INFERENCE IN FIRST-ORDER LOGIC

CHAPTER 9

Outline

- ♢ Reducing first-order inference to propositional inference
- ♢ Unification
- ♢ Generalized Modus Ponens
- ♢ Forward and backward chaining
- ♢ Logic programming
- ♢ Resolution

Universal instantiation (UI)

Every instantiation of a universally quantified sentence is entailed by it:

$$\frac{\forall\, v \;\; \alpha}{\textsc{Subst}(\{v/g\}, \alpha)}$$

for any variable v and ground term g

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

$$\begin{array}{l} King(John) \land Greedy(John) \;\Rightarrow\; Evil(John) \\ King(Richard) \land Greedy(Richard) \;\Rightarrow\; Evil(Richard) \\ King(Father(John)) \land Greedy(Father(John)) \;\Rightarrow\; Evil(Father(John)) \\ \quad\vdots \end{array}$$

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., $\exists\, x\ \ Crown(x) \land OnHead(x, John)$ yields

$$Crown(C_1) \land OnHead(C_1, John)$$

provided C_1 is a new constant symbol, called a Skolem constant

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

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

The new KB is propositionalized: proposition symbols are

$King(John),\; Greedy(John),\; Evil(John), King(Richard)$ etc.

Reduction contd.

Idea: propositionalize KB and query, apply resolution, return 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 a FOL KB,
it is entailed by a **finite** subset of the propositional KB

Idea: For $n = 0$ to ∞ do
create a propositional KB by instantiating with depth-n terms
see if α is entailed by this KB

Problem: works if α is entailed, loops if α is not entailed

Theorem: Turing (1936), Church (1936), entailment in FOL is semidecidable

Problems with propositionalization

Propositionalization seems to generate lots of irrelevant sentences.
E.g., from

$$\begin{array}{l} \forall\, x \;\; King(x) \land Greedy(x) \;\Rightarrow\; Evil(x) \\ King(John) \\ \forall\, y \;\; Greedy(y) \\ Brother(Richard, John) \end{array}$$

it seems obvious that $Evil(John)$, but propositionalization produces lots of facts such as $Greedy(Richard)$ that are irrelevant

With p k-ary predicates and n constants, there are $p \cdot n^k$ instantiations

With function symbols, it gets much much worse!

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

$\textsc{Unify}(\alpha, \beta) = \theta$ if $\alpha\theta = \beta\theta$

p	q	θ
$Knows(John, x)$	$Knows(John, Jane)$	
$Knows(John, x)$	$Knows(y, OJ)$	
$Knows(John, x)$	$Knows(y, Mother(y))$	
$Knows(John, x)$	$Knows(x, OJ)$	

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

$\textsc{Unify}(\alpha, \beta) = \theta$ if $\alpha\theta = \beta\theta$

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

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

$\textsc{Unify}(\alpha, \beta) = \theta$ if $\alpha\theta = \beta\theta$

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

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

$\textsc{Unify}(\alpha, \beta) = \theta$ if $\alpha\theta = \beta\theta$

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

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

$\textsc{Unify}(\alpha, \beta) = \theta$ if $\alpha\theta = \beta\theta$

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

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

Generalized Modus Ponens (GMP)

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

$p_1{}'$ 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 $Evil(x)$
$q\theta$ is $Evil(John)$	

GMP used with KB of definite clauses (**exactly** one positive literal)
All variables assumed universally quantified

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

Example knowledge base contd.

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

Example knowledge base contd.

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

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

Nono . . . has some missiles

Example knowledge base contd.

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

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

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

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

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

Example knowledge base contd.

. . . 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

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

Missiles are weapons:

Example knowledge base contd.

. . . 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

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

Missiles are weapons:

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

An enemy of America counts as "hostile":

Example knowledge base contd.

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

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

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

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

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

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

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 algorithm

```
function FOL-FC-Ask($KB, \alpha$) returns a substitution or false
    repeat until new is empty
        new ← { }
        for each sentence r in KB do
            ($p_1 \wedge \ldots \wedge p_n \Rightarrow q$) ← Standardize-Apart(r)
            for each $\theta$ such that $(p_1 \wedge \ldots \wedge p_n)\theta = (p'_1 \wedge \ldots \wedge p'_n)\theta$
                            for some $p'_1, \ldots, p'_n$ in KB
                $q'$ ← Subst($\theta$, q)
                if $q'$ is not a renaming of a sentence already in KB or new then do
                    add $q'$ to new
                    $\phi$ ← Unify($q'$, $\alpha$)
                    if $\phi$ is not fail then return $\phi$
        add new to KB
    return false
```

