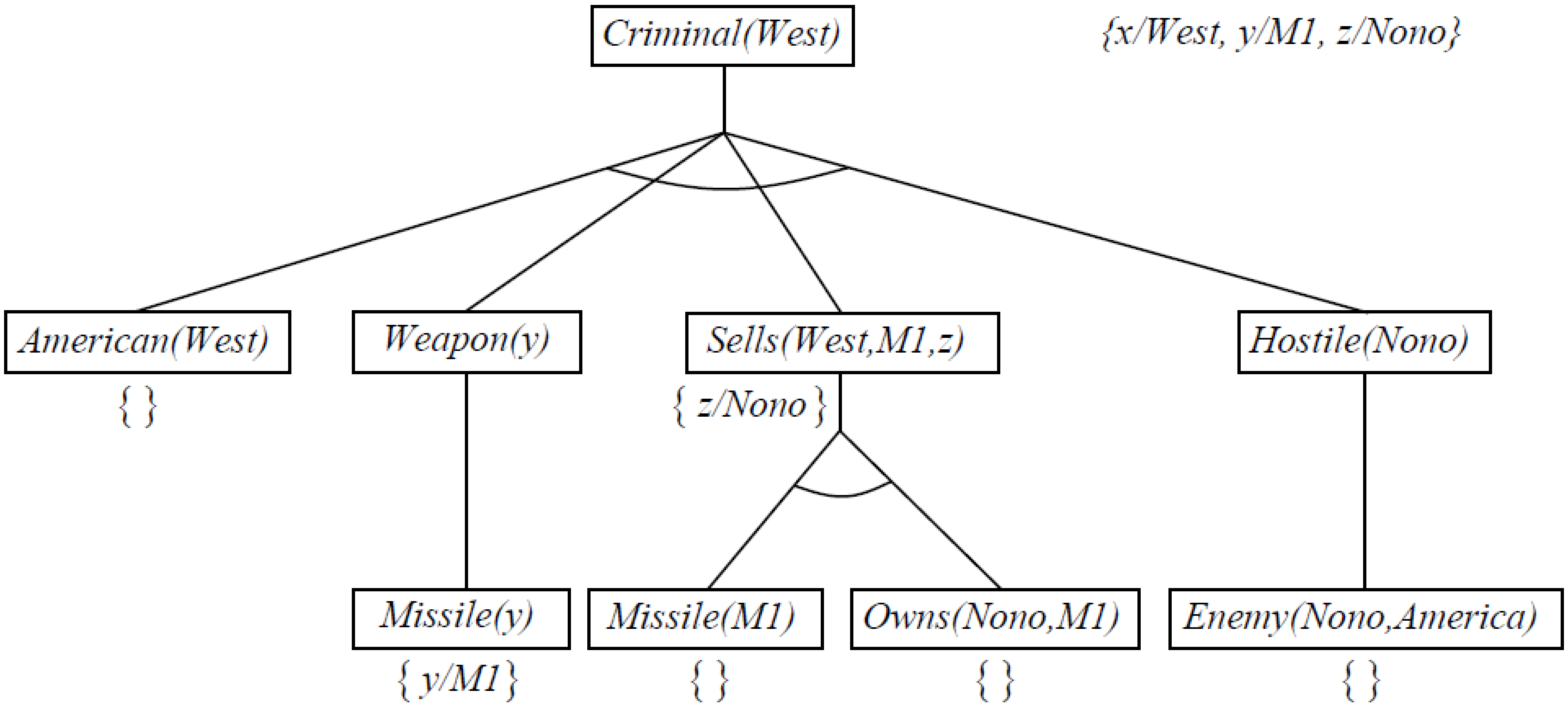
Backward Chaining Algorithm: Example



Backward Chaining Algorithm: Example



Backward Chaining Algorithm: Example



FOL Resolution: Brief Summary

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 $\textsc{Unify}(\ell_i, \neg m_j) = \theta$.

For example,

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

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

Apply resolution steps to $CNF(KB \wedge \neg\alpha)$; complete for FOL

Conversion to CNF

Everyone who loves all animals is loved by someone:

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

1. Eliminate biconditionals and implications

$$\forall x\ [\neg\forall y\ \ \neg Animal(y) \lor Loves(x,y)] \lor [\exists y\ \ Loves(y,x)]$$

2. Move $\neg$ inwards: $\neg\forall x, p\ \equiv \exists x\ \neg p$, $\neg\exists x, p\ \equiv \forall x\ \neg p$:

$$\forall x\ [\exists y\ \ \neg(\neg Animal(y) \lor Loves(x,y))] \lor [\exists y\ \ Loves(y,x)]$$
$$\forall x\ [\exists y\ \ \neg\neg Animal(y) \land \neg Loves(x,y)] \lor [\exists y\ \ Loves(y,x)]$$
$$\forall x\ [\exists y\ \ Animal(y) \land \neg Loves(x,y)] \lor [\exists y\ \ Loves(y,x)]$$

Conversion to CNF

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

$$\forall x \ \ [\exists y \ \ Animal(y) \land \neg Loves(x,y)] \lor [\exists z \ \ Loves(z,x)]$$

4. Skolemize: a more general form of existential instantiation.
   Each existential variable is replaced by a Skolem function
   of the enclosing universally quantified variables:

$$\forall x \ \ [Animal(F(x)) \land \neg Loves(x, F(x))] \lor Loves(G(x), x)$$

5. Drop universal quantifiers:

$$[Animal(F(x)) \land \neg Loves(x, F(x))] \lor Loves(G(x), x)$$

6. Distribute $\land$ over $\lor$:

$$[Animal(F(x)) \lor Loves(G(x), x)] \land [\neg Loves(x, F(x)) \lor Loves(G(x), x)]$$

Resolution Proof: Definite Clauses