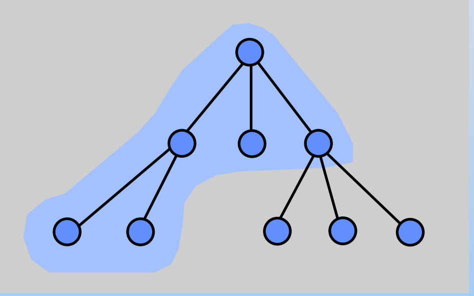
Part III – Knowledge and Reasoning

- **9 Inference in First-Order Logic**
  - Inference Rules. – Generalised Modus Ponens.
  - Forward and Backward Chaining. – Resolution.
- **10 Logical Reasoning Systems**
  - Indexing, Retrieval and Unification. – Logic Programming / Prolog. – Production Systems.
  - Frames and Semantic / Conceptual Networks.
  - Managing Retractions, Assumptions, and Explanations.

Forward and Backward Chaining

- **Reasoning**
  - Knowledge representation language (First-Order Logic)
  - Efficient inference rule (Generalised Modus Ponens)
  - *> Generate the proof*
- **Using the GMP**
  - Forward chaining (data-driven): KB, α |– ?
    - Start with the KB and generate new sentences e.g. to derive the consequences of newly added facts.
  - Backward chaining (goal-driven): KB |– α?
    - Start with a sentence not in the KB and attempt to establish its premises, e.g. to prove some new fact.

Forward Chaining



- **Idea: inferring consequences**
  – TELLing a new sentence α, KB, $\alpha \vdash ?$
- **Pseudo-algorithm**
  – If α already in the KB, do nothing
  – Find all implications that have α as a premise, i.e.
  $\alpha \wedge \alpha_1 \wedge \ldots \wedge \alpha_n \Rightarrow \beta$ then
  – If all other premises α_i are known under some MGU θ, infer the conclusion β under θ
  – If some premises α_i can be matched several ways, then infer each corresponding conclusion

Variable Substitution

- **Renaming**
  - Sentence identical to another, except for variable names
    - e.g. Hates(x,Elizabeth) and Hates(y,Elizabeth)
- **Composition of substitutions**
  - *Substitution with composed unifier identical to the sequence of substitutions with each unifier*

    i.e.

    $$\text{Subst}(\text{Compose}(\theta_1,\theta_2),\alpha) = \text{Subst}(\theta_2\ ,\text{Subst}(\theta_1,\alpha))$$

    - e.g. α = Knows(x,y), θ_1 = {x/John}, θ_2 = {y/ Elizabeth}
      Subst(θ_2,Subst(θ_1 ,α)) = Subst(θ_2,Knows(John,y)) =
      Subst({x/John,y/Elizabeth}, Knows(x,y)) = Knows(John,Elizabeth)

Example of Forward Chaining

Knowledge Base (HNF)

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

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

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

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

Adding Atomic Sentences

Forward-Chain(KB, American(West)):

(5) American(West)

Unifies with a premise of (1), others not known: no new inference.

Forward-Chain(KB, Nation(Nono)):

(6) Nation(Nono) id.

Forward-Chain(KB, Enemy(Nono,America)):

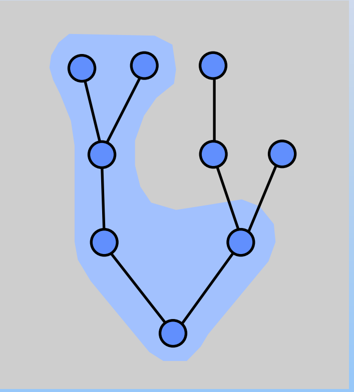
(7) Enemy(Nono,America)

Unifies with (4), with unifier {x/Nono}; call …

Knowledge Base (HNF)	Inferences
(1) American(x) $\wedge$ Weapon(y) $\wedge$ Nation (z) $\wedge$ Hostile(z) $\wedge$ Sells(x,z,y) $\Rightarrow$ Criminal(x) **(2)** Owns(Nono,x) $\wedge$ Missile(x) $\Rightarrow$ Sells(West,Nono,x) **(3)** Missile(x) $\Rightarrow$ Weapon(x) **(4)** Enemy(x,America) $\Rightarrow$ Hostile(x) **(5)** American(West) **(6)** Nation(Nono) **(7)** Enemy(Nono,America)	Forward-Chain(KB, Hostile(Nono)): **(8)** Hostile(Nono) Unifies with (1), no new inference. Forward-Chain(KB, Owns(Nono,M1)): **(9)** Owns(Nono,M1) Unifies with (2), no new inference. Forward-Chain(KB, Missile(M1)): **(10)** Missile(M1) Unifies with (2), with unifier {x/M1}. Other premise known; call Forward-Chain(KB, Sells(West, Nono,M1)):