Forward chaining proof

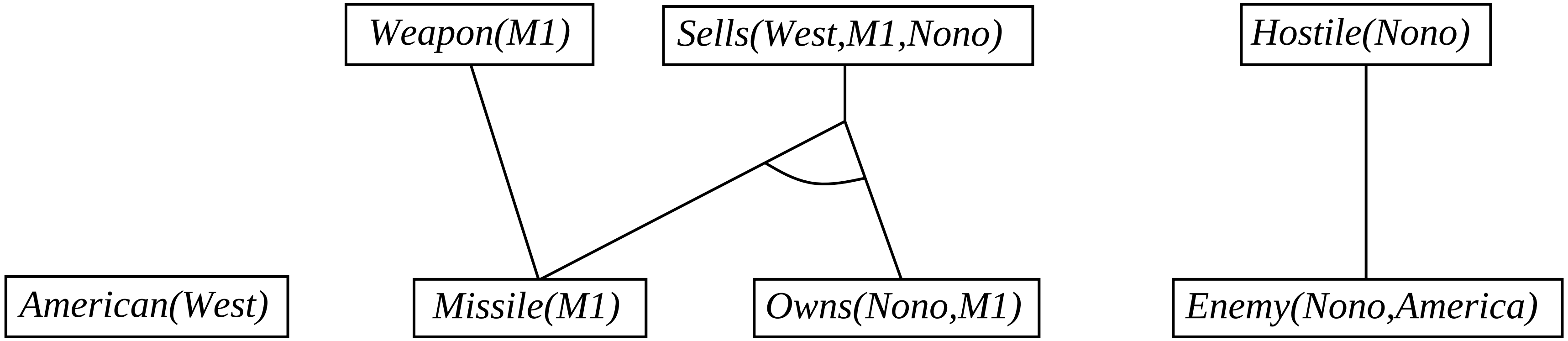
American(West)

Missile(M1)

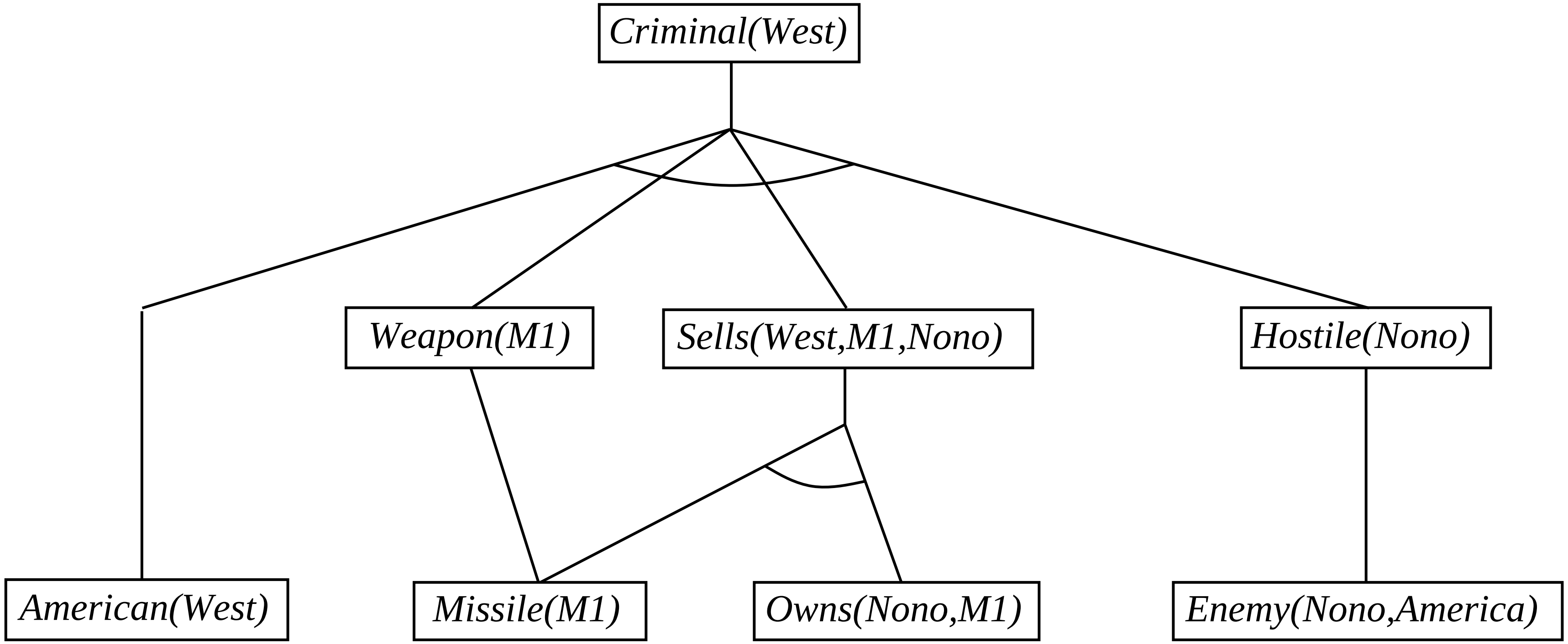
Owns(Nono,M1)

