Logic in AI

Contd.

Agenda: Inference in FIRST ORDER LOGIC

Rules of Inference

1. Modus ponens

$$\frac{A \to B, A}{B}$$

$$\frac{A \qquad \neg A \lor B}{B}$$

2. Modus tolens

$$\frac{A \to B, \neg B}{\neg A}$$

$$\frac{\neg A \lor B \qquad \neg B}{\neg A}$$

3. Resolution

$$\frac{A \lor B, \neg B \lor C}{A \lor C}$$

$$\frac{A \lor B \qquad \neg B \lor C}{A \lor C}$$

Rules of Inference

- Rules of Inference introduced in Propositional Logic can be also used in Predicate Logic
  - One would need to learn **how to deal with formulas that contain variables**.

1. Universal Specialization – Universal Instantiation
2. Existential Instantiation
3. Existential Generalization
4. Universal Generalization - Universal Introduction

Universal Specialization

- $$\frac{\forall x\ P(x)}{P(C)}$$

  where C is *any* constant symbol.

Universal Specialization is also referred to as Universal Instantiation.

- Example:
  - $\forall x\ eats(Zen, x) \rightarrow eats(Zen, IceCream)$

The **variable symbol can be replaced by any ground term**, i.e., any constant symbol or function symbol applied to ground terms only.

Existential Instantiation

- $$\frac{\exists x\, P(x)}{P(A)}$$

  Where A is a *brand-new* constant symbol.

- Example:
  - ∃x likes(Zen, x) → likes(Zen, Stuff)

Also known as skolemization; constant is a **skolem constant**. Convenient to reason about the unknown object, rather than the existential quantifier.

Existential Generalization

- $$\frac{P(c)}{\exists x\ P(x)}$$

- Example
  - eats(Zen, IceCream) $\rightarrow \exists x$ eats(Zen, x)

All instances of the given **constant symbol are replaced by the new variable symbol**. Note that the variable symbol cannot already exist anywhere in the expression.

Universal Generalization

- $$\frac{P(c)}{\forall x\, P(x)}$$

  If P(c) must be true, and we have assumed nothing about c, then $\forall$ x P(x) is true.

  Universal generalization is the rule of inference that states that $\forall$ xP(x) is true, given the premise that P(c) is true for all elements c in the domain.

  Universal generalization is used when we show that $\forall$ xP(x) is true by taking an arbitrary element c from the domain and showing that P(c) is true.

Unification

- Example

$$\neg W1(x) \vee W2(x) \qquad\qquad W1(A)$$

$$W2(A)$$

1. $\forall x\, [\, W1(x) \rightarrow W2(x)]$
2. $W1(A)$

For *universal specialization* to produce W2(A) from 1 and 2 above; it is necessary to find the substitution A/x.

- Finding **substitutions of terms for variables** to make expressions identical is an extremely important process and is called **unification**.
- The **set of substitutions is called a unifier**.

Unification

- The terms of an expression can be variable symbols, constant symbols or functional expressions, the latter consisting of function symbols and terms.

- A **substitution instance** of an expression is obtained by substituting terms for variables in that expression.

Example: Four instances of substitution of P[x, f(y), B].

P[z, f(w), B]	Alphabetic variant
P[x, f(A), B]	
P[g(z),f(A),B]	
P[C, f(A), B]	Ground Instance

The last of the four instances shown is called a ground instance, since none of the terms in the literal contains variables.

Ques:- Describe Unification Algorithm with an example.

Unifications means making expressions looks identical.

↳ Can be done with the process of Substitution.

Simple Eg:- $p(x, F(y))$ — ① $p(a, F(g(z)))$ — ②

↳ ① and ② are identical if x is replace with a and y is replace with $g(z)$

$p(a, F(g(z)))$ $[a/x, g(z)/y]$

Substitution Set.

Unification Condⁿ:-

① Predicate Symbol must be Same.

② No. of arguments in both expressions must be identical. → Same

③ If two Similar Variables present in Same expression, then Unification fails.

