

Inference in First Order Logic

Xiaojin Zhu

`jerryzhu@cs.wisc.edu`

**Computer Sciences Department
University of Wisconsin, Madison**

“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 an American.”

The question: is Colonel West a criminal?



The difficulty of FOL inference

- Recall that in propositional logic, inference is easy
 - Enumerate all possibilities (truth tables)
 - Apply sound inference rules on facts
- But in FOL, we have the concepts of variables, relations, and quantification
 - This complicates things quite a bit!
- First let's see how we can convert FOL into propositional logic, and use PL inference
 - ... and why this doesn't work in practice

FOL inference method #1 (doesn't really work...)

CONVERTING FOL TO PL

Convert FOL into PL

Get rid of quantifiers

- Universal Instantiation

$$\forall \mathbf{x} \text{ Eats}(\text{Jim}, \mathbf{x})$$

- This infers (variable substituted with ground term)

$$\text{Eats}(\text{Jim}, \text{Cake})$$

- ...and other million things in KB
- Use substitution to represent this as (g is a ground term)

$$\frac{\forall v \alpha}{\text{SUBST}(\{v/g\}, \alpha)}$$

Convert FOL into PL

- Existential Instantiation

- Variable substituted with a **new** term

$\exists x \text{ Eats}(\text{Jim}, x)$ infer $\text{Eats}(\text{Jim}, \text{NewFood})$

- Why need a **new** term?

- “Jim eats something and Bill eats something.”

$\exists x \exists y \text{ Eats}(\text{Jim}, x) \wedge \text{Eats}(\text{Bill}, y)$

- Use substitution to represent this as

$$\frac{\exists v \alpha}{\text{SUBST}(\{v/k\}, \alpha)}$$

k is a new term

Convert FOL into PL

- Then treat predicates like propositional symbols.
- Example:
 - “Thom is a turtle.”
 1. `turtle(Thom)`
 - “Rob is a rabbit.”
 2. `rabbit(Rob)`
 - “Turtles outlast rabbits.”
 3. $\forall x,y \text{ turtle}(x) \wedge \text{rabbit}(y) \Rightarrow \text{outlasts}(x,y)$
 - **Prove:** “Thom outlasts Rob.”
 - ? `outlasts(Thom,Rob)`

Convert FOL into PL

- Proof:
 - And-Introduction: 1, 2
 - 4. `turtle(Thom) ∧ rabbit(Rob)`
 - Universal Elimination: 3 `{x/Thom, y/Rob}`, among million things in KB
 - 5. `turtle(Thom) ∧ rabbit(Rob) ⇒ outlasts(Thom,Rob)`
 - Modus Ponens: 4, 5
 - 6. `outlasts(Thom,Rob)`
- This doesn't seem to be efficient... We need new, powerful inference rules for FOL
- Observation: why bother with the million things? Find a substitution that makes the rule premise match known facts

FOL inference method #2 (fast but incomplete for FOL)

FORWARD AND BACKWARD CHAINING ON HORN CLAUSES

Substitution

- Substitution list $\theta = \{v_1/t_1, v_2/t_2, \dots, v_n/t_n\}$ means
 - Replace all occurrences of variable v_i with term t_i
 - Substitutions are made in left to right order
- $\text{SUBST}(\theta, \alpha)$ means apply substitutions in θ to sentence α

Unification

- Substitution θ is said to **unify** p and q , if $\text{SUBST}(\theta, p) = \text{SUBST}(\theta, q)$

p	q	θ
<code>turtle(y)</code>	<code>turtle(Thom)</code>	<code>{y/Thom}</code>
<code>loves(Jerry, x)</code>	<code>loves(Jerry, Jing)</code>	<code>{x/Jing}</code>
<code>friends(Bart, x)</code>	<code>friends(x, Milhouse)</code>	<code>{y/Bart, x/Milhouse}</code>
<code>obeys(Ron, x)</code>	<code>obeys(z, mother(z))</code>	<code>{z/Ron, x/mother(Ron)}</code>
<code>eats(y, y)</code>	<code>eats(z, Fish)</code>	<code>{y/z, z/Fish}</code>
<code>sees(JD, x, y)</code>	<code>sees(z, DJ, home(z))</code>	<code>{z/JD, x/DJ, y/home(JD)}</code>
<code>sees(x, id(x), home(JD))</code>	<code>sees(DJ, id(y), home(y))</code>	<code>failure, assuming home(JD) ≠ home(DJ)</code>