Enemy(Nono,America)

Forward chaining proof



Forward chaining proof



Criminal(West)

Weapon(M1)

Sells(West,M1,Nono)

Hostile(Nono)

American(West)

Missile(M1)

Owns(Nono,M1)

Enemy(Nono,America)

Properties of forward chaining

Sound and complete for first-order definite clauses

Datalog = first-order definite clauses + **no functions** (e.g., crime KB)
FC terminates for Datalog in poly iterations: at most $p \cdot n^k$ literals

May not terminate in general if α is not entailed

This is unavoidable: entailment with definite clauses is semidecidable

Backward chaining algorithm

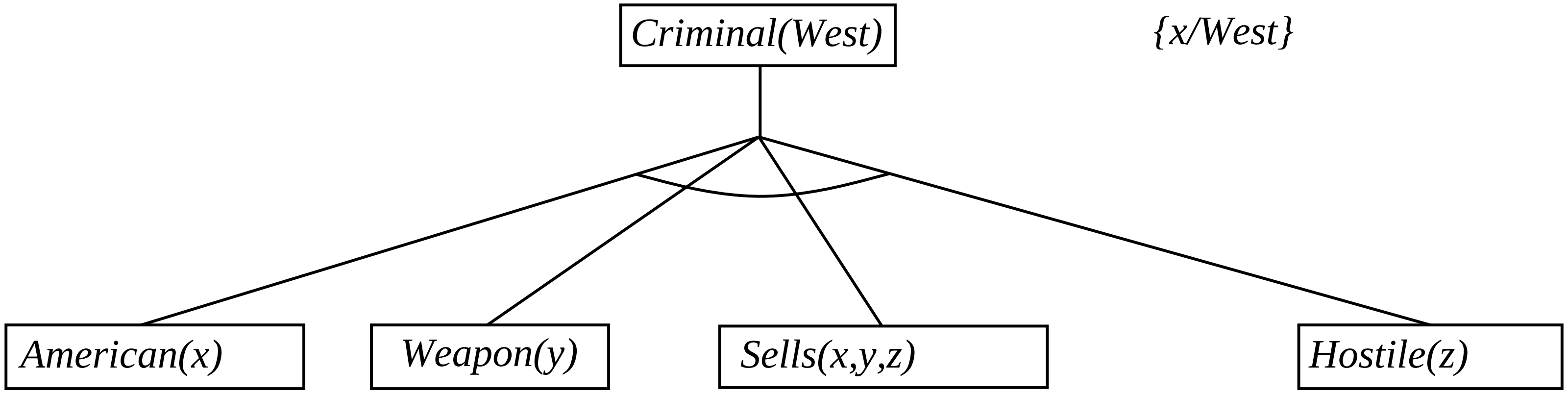
```
function FOL-BC-Ask($KB, goals, \theta$) returns a set of substitutions
    inputs: $KB$, a knowledge base
            $goals$, a list of conjuncts forming a query
            $\theta$, the current substitution, initially the empty substitution $\{\,\}$
    local variables: $answers$, a set of substitutions,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

    if $goals$ is empty then return $\{\theta\}$
    $q' \leftarrow$ Subst($\theta$, 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
        $new\_goals \leftarrow [p_1, \ldots, p_n |$Rest($goals$)$]$
        $answers \leftarrow$ FOL-BC-Ask($KB$, $new\_goals$, Compose($\theta', \theta$)) $\cup$ $answers$
    return $answers$
```

Backward chaining example

Criminal(West)

Backward chaining example



Criminal(West)

