

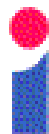


Resolution-Based Inference

R&N: §9.3-9.6

Jacques Fleuriot

 School of
informatics
University of Edinburgh



Outline

- Forward chaining
- Backward chaining
- Resolution

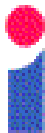
Forward chaining algorithm

```
function FOL-FC-ASK( $KB, \alpha$ ) returns a substitution or false
  inputs:  $KB$ , the knowledge base, a set of first-order definite clauses
            $\alpha$ , 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
       $(p_1 \wedge \dots \wedge p_n \Rightarrow q) \leftarrow \text{STANDARDIZE-VARIABLES}(\textit{rule})$ 
      for each  $\theta$  such that  $\text{SUBST}(\theta, p_1 \wedge \dots \wedge p_n) = \text{SUBST}(\theta, p'_1 \wedge \dots \wedge p'_n)$ 
        for some  $p'_1, \dots, p'_n$  in  $KB$ 
           $q' \leftarrow \text{SUBST}(\theta, q)$ 
          if  $q'$  does not unify with some sentence already in  $KB$  or new then
            add  $q'$  to new
             $\phi \leftarrow \text{UNIFY}(q', \alpha)$ 
            if  $\phi$  is not fail then return  $\phi$ 
      add new to  $KB$ 
  return false
```

← Pattern-matching

← Facts irrelevant to the goal can be generated



'West' Knowledge Base

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

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

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

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

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

$American(West)$ and $Enemy(Nono,America)$

Forward chaining proof

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

Owns(Nono, M₁) and *Missile(M₁)*

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

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

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

American(West) and *Enemy(Nono,America)*

American(West)

Missile(M₁)

Owns(Nono, M₁)

Enemy(Nono,America)