Unification

- $\text{UNIFY}(\text{Knows}(\text{John},x), \text{Knows}(y,z)) = ?$
 - $\{y/\text{John}, x/z\}$ gives $\text{Knows}(\text{John},z)$
 - $\{y/\text{John}, x/\text{John}, z/\text{John}\}$ gives $\text{Knows}(\text{John},\text{John})$
- The first one is more general. In fact it's the **most general unifier** (MGU).
- MGU is unique up to renaming of variables
- Cannot unify if a variable itself occurs in the other term: $\text{UNIFY}(x, F(x)) = \text{fail}$
- Cannot unify different ground terms: $\text{UNIFY}(\text{John}, \text{Bill}) = \text{fail}$
- MGU Algorithm on R&N p.278

Generalized Modus Ponens (GMP)

- Unify rule premises with known facts and apply unifier to conclusion
- Rule: $\forall x, y \text{ turtle}(x) \wedge \text{rabbit}(y) \Rightarrow \text{outlasts}(x, y)$
 - Known facts: `turtle(Thom)` and `rabbit(Rob)`
 - Unifier `{x/Thom, y/Rob}`
- Apply unifier to conclusion: `outlasts(Thom, Rob)`

Generalized Modus Ponens

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

(where $SUBST(\theta, p_i') = SUBST(\theta, p_i)$ for all i)

- All variables assumed to be **universally** quantified
- Used with a KB in **Horn normal form**

Generalized Modus Ponens

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

(where $SUBST(\theta, p_i') = SUBST(\theta, p_i)$ for all i)

Example:

$$p_1' = \text{taller}(\text{Larry}, \text{Curly})$$

$$p_2' = \text{taller}(\text{Curly}, \text{Moe})$$

$$p_1 \wedge p_2 \Rightarrow q = \text{taller}(\mathbf{x}, \mathbf{y}) \wedge \text{taller}(\mathbf{y}, \mathbf{z}) \Rightarrow \text{taller}(\mathbf{x}, \mathbf{z})$$

$$\theta = \{\mathbf{x}/\text{Larry}, \mathbf{y}/\text{Curly}, \mathbf{z}/\text{Moe}\}$$

$$SUBST(\theta, q) = \text{taller}(\text{Larry}, \text{Moe})$$

Completeness of FOL inference

- Truth table enumeration is **incomplete** for FOL (table may be infinite in size)
- Generalized Modus Ponens is **sound**
- Generalized Modus Ponens is **incomplete** for FOL (not all sentences can be converted to Horn form)
- Generalized Modus Ponens is complete for FOL KB in HNF
 - Forward chaining
 - Backward chaining

FOL inference example

“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 an American.”

1. $\text{american}(x) \wedge \text{weapon}(y) \wedge \text{sells}(x,y,z) \wedge \text{hostile}(z) \Rightarrow \text{criminal}(x)$