{x/West}

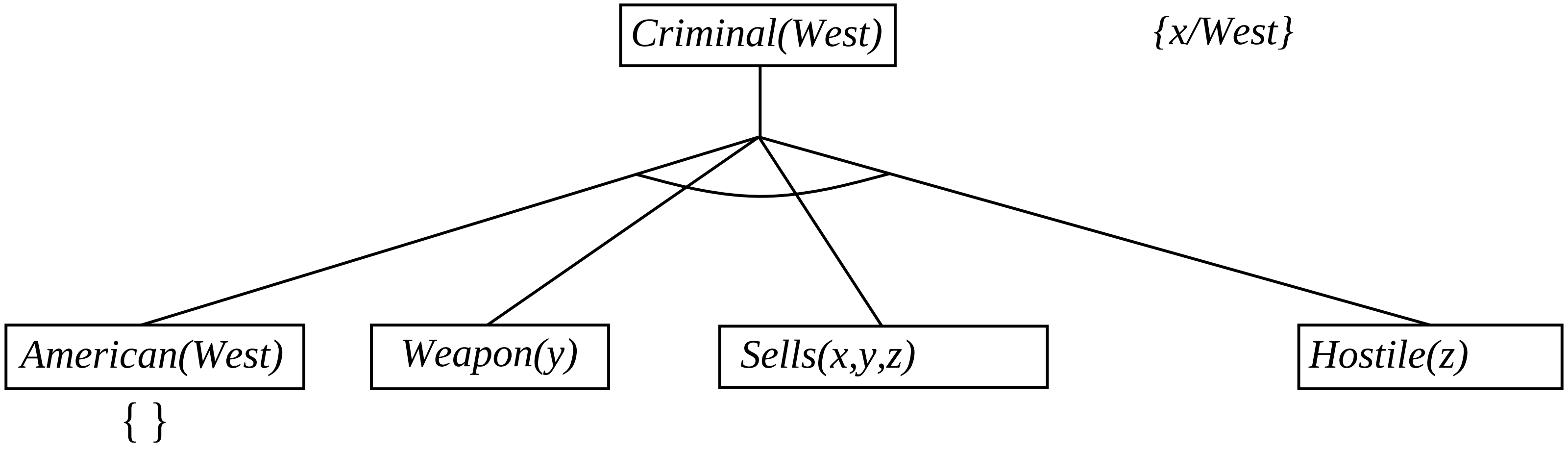
American(x)

Weapon(y)

Sells(x,y,z)

Hostile(z)

Backward chaining example



Criminal(West)

{x/West}

American(West)

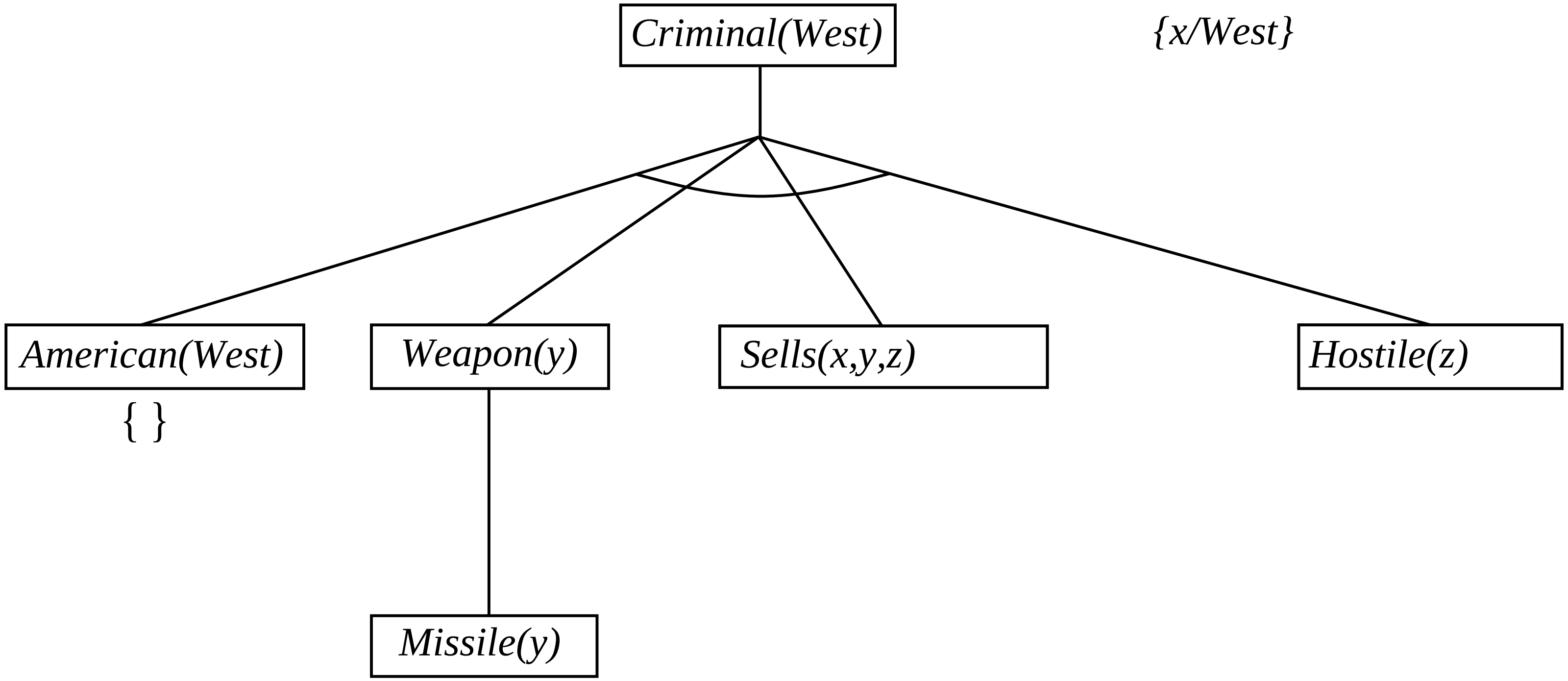
{ }

Weapon(y)