Knowledge Base (HNF)	Inferences
(1) American(x) Λ Weapon(y) Λ Nation (z) Λ Hostile(z) Λ Sells(x,z,y) ⇒ Criminal(x) **(2)** Owns(Nono,x) Λ Missile(x) ⇒ Sells(West,Nono,x) **(3)** Missile(x) ⇒ Weapon(x) **(4)** Enemy(x,America) ⇒ Hostile(x) **(5)** American(West) **(6)** Nation(Nono) **(7)** Enemy(Nono,America) **(8)** Hostile(Nono) **(9)** Owns(Nono,M1) **(10)** Missile(M1)	**(11)** Sells(West,Nono,M1) Unifies with (1), no new inference. Back to **(10)** Missile(M1): Unifies with (3), w/ unifier {x/M1}; call Forward-Chain(KB, Weapon(M1)): **(12)** Weapon(M1) Unifies with (1), all other premises known, with {x/West, y/M1, z/Nono}; Call Forward-Chain(KB, Criminal(West)): **(13)** **Criminal(West)**

Backward Chaining



- **Idea: checking for causes**
  - ASKing a query β, KB |– β?

- **Pseudo-algorithm**
  - If β already in the KB, proof immediate
  - Find all implications that have β as a conclusion, i.e.
    $\alpha_1 \wedge \alpha_2 \wedge \ldots \wedge \alpha_n \Rightarrow \beta$ then
  - Try and establish all the (qlist) premises [α_i] then infer β

function Backward-Chain (KB, β) **returns** substitutions

return Back-Chain-List(KB, [β], { })

Example of Backward Chaining

Knowledge Base (HNF)	Establishing Premises
(1) American(x) $\wedge$ Weapon(y) $\wedge$ Nation (z) $\wedge$ Hostile(z) $\wedge$ Sells(x,z,y) $\Rightarrow$ Criminal(x) **(2a)** Owns(Nono, M1) **(2b)** Missile(M1) **(3)** Owns(Nono,u) $\wedge$ Missile(u) $\Rightarrow$ Sells(West,Nono,u) **(4)** Missile(v) $\Rightarrow$ Weapon(v) **(5)** Enemy(c,America) $\Rightarrow$ Hostile(c) **(6)** American(West) **(7)** Nation(Nono) **(8)** Enemy(Nono,America) **(9)** Nation(America)	Backward-Chain(KB,Criminal(West)): Call B-Chain-List(KB, [Criminal(West)], {}) Unifies with conclusion of (1); answers = {x/West}; call B-Chain-List(KB, [American(West), Weapon(y), Nation(z), Hostile(z), Sells(West,z,y)], {x/West}): American(West) as **(6)**; Weapon(y) unifies with conclusion of (4) …

Knowledge Base (HNF)	Establishing Premises
(1) American(x) $\wedge$ Weapon(y) $\wedge$ Nation (z) $\wedge$ Hostile(z) $\wedge$ Sells(x,z,y) $\Rightarrow$ Criminal(x)	answers = {x/West, y/v}; call B-Chain-List(KB, [Missile(v)], {x/West, y/v}):
(2a) Owns(Nono, M1)	Missile(v) unifies with **(2b)**;
(2b) Missile(M1)	answers = {x/West, y/M1};
(3) Owns(Nono,u) $\wedge$ Missile(u) $\Rightarrow$ Sells(West,Nono,u)	Weapon(M1) established; back, call
(4) Missile(v) $\Rightarrow$ Weapon(v)	B-Chain-List(KB, [Nation(z)], {x/West, y/M1}):
(5) Enemy(c,America) $\Rightarrow$ Hostile(c)	Nation(z) unifies with **(7)**;
(6) American(West)	answers = {x/West, y/M1, z/Nono};
(7) Nation(Nono)	Hostile(Nono) unifies with conclusion of **(5)**; call …
(8) Enemy(Nono,America)	
(9) Nation(America)	

Knowledge Base (HNF)

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

(2a) Owns(Nono, M1)

(2b) Missile(M1)

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

(4) Missile(v) $\Rightarrow$ Weapon(v)

(5) Enemy(c,America) $\Rightarrow$ Hostile(c)