$p(\ldots)$
$p(\ldots)$] ✗

Numericals

ALGORITHM:-

Unify (A1, A2)

① if A1 or A2 is Variable / Constant

↳ if A1 and A2 are identical return NIL

↳ Else if A1 occurs in A2 return fail

↳ Else return $\boxed{A2/A1}$

↳ Check for A2 in A1

↳ fail if A2 occurs in A1

↳ Else return $\boxed{A1/A2}$

② if Predicate not Same] fail

③ if diff. arguments] fail

④ Else SUBST to NIL

⑤ Loop

⑥ Return SUBST.

Ques:- $\underbrace{Q(a, g(x,a), f(y))}_{A_1}$, $\underbrace{Q(a, g(f(b),a), x\}}_{A_2}$

Substitute x with $f(b)$ $[f(b)/x]$

$Q(a, g(f(b),a), f(y))$, $Q(a, g(f(b),a), f(b)\}$

Substitute (b/y) [y is substituted with b].

$[Q(a, g(f(b),a), f(b)), Q(a, g(f(b),a), f(b)]$

Unified Successfully.

Resolution

Resolution Refutation

Basic steps for proving a conclusion S given premises

Premise_1 , …, Premise_n (all expressed in FOL):

1. Convert all sentences to **Clausal Normal Form** (CNF)
2. Negate conclusion T and convert result to CNF
3. Add negated conclusion T to the premise clauses
4. Repeat until **contradiction** or no progress is made:
    a. Select 2 clauses (call them parent clauses)
    b. Resolve them together, performing all required unifications
    c. If resolvent is the empty clause, a contradiction has been found (i.e., T follows from the premises)
    d. If not, add resolvent to the premises.

If we succeed in Step 4, we have proved the conclusion.

Converting to Clausal Form

Step – I : Eliminate Implication Symbols

Example $\forall x\ [\ W1(x) \rightarrow [\forall y\ [W2(y) \rightarrow W3(f(x,y))]]]$

$\forall x\ [\neg W1(x) \vee [\forall y\ [\neg W2(y) \vee W3(f(x,y))]]]$

All occurrences of the $\rightarrow$ symbol in a well-formed formula are eliminated by making the substitution

$[\neg X \vee Y]$ for $[X \rightarrow Y]$

Step – II : Reduce scopes of Negation Symbols

Example $\neg \forall y\ [Q(x,y) \rightarrow P(y)]$

$\exists y\ [Q(x,y) \wedge \neg P(y)]$

$\exists y \neg [Q(x,y) \rightarrow P(y)]$
$\exists y \neg [\neg Q(x,y) \vee P(y)]$
$\exists y\ [Q(x,y) \wedge \neg P(y)]$

We want each negation symbol to apply to at most one atomic formula. Achieve this by repeated use of De Morgan's Laws and other equivalences.

Converting to Clausal Form

Step – III : Standardize variables

Example $\forall x\, [\, W1(x) \rightarrow \exists x\, W2(x)]$

$\forall x\, [\, W1(x) \rightarrow \exists y\, W2(y)]$

The scope of a variable is the sentence to which the quantifier syntactically applies.

Within the scope of any quantifier, a variable bound by the quantifier is a dummy variable. It can be uniformly replaced by any other (non-occurring) variable throughout the scope of the quantifier without changing the truth value of the well-formed formula.

Standardizing variable refers to renaming the dummy variables to ensure that each quantifier has its own unique dummy variable.

Converting to Clausal Form

Step – IV : Eliminate Existential Quantifiers

Example 1. $\forall y\ [\exists x\ P(x,y)]$

$\forall y\ [P(g(y),y)]$

2. $\exists x\ P(x)$

$P(A)$

In Example 2 the existential quantifier being eliminated is not within the scope of the universal quantifier. We use a Skolem function of no arguments.

Explicitly state a constant A, used to refer to the entity that we know `exists'. Such a constant is called a **Skolem constant**.

It is important that A be a new constant symbol; one not used in other formulas to refer to known entities.

Converting to Clausal Form

Step – V : Convert to Prenex Form

There are no remaining existential quantifier; Each Universal quantifier has its own variable.