Sells(x,y,z)

Hostile(z)

Backward chaining example



Criminal(West)

{x/West}

American(West)

{ }

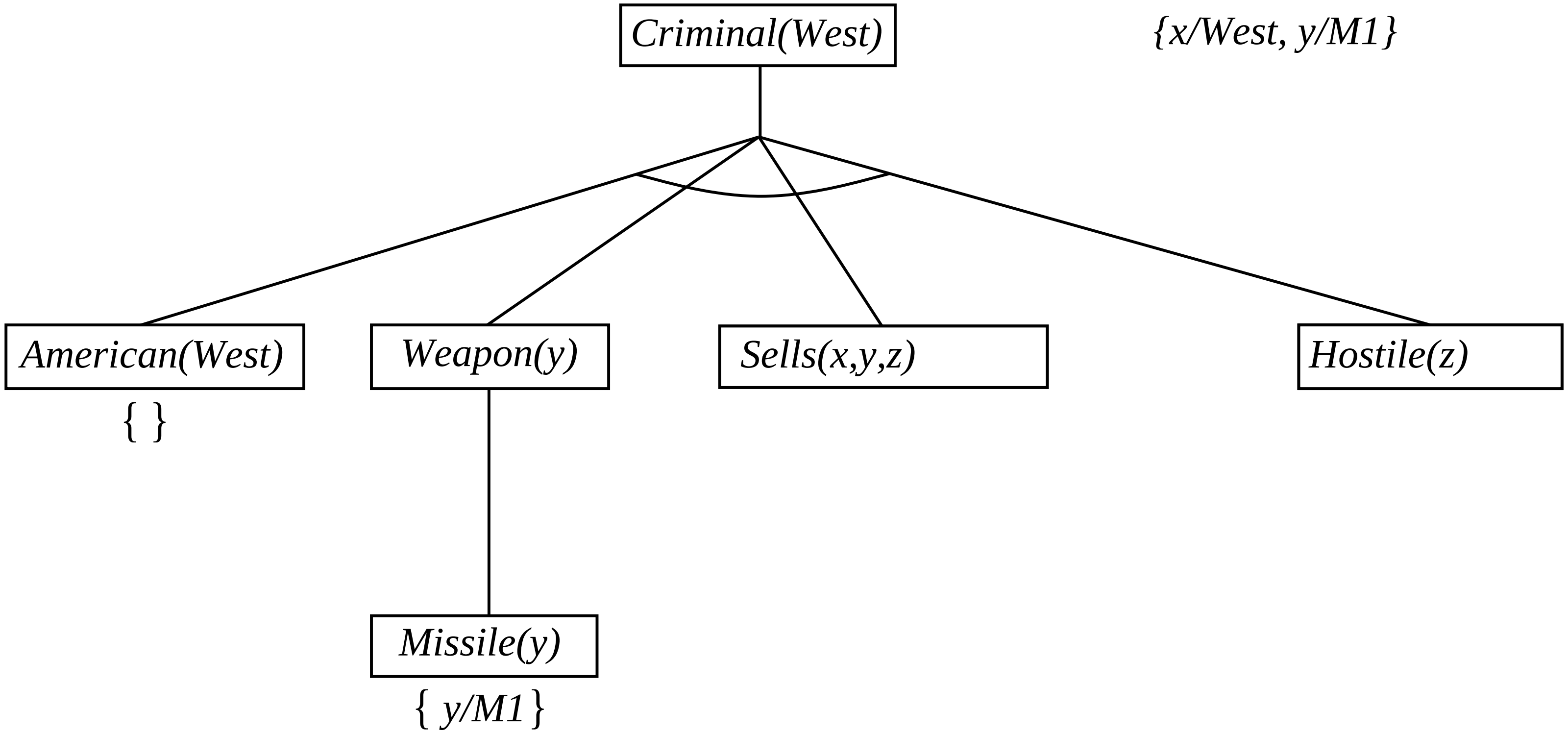
Weapon(y)

Sells(x,y,z)