2. $\text{enemy}(\text{Nono}, \text{America})$

$\exists x \text{ owns}(\text{Nono}, x) \wedge \text{missile}(x)$

← *Must be in HNF!*

3. $\text{owns}(\text{Nono}, M)$

← *Use EE and generate*

4. $\text{missile}(M)$

← *2 sentences*

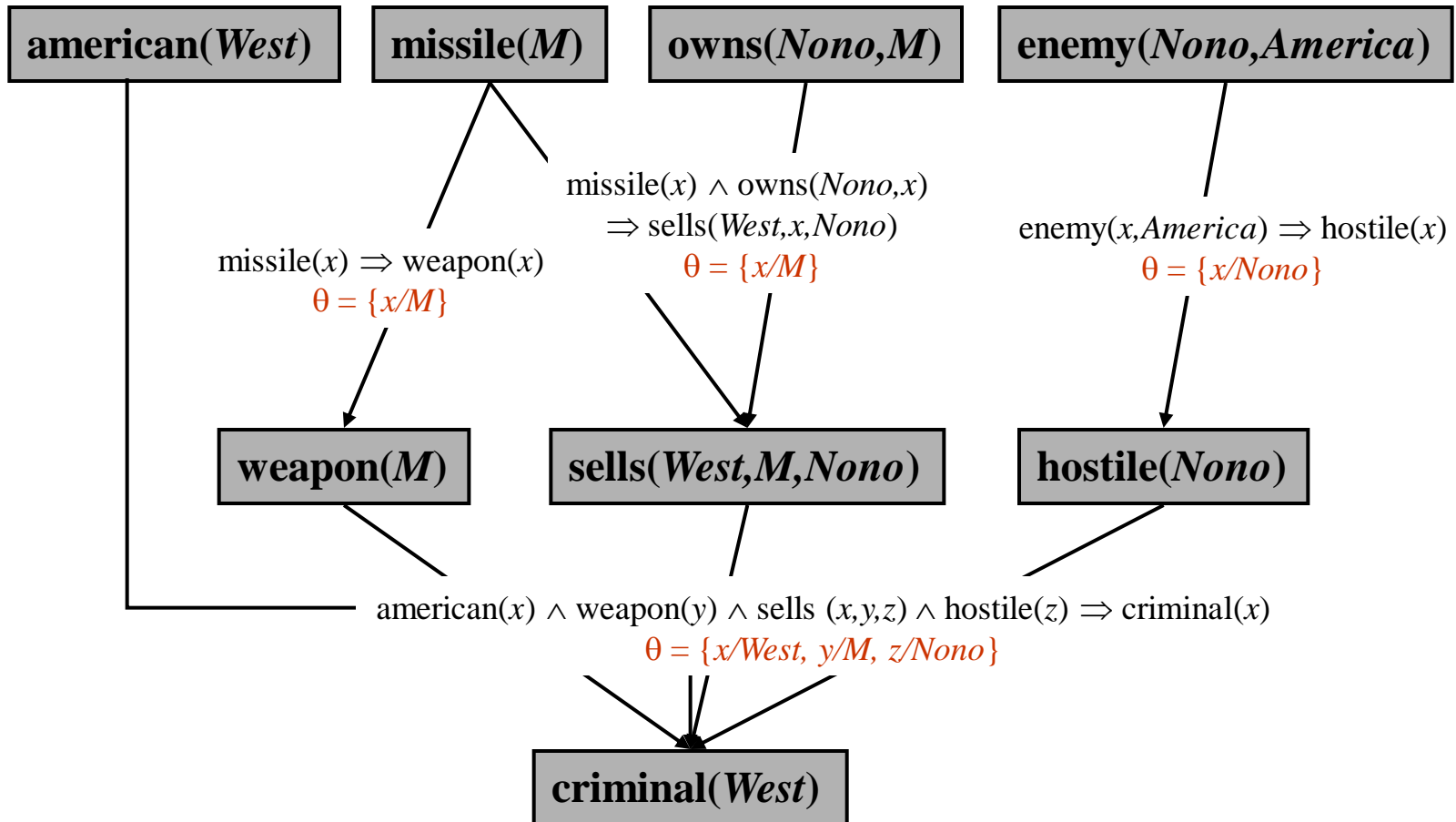
5. $\text{missile}(x) \wedge \text{owns}(\text{Nono}, x) \Rightarrow \text{sells}(\text{West}, x, \text{Nono})$

