Cooperating Intelligent Systems

Inference in first-order logic

Chapter 9, AIMA

Reduce to propositional logic

- Reduce the first order logic sentences to propositional (boolean) logic sentences
- It is then possible to use the propositional logic inference systems
  - model checking
  - resolution
  - …

Basically, we need a way of transforming sentences with quantifiers to sentences without quantifiers

FOL inference rules

All the propositional rules (Modus Ponens, And Elimination, And Introduction, etc.) plus:

Universal Instantiation (UI)

$$\frac{\cdot \;\; x\, w(x)}{w(a)}$$

Where the variable x is replaced by the ground term a everywhere in the sentence w.

Example:

∀x P(x,f(x),B) ⇒ P(A,f(A),B)

Existential Instantiation (EI)

$$\frac{{}^{\wedge} \;\; x\, w(x)}{w(a)}$$

Where the variable x is replaced by the ground term a everywhere in the sentence w.

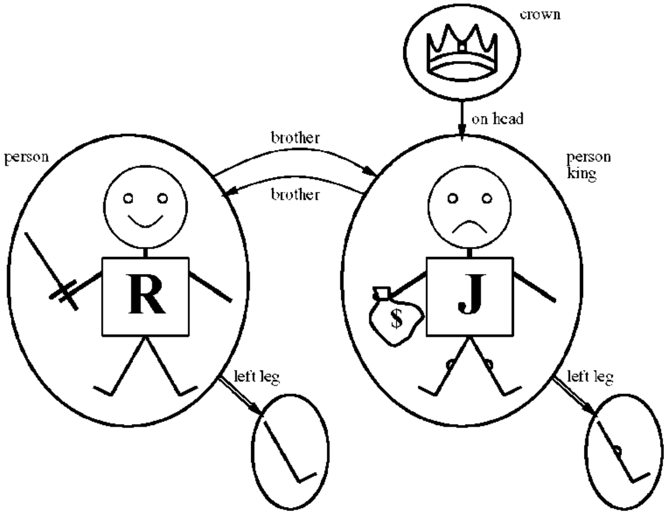
A must be a new symbol

Example:

∃x Q(x,g(x),B) ⇒ Q(A,g(A),B)

Ground term = a term without variables

Example: Kings...



UI: (Universal Instantiation)

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

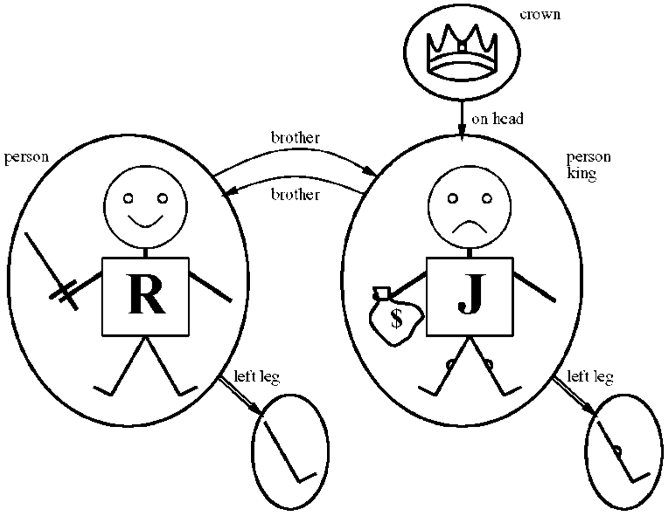
$King(John) \wedge Greedy(John) \Rightarrow Evil(John)$

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

$King(Crown) \wedge Greedy(Crown) \Rightarrow Evil(Crown)$

⋮

Example: Kings...



UI: (Universal Instantiation)

∀x (King(x) ∧ Greedy(x)) ⇒ Evil(x)

King(John) ∧ Greedy(John) ⇒ Evil(John)

King(Richard) ∧ Greedy(Richard) ⇒ Evil(Richard)

King(Crown) ∧ Greedy(Crown) ⇒ Evil(Crown)

⋮

EI: (Existential Instantiation)

∃x (Crown(x) ∧ OnHead(x,John))

Crown(C) ∧ OnHead(C,John)

C is called a *Skolem constant*

Making up names is called *skolemization*

Propositionalization

Keep applying Universal Instantiation (UI) and Existential Instantiation (EI) – eventually, every FOL sentence in the KB will be made into a propositional sentence

- propositional logic tools can be used to prove theorems

Problem with function constants: Father(A), Father(Father(A)), Father(Father(Father(A))), etc...

- we can end up with infinite number of sentences...
- how can we prove things in finite time?

Theorem [Gödel, Herbrand]: We can find every entailed sentence, but the search is not guaranteed to stop for nonentailed sentences – FOL is *semi-decidable*

”Solution”: stop after a certain time and assume the sentence is false

Still, Propositionalization is inefficient
generalized (lifted) inference rules are better

Notation: Substitution

Subst(Ø Ò) = Apply the substitution Ø to the sentence Ò.

Example:
Ø = {x/John} (replace all occurences of ”x” with ”John”)

Ò = (King(x) ∧ Greedy(x)) ⇒ Evil(x)

Subst(Ø Ò) = (King(John) ∧ Greedy(John)) ⇒ Evil(John)

General form: Ø = $\{x_0/g_0,x_1/g_1,\ldots,x_n/g_n\}$

- where x are variables and g are terms

Generalized (lifted) Modus Ponens

If there exists a substitution Ø such that
for every pair of atomic sentences p_i and q_i
Subst(Ø,p_i) = Subst(Ø,q_i), then:

$$\frac{p_1, p_2, \ldots, p_n, (q_1 \text{ ť } q_2 \text{ ť } \cdots \text{ ť } q_n \text{ Ŭ } r)}{\text{Subst}(Ø, r)}$$

KB

p_1 = King(John)

p_2 = Greedy(John)

∀x (King(x) ∧ Greedy(x) ⇒ Evil(x))

We have John who is King and is Greedy.
If someone is King and Greedy then he/she/it is also Evil.

Generalized (lifted) Modus Ponens

If there exists a substitution Ø such that

for every pair of atomic sentences p_i and q_i

Subst(Ø,p_i) = Subst(Ø,q_i), then:

$$\frac{p_1, p_2, \ldots, p_n, (q_1 \text{ ť } q_2 \text{ ť } \cdots \text{ ť } q_n \text{ Ŭ } r)}{\text{Subst}(\varnothing, r)}$$

KB

p_1 = King(John)

p_2 = Greedy(John)

Ø = {x/John}

q_1 = King(x)

q_2 = Greedy(x)

r = Evil(x)

∀x (King(x) ∧ Greedy(x) ⇒ Evil(x))

Subst(Ø,p_1) = Subst(Ø,q_1)

Generalized (lifted) Modus Ponens

If there exists a substitution Ø such that for every pair of atomic sentences p_i and q_i Subst(Ø,p_i) = Subst(Ø,q_i), then:

$$\frac{p_1, p_2, \ldots, p_n, (q_1 \text{ ť } q_2 \text{ ť } \cdots \text{ ť } q_n \text{ Ŭ } r)}{\text{Subst}(Ø, r)}$$

KB

p_1 = King(John)	q_1 = King(x)	∀x (King(x) ∧ Greedy(x) ⇒ Evil(x))
p_2 = Greedy(John)	q_2 = Greedy(x)	
Ø = {x/John}	r = Evil(x)	

Subst(Ø,p_2) = Subst(Ø,q_2)

Generalized (lifted) Modus Ponens

If there exists a substitution Ø such that

for every pair of atomic sentences p_i and q_i

Subst(Ø,p_i) = Subst(Ø,q_i), then:

$$\frac{p_1, p_2, \ldots, p_n, (q_1 \text{ ť } q_2 \text{ ť } \cdots \text{ ť } q_n \text{ Ŭ } r)}{\text{Subst}(\text{Ø}, r)}$$

KB

p_1 = King(John)	q_1 = King(x)	∀x (King(x) ∧ Greedy(x) ⇒ Evil(x))
p_2 = Greedy(John)	q_2 = Greedy(x)	
Ø = {x/John}	r = Evil(x)	King(John), Greedy(John)
Subst(Ø,r) = Evil(John)		⇒Evil(John)

Generalized (lifted) Modus Ponens

If there exists a substitution Ø such that for every pair of atomic sentences p_i and q_i Subst(Ø,p_i) = Subst(Ø,q_i), then:

$$\frac{p_1, p_2, \ldots, p_n, (q_1 \text{ ť } q_2 \text{ ť } \cdots \text{ ť } q_n \text{ Ŭ } r)}{\text{Subst}(Ø, r)}$$

KB

p_1 = King(John)	q_1 = King(x)	∀x (King(x) ∧ Greedy(x) ⇒ Evil(x))
p_2 = Greedy(John)	q_2 = Greedy(x)	
Ø = {x/John}	r = Evil(x)	King(John), Greedy(John)
Subst(Ø,r) = Evil(John)		⇒Evil(John)

Lifted inference rules make only the necessary substitutions

Forward chaining example

KB:

1. All cats like fish
2. Cats eat everything they like
3. Ziggy is a cat

Example from Tuomas Sandholm @ CMU

Forward chaining example

KB:

