CSC421 Intro to Artificial Intelligence

UNIT 14:
Inference in First-Order Logic

Outline

- Generalized Modus Ponens
- Forward and backward chaining
- Resolution
- Logic Programming

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
- $Unify(a,b)=\theta$ if $a\theta = b\theta$

Unification examples

P	Q	θ ?
Knows(John, x)	Knows(John, Jane)	
Knows(John, x)	Knows(y, Bob)	
Knows(John, x)	Knows(y, Mother(y))	
Knows(John, x)	Knows(x, Bob)	

Knows(John,x) Knows(y,z)

Most general unifier Knows(John, z) rather than Knows(John, John)

Generalized Modus Ponens

$$\frac{p_1', p_2', \ldots p_n' \qquad (p_1 \wedge p_2 \wedge \ldots p_n \Rightarrow q)}{q\theta}$$

where $p_i'\theta = p_i\theta$ for all i

p_1' is King(John) p_1 is King(x)

p_2' is Greedy(y) p_2 is Greedy(x)

θ {x/John, y/John} q is Evil(x)

$q\theta$ is Evil(John)

GMP is used with KB of definiteclauses (exactly one positive literal) All variables are assumed universally quanitifed.

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 Colonel West is criminal

Example

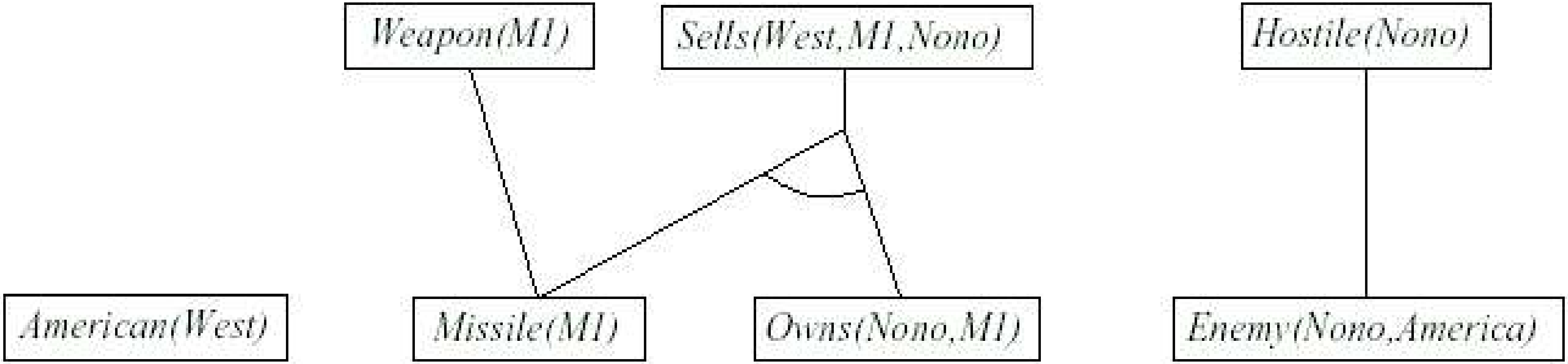
- It is a crime for an american to sell weapons to hostile nations:
  - $American(x) \wedge Weapon(y) \wedge Sells(x,y,z) \wedge Hostile(z) => Criminal(x)$
- Nono has some missiles
  - Owns(Nono, M1) and Missile(M1)
- All of it's missiles were sold by Colonel West
  - $Missile(x) \wedge Owns(Nono, x) => Sell(West, x, Nono)$
- Missiles are weapons
  - Missile(x) => Weapon(x)
- An enemy of America counts as hostile
  - Enemy(x, America) => Hostile(x)