6. $\text{american}(\text{West})$

7. $\text{enemy}(x, \text{America}) \Rightarrow \text{hostile}(x)$

8. $\text{missile}(x) \Rightarrow \text{weapon}(x)$

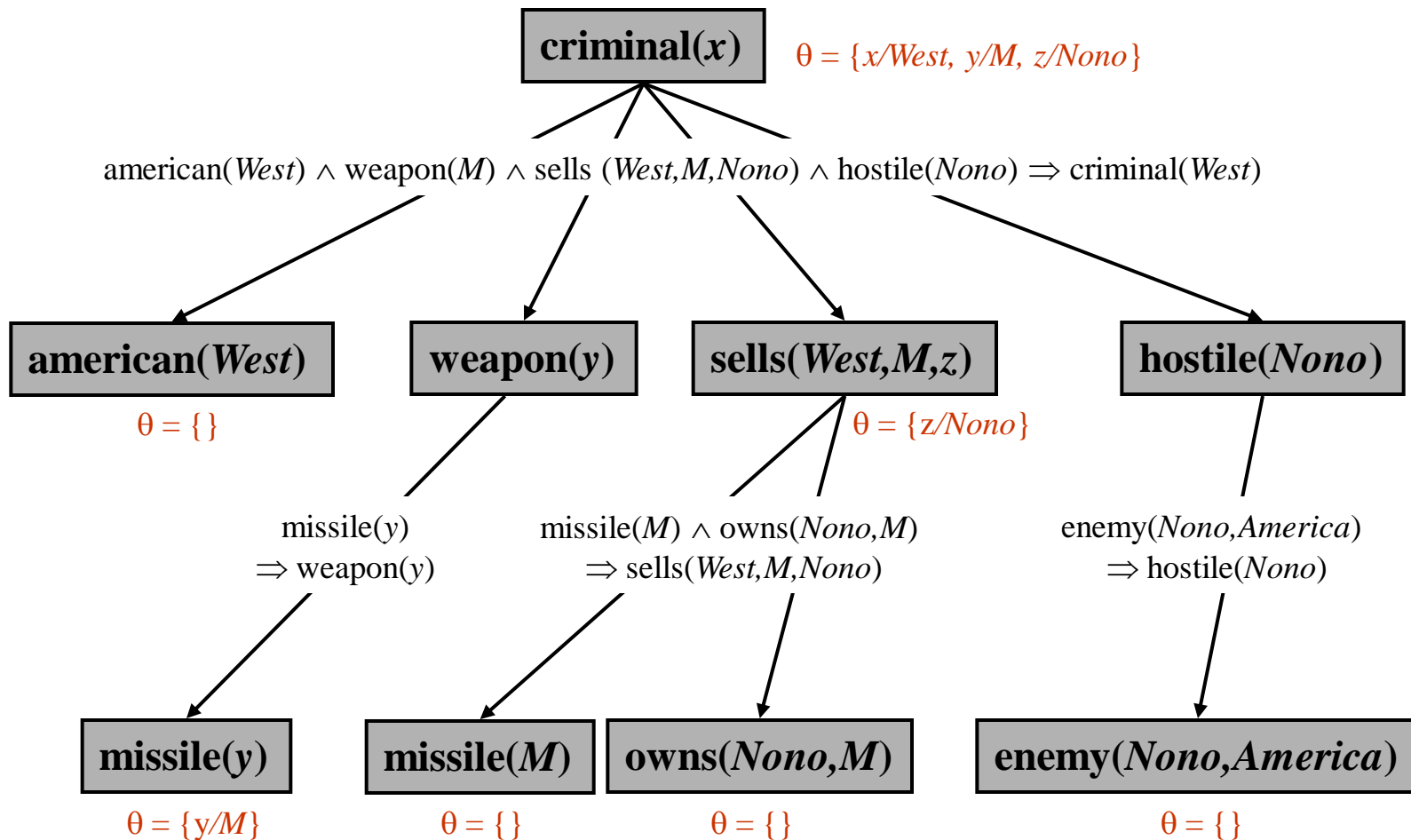
Forward chaining with GMP



Forward chaining with GMP

- The full FC algorithm is on R&N p.282
- Sound and complete for first-order definite clauses
- **Datalog**: FOL clauses with no functions (e.g. the crime KB)
 - FC is efficient for Datalog
- FC typically adds all sentences that can be inferred

Backward chaining with GMP



Backward chaining with GMP

- The full BC algorithm is on R&N p. 288
- BC is depth first search
 - Space is linear in the size of the proof
 - But incomplete even for Datalog
- BC used extensively for logic programming systems (e.g. Prolog)

FOL inference method #3

RESOLUTION

Resolution in FOL

- FC and BC are not complete for FOL
- Resolution (a.k.a. resolution refutation) is refutation-complete but slower.
 - To prove $\mathbf{KB} \models \alpha$, show that $\mathbf{KB} \wedge \neg\alpha$ is unsatisfiable
 - \mathbf{KB} and $\neg\alpha$ need to be in CNF: conjunction of clauses that are disjunction of literals. Any FOL KB can be converted into CNF
 - Repeatedly combines two clauses to make a new one until an empty clause is derived: a contradiction

Resolution

- Resolution in PL

$$\frac{\alpha \vee \beta, \neg \beta \vee \gamma}{\alpha \vee \gamma}$$

equivalently

$$\frac{\neg \alpha \Rightarrow \beta, \beta \Rightarrow \gamma}{\neg \alpha \Rightarrow \gamma}$$