1. All cats like fish
2. Cats eat everything they like
3. Ziggy is a cat

$$\forall x\ Cat(x) \Rightarrow Likes(x, Fish)$$

$$\forall x \forall y\ Cat(x) \wedge Likes(x, y) \Rightarrow Eats(x, y)$$

$$Cat(Ziggy)$$

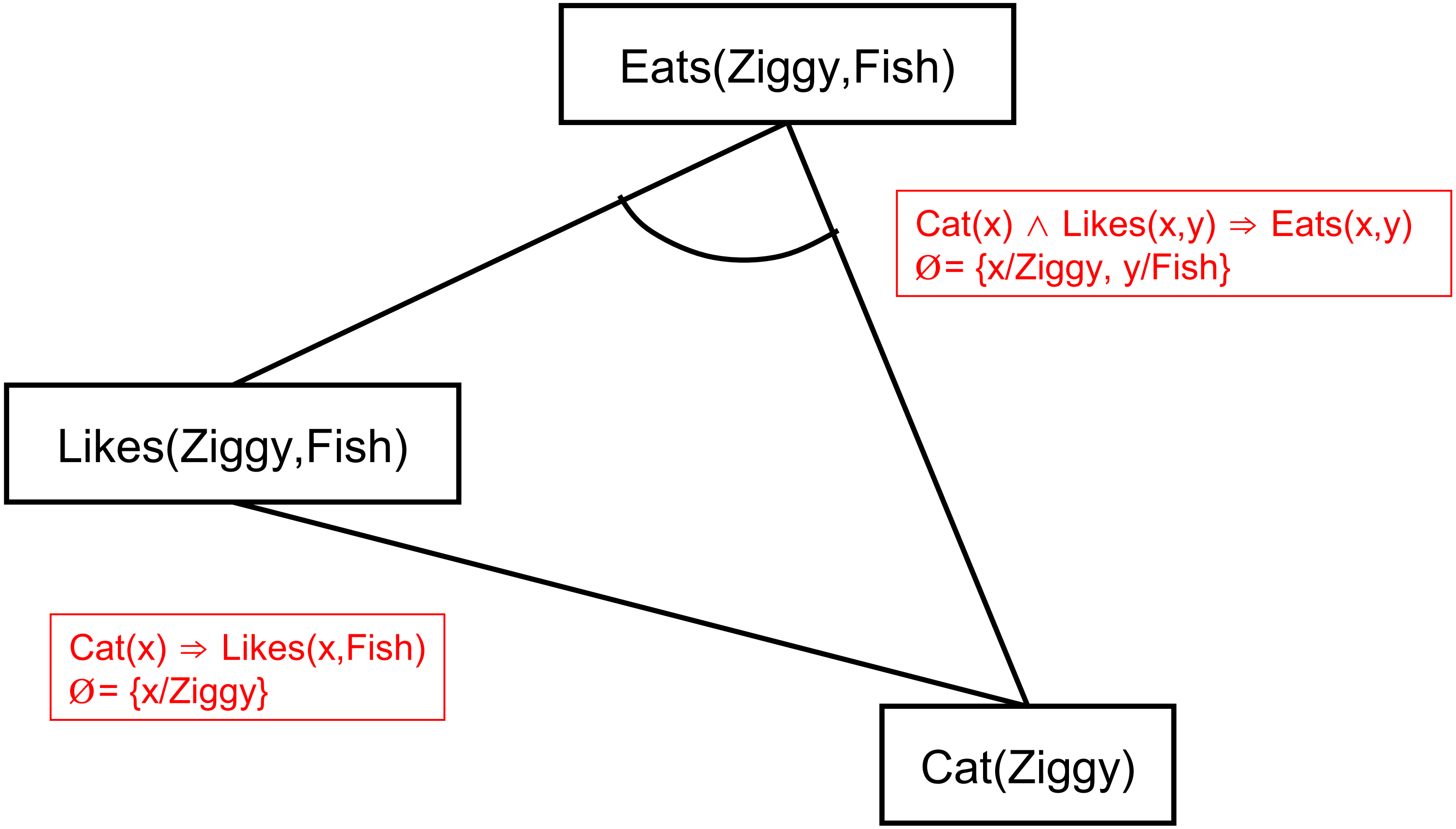
Example from Tuomas Sandholm @ CMU

$\cdot\ x\ \ Cat(x)$ Ů $\ Likes(x, Fish)$

$\cdot\ x\cdot\ \ y\ \ Cat(x)$ ť $\ Likes(x, y)$ Ů $\ Eats(x, y)$

$Cat(Ziggy)$

Ziggy the cat eats the fish!



Example: Arms dealer

KB in Horn Form

(1) $\forall x\ (\text{American}(x) \wedge \text{Weapon}(y) \wedge \text{Hostile}(z) \wedge \text{Sells}(x,y,z) \Rightarrow \text{Criminal}(x))$

(2) Owns(NoNo,M)

(3) Missile(M)

(4) $\forall x\ (\text{Missile}(x) \wedge \text{Owns}(\text{NoNo},x) \Rightarrow \text{Sells}(\text{West},x,\text{NoNo}))$

(5) $\forall x\ (\text{Missile}(x) \Rightarrow \text{Weapon}(x))$

(6) $\forall x\ (\text{Enemy}(x,\text{America}) \Rightarrow \text{Hostile}(x))$

(7) American(West)

(8) Enemy(NoNo,America)

Example: Arms dealer

KB in Horn Form

(1) $\forall x\ (\text{American}(x) \wedge \text{Weapon}(y) \wedge \text{Hostile}(z) \wedge \text{Sells}(x,y,z) \Rightarrow \text{Criminal}(x))$

(2) $\text{Owns}(\text{NoNo},M)$

(3) $\text{Missile}(M)$

(4) $\forall x\ (\text{Missile}(x) \wedge \text{Owns}(\text{NoNo},x) \Rightarrow \text{Sells}(\text{West},x,\text{NoNo}))$

(5) $\forall x\ (\text{Missile}(x) \Rightarrow \text{Weapon}(x))$

(6) $\forall x\ (\text{Enemy}(x,\text{America}) \Rightarrow \text{Hostile}(x))$

(7) $\text{American}(\text{West})$

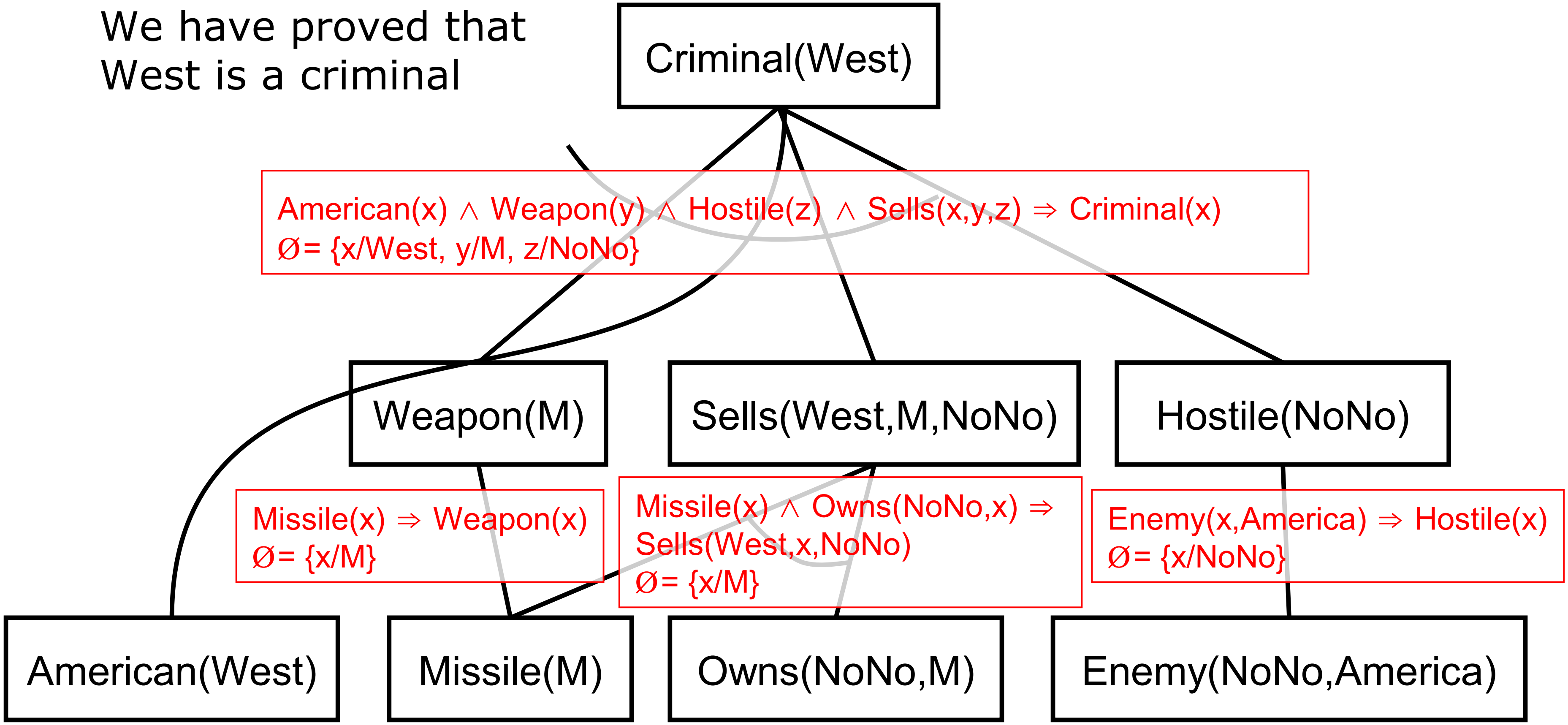
(8) $\text{Enemy}(\text{NoNo},\text{America})$

Facts

Forward chaining: Arms dealer

Forward chaining generates all inferences (also irrelevant ones)

We have proved that
West is a criminal



Criminal(West)

American(x) ∧ Weapon(y) ∧ Hostile(z) ∧ Sells(x,y,z) ⇒ Criminal(x)
Ø= {x/West, y/M, z/NoNo}

Weapon(M)

Sells(West,M,NoNo)

Hostile(NoNo)

Missile(x) ⇒ Weapon(x)
Ø= {x/M}

Missile(x) ∧ Owns(NoNo,x) ⇒ Sells(West,x,NoNo)
Ø= {x/M}

Enemy(x,America) ⇒ Hostile(x)
Ø= {x/NoNo}

American(West)

Missile(M)

Owns(NoNo,M)

Enemy(NoNo,America)

Example: Financial advisor

KB in Horn Form

1) SavingsAccount(Inadequate) ⇒ Investments(Bank)

2) SavingsAccount(Adequate) ∧ Income(Adequate) ⇒ Investments(Stocks)

3) SavingsAccount(Adequate) ∧ Income(Inadequate) ⇒ Investments(Mixed)

4) ∀x (AmountSaved(x) ∧ ∃y (Dependents(y) ∧ Greater(x,MinSavings(y))) ⇒ SavingsAccount(Adequate))

5) ∀x (AmountSaved(x) ∧ ∃y (Dependents(y) ∧ ¬Greater(x,MinSavings(y))) ⇒ SavingsAccount(Inadequate))