Example

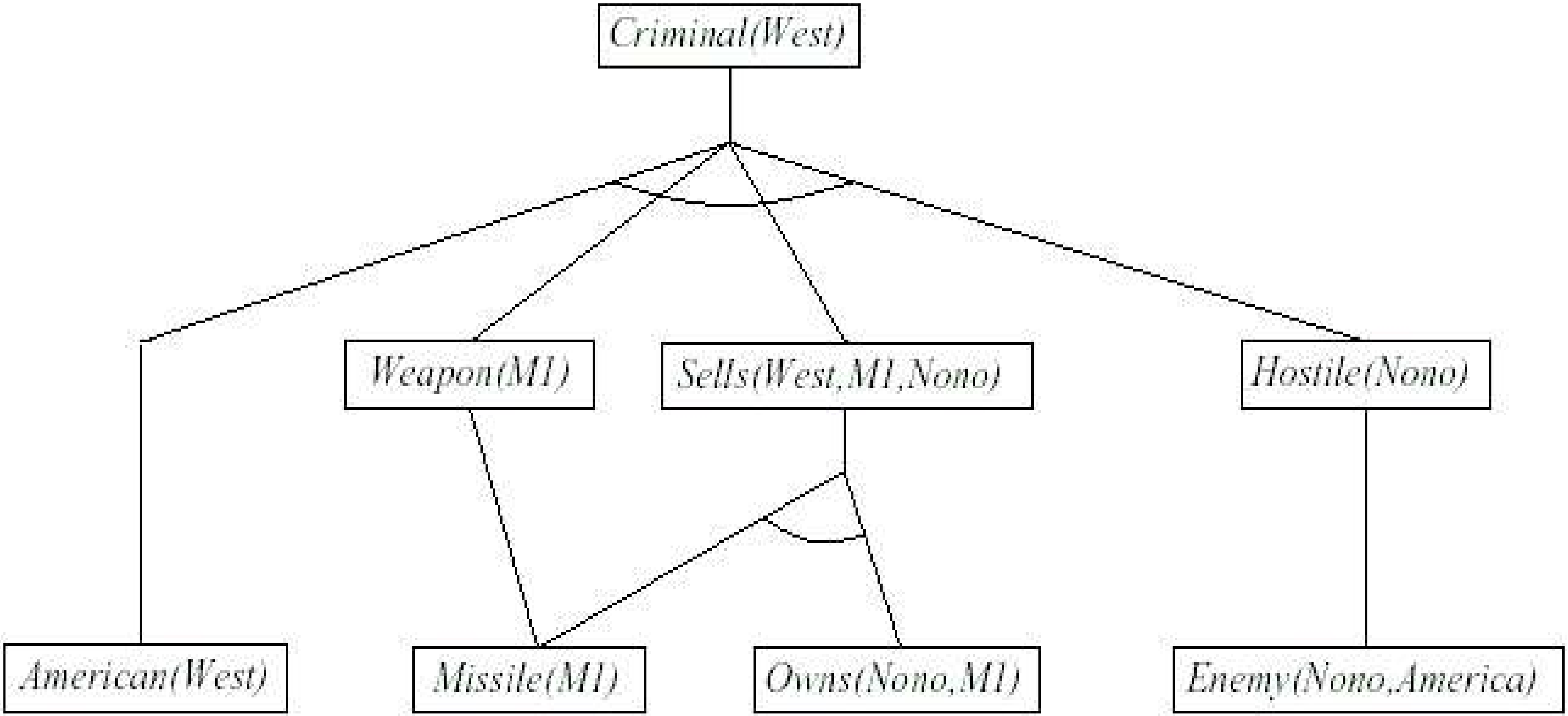
- West is American
  - American(West)
- The country Nono an enemy of America
  - Enemy(Nono, America)