Forward chaining proof

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

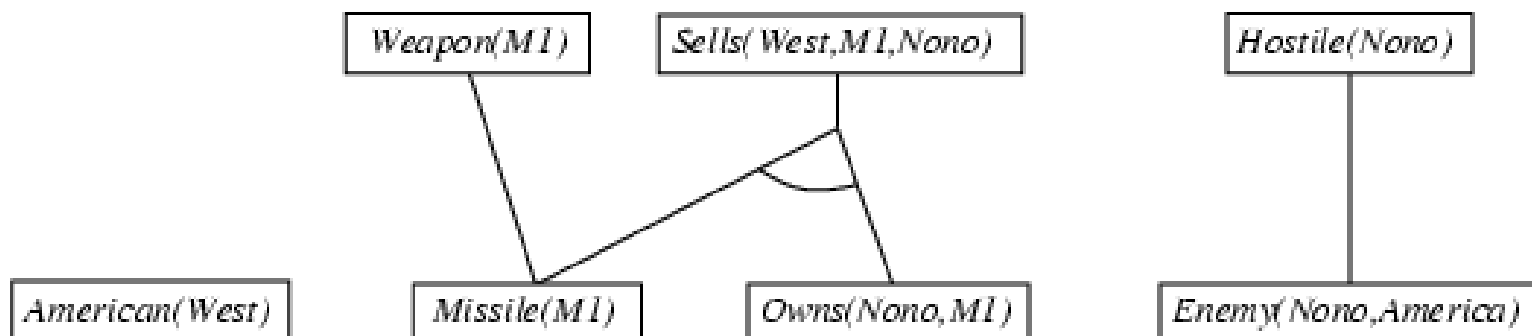
$Owens(Nono,M_1)$ and $Missile(M_1)$

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

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

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

$American(West)$ and $Enemy(Nono,America)$



Forward chaining proof

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

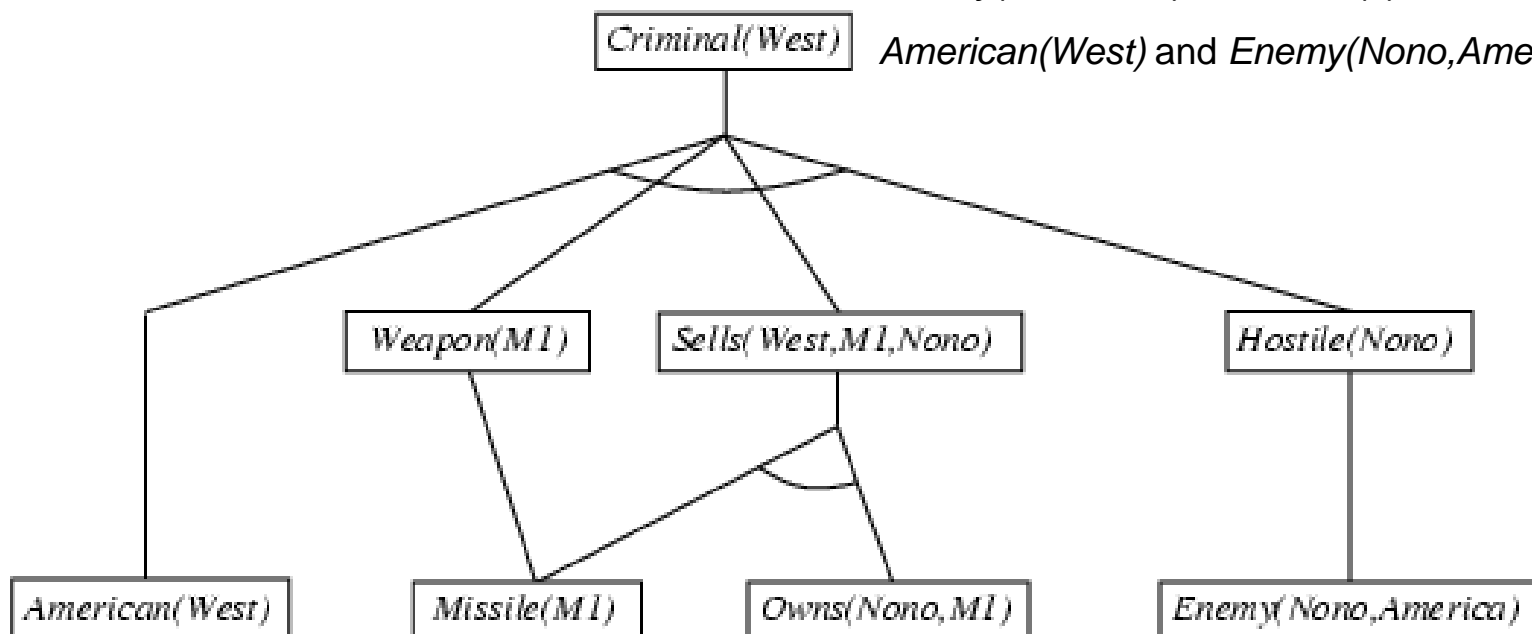
$Owns(Nono,M_1)$ and $Missile(M_1)$

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

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

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

$American(West)$ and $Enemy(Nono,America)$



Properties of forward chaining II

- Sound and complete for first-order definite clauses
 - Definite clause = exactly one positive literal.
- **Datalog** = first-order definite clauses + no functions
 - FC terminates for Datalog in finite number of iterations
- May not terminate in general if α is **not** entailed
- This is unavoidable: entailment with definite clauses is **semi-decidable**

Efficiency of forward chaining I

Incremental forward chaining: no need to match a rule on iteration k if a premise wasn't added on iteration $k-1$

⇒ match each rule whose premise contains a newly added positive literal

Matching itself can be expensive:

Database indexing allows $O(1)$ retrieval of known facts

– e.g., query $Missile(x)$ retrieves $Missile(M_1)$

Forward chaining is widely used in **deductive databases**

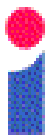
Efficiency of forward chaining II

- Pattern Matching

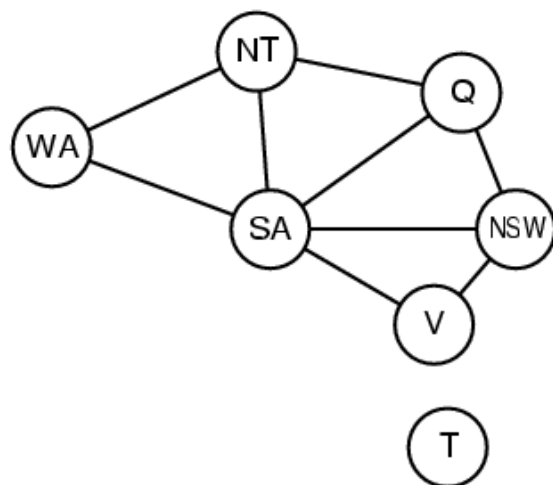
- For each θ such that $\text{SUBST}(\theta, p_1 \wedge \dots \wedge p_n) = \text{SUBST}(\theta, p'_1 \wedge \dots \wedge p'_n)$ for some p'_1, \dots, p'_n in KB
- Finding all possible unifiers can be very expensive

- Example

- $\text{Missile}(x) \wedge \text{Owns}(\text{Nono}, x) \Rightarrow \text{Sells}(\text{West}, x, \text{Nono})$
- Can find each object owned by Nono in constant time and then check if it is a missile
- But what if Nono owns many objects but very few missiles?
- Conjunct Ordering: Better (cost-wise) to find all missiles first and then check whether they are owned by Nono
- Optimal ordering is NP-hard. Heuristics available: e.g. MRV from CSP if each conjunct is viewed as a constraint on its vars