6) ∀x (Earnings(x,Steady) ∧ ∃y (Dependents(y) ∧ Greater(x,MinIncome(y))) ⇒ Income(Adequate))

7) ∀x (Earnings(x,Steady) ∧ ∃y (Dependents(y) ∧ ¬Greater(x,MinIncome(y))) ⇒ Income(Inadequate))

8) ∀x (Earnings(x,UnSteady) ⇒ Income(Inadequate))

9) AmountSaved($22000)

10) Earnings($25000,Steady)

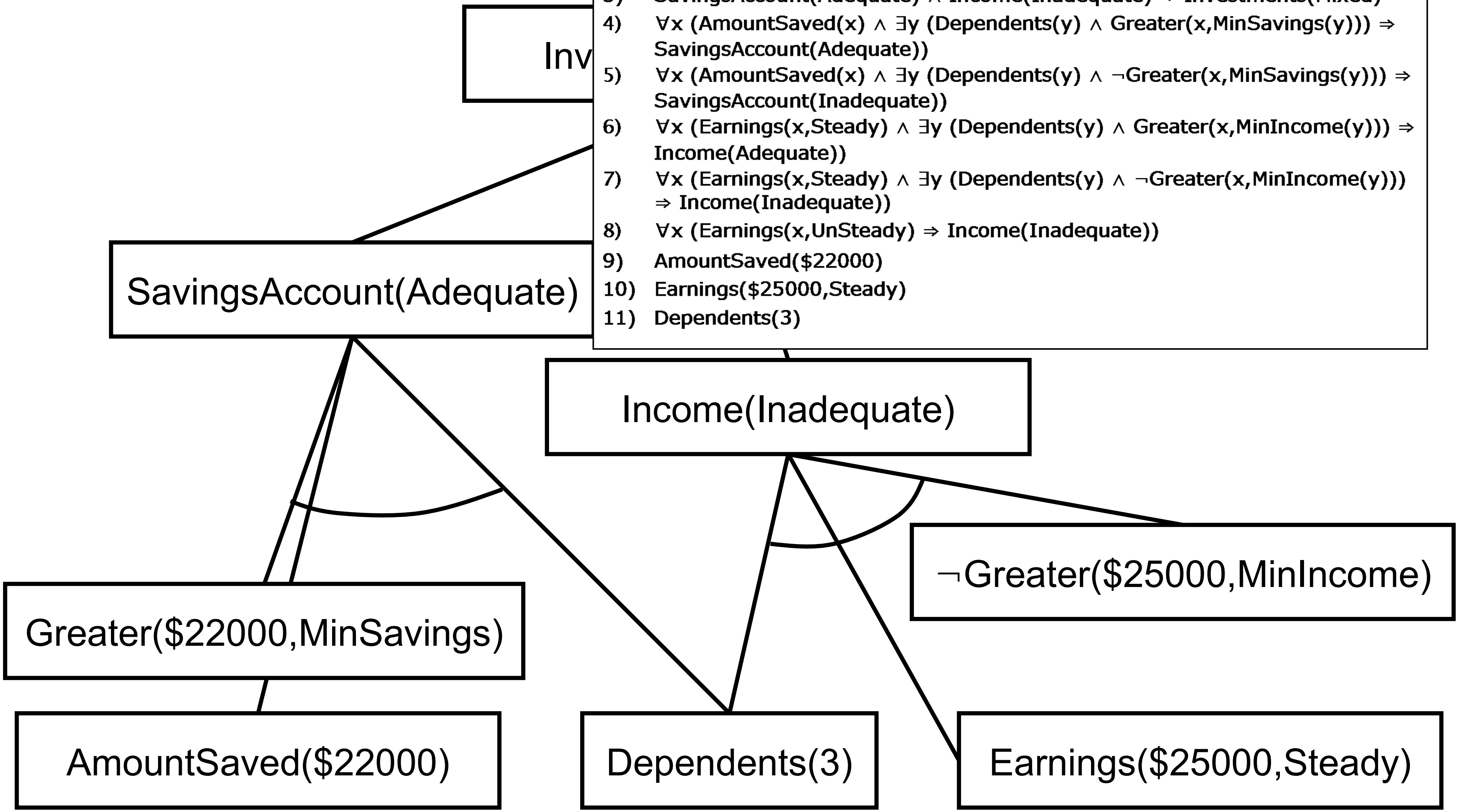
11) Dependents(3)

$\text{MinSavings}(x) \equiv \$5000 \cdot x$
$\text{MinIncome}(x) \equiv \$15000 + (\$4000 \cdot x)$

Example from G.F. Luger, ”Artificial Intelligence” 2002

FC financial advisor

1) SavingsAccount(Inadequate) ⇒ Investments(Bank)
2) SavingsAccount(Adequate) ∧ Income(Adequate) ⇒ Investments(Stocks)
3) SavingsAccount(Adequate) ∧ Income(Inadequate) ⇒ Investments(Mixed)
4) ∀x (AmountSaved(x) ∧ ∃y (Dependents(y) ∧ Greater(x,MinSavings(y))) ⇒ SavingsAccount(Adequate))
5) ∀x (AmountSaved(x) ∧ ∃y (Dependents(y) ∧ ¬Greater(x,MinSavings(y))) ⇒ SavingsAccount(Inadequate))
6) ∀x (Earnings(x,Steady) ∧ ∃y (Dependents(y) ∧ Greater(x,MinIncome(y))) ⇒ Income(Adequate))
7) ∀x (Earnings(x,Steady) ∧ ∃y (Dependents(y) ∧ ¬Greater(x,MinIncome(y))) ⇒ Income(Inadequate))
8) ∀x (Earnings(x,UnSteady) ⇒ Income(Inadequate))
9) AmountSaved($22000)
10) Earnings($25000,Steady)
11) Dependents(3)



Inv

SavingsAccount(Adequate)

Income(Inadequate)

¬Greater($25000,MinIncome)

Greater($22000,MinSavings)

AmountSaved($22000)

Dependents(3)

Earnings($25000,Steady)

FOL CNF (Conjunctive Normal Form)

Literal = (possibly negated) atomic sentence, e.g., ¬Rich(Me)

Clause = disjunction of literals, e.g. ¬Rich(Me) ∨ Unhappy(Me)

The KB is a conjunction of clauses

Any FOL KB can be converted to CNF as follows:

1. Replace (P ⇒ Q) by (¬P ∨ Q)   (implication elimination)
2. Move ¬ inwards, e.g., ¬∀x P(x) becomes ∃x ¬P(x)
3. Standardize variable names apart
   – e.g., (∀x P(x) ∨ ∃x Q(x)) becomes (∀x P(x) ∨ ∃y Q(y))
4. Move quantifiers left, e.g., (∀x P(x) ∨ ∃y Q(y)) becomes ∀x ∃y (P(x) ∨ Q(y))
5. Eliminate ∃ by *Skolemization*
6. Drop universal quantifiers
7. Distribute ∧ over ∨, e.g., (P ∧ Q) ∨ R becomes (P ∨ R) ∧ (Q ∨ R)

Slide from S. Russel

CNF example

”Everyone who loves all animals is loved by someone”

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

Implication elimination

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

Move $\neg$ inwards ($\neg\forall y\ P$ becomes $\exists y\ \neg P$)

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

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

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

Standardize variables individually

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

Skolemize (Replace $\exists$ with constants)

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

Why not $\forall x\ [\text{Animal}(A) \wedge \neg\text{Loves}(x,A)] \vee \text{Loves}(B,x)$??

Drop $\forall$

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

Distribute $\vee$ over $\wedge$

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

CNF example

”Everyone who loves all animals is loved by someone”

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

The lower (green) sentence says that everyone (x) either fails to love one particular animal (A) or is loved by one particular person (B).

However, the original sentence (above) says that everyone could either fail to love an animal, different for different people, or be loved by someone, different for different people. Therefore we introduce Skolem functions F(x) and G(x) that depend on the individual (x).

Skolemize (Replace ∃ with constants)

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

Why not $\forall x\ [Animal(A) \wedge \neg Loves(x,A)] \vee Loves(B,x)$??

Notation: Unification

Unify(p,q) = Ø

means that

Subst(Ø,p) = Subst(Ø,q)

FOL resolution inference rule

First-order literals are *complementary* if one unifies with the negation of the other

$$\frac{(l_1 \text{ Ţ } l_2 \text{ Ţ } \cdots \text{ Ţ } l_k),\ (m_1 \text{ Ţ } m_2 \text{ Ţ } \cdots \text{ Ţ } m_n)}{Subst(Ø, l_1 \text{ Ţ } \cdots \text{ Ţ } l_{i,1} \text{ Ţ } l_{i'1} \text{ Ţ } \cdots \text{ Ţ } l_k \text{ Ţ } m_1 \text{ Ţ } \cdots \text{ Ţ } m_{j,1} \text{ Ţ } m_{j'1} \text{ Ţ } \cdots \text{ Ţ } m_n)}$$