Forward Chaining Proof

Forward Chaining Proof



Forward Chaining



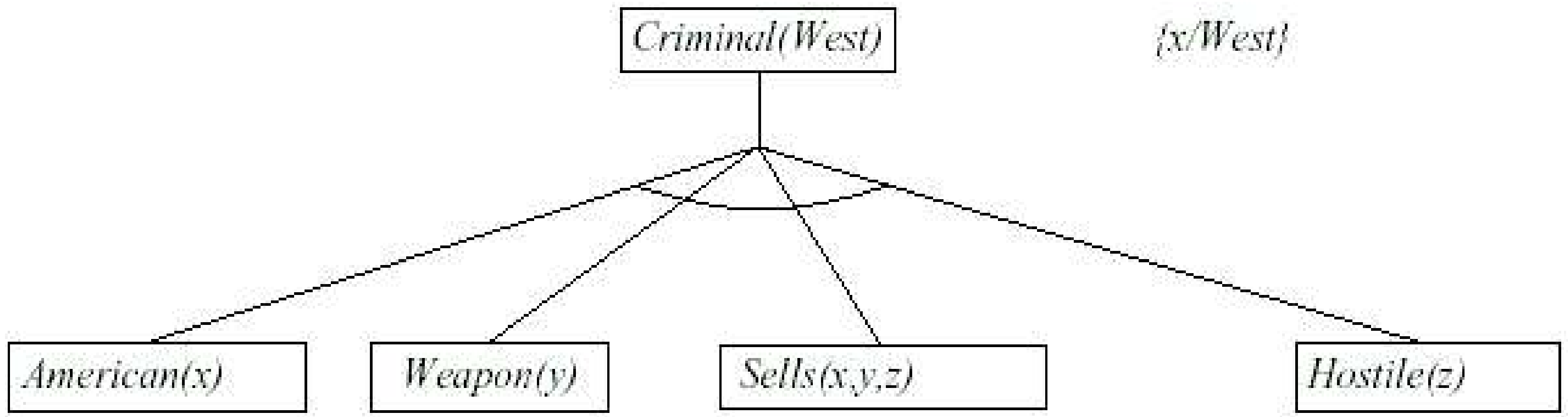
Properties of FC

- Sound & complete for FOL definite clauses
- Datalog = first-order definite clauses, no functions. FC terminates for Datalog in poly iterations
- May not terminate in general if a is not entailed
- Unavoidable: entailment with definite clauses is semidecidable