(6) American(West)

(7) Nation(Nono)

(8) Enemy(Nono,America)

(9) Nation(America)

Establishing Premises

B-Chain-List(KB, [Enemy(Nono, America)], {x/West, y/M1, z/Nono}):

Enemy(Nono,America) as **(8)**;
Hostile(Nono) established; back, call

B-Chain-List(KB, [Sells(West,Nono, M1), {x/West, y/M1, z/Nono}):

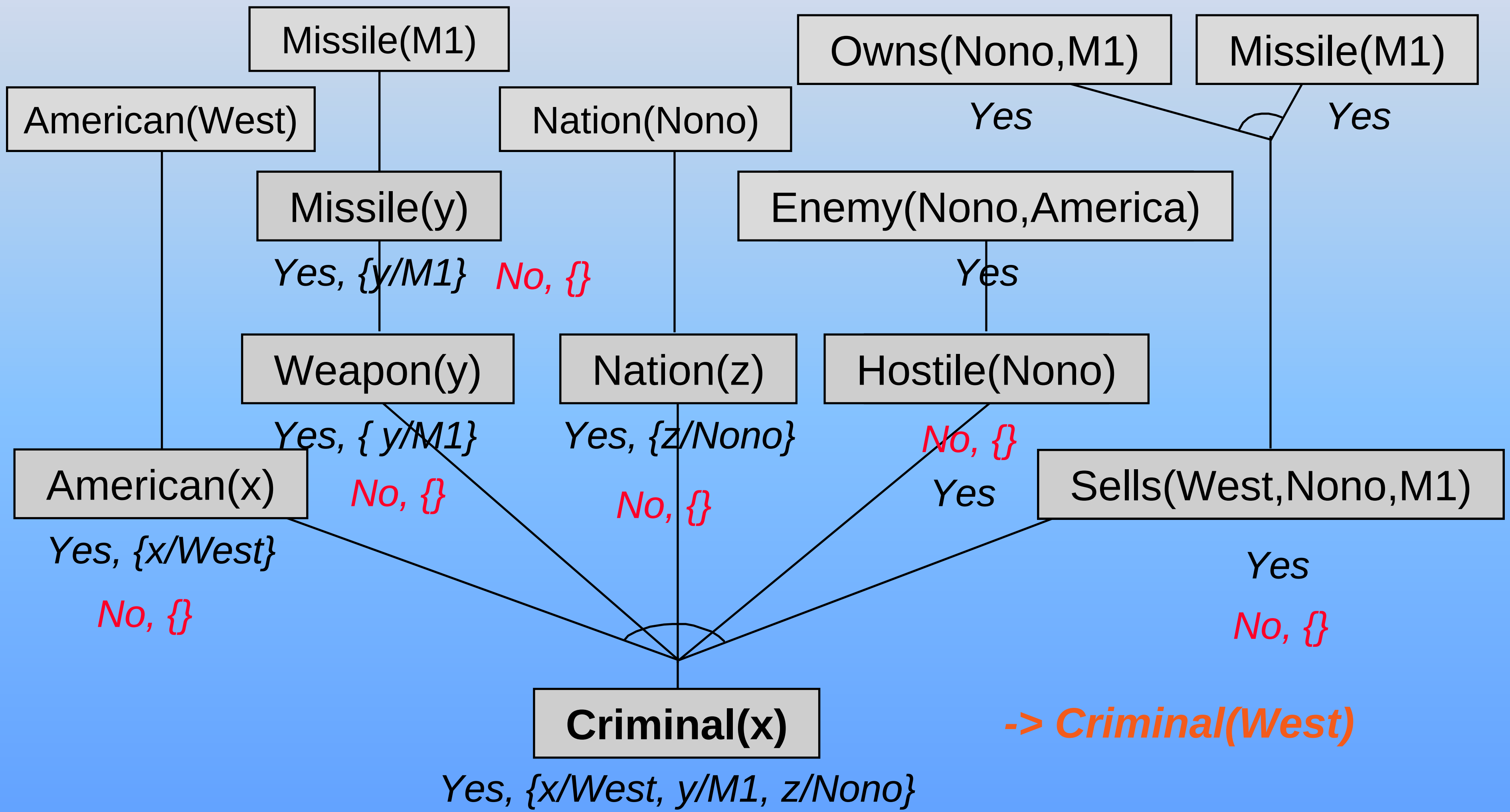
Sells(West,Nono,M1) unifies with **(3)**; call
B-Chain-List(KB,[Owns(Nono,M1)] …)