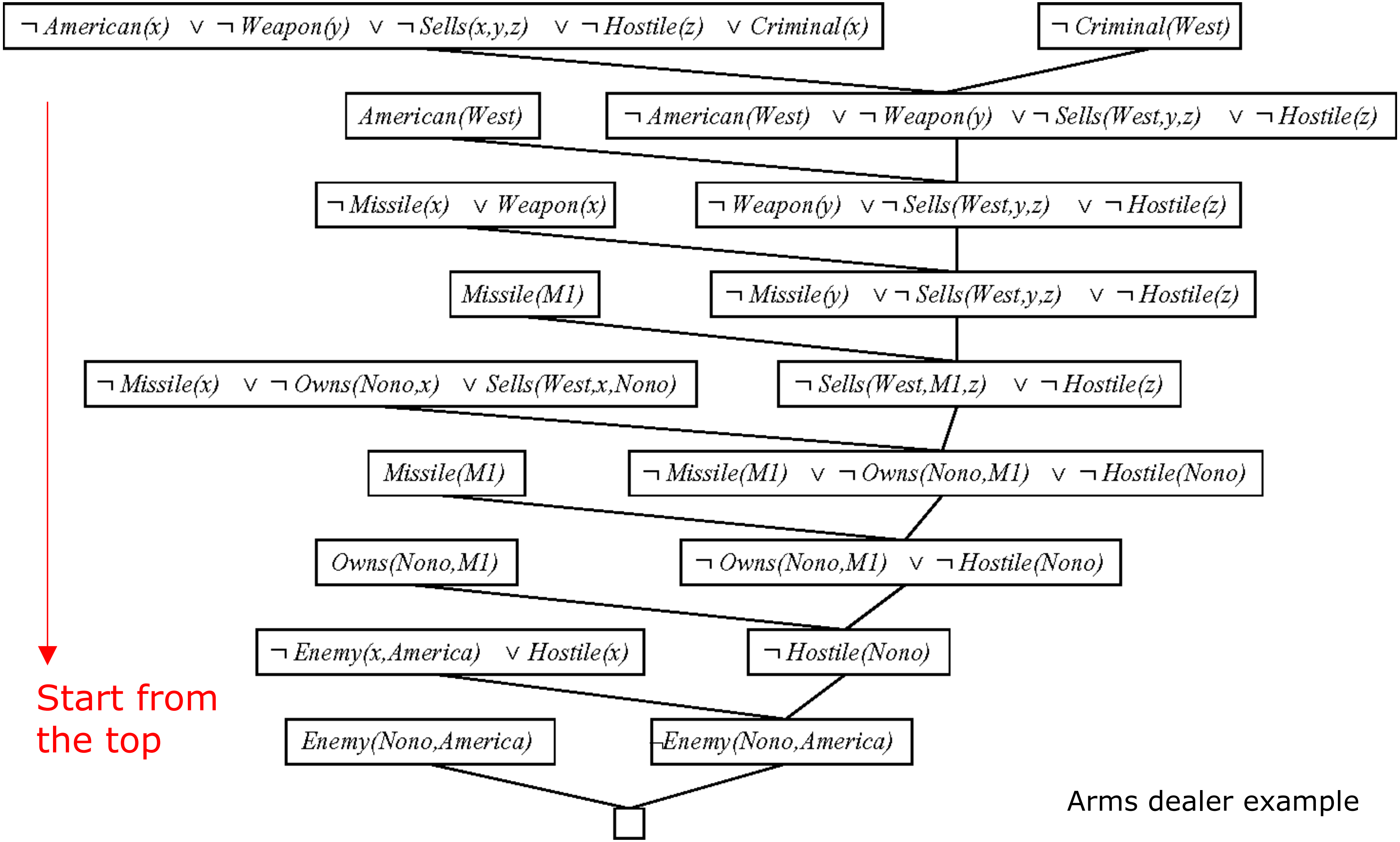
Where Unify(l_i,¬m_j) = Ø.

Note that l_i and m_j are removed

$$\frac{\text{[Animal(F(x)) ∨ Loves(G(x),x)], [¬Loves(u,v) ∨ ¬Kills(u,v)]}}{\text{Subst(\{u/G(x),v/x\}, [Animal(F(x)) ∨ ¬Kills(u,v)])}}$$

Which produces resolvent [Animal(F(x)) ∨ ¬Kills(G(x),x)]

Resolution proves KB ⊨ Ò by proving (KB ∧ ¬Ò) is unsatisfiable



$\neg American(x) \lor \neg Weapon(y) \lor \neg Sells(x,y,z) \lor \neg Hostile(z) \lor Criminal(x)$

$\neg Criminal(West)$

$American(West)$

$\neg American(West) \lor \neg Weapon(y) \lor \neg Sells(West,y,z) \lor \neg Hostile(z)$

$\neg Missile(x) \lor Weapon(x)$

$\neg Weapon(y) \lor \neg Sells(West,y,z) \lor \neg Hostile(z)$

$Missile(M1)$

$\neg Missile(y) \lor \neg Sells(West,y,z) \lor \neg Hostile(z)$

$\neg Missile(x) \lor \neg Owns(Nono,x) \lor Sells(West,x,Nono)$

$\neg Sells(West,M1,z) \lor \neg Hostile(z)$

$Missile(M1)$

$\neg Missile(M1) \lor \neg Owns(Nono,M1) \lor \neg Hostile(Nono)$

$Owns(Nono,M1)$

$\neg Owns(Nono,M1) \lor \neg Hostile(Nono)$

$\neg Enemy(x,America) \lor Hostile(x)$

$\neg Hostile(Nono)$

$Enemy(Nono,America)$

$\neg Enemy(Nono,America)$

□

Start from the top

Arms dealer example

Resolution proves KB ⊨ Ò by proving (KB ∧ ¬Ò) is unsatisfiable

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

$\neg Criminal(West)$

$$\forall x\,(\text{American}(x) \wedge \text{Weapon}(y) \wedge \text{Hostile}(z) \wedge \text{Sells}(x,y,z) \Rightarrow \text{Criminal}(x))$$

Translate to CNF:

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

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

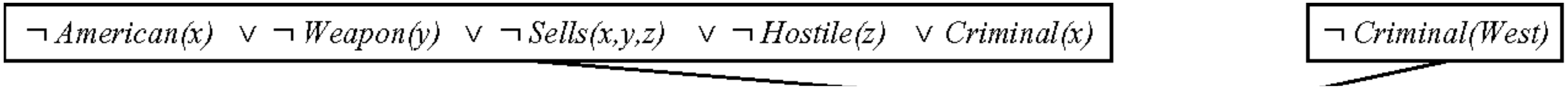
$\forall x\,(\neg\text{American}(x) \vee \neg\text{Weapon}(y) \vee \neg\text{Hostile}(z) \vee \neg\text{Sells}(x,y,z) \vee \text{Criminal}(x))$

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

Any FOL KB can be converted to CNF as follows:

1. Replace (P ⇒ Q) by (¬P ∨ Q) (implication elimination)
2. Move ¬ inwards, e.g., ¬∀x P(x) becomes ∃x ¬P(x)
3. Standardize variables apart, e.g., (∀x P(x) ∨ ∃x Q(x)) becomes (∀x P(x) ∨ ∃y Q(y))
4. Move quantifiers left, e.g., (∀x P(x) ∨ ∃y Q(y)) becomes ∀x ∃y (P(x) ∨ Q(y))
5. Eliminate ∃ by Skolemization
6. Drop universal quantifiers
7. Distribute ∧ over ∨, e.g., (P ∧ Q) ∨ R becomes (P ∨ R) ∧ (Q ∨ R)

Resolution proves KB ⊨ Ò by proving (KB ∧ ¬Ò) is unsatisfiable



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

¬Criminal(West)

∀x (American(x) ∧ Weapon(y) ∧ Hostile(z) ∧ Sells(x,y,z) ⇒ Criminal(x))

Translate to CNF:

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

Any FOL KB can be converted to CNF as follows:

1. Replace (P ⇒ Q) by (¬P ∨ Q) (implication elimination)
2. Move ¬ inwards, e.g., ¬∀x P(x) becomes ∃x ¬P(x)
3. Standardize variables apart, e.g., (∀x P(x) ∨ ∃x Q(x)) becomes (∀x P(x) ∨ ∃y Q(y))
4. Move quantifiers left, e.g., (∀x P(x) ∨ ∃y Q(y)) becomes ∀x ∃y (P(x) ∨ Q(y))
5. Eliminate ∃ by Skolemization
6. Drop universal quantifiers
7. Distribute ∧ over ∨, e.g., (P ∧ Q) ∨ R becomes (P ∨ R) ∧ (Q ∨ R)

Resolution proves KB ⊨ Ò by proving (KB ∧ ¬Ò) is unsatisfiable

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

¬ *Criminal(West)*

$$\frac{(l_1 \text{Ţ } l_2 \text{Ţ } \cdots \text{Ţ } l_k),\ (m_1 \text{Ţ } m_2 \text{Ţ } \cdots \text{Ţ } m_n)}{Subst(\text{Ø}, l_1 \text{Ţ } \cdots \text{Ţ } l_{i,\,1} \text{Ţ } l_{i'\,1} \text{Ţ } \cdots \text{Ţ } l_k \text{Ţ } m_1 \text{Ţ } \cdots \text{Ţ } m_{j,\,1} \text{Ţ } m_{j'\,1} \text{Ţ } \cdots \text{Ţ } m_n)}$$

Where Unify(l_i,¬m_j) = Ø.

Resolution proves KB ⊨ Ò by proving (KB ∧ ¬Ò) is unsatisfiable

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

¬Criminal(West)

$$\frac{(l_1 \text{Ţ}\ l_2 \text{Ţ} \cdots \text{Ţ}\ l_k),\ (m_1 \text{Ţ}\ m_2 \text{Ţ} \cdots \text{Ţ}\ m_n)}{Subst(\text{Ø}\ , l_1 \text{Ţ} \cdots \text{Ţ}\ l_{i,\ 1} \text{Ţ}\ l_{i'\ 1} \text{Ţ} \cdots \text{Ţ}\ l_k \text{Ţ}\ m_1 \text{Ţ} \cdots \text{Ţ}\ m_{j,\ 1} \text{Ţ}\ m_{j'\ 1} \text{Ţ} \cdots \text{Ţ}\ m_n)}$$

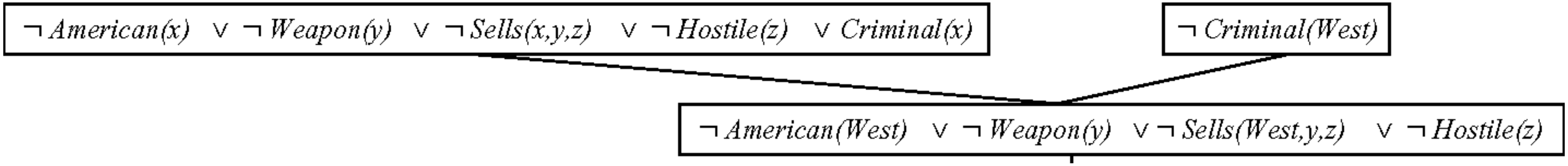
Where Unify(l_i,¬m_j) = Ø.

l_1 = ¬American(x)
l_2 = ¬Weapon(y)
l_3 = ¬Sells(x,y,z)
l_4 = ¬Hostile(z)
l_5 = Criminal(x)
m_1 = Criminal(West)

Unify(l_5,¬m_1) = Ø= {x/West}

Subst(Ø $l_1 \vee l_2 \vee l_3 \vee l_4$) =...