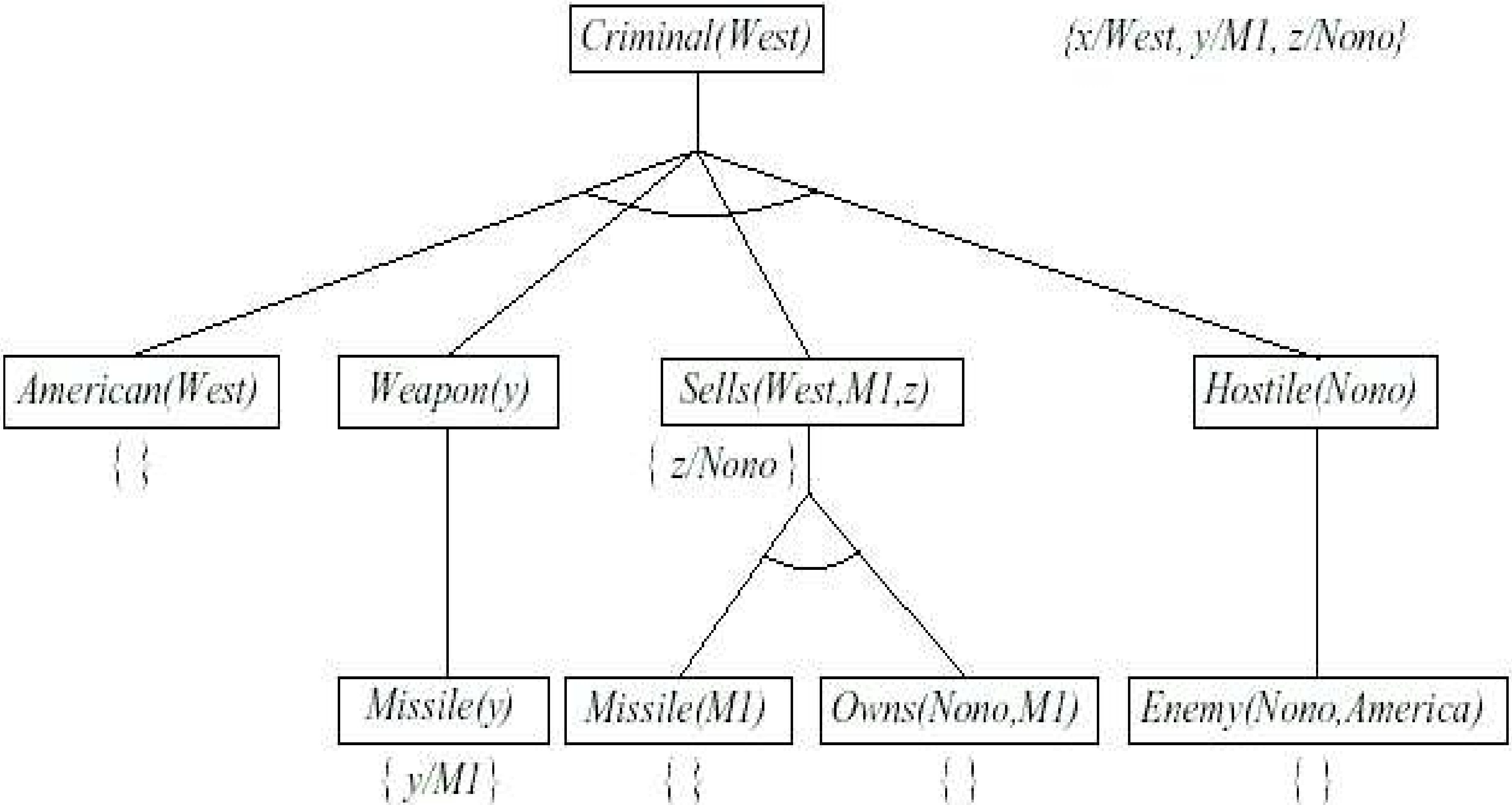
Backward Chaining

$\boxed{Criminal(West)}$

Backward Chaining



Backward Chaining



Properties of BC

- Depth-first recursive proof search – space is linear in size of proof
- Incomplete due to infinite loops
- Inefficient due to repeated subgoals
- Widely used for logic programming

Logic Programming

- Computation as inference in logical KB

Logic Programming	Ordinary Programming
1) Identify problem	Identify problem
2) Assemble information	Assemble information
3) Tea break	Figure out solution
4) Encode info in KB	Program solution
5) Encode problem instance as facts	Encode problem instance as data
6) Ask queries	Apply program to data
7) Find false facts	Debug procedural errors

Prolog systems

- Basis: backward chaining with Horn clauses + bells and whistles
- Widely used in Europe, Japan (5th gen.)
- Program = set of clauses
  - head :- literal1,....,literaln
  - criminal(X) :- american(X), weapon(Y), sells (X,Y<Z), hostile(Z)
- Efficient unification, retrieval
- Depth-first, left-to-right BC
- Predicates for arithmetic X is Y * Z + 3
- Closed world assumption (negation as failure)

Prolog examples

- DFS
  - dfs(X) :- goal(X).
  - dfs(X) :- succesor(X, S), dfs(S).
- Loops expressed as recursions
- 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]) ?
  - A = [], B=[1,2]
  - A = [1], B = [2]
  - A = [1,2], B = []

Resolution

Full first-order version

$$\frac{l_1 \vee l_2 \vee \ldots \vee l_k \qquad m_1 \vee m_2 \vee \ldots \vee m_n}{(l_1 \vee l_2 \vee l_{i-1} \vee l_{i+1} \ldots \vee l_k \vee m_1 \vee m_2 \vee m_{j-1} \vee m_{j+1} \ldots \vee m_n)\theta}$$

Where UNIFY(l_i, $\neg m_j$)= θ

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

Resolution Example

- $\neg$ Rich(x) $\vee$ Unhappy(x)   Rich(Ken)
- Unhappy(Ken) with $\theta = \{x/Ken\}$
- Conversion to CNF
  - Eliminate biconditionals and implications
  - Move negations inwards
  - Standarize variables
  - Skolemize
  - Drop universal quantifiers
  - Distribute $\wedge$ over $\vee$