- Generalized Resolution (GR) for FOL
 - Where p_i and q_i are **literals** (a positive or negated predicate with its terms) for all i
 - Where $\text{UNIFY}(p_j, q_k) = \theta$, and q_k is the negation of p_j

$$\frac{p_1 \vee \dots p_j \vee \dots \vee p_m, q_1 \vee \dots q_k \vee \dots \vee q_n}{\text{SUBST}(\theta, p_1 \vee \dots p_{j-1} \vee p_{j+1} \dots \vee p_m \vee q_1 \vee \dots q_{k-1} \vee q_{k+1} \dots \vee q_n)}$$

Resolution

$\text{well-fed}(\text{Me}), \neg\text{well-fed}(\mathbf{x}) \vee \text{happy}(\mathbf{x})$

$\text{SUBST}(\theta, \text{happy}(\mathbf{x}))$

- p_j is $\text{well-fed}(\text{Me})$
 q_k is $\neg\text{well-fed}(\mathbf{x})$
- $\text{UNIFY}(p_j, q_k)$ result in $\theta = \{\mathbf{x}/\text{Me}\}$
 $\text{SUBST}(\theta, \text{happy}(\mathbf{x}))$ result in $\text{happy}(\text{Me})$

Inferred sentence: $\text{happy}(\text{Me})$

- * *GMP is a special case of generalized resolution (for KBs in HNF)*

Resolution refutation

- Going back to the “West is a criminal” example, let’s begin by making sure that all the facts and rules in the KB are in CNF. The following are already so:

$enemy(Nono, America)$ $owns(Nono, M)$
 $missile(M)$ $american(West)$

- The remaining four need to be converted to CNF:

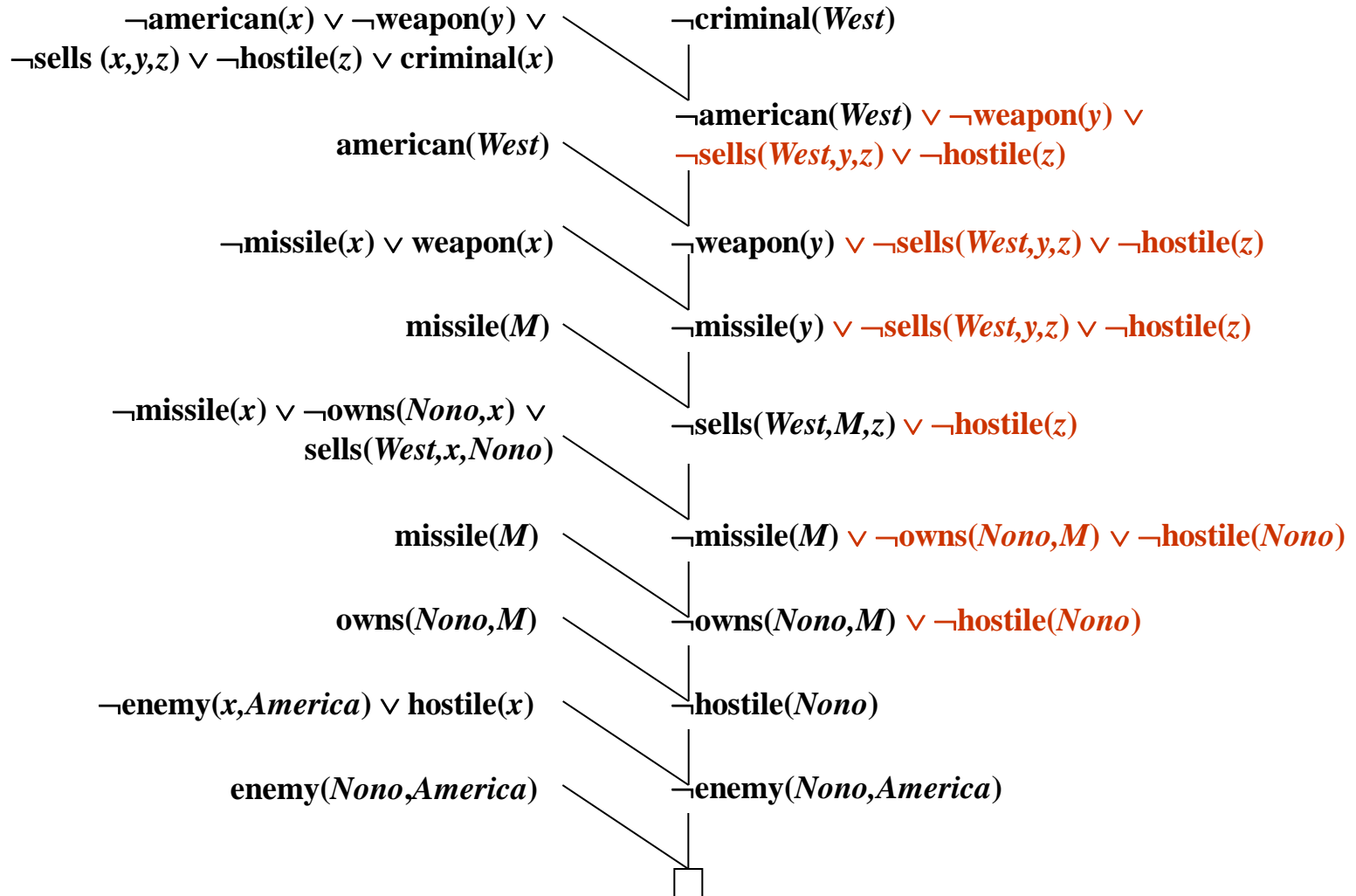
$american(x) \wedge weapon(y) \wedge sells(x, y, z) \wedge hostile(z) \Rightarrow criminal(x)$
 $\neg american(x) \vee \neg weapon(y) \vee \neg sells(x, y, z) \vee \neg hostile(z) \vee criminal(x)$