Resolution proves KB ⊨ Ò by proving (KB ∧ ¬Ò) is unsatisfiable

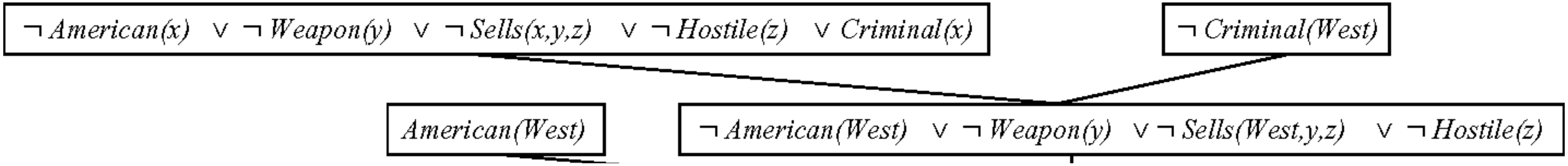


$\neg American(x) \vee \neg Weapon(y) \vee \neg Sells(x,y,z) \vee \neg Hostile(z) \vee Criminal(x)$

$\neg Criminal(West)$

$\neg American(West) \vee \neg Weapon(y) \vee \neg Sells(West,y,z) \vee \neg Hostile(z)$

Resolution proves KB ⊨ Ò by proving (KB ∧ ¬Ò) is unsatisfiable



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

¬ *Criminal(West)*

American(West)

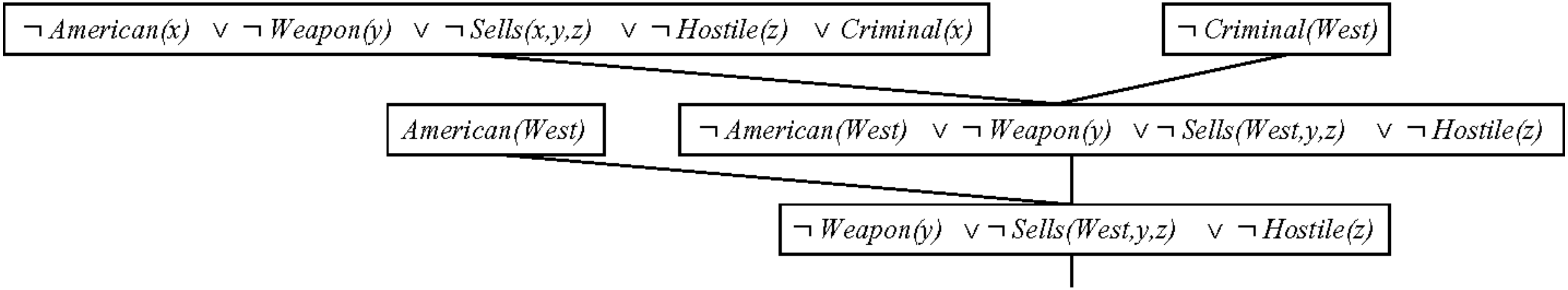
¬ *American(West)* ∨ ¬ *Weapon(y)* ∨ ¬ *Sells(West,y,z)* ∨ ¬ *Hostile(z)*

l_1 = ¬American(x)
l_2 = ¬Weapon(y)
l_3 = ¬Sells(x,y,z)
l_4 = ¬Hostile(z)
l_5 = Criminal(x)
m_2 = American(West)

Unify(l_1,¬m_2) = Ø= {x/West}

Subst(Ø $l_2 \vee l_3 \vee l_4$) =...

Resolution proves KB ⊨ Ò by proving (KB ∧ ¬Ò) is unsatisfiable



$\neg American(x) \lor \neg Weapon(y) \lor \neg Sells(x,y,z) \lor \neg Hostile(z) \lor Criminal(x)$

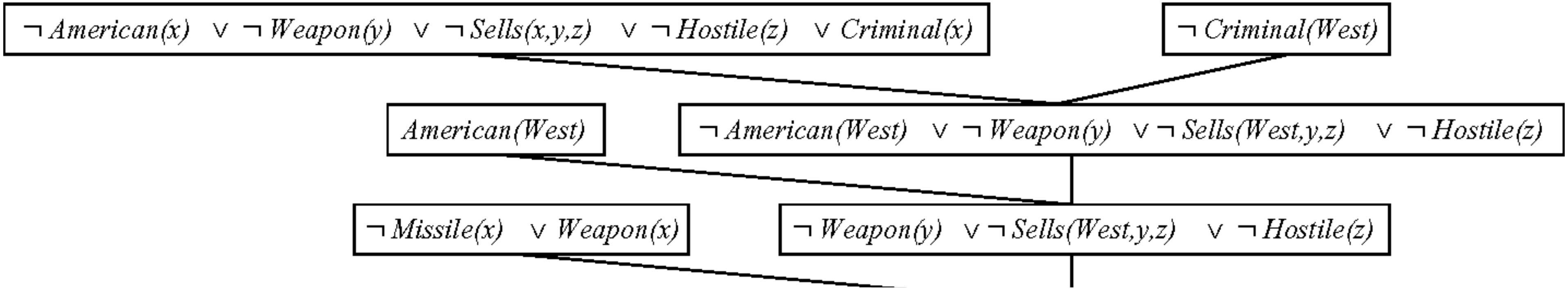
$\neg Criminal(West)$

$American(West)$

$\neg American(West) \lor \neg Weapon(y) \lor \neg Sells(West,y,z) \lor \neg Hostile(z)$

$\neg Weapon(y) \lor \neg Sells(West,y,z) \lor \neg Hostile(z)$

Resolution proves KB ⊨ Ò by proving (KB ∧ ¬Ò) is unsatisfiable



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

¬*Criminal*(*West*)

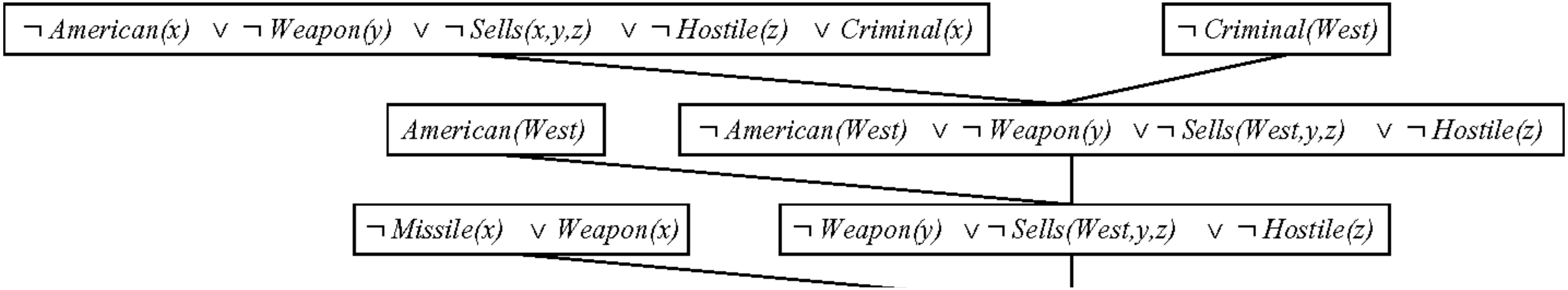
American(*West*)

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

¬*Missile*(*x*) ∨ *Weapon*(*x*)

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

Resolution proves KB ⊨ Ò by proving (KB ∧ ¬Ò) is unsatisfiable



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

¬ *Criminal(West)*

American(West)

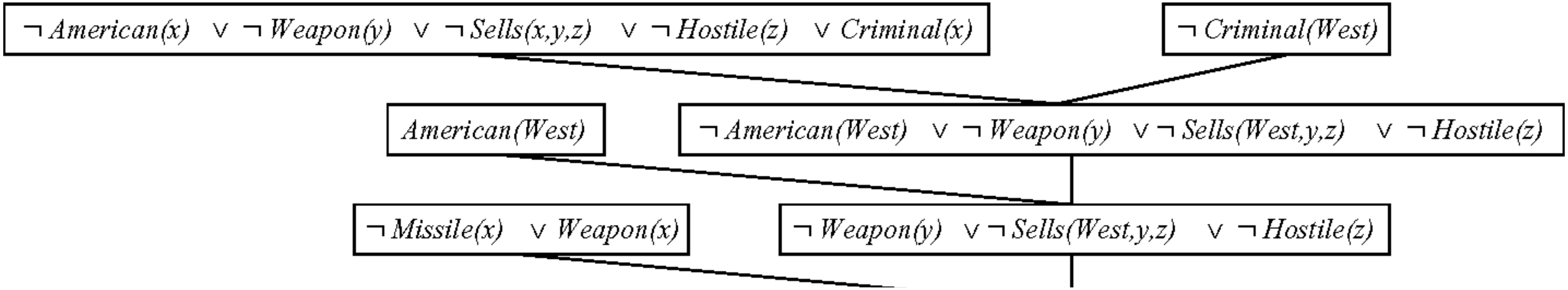
¬ *American(West)* ∨ ¬ *Weapon(y)* ∨ ¬ *Sells(West,y,z)* ∨ ¬ *Hostile(z)*

¬ *Missile(x)* ∨ *Weapon(x)*

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

?