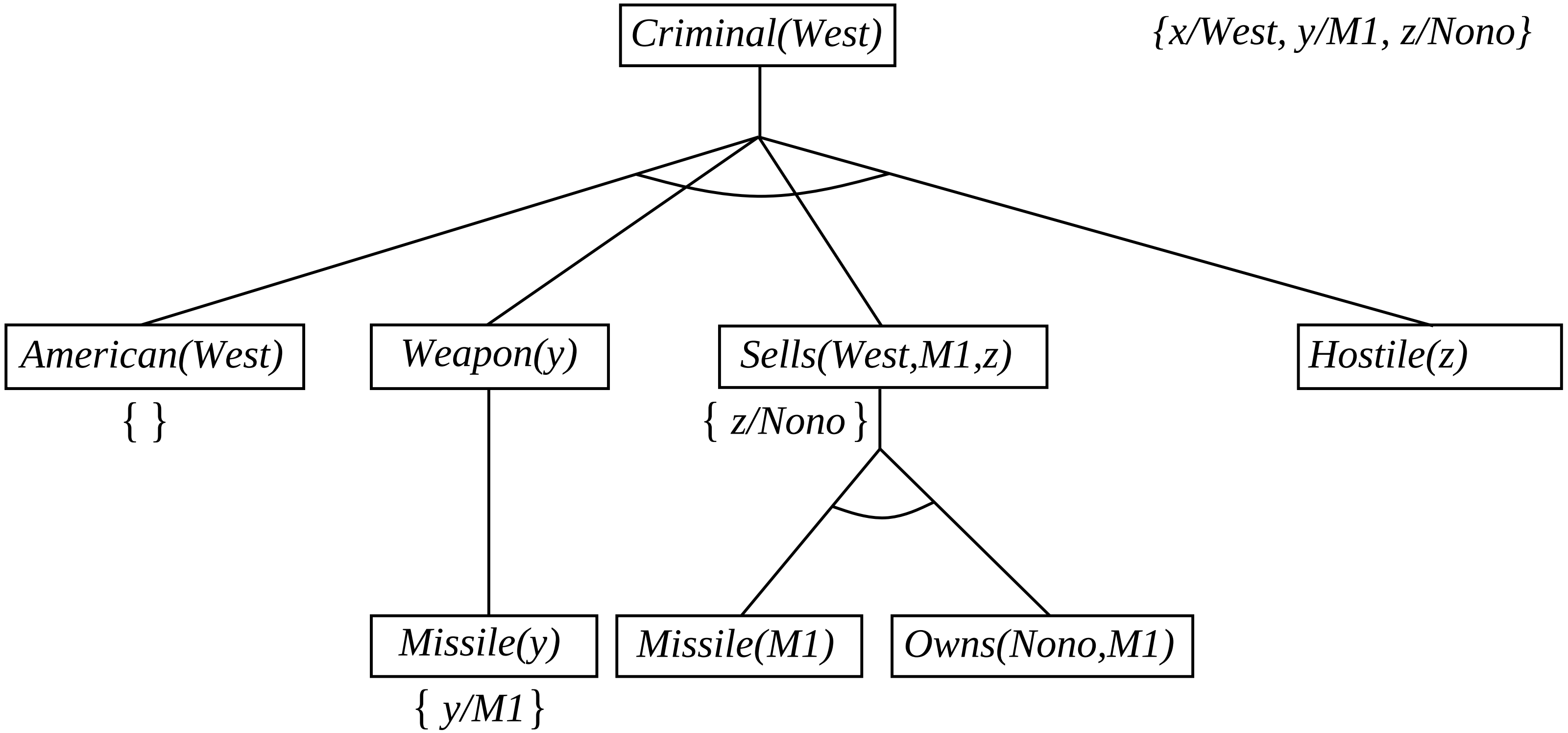
Hostile(z)

Missile(y)