Hard matching example



$$\begin{aligned} & Diff(wa,nt) \wedge Diff(wa,sa) \wedge Diff(nt,q) \wedge \\ & Diff(nt,sa) \wedge Diff(q,nsw) \wedge Diff(q,sa) \wedge \\ & Diff(nsw,v) \wedge Diff(nsw,sa) \wedge Diff(v,sa) \Rightarrow \\ & \text{Colourable} \end{aligned}$$

$$\begin{aligned} & Diff(\text{Red},\text{Blue}) \quad Diff(\text{Red},\text{Green}) \\ & Diff(\text{Green},\text{Red}) \quad Diff(\text{Green},\text{Blue}) \\ & Diff(\text{Blue},\text{Red}) \quad Diff(\text{Blue},\text{Green}) \end{aligned}$$

- Every finite domain CSP can be expressed as a single definite clause + ground facts
- *Colourable* is inferred iff the CSP has a solution
- CSPs include 3SAT as a special case, hence matching is NP-hard

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,  $\theta$ ) 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  $\theta'$  in FOL-BC-AND(KB, lhs, UNIFY(rhs, goal,  $\theta$ )) do  
    yield  $\theta'$ 
```

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

Fetch rules that **might** unify

A function that returns multiple times, each time giving one possible result

$$\text{SUBST}(\text{COMPOSE}(\theta_1, \theta_2), p) = \text{SUBST}(\theta_2, \text{SUBST}(\theta_1, p))$$

Backward chaining example