Move all universal quantifiers to front of well-formed formula; scope of each quantifier is the entirety of the formula.

The resulting well-formed formula is in **prenex form**.

The prenex form consists of a **string of quantifiers called prefix** followed by a quantifier-free formula called the matrix.

$$\forall x \; \forall y \; \forall z \; \forall w \; \ldots\ldots \; [P(x,y)Q(g(z),y)R(w) \; \ldots\ldots]$$

Converting to Clausal Form

Step – VI : Put in Conjunctive Normal Form

Example $P \vee (Q \wedge R)$

Conjunction of a finite set of disjunctions of literals

$(P \vee Q) \wedge (P \vee R)$

Any matrix may be written as the **conjunction of a finite set of disjunction of literals**. Such a matrix is said to be in **conjunctive normal form**.

Recall that a quantifier-free formula called the matrix.

May put any matrix into a conjunctive normal form by repeatedly using one of the distributive rules as highlighted above.

Converting to Clausal Form

Step – VII : Eliminate Universal Quantifiers

All variables remaining at this stage are universally quantified; bound. Eliminate the explicit reference.

Left with a matrix in Conjunctive Normal Form.

Step – VIII : Eliminate ∧ Symbols

Example $P \wedge (Q \vee R)$

1. P
2. $Q \vee R$

Eliminate the explicit reference of AND. Result of repeated replacement is to obtain a finite set of well-formed formula, each of which is a disjunction of literals.

Step – IX : Rename variables

Variables symbols may be renamed so that no variable symbol appears in more than one clause; Standardizing variables apart.

An Illustrative Example

Example

1. Whoever can read is literate.
2. Dolphins are not literate.
3. Some dolphins are intelligent.

Prove: Some who are intelligent cannot read.

Predicates -	$R(x)$	:	x can read.
	$L(x)$	:	x is literate.
	$D(x)$	:	x is a dolphin.
	$I(x)$	:	x is intelligent.