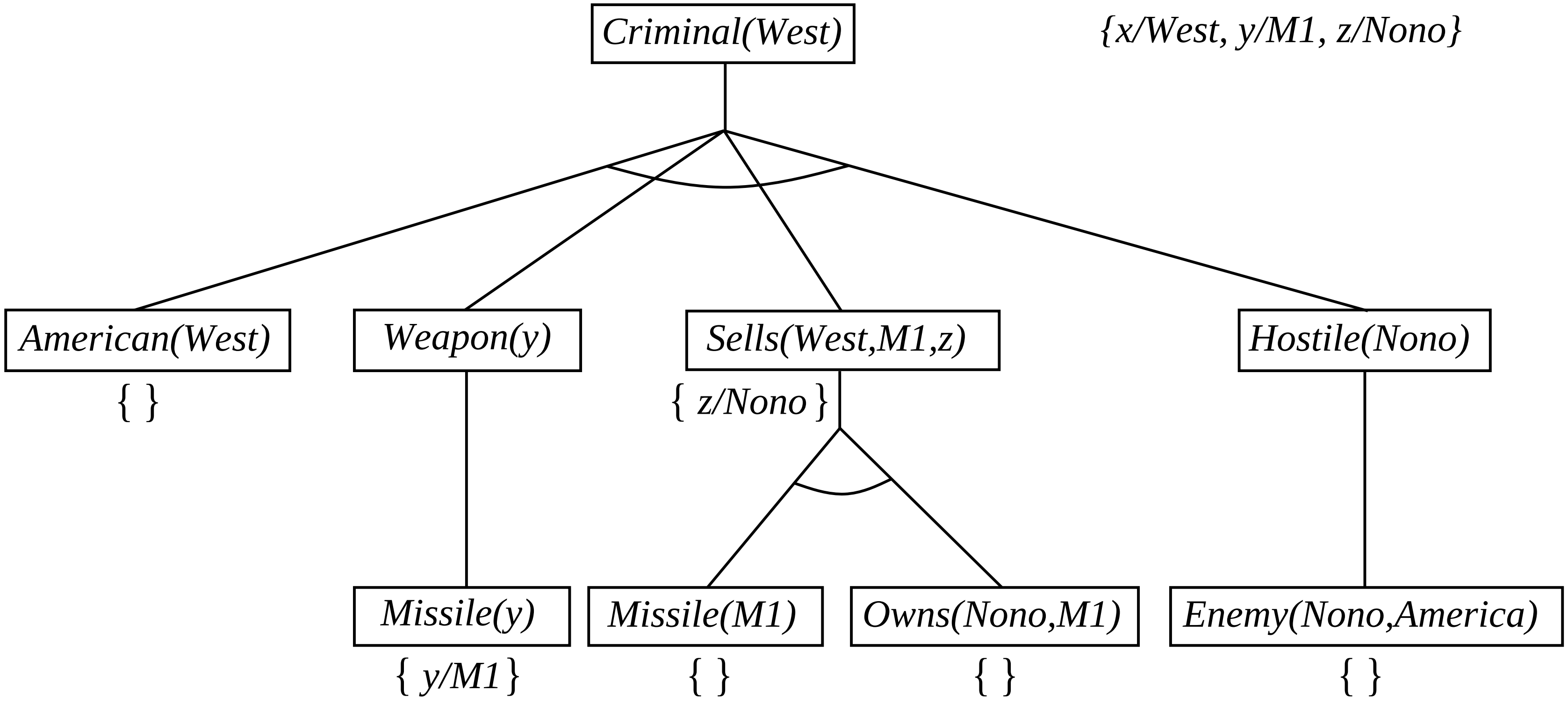
Backward chaining example



Backward chaining example



Backward chaining example


Criminal(West)

{x/West, y/M1, z/Nono}

American(West)

{ }

Weapon(y)

Sells(West,M1,z)

{ z/Nono }

Hostile(Nono)

Missile(y)

{ y/M1}

Missile(M1)

{ }

Owns(Nono,M1)

{ }

Enemy(Nono,America)

{ }

Properties of backward chaining

Depth-first recursive proof search: space is linear in size of proof

Incomplete due to infinite loops
$\Rightarrow$ fix by checking current goal against every goal on stack

Inefficient due to repeated subgoals (both success and failure)
$\Rightarrow$ fix using caching of previous results (extra space!)

Widely used for logic programming

Prolog systems

Basis: backward chaining with Horn clauses + bells & whistles

Program = set of clauses = `head :- literal`$_1$`, ... literal`$_n$`.`

```
criminal(X) :- american(X), weapon(Y), sells(X,Y,Z), hostile(Z).
```

Built-in predicates for arithmetic etc., e.g., `X is Y*Z+3`
Closed-world assumption ("negation as failure")
   e.g., given `alive(X) :- not dead(X).`
   `alive(joe)` succeeds if `dead(joe)` fails

Prolog examples

Depth-first search from a start state `X`:

```
dfs(X) :- goal(X).
dfs(X) :- successor(X,S),dfs(S).
```

No need to loop over `S`: `successor` succeeds for each

Appending two lists to produce a third:

```
append([],Y,Y).
append([X|L],Y,[X|Z]) :- append(L,Y,Z).

query:   append(A,B,[1,2]) ?
answers: A=[]    B=[1,2]
         A=[1]   B=[2]
         A=[1,2] B=[]
```

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 contd.