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

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

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

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

$Criminal(West)$

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

$American(West)$ and $Enemy(Nono,America)$

Backward chaining example

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

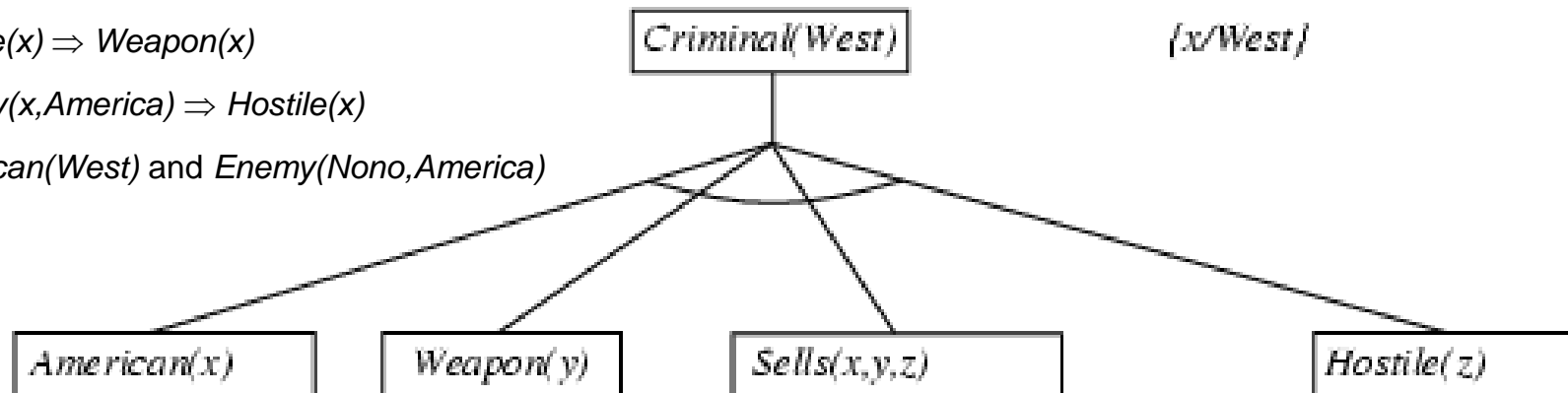
$Owns(Nono,M_1)$ and $Missile(M_1)$

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

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

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

$American(West)$ and $Enemy(Nono,America)$



Backward chaining example

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

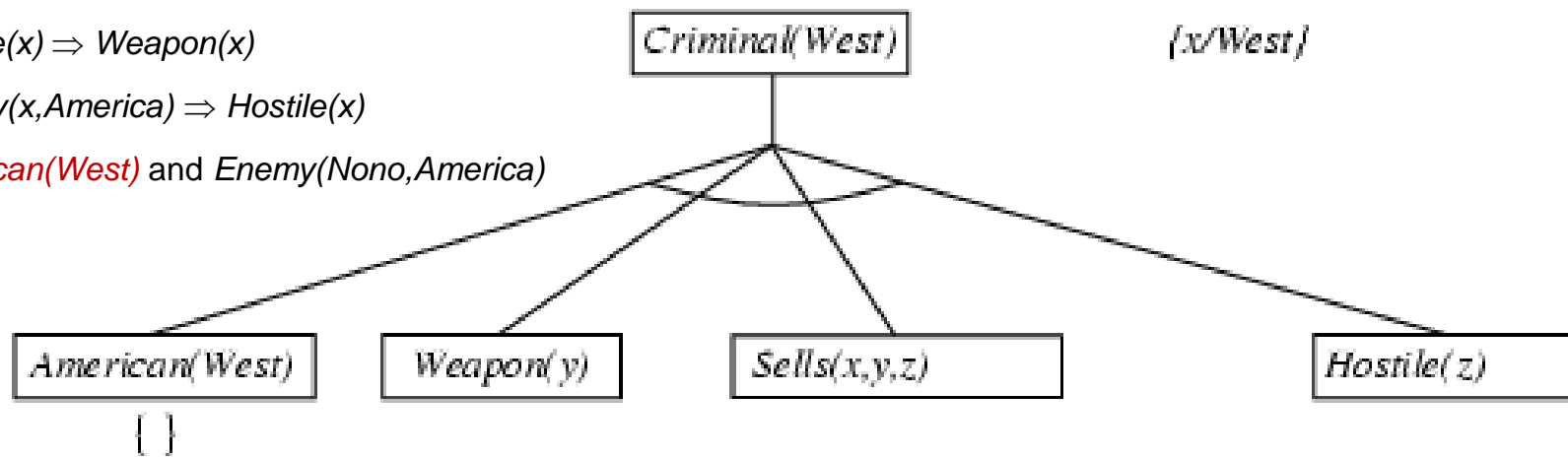
$Owens(Nono,M_1)$ and $Missile(M_1)$

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

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

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

$American(West)$ and $Enemy(Nono,America)$



Backward chaining example

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

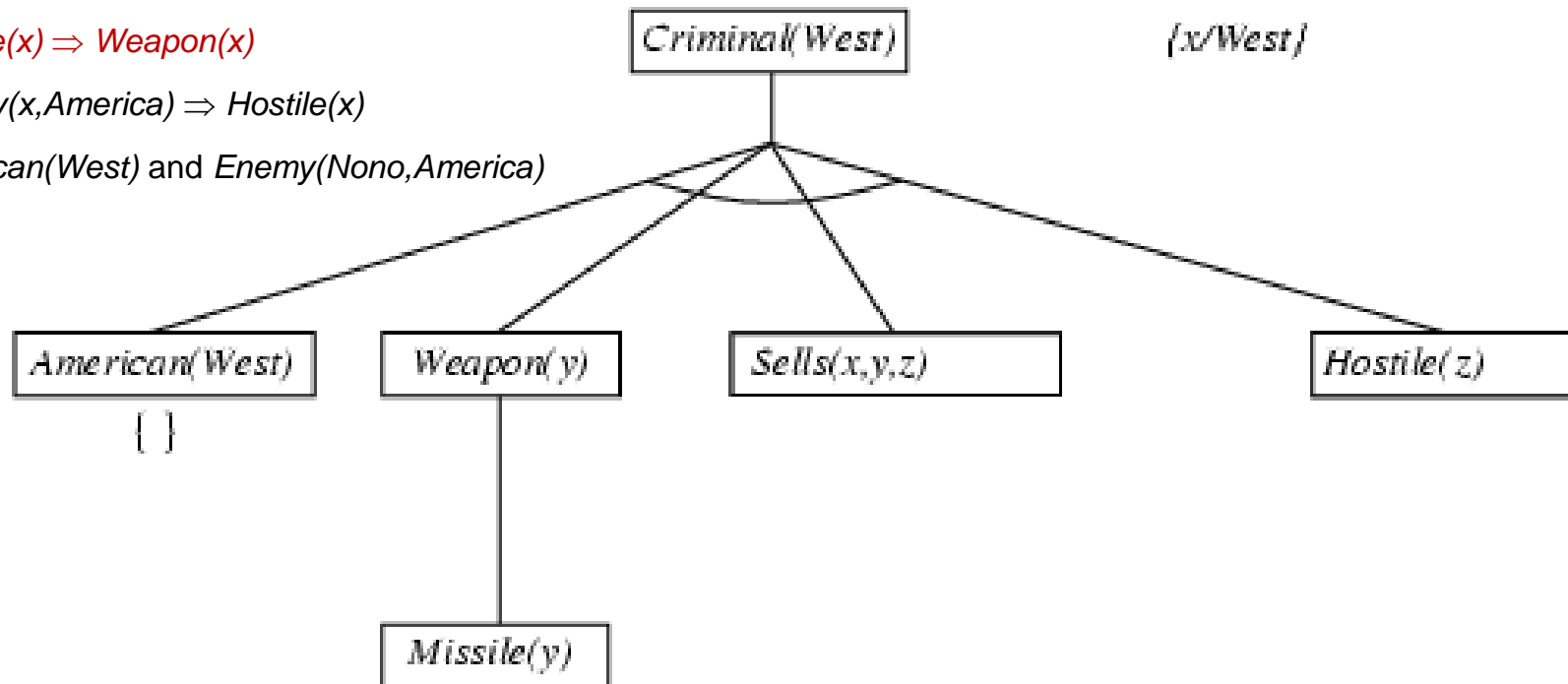
$Owns(Nono,M_1)$ and $Missile(M_1)$