Owns(Nono,M1) as **(2a)**;

B-Chain-List(KB,[Missile(M1)], …)

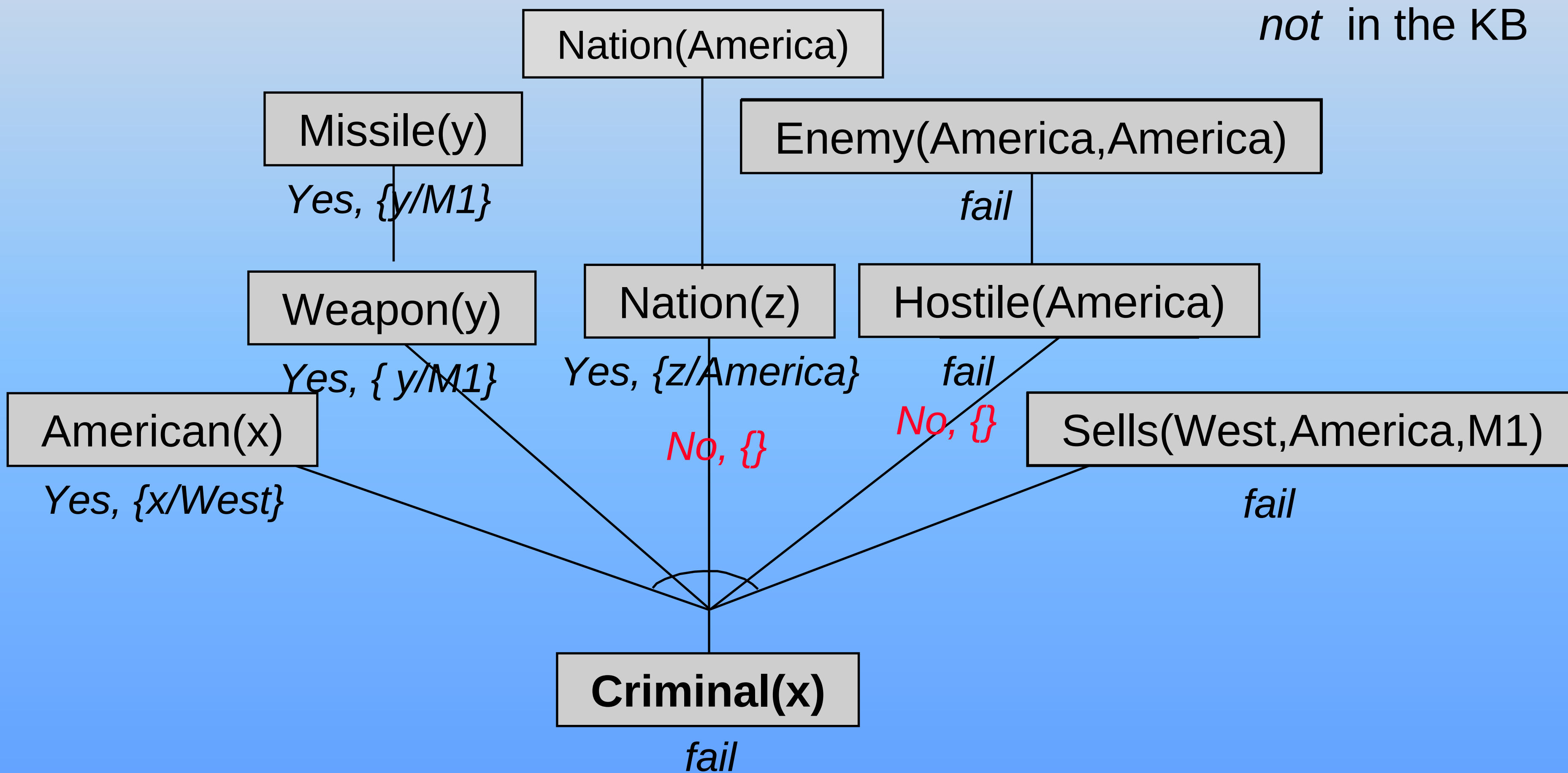
Missile(M1) as **(2b)**. **end**

Chaining as a Proof Tree
(with Success)