$missile(x) \wedge owns(Nono, x) \Rightarrow sells(West, x, Nono)$
 $\neg missile(x) \vee \neg owns(Nono, x) \vee sells(West, x, Nono)$

$enemy(x, America) \Rightarrow hostile(x)$ $missile(x) \Rightarrow weapon(x)$
 $\neg enemy(x, America) \vee hostile(x)$ $\neg missile(x) \vee weapon(x)$

- And we need to negate our query: $\neg criminal(West)$

Resolution refutation



Converting FOL to CNF

- Conjunctive normal form (CNF): disjunction of literals (may be negated)

- Convert FOL to CNF in 8 easy steps!

1. $P \Leftrightarrow Q$ becomes $P \Rightarrow Q \wedge Q \Rightarrow P$

2. $P \Rightarrow Q$ becomes $\neg P \vee Q$

3. Reduce scope of \neg to single literals:

$\neg\neg P$ becomes P (DNE)

$\neg(P \vee Q)$ becomes $\neg P \wedge \neg Q$ (de Morgan's)

$\neg(P \wedge Q)$ becomes $\neg P \vee \neg Q$ (de Morgan's)

$\neg\forall x P$ becomes $\exists x \neg P$

$\neg\exists x P$ becomes $\forall x \neg P$

4. Standardize variables apart:

- Each quantifier must have a unique variable name
- Avoids confusion in steps 5 and 6
- e.g. $[\forall x P] \vee [\exists x Q]$ becomes $\forall x P \vee \exists y Q$

Converting FOL to CNF

5. Eliminate existential quantifiers (Skolemize):

$\exists \mathbf{x} P(\mathbf{x})$ becomes $P(\mathbf{K})$ (EE)

\mathbf{K} is some new constant (Skolem constant)

- e.g. $\forall \mathbf{x} \exists \mathbf{y} P(\mathbf{x}, \mathbf{y})$ becomes $\forall \mathbf{x} P(\mathbf{x}, F(\mathbf{x}))$
 $F()$ must be a new function (Skolem function) with arguments that are all enclosing universally quantified variables

- Everyone has a name.

$\forall \mathbf{x} \text{ person}(\mathbf{x}) \Rightarrow \exists \mathbf{y} \text{ name}(\mathbf{y}) \wedge \text{has}(\mathbf{x}, \mathbf{y})$

wrong: $\forall \mathbf{x} \text{ person}(\mathbf{x}) \Rightarrow \text{name}(\mathbf{K}) \wedge \text{has}(\mathbf{x}, \mathbf{K})$

Everyone has the same name K!!