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

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

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

$American(West)$ and $Enemy(Nono,America)$



Backward chaining example

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

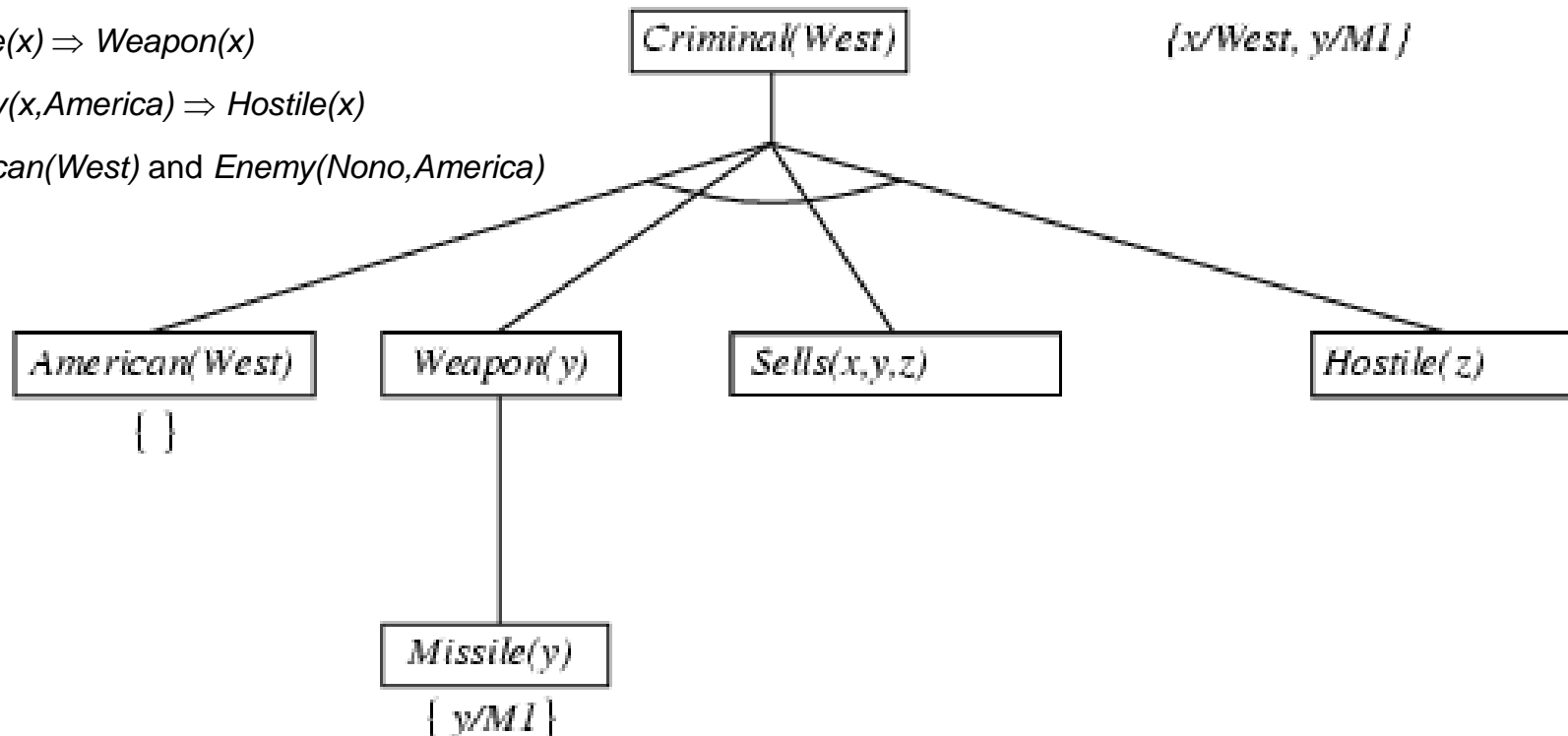
$Owns(Nono,M_1)$ and *Missile(M₁)*

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

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

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

$American(West)$ and $Enemy(Nono,America)$



Backward chaining example

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

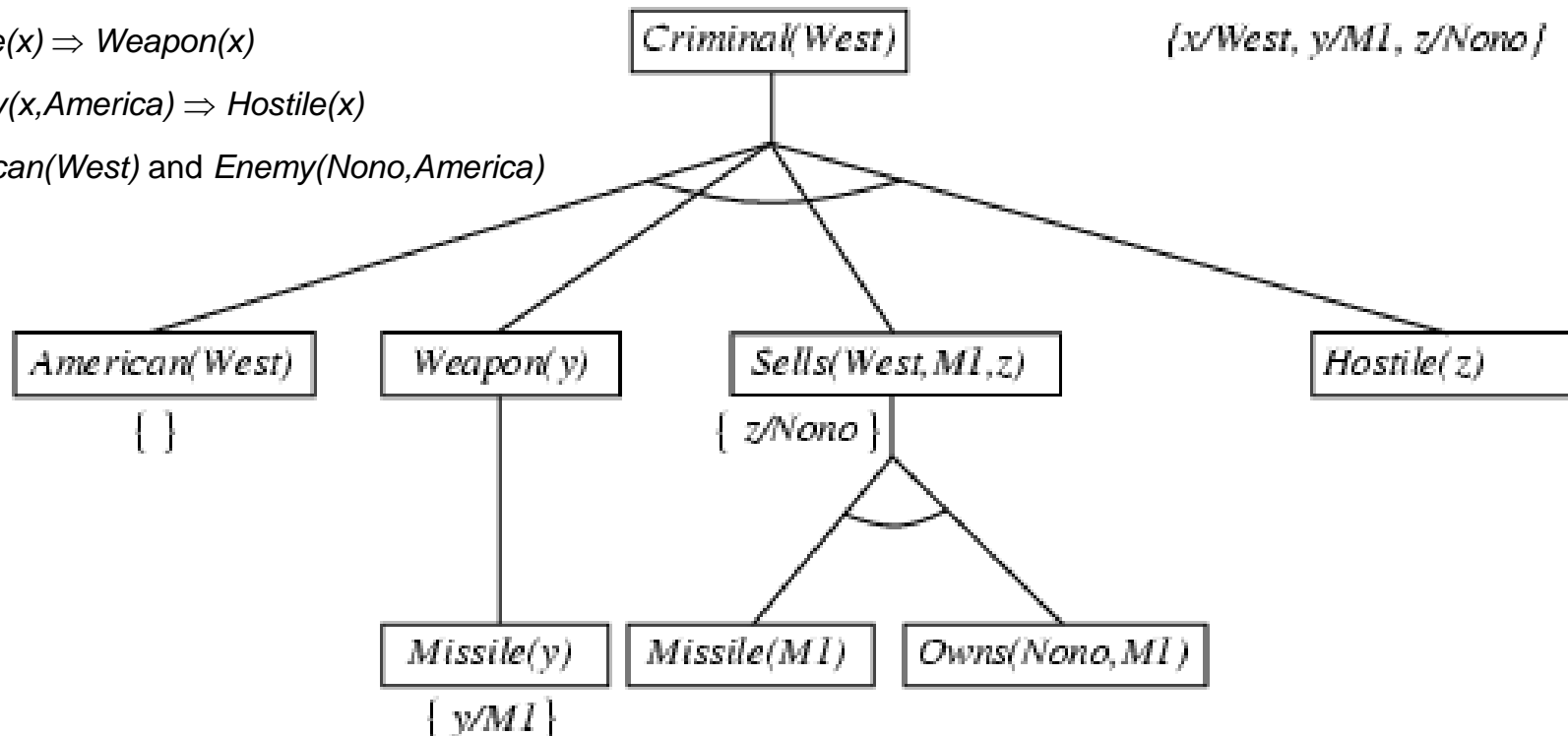
$Owens(Nono,M_1)$ and $Missile(M_1)$

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

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

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

$American(West)$ and $Enemy(Nono,America)$



Backward chaining example

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

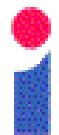
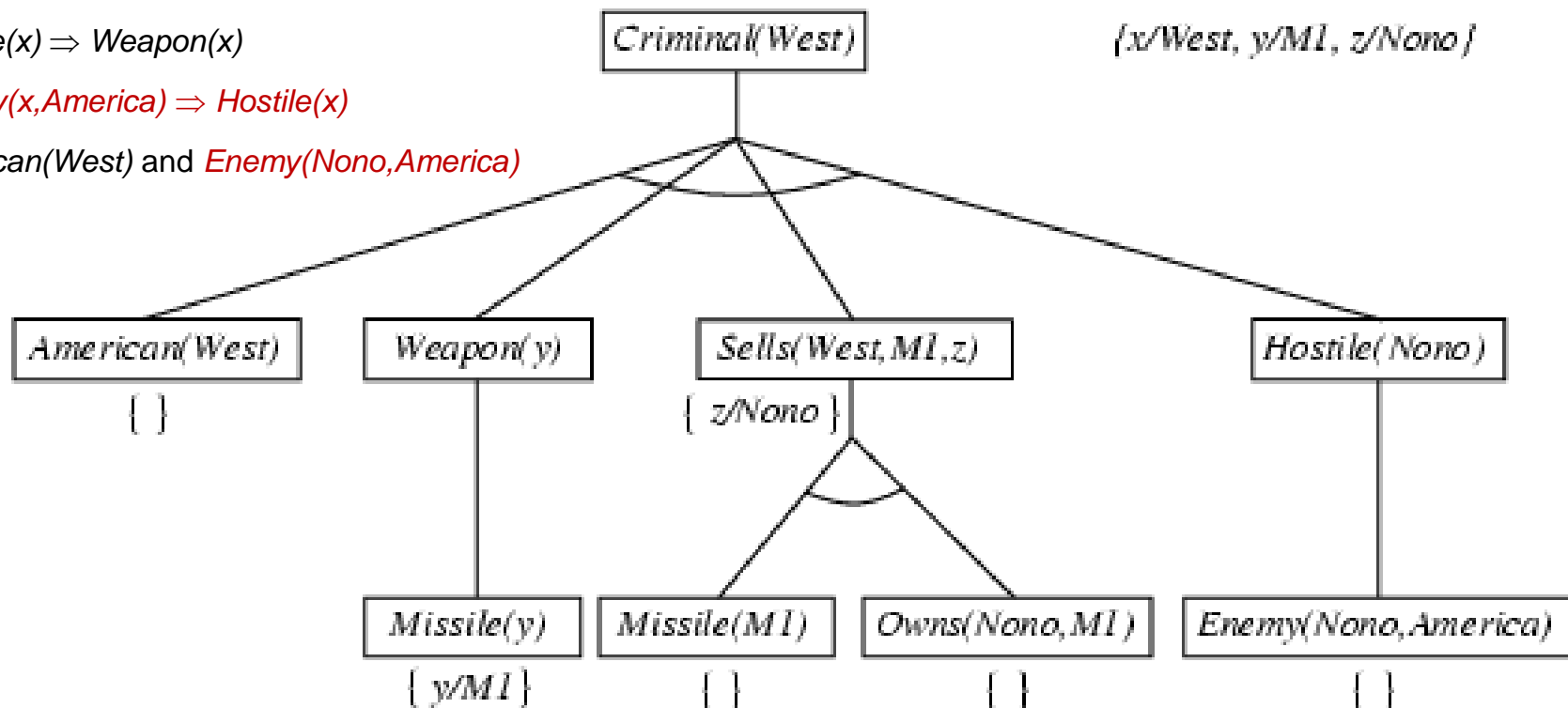
$Owns(Nono,M_1)$ and $Missile(M_1)$

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