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:

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

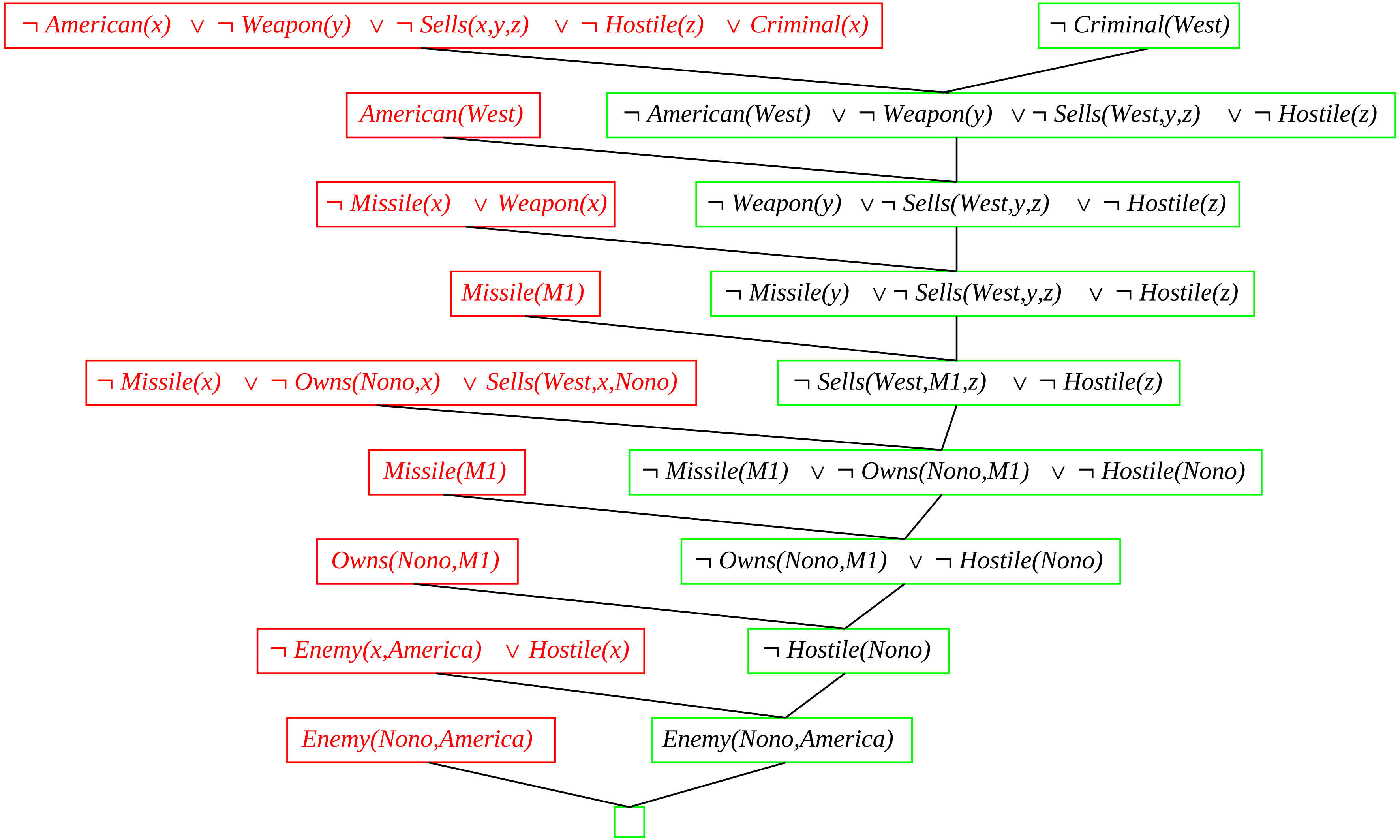
5. Drop universal quantifiers:

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

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



¬ American(x) ∨ ¬ Weapon(y) ∨ ¬ Sells(x,y,z) ∨ ¬ Hostile(z) ∨ Criminal(x)

¬ Criminal(West)

American(West)

¬ American(West) ∨ ¬ Weapon(y) ∨ ¬ Sells(West,y,z) ∨ ¬ Hostile(z)

¬ Missile(x) ∨ Weapon(x)

¬ Weapon(y) ∨ ¬ Sells(West,y,z) ∨ ¬ Hostile(z)

Missile(M1)

¬ Missile(y) ∨ ¬ Sells(West,y,z) ∨ ¬ Hostile(z)

¬ Missile(x) ∨ ¬ Owns(Nono,x) ∨ Sells(West,x,Nono)

¬ Sells(West,M1,z) ∨ ¬ Hostile(z)

Missile(M1)

¬ Missile(M1) ∨ ¬ Owns(Nono,M1) ∨ ¬ Hostile(Nono)

Owns(Nono,M1)

¬ Owns(Nono,M1) ∨ ¬ Hostile(Nono)

¬ Enemy(x,America) ∨ Hostile(x)

¬ Hostile(Nono)

Enemy(Nono,America)

Enemy(Nono,America)