Resolution proves KB ⊨ Ò by proving (KB ∧ ¬Ò) is unsatisfiable



¬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)

l_2 = ¬Weapon(y)

l_3 = ¬Sells(x,y,z)

l_4 = ¬Hostile(z)

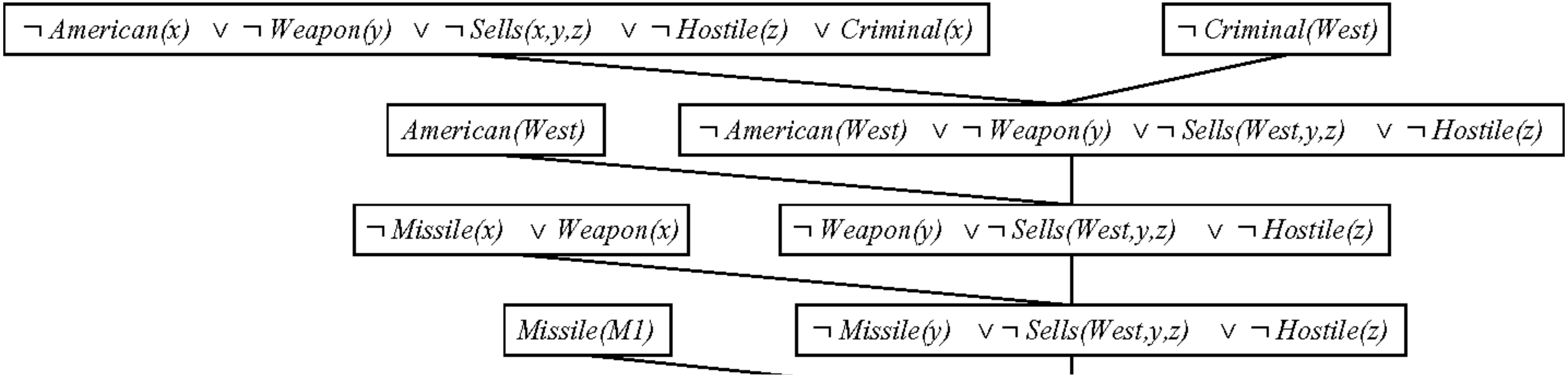
m_3 = Weapon(x)

m_4 = Missile(x)

Unify(l_2,¬m_3) = Ø= {y/x}

Subst(Ø $l_2 \vee l_3 \vee l_4 \vee m_4$) =...

Resolution proves KB ⊨ Ò by proving (KB ∧ ¬Ò) is unsatisfiable



¬ *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)*

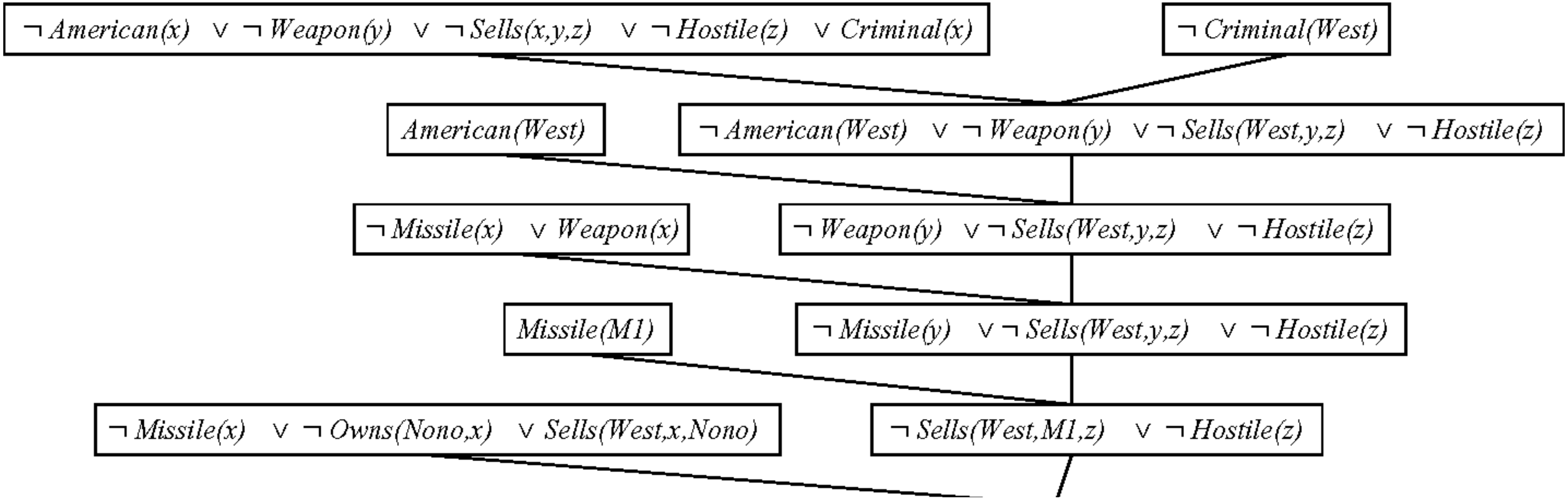
¬ *Weapon(y)* ∨ ¬ *Sells(West,y,z)* ∨ ¬ *Hostile(z)*

Missile(M1)

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

?

Resolution proves KB ⊨ Ò by proving (KB ∧ ¬Ò) is unsatisfiable



¬ *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)*

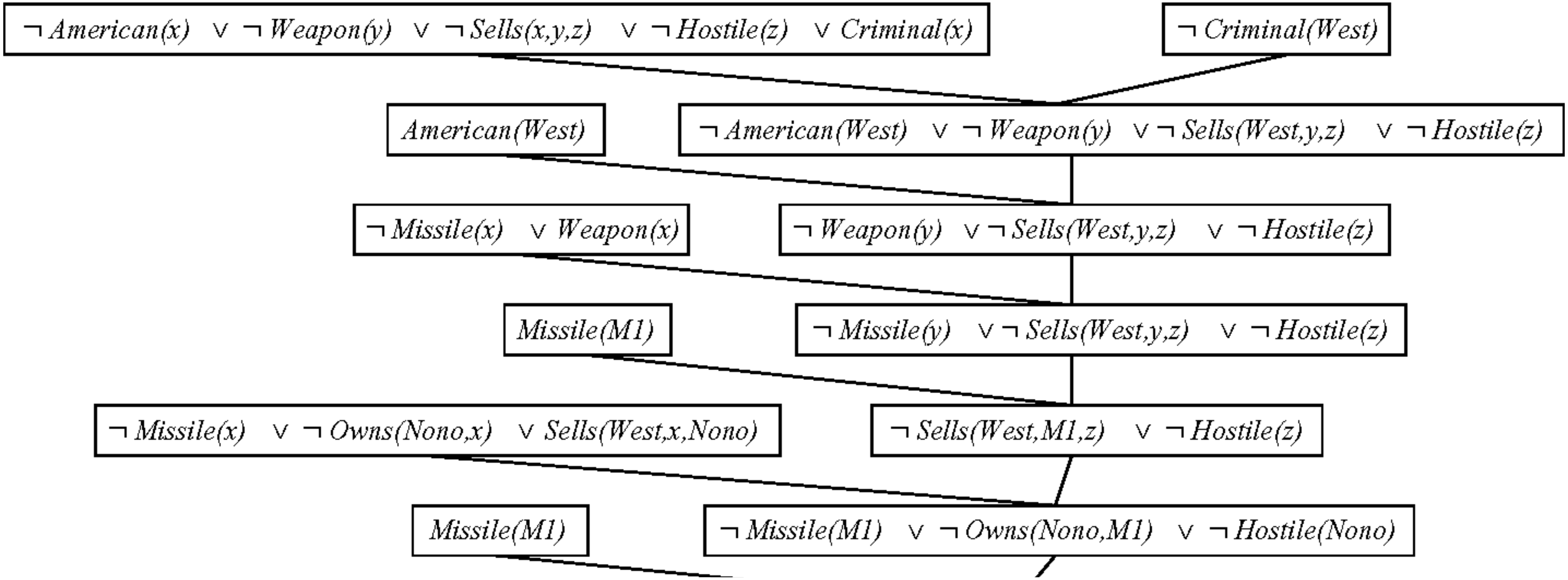
Missile(M1)

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

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

¬ *Sells(West,M1,z)* ∨ ¬ *Hostile(z)*

Resolution proves KB ⊨ Ò by proving (KB ∧ ¬Ò) is unsatisfiable



¬ *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)*

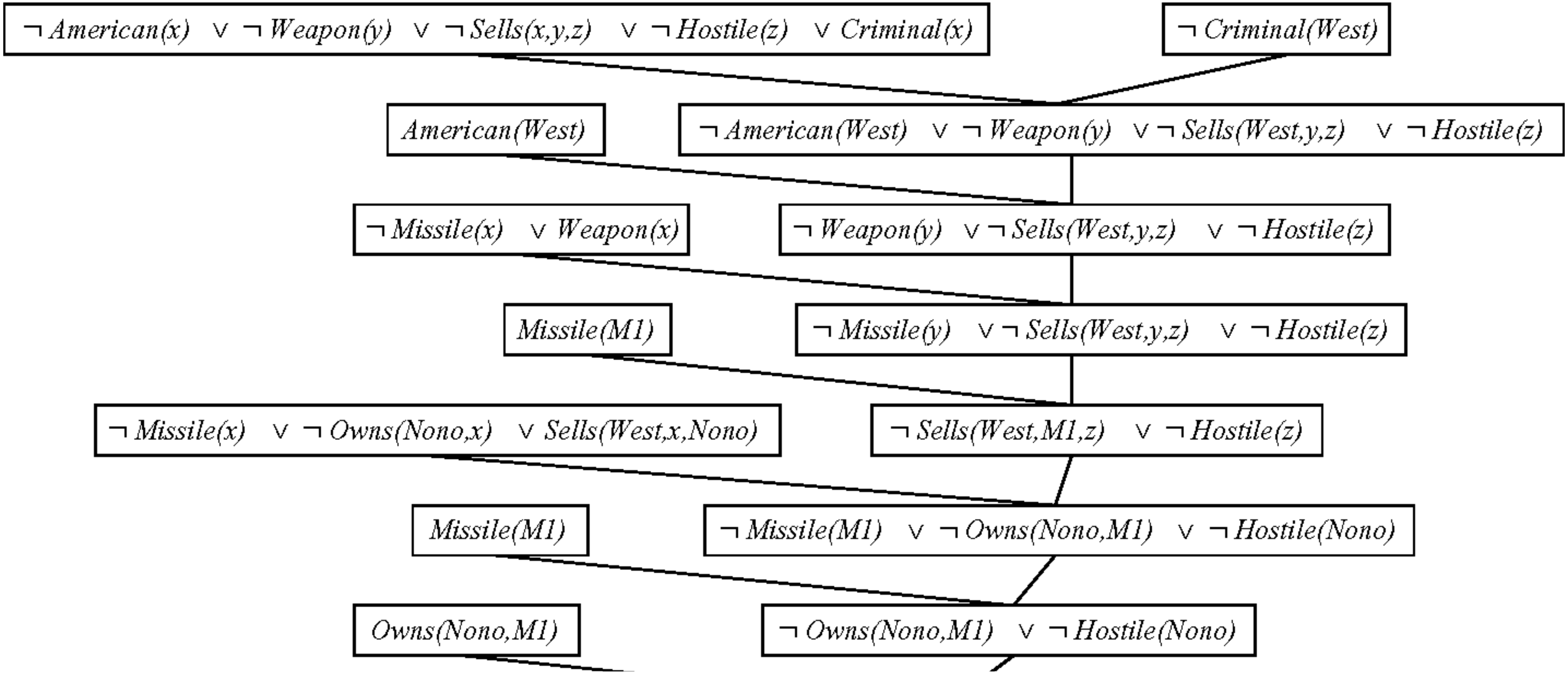
¬ *Missile(x)* ∨ ¬ *Owns(Nono,x)* ∨ *Sells(West,x,Nono)*