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

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

$American(West)$ and $Enemy(Nono,America)$



Properties of backward chaining

- Depth-first recursive proof search: space is linear in size of proof
- Incomplete due to infinite loops
 - partial fix by checking current goal against every goal on stack
- Inefficient due to repeated subgoals (both success and failure)
 - fix using caching of previous results (extra space)
- Widely used for **logic programming**

Ground Binary Resolution

$$\frac{C \vee P \quad D \vee \neg P}{C \vee D}$$

Soundness:

$C \vee P$ iff $\neg C \Rightarrow P$

$D \vee \neg P$ iff $P \Rightarrow D$

– Therefore, $\neg C \Rightarrow D$

– Which is equivalent to $C \vee D$

Note: if both C and D are empty then resolution deduces the *empty clause*, i.e. **false**.

Non-Ground Binary Resolution

$$\frac{C \vee P \qquad D \vee \neg P'}{(C \vee D)\theta}$$

where θ is the mgu of P and P'

- The two clauses are assumed to be **standardized apart** so that they share **no** variables.
- **Soundness**: apply θ to premises then appeal to ground binary resolution.

$$\frac{C\theta \vee P\theta \qquad D\theta \vee \neg P\theta}{C\theta \vee D\theta}$$

Example

$\neg Rich(x) \vee Unhappy(x)$

$Rich(Ken)$

$Unhappy(Ken)$

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

Factoring

$$\frac{C \vee P_1 \vee \dots \vee P_m}{(C \vee P_1)\theta}$$

- where θ is the mgu of the P_i
- **Soundness**: by universal instantiation and deletion of duplicates.

Full Resolution

$$\frac{C \vee P_1 \vee \dots \vee P_m \quad D \vee \neg P_1' \vee \dots \vee \neg P_n'}{(C \vee D)\theta}$$

where θ is mgu of all P_i and P_i'

- **Soundness**: by combination of factoring and binary resolution.
- To prove α : apply resolution steps to $\text{CNF}(KB \wedge \neg\alpha)$;
 - **complete** for FOL, if **full resolution** or **binary resolution + factoring** is used

Conversion to CNF

Example:

Everyone who loves all animals is loved by someone:

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

1. Eliminate all biconditionals and implications

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

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

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

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

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

Conversion to CNF contd.

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