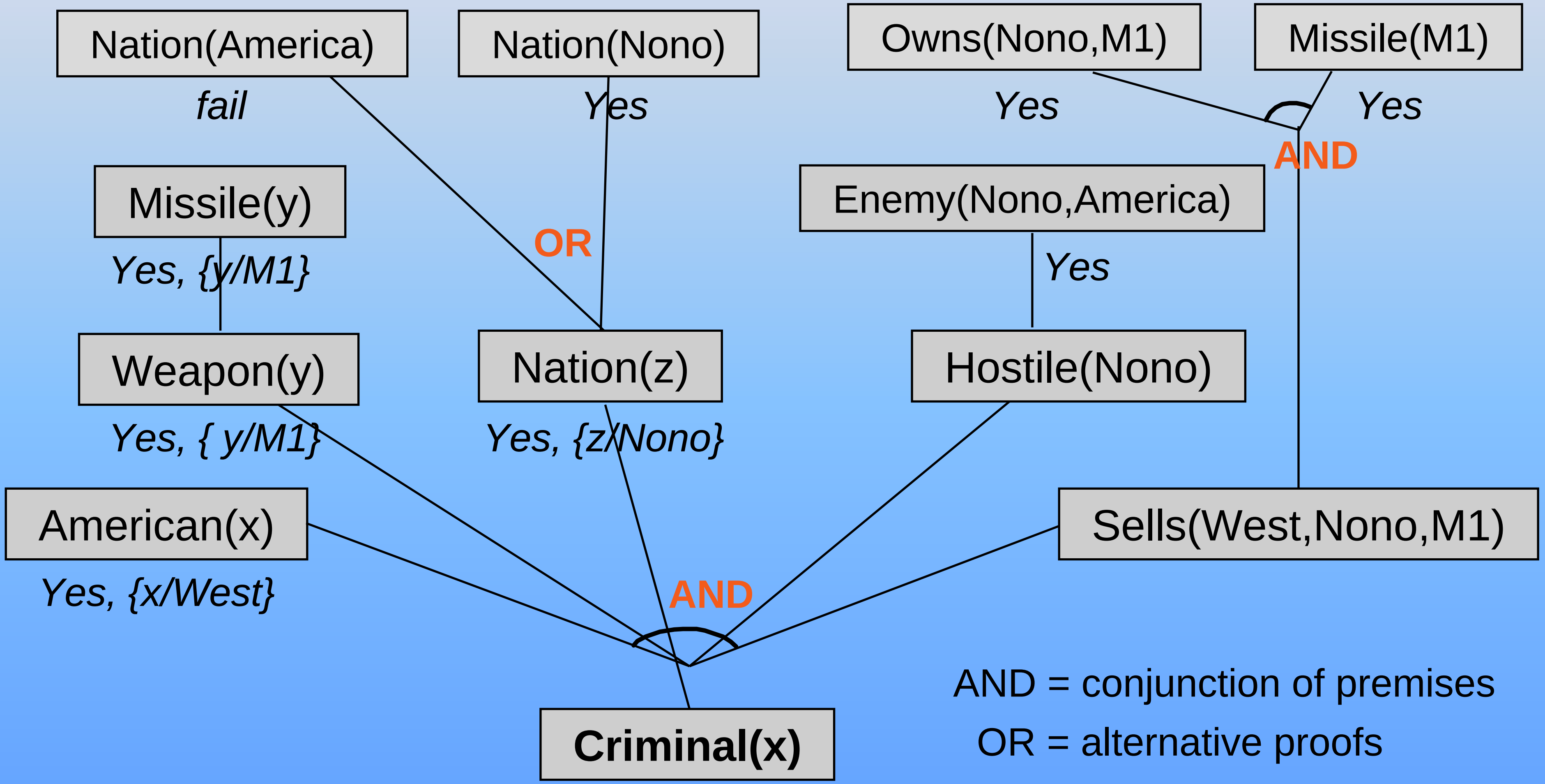


Chaining as a Proof Tree
(with Failure)

Nation(Nono)
not in the KB



And-Or Proof Tree
(with Backtracking)



AND = conjunction of premises

OR = alternative proofs

Completeness

- **GMP is not complete**
  - There exist entailed sentences that GMP cannot prove
    - Some sentences cannot be converted to Horn sentences e.g.

KB: $\forall x \; P(x) \Rightarrow Q(x)$ $\forall x \; Q(x) \Rightarrow S(x)$ entails:

$\forall x \; \neg P(x) \Rightarrow R(x)$ $\forall x \; R(x) \Rightarrow S(x)$ $\forall x \; S(x)$

no Horn form

- **Need for complete inference rules**
  - Completeness theorem (Gödel, 1930)
    - There are complete inference rules for first-order logic
  - Resolution algorithm (Robinson, 1965)

end