¬ *Sells(West,M1,z)* ∨ ¬ *Hostile(z)*

Missile(M1)

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

Resolution proves KB ⊨ Ò by proving (KB ∧ ¬Ò) is unsatisfiable



¬ *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)*

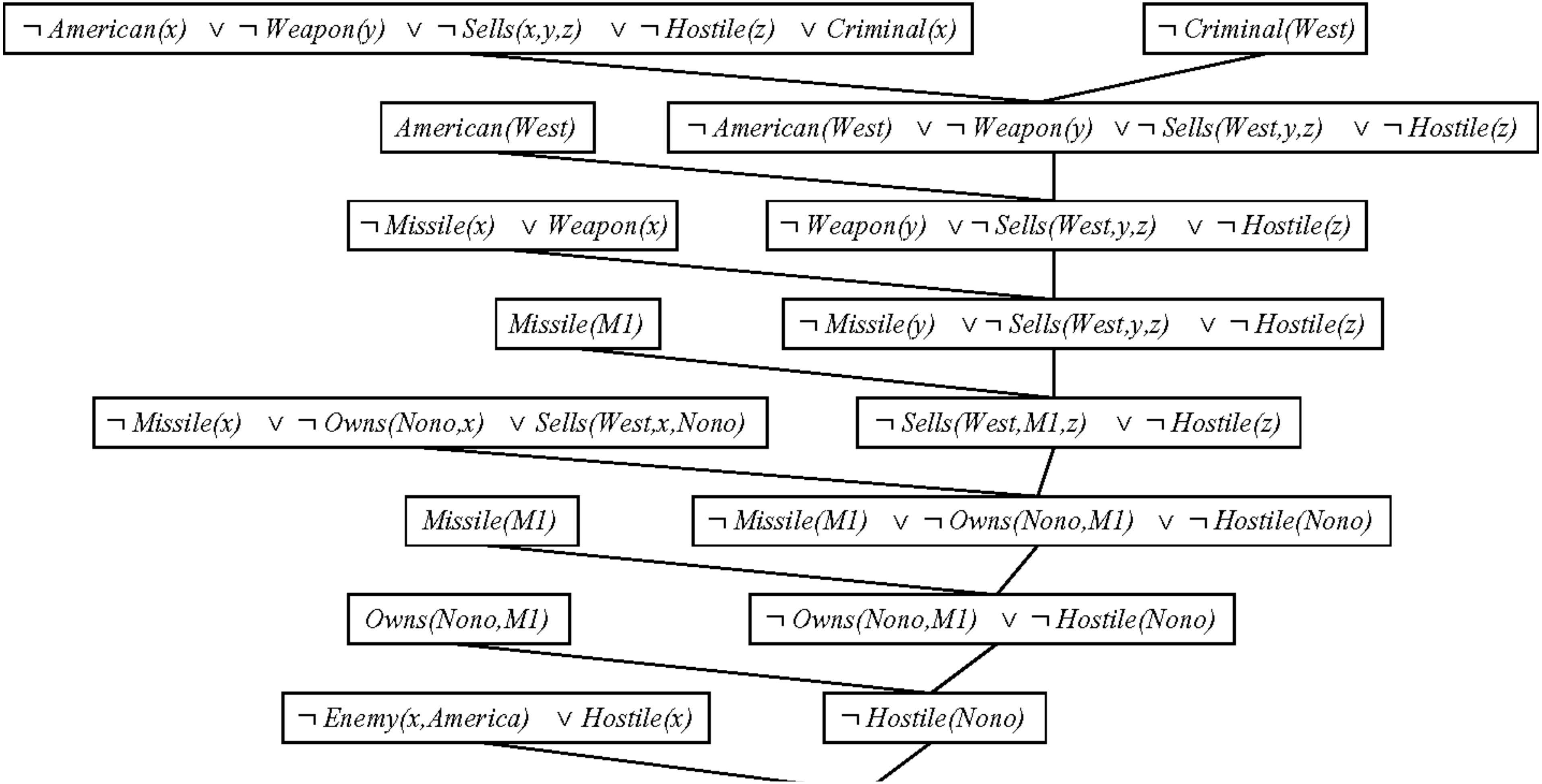
Missile(M1)

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

Owns(Nono,M1)

¬ *Owns(Nono,M1)* ∨ ¬ *Hostile(Nono)*

Resolution proves KB ⊨ Ò by proving (KB ∧ ¬Ò) is unsatisfiable



¬ *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)*

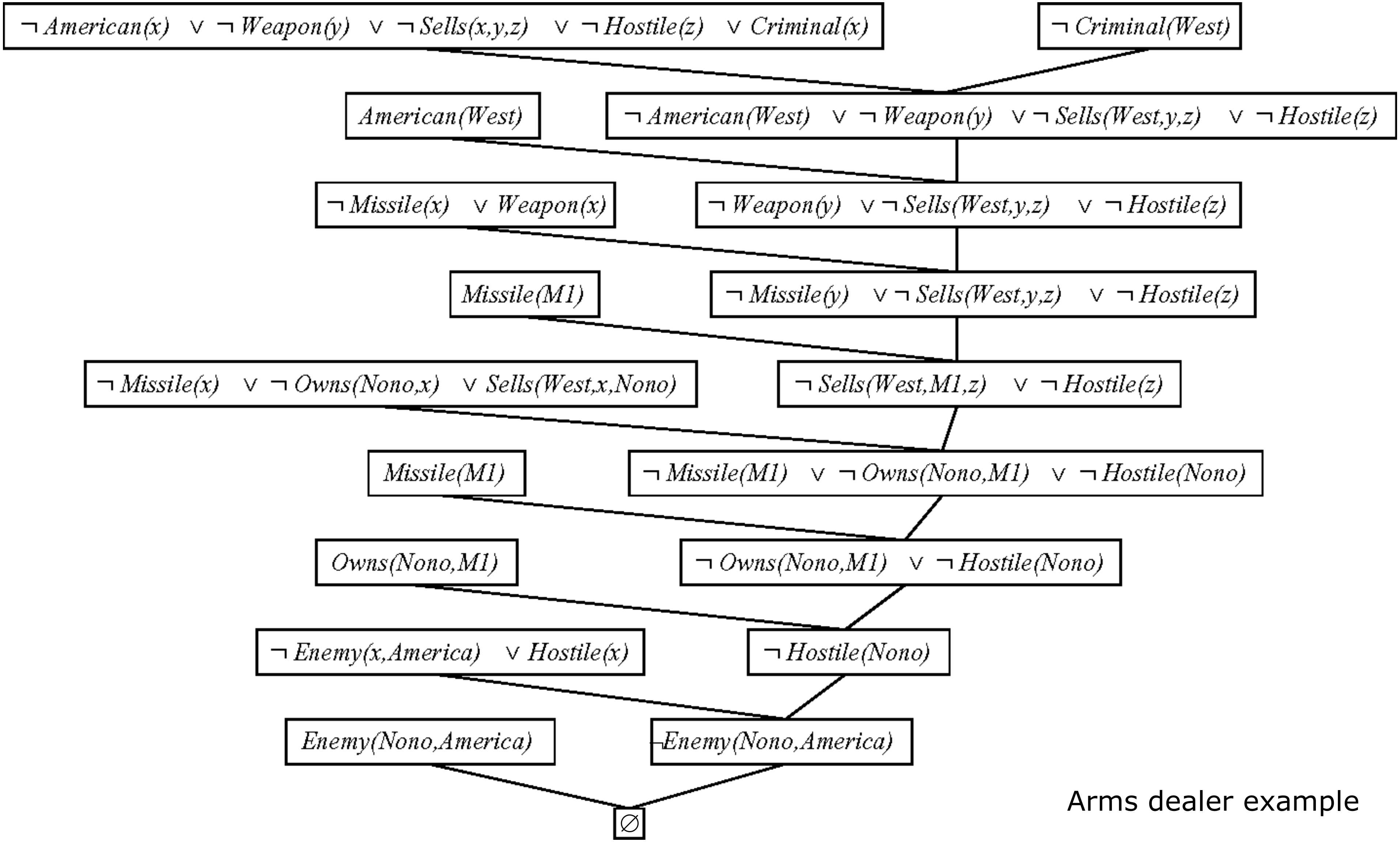
Owns(Nono,M1)

¬ *Owns(Nono,M1)* ∨ ¬ *Hostile(Nono)*

¬ *Enemy(x,America)* ∨ *Hostile(x)*

¬ *Hostile(Nono)*

Resolution proves KB ⊨ Ò by proving (KB ∧ ¬Ò) is unsatisfiable



¬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)

∅

Arms dealer example

Resolution example II

- **Problem Statement:** Tony, Shikuo and Ellen belong to the Hoofers Club. Every member of the Hoofers Club is either a skier or a mountain climber or both. No mountain climber likes rain, and all skiers like snow. Ellen dislikes whatever Tony likes and likes whatever Tony dislikes. Tony likes rain and snow.

- **Query:** Is there a member of the Hoofers Club who is a mountain climber but not a skier?

Example from Charles Dyer (referenced by Tuomas Sandhom @ CMU)

KB

The rules only apply to members of the Hoofers club (our domain).