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

4. Skolemize: a more general form of **existential instantiation**

Each existential variable is replaced by a **Skolem function** of the **enclosing** universally quantified variables:

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

← No enclosing universal quantifier? Just replace with Skolem constant i.e. a function with no argument

5. Drop universal quantifiers:

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

6. Distribute \vee over \wedge :

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

'West' Clauses

$\neg \text{American}(x) \vee \neg \text{Weapon}(y) \vee \neg \text{Sells}(x,y,z) \vee$
 $\neg \text{Hostile}(z) \vee \text{Criminal}(x)$

$\text{Owns}(\text{Nono}, M_1)$ and $\text{Missile}(M_1)$

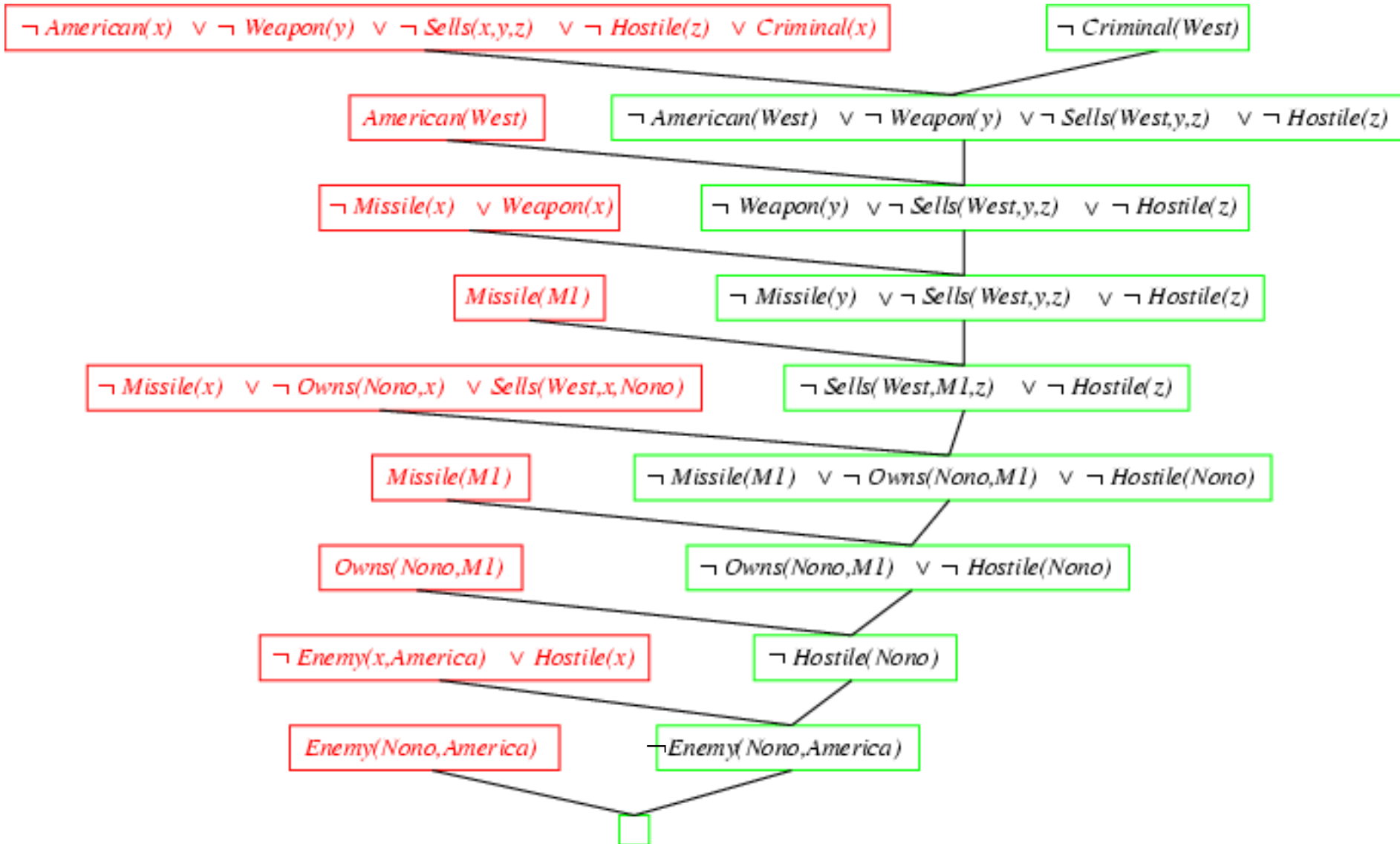
$\neg \text{Missile}(x) \vee \neg \text{Owns}(\text{Nono}, x) \vee \text{Sells}(\text{West}, x, \text{Nono})$

$\neg \text{Missile}(x) \vee \text{Weapon}(x)$

$\neg \text{Enemy}(x, \text{America}) \vee \text{Hostile}(x)$

$\text{American}(\text{West})$ and $\text{Enemy}(\text{Nono}, \text{America})$

Resolution proof: definite clauses



Summary

- Forward chaining
- Backward chaining
- Resolution