We want everyone to have a name based on who they are

right: $\forall \mathbf{x} \text{ person}(\mathbf{x}) \Rightarrow \text{name}(F(\mathbf{x})) \wedge \text{has}(\mathbf{x}, F(\mathbf{x}))$

Converting FOL to CNF

6. Hide universal quantifiers:

- All variables are only universally quantified after step 5
- e.g. $\forall x P(x) \vee \forall y Q(y)$ becomes $P(x) \vee Q(y)$
- All variables in KB will be assumed to be universally quantified

7. Distribute \vee over \wedge :

$$(P \wedge Q) \vee R \text{ becomes } (P \vee R) \wedge (Q \vee R)$$

8. Group conjunctions/disjunctions together:

$$(P \wedge Q) \wedge R \text{ becomes } (P \wedge Q \wedge R)$$

$$(P \vee Q) \vee R \text{ becomes } (P \vee Q \vee R)$$

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&2. Eliminate biconditionals and implications

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

3. Reduce scope of \neg to single literals

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

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

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

CNF example

4. Standardize variables apart

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

5. Eliminate Existentials by skolemizing

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

6. Drop universals

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

7&8. Distribute \vee over \wedge , group conjunctions/disjunctions:

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

Example: Hoofers Club

- Tony, Shi-Kuo 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?

Translation into FOL Sentences

- $S(x)$: x is a skier,
 - $M(x)$: x is a mountain climber
 - $L(x,y)$: x likes y , where the domain of the first variable is Hoofers Club members, and the domain of the second variable is snow and rain
1. $(\forall x) S(x) \vee M(x)$
 2. $\sim(\exists x) M(x) \wedge L(x, \text{Rain})$
 3. $(\forall x) S(x) \Rightarrow L(x, \text{Snow})$
 4. $(\forall y) L(\text{Ellen}, y) \Leftrightarrow \sim L(\text{Tony}, y)$
 5. $L(\text{Tony}, \text{Rain})$
 6. $L(\text{Tony}, \text{Snow})$
 7. **Query:** $(\exists x) M(x) \wedge \sim S(x)$
 8. **Negation of the Query:** $\sim(\exists x) M(x) \wedge \sim S(x)$

Convert to CNF

1. $S(x_1) \vee M(x_1)$
2. $\sim M(x_2) \vee \sim L(x_2, \text{Rain})$
3. $\sim S(x_3) \vee L(x_3, \text{Snow})$
4. $\sim L(\text{Tony}, x_4) \vee \sim L(\text{Ellen}, x_4)$
5. $L(\text{Tony}, x_5) \vee L(\text{Ellen}, x_5)$
6. $L(\text{Tony}, \text{Rain})$
7. $L(\text{Tony}, \text{Snow})$
8. **Negation of the Query:** $\sim M(x_7) \vee S(x_7)$

Resolution Refutation Proof

Clause 1	Clause 2	Resolvent	MGU (i.e., Theta)
8	1	9. $S(x_1)$	$\{x_7/x_1\}$
9	3	10. $L(x_1, \text{Snow})$	$\{x_3/x_1\}$
10	4	11. $\sim L(\text{Tony}, \text{Snow})$	$\{x_4/\text{Snow}, x_1/\text{Ellen}\}$
11	7	12. False	$\{\}$

Answer extraction

- Add a **special answer literal** to the negated goal
- Resolution generates an answer whenever a clause is generated containing a **single answer literal**

Clause 1	Clause 2	Resolvent	MGU (i.e., Theta)
$\sim M(x7) \vee S(x7) \vee (M(x7) \wedge \sim S(x7))$	1	9. $S(x1) \vee (M(x1) \wedge \sim S(x1))$	$\{x7/x1\}$
9	3	10. $L(x1, \text{Snow}) \vee (M(x1) \wedge \sim S(x1))$	$\{x3/x1\}$
10	4	11. $\sim L(\text{Tony}, \text{Snow}) \vee (M(\text{Ellen}) \wedge \sim S(\text{Ellen}))$	$\{x4/\text{Snow}, x1/\text{Ellen}\}$
11	7	12. $M(\text{Ellen}) \wedge \sim S(\text{Ellen})$	$\{\}$

Answer to the Query: Ellen!

Summary

- FOL is a very expressive language, but difficult to perform inference with
 - One inference method is removing all variables / quantifiers (**propositionalizing**), which is slow
 - We can also use **unification** to identify appropriate substitutions with **generalized Modus Ponens**, which is complete for HNF but not general FOL
 - The **forward chaining** and **backward chaining** algorithms use GMP to KBs in HNF
 - Generalized **resolution** inference is complete for all of FOL. KB must be in CNF. Use refutation