An Illustrative Example

1. Whoever can read is literate.

   $\forall x[\ R(x) \rightarrow L(x)]$ ✓C1. $\neg R(x) \vee L(x)$

2. Dolphins are not literate.

   $\forall x[D(x) \rightarrow \neg L(x)]$ ✓C2. $\neg D(y) \vee \neg L(y)$

3. Some dolphins are intelligent.

   $\exists x[D(x) \wedge I(x)]$ C3. $D(A)$

   C4. $I(A)$

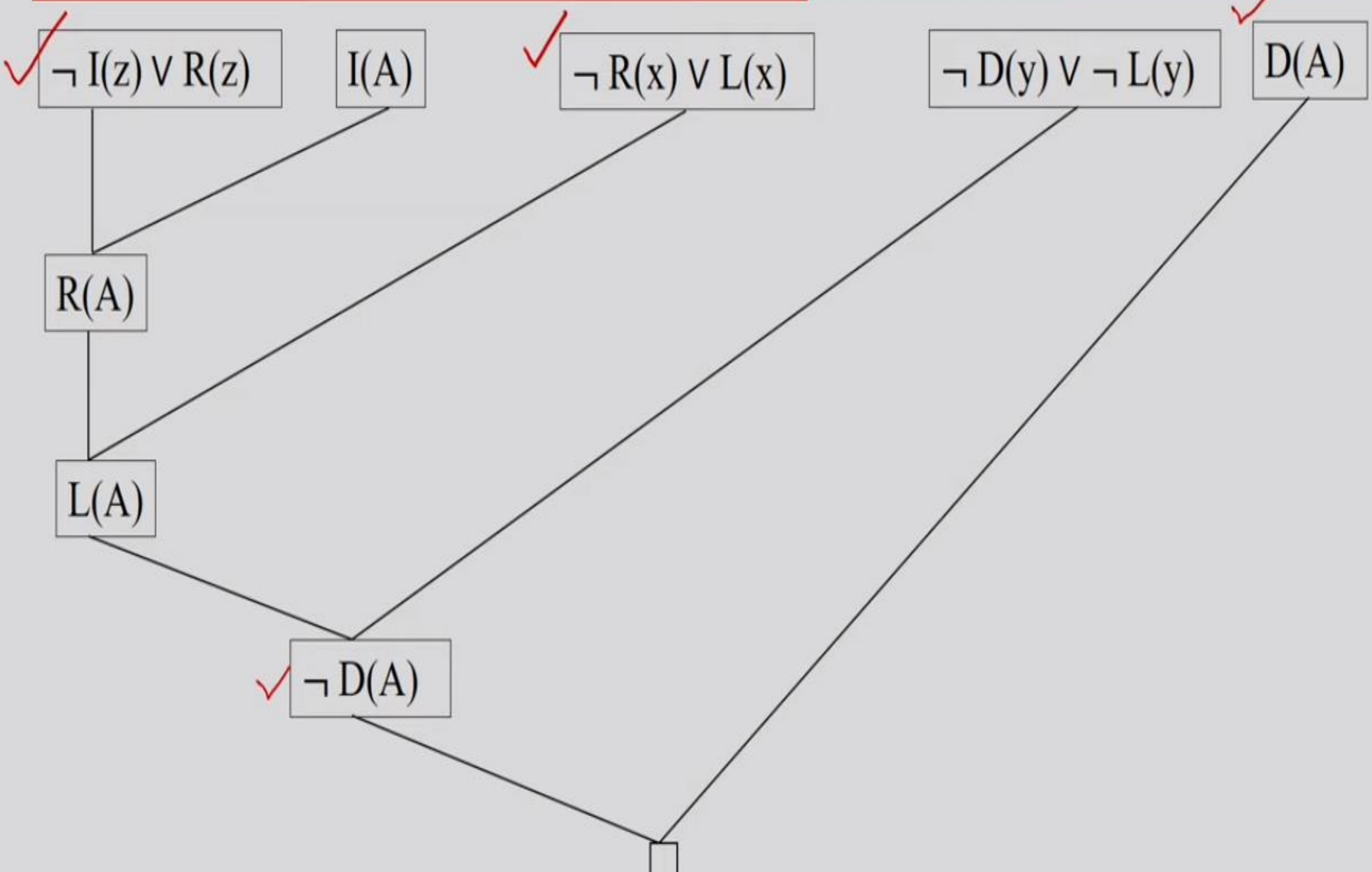
Prove   Some who are intelligent cannot read.

$\exists x[I(x) \wedge \neg R(x)]$

Negation $\neg \exists x[I(x) \wedge \neg R(x)]$

$\forall x[\neg I(x) \vee R(x)]$ ✓C5. $\neg I(z) \vee R(z)$

An Illustrative Example



Resolution Refutation Proofs

Example 1

Premise

1. If a course is easy, some students are happy.
2. If a course has a final exam, no students are happy.

Prove: If a course has a final exam, the course is not easy.

Recall that a predicate is an assertion that some property or relationship holds for one or more arguments,

Predicates

easy(x) : Course `x' is a easy.

happy(x) : Student `x' is happy.

final(x) : Course `x' has a final exam.

Date

(1) If course is easy, some students are happy

$\forall x_1 [easy(x_1) \rightarrow \exists y_1 happy(y_1)]$

$\forall x_1 (\sim easy(x_1) \vee \exists y_1 happy(y_1))$

$\forall x_1 (\sim easy(x_1) \vee \exists y_1 happy(y_1))$

C1: $\sim easy(x_1) \vee happy(y_2) \quad (y_2/y_1)$.

(2) If course has a final exam, no students are happy

$\forall x [final(x) \rightarrow \sim \exists y Happy(y)]$

$\forall x [final(x) \rightarrow \forall y \sim happy(y)]$

$\forall x \forall y [final(x) \rightarrow \sim happy(y)]$

C2: $\sim final(x) \vee \sim happy(y)$

goal:

If course has final exam, then course is not easy.

$\forall x_3 (final(x_3) \longrightarrow \neg easy(x_3))$

$\forall x_2 (\neg final(x_3) \vee \neg easy(x_3))$

Goal: $\neg final(x_3) \vee \neg easy(x_3)$

Negate the goal

$\neg(\neg final(x_3) \vee \neg easy(x_3))$

$final(x_3) \wedge easy(x_3)$

C_3 : $final(x_2)$, C_4 : $easy(x_3)$

C_1 : $\neg easy(x_1) \vee happy(y_2)$

C_2 : $\neg final(x) \vee \neg happy(y)$.

Resolution trace

C_1 C_2 C_3 C_4

x/x_3 x_1/x_4

$\neg happy(y)$ $happy(y_2)$

y/y_2

$\square$

Resolution Refutation Proofs

Example 2

Premise

1. The father of someone or the mother of someone is an ancestor of that person.
2. An ancestor of someone's ancestor is also the ancestor of that person.
3. Jesse is the father of David.
4. David is the ancestor of Mary
5. Mary is the mother of Jesus

Prove: Jesse is an ancestor of Jesus.

Resolution Refutation Proofs

Predicates

A predicate is an assertion that some property or relationship holds for one or more arguments,

1. father(x,y) : `x' is the father of `y'.
2. mother(x,y) : `x' is the mother of `y'.
3. ancestor(x,y) : `x' is the ancestor of `y'.

Constants

1. Jesse
2. David
3. Mary
4. Jesus

A constant is a symbolic name for a real-world person, object or event.

Resolution Refutation Proofs

1. The father of someone or the mother of someone is an ancestor of that person

$\forall x \forall y[(father(x,y) \vee mother(x,y)) \rightarrow ancestor(x,y)]$

$\forall x \forall y[\neg (father(x,y) \vee mother(x,y)) \vee ancestor(x,y)]$

$\forall x \forall y[(\neg father(x,y) \wedge \neg mother(x,y)) \vee ancestor(x,y)]$

$\forall x \forall y[(\neg father(x,y) \vee ancestor(x,y)) \wedge (\neg mother(x,y)) \vee ancestor(x,y)]$

C1. $[\neg father(x_1,y_1) \vee ancestor(x_1,y_1)]$

C2. $[\neg mother(x_2,y_2) \vee ancestor(x_2,y_2)]$

2. An ancestor of someone's ancestor is also an ancestor of that person.

$\forall r \forall s \forall t[(ancestor(r,s) \wedge ancestor(s,t)) \rightarrow ancestor(r,t)]$

$\forall r \forall s \forall t[\neg (ancestor(r,s) \wedge ancestor(s,t)) \vee ancestor(r,t)]$

$\forall r \forall s \forall t[\neg ancestor(r,s) \vee \neg ancestor(s,t) \vee ancestor(r,t)]$

C3. $[\neg ancestor(r,s) \vee \neg ancestor(s,t) \vee ancestor(r,t)]$

Resolution Refutation Proofs

3. Jesse is the father of David.

   C4. father(Jesse, David).

4. David is an ancestor of Mary.

   C5. ancestor(David, Mary).

5. Mary is the mother of Jesus.

   C6. mother(Mary, Jesus).

Φ: Jesse is an ancestor of Jesus.

ancestor(Jesse, Jesus).

Negate the goal: ¬ ancestor(Jesse, Jesus).

C7. ¬ ancestor(Jesse, Jesus).

Resolution Refutation Proofs

Resolution Trace

1.	$[\neg\, father(x_1,y_1) \vee ancestor(x_1,y_1)]$	C1
2.	$[\neg\, mother(x_2,y_2) \vee ancestor(x_2,y_2)]$	C2
3.	$[\neg\, ancestor(r,s) \vee \neg\, ancestor(s,t) \vee ancestor(r,t)]$	C3
4.	father(Jesse, David)	C4
5.	ancestor(David, Mary)	C5
6.	mother(Mary, Jesus)	C6
7.	$\neg$ ancestor(Jesse, Jesus).	C7
8.	$\neg$ ancestor(Jesse,s) $\vee$ $\neg$ ancestor(s,Jesus)	3,7 {Jesse/r, Jesus/t}

Resolution Refutation Proofs