Problem Statement: Tony, Shikuo and Ellen belong to the Hoofers Club. Every member of the Hoofers Club is either a skier or a mountain climber or both. No mountain climber likes rain, and all skiers like snow. Ellen dislikes whatever Tony likes and likes whatever Tony dislikes. Tony likes rain and snow.

$$Tony$$

$$Shikuo$$

$$Ellen$$

$$\forall x \;\; Skier(x) \vee MountainC(x)$$

$$\neg \exists x \;\; MountainC(x) \wedge Likes(x, Rain)$$

$$\forall x \;\; Skier(x) \Rightarrow Likes(x, Snow)$$

$$\forall x \;\; Likes(Tony, x) \Leftrightarrow \neg Likes(Ellen, x)$$

$$Likes(Tony, Rain)$$

$$Likes(Tony, Snow)$$

Query

Query: Is there a member of the Hoofers Club who is a mountain climber but not a skier?

$$\exists x \ \ MountainC(x) \wedge \neg Skier(x)$$

KB + the negation of the Query

$Tony$

$Shikuo$

$Ellen$

$\forall x \; Skier(x) \lor MountainC(x)$

$\neg \exists x \; MountainC(x) \land Likes(x, Rain)$

$\forall x \; Skier(x) \Rightarrow Likes(x, Snow)$

$\forall x \; Likes(Tony, x) \Leftrightarrow \neg Likes(Ellen, x)$

$Likes(Tony, Rain)$

$Likes(Tony, Snow)$

$\neg \exists x \; MountainC(x) \land \neg Skier(x)$

(KB ∧¬Q) to Clause form...(I)

Tony · *x* Ť *Skier*(*x*) Ţ *Likes*(*x*, *Snow*)

Shikuo

Ellen

· *x* *Skier*(*x*) Ţ *MountainC*(*x*)

Ť ˆ *x* *MountainC*(*x*) ť *Likes*(*x*, *Rain*)

· *x* *Skier*(*x*) Ŭ *Likes*(*x*, *Snow*)

· *x* *Likes*(*Tony*, *x*) ţ Ť *Likes*(*Ellen*, *x*)

Likes(*Tony*, *Rain*)

Likes(*Tony*, *Snow*)

Ť ˆ *x* *MountainC*(*x*) ť Ť *Skier*(*x*)

(KB ∧¬Q) to Clause form…(II)

· *x* Ť " *MountainC*(*x*) ť *Likes*(*x*, *Rain*)ˉ

· *x* Ť *MountainC*(*x*) Ţ Ť *Likes*(*x*, *Rain*)

Tony

Shikuo

Ellen

· *x* *Skier*(*x*) Ţ *MountainC*(*x*)

Ť ˆ *x* *MountainC*(*x*) ť *Likes*(*x*, *Rain*)

· *x* *Skier*(*x*) Ů *Likes*(*x*, *Snow*)

· *x* *Likes*(*Tony*, *x*) ţ Ť *Likes*(*Ellen*, *x*)

Likes(*Tony*, *Rain*)

Likes(*Tony*, *Snow*)

Ť ˆ *x* *MountainC*(*x*) ť Ť *Skier*(*x*)

(KB ∧¬Q) to Clause form…(III)

Ŷ· *x* *Likes*(*Tony*, *x*) Ŭ Ť *Likes*(*Ellen*, *x*)
ŷ
Ž· *x* Ť *Likes*(*Ellen*, *x*) Ŭ *Likes*(*Tony*, *x*)

Ŷ· *x* Ť *Likes*(*Tony*, *x*) Ţ Ť *Likes*(*Ellen*, *x*)
ŷ
Ž · *x* *Likes*(*Ellen*, *x*) Ţ *Likes*(*Tony*, *x*)

Tony

Shikuo

Ellen

· *x* *Skier*(*x*) Ţ *MountainC*(*x*)

Ť ˆ *x* *MountainC*(*x*) ť *Likes*(*x*, *Rain*)

· *x* *Skier*(*x*) Ŭ *Likes*(*x*, *Snow*)

· *x* *Likes*(*Tony*, *x*) ţ Ť *Likes*(*Ellen*, *x*)

Likes(*Tony*, *Rain*)

Likes(*Tony*, *Snow*)

Ť ˆ *x* *MountainC*(*x*) ť Ť *Skier*(*x*)

(KB ∧¬Q) to Clause form…(IV)

· *x* Ť ˝*MountainC*(*x*) ť Ť *Skier*(*x*)¯

Tony

· *x* Ť *MountainC*(*x*) Ţ *Skier*(*x*)

Shikuo

Ellen

· *x* *Skier*(*x*) Ţ *MountainC*(*x*)

Ť ˆ *x* *MountainC*(*x*) ť *Likes*(*x*, *Rain*)

· *x* *Skier*(*x*) Ů *Likes*(*x*, *Snow*)

· *x* *Likes*(*Tony*, *x*) ţ Ť *Likes*(*Ellen*, *x*)

Likes(*Tony*, *Rain*)

Likes(*Tony*, *Snow*)

Ť ˆ *x* *MountainC*(*x*) ť Ť *Skier*(*x*)

(KB ∧¬Q) in Clause form

We drop the universal quantifiers...
We also change variable names...

$$Tony$$

$$Shikuo$$

$$Ellen$$

$$Skier(x) \lor MountainC(x)$$

$$\neg MountainC(y) \lor \neg Likes(y, Rain)$$

$$\neg Skier(z) \lor Likes(z, Snow)$$

$$Likes(Tony, w) \lor Likes(Ellen, w)$$

$$\neg Likes(Tony, v) \lor \neg Likes(Ellen, v)$$

$$Likes(Tony, Rain)$$

$$Likes(Tony, Snow)$$

$$\neg MountainC(s) \lor Skier(s)$$

1 $Tony$

2 $Shikuo$

3 $Ellen$

4 $Skier(x) \lor MountainC(x)$

5 $\neg MountainC(y) \lor \neg Likes(y, Rain)$

6 $\neg Skier(z) \lor Likes(z, Snow)$

7 $Likes(Tony, w) \lor Likes(Ellen, w)$

8 $\neg Likes(Tony, v) \lor \neg Likes(Ellen, v)$

9 $Likes(Tony, Rain)$

10 $Likes(Tony, Snow)$

11 $\neg MountainC(s) \lor Skier(s)$

1 $Tony$

2 $Shikuo$

3 $Ellen$

4 $Skier(x) \lor MountainC(x)$

5 $\neg MountainC(y) \lor \neg Likes(y, Rain)$

6 $\neg Skier(z) \lor Likes(z, Snow)$

7 $Likes(Tony, w) \lor Likes(Ellen, w)$

8 $\neg Likes(Tony, v) \lor \neg Likes(Ellen, v)$

9 $Likes(Tony, Rain)$

10 $Likes(Tony, Snow)$

11 $\neg MountainC(s) \lor Skier(s)$

$$\frac{\neg MountainC(s) \lor Skier(s), Skier(x) \lor MountainC(x)}{Skier(x)}$$

Unify(p_4,$\neg p_{11}$) = Ø= {x/s}

The resolvent becomes our clause # 12

1 $Tony$

2 $Shikuo$

3 $Ellen$

4 $Skier(x) \lor MountainC(x)$

5 $\neg MountainC(y) \lor \neg Likes(y, Rain)$

6 $\neg Skier(z) \lor Likes(z, Snow)$

7 $Likes(Tony, w) \lor Likes(Ellen, w)$

8 $\neg Likes(Tony, v) \lor \neg Likes(Ellen, v)$

9 $Likes(Tony, Rain)$

10 $Likes(Tony, Snow)$

11 $\neg MountainC(x) \lor Skier(x)$

12 $Skier(x)$

$$\frac{Skier(x), \neg Skier(z) \lor Likes(z, Snow)}{Likes(x, Snow)}$$

Unify(p_6, $\neg p_{12}$) = Ø= {x/z}

The resolvent becomes our clause # 13

1 $Tony$

2 $Shikuo$

3 $Ellen$

4 $Skier(x)\text{Ţ}\ MountainC(x)$

5 $\text{Ť}\ MountainC(y)\text{Ţ}\ \text{Ť}\ Likes(y, Rain)$

6 $\text{Ť}\ Skier(z)\text{Ţ}\ Likes(z, Snow)$

7 $Likes(Tony, w)\text{Ţ}\ Likes(Ellen, w)$

8 $\text{Ť}\ Likes(Tony, v)\text{Ţ}\ \text{Ť}\ Likes(Ellen, v)$

9 $Likes(Tony, Rain)$

10 $Likes(Tony, Snow)$

11 $\text{Ť}\ MountainC(x)\text{Ţ}\ Skier(x)$

12 $Skier(x)$

13 $Likes(x, Snow)$

$$\frac{Likes(Tony, Snow), \text{Ť}\ Likes(Tony, v)\ \text{Ţ}\ \text{Ť}\ Likes(Ellen, v)}{\text{Ť}\ Likes(Ellen, Snow)}$$

Unify(p_{10}, ¬p_{8}) = Ø= {v/Snow}

The resolvent becomes our clause # 14

1 *Tony*

2 *Shikuo*

3 *Ellen*

4 *Skier*(*x*) Ţ *MountainC*(*x*)

5 Ť *MountainC*(*y*) Ţ Ť *Likes*(*y*, *Rain*)

6 Ť *Skier*(*z*) Ţ *Likes*(*z*, *Snow*)

7 *Likes*(*Tony*, *w*) Ţ *Likes*(*Ellen*, *w*)

8 Ť *Likes*(*Tony*, *v*) Ţ Ť *Likes*(*Ellen*, *v*)

9 *Likes*(*Tony*, *Rain*)

10 *Likes*(*Tony*, *Snow*)

11 Ť *MountainC*(*x*) Ţ *Skier*(*x*)

12 *Skier*(*x*)

13 *Likes*(*x*, *Snow*)

14 Ť *Likes*(*Ellen*, *Snow*)

We have proved that there is a member of the Hoofers club who is a mountain climber but not a skier.

$$\frac{\text{Ť}\ Likes(Ellen, Snow), Likes(x, Snow)}{\text{Ō}}$$

Unify(p_{13}, $\neg p_{14}$) = Ø= {x/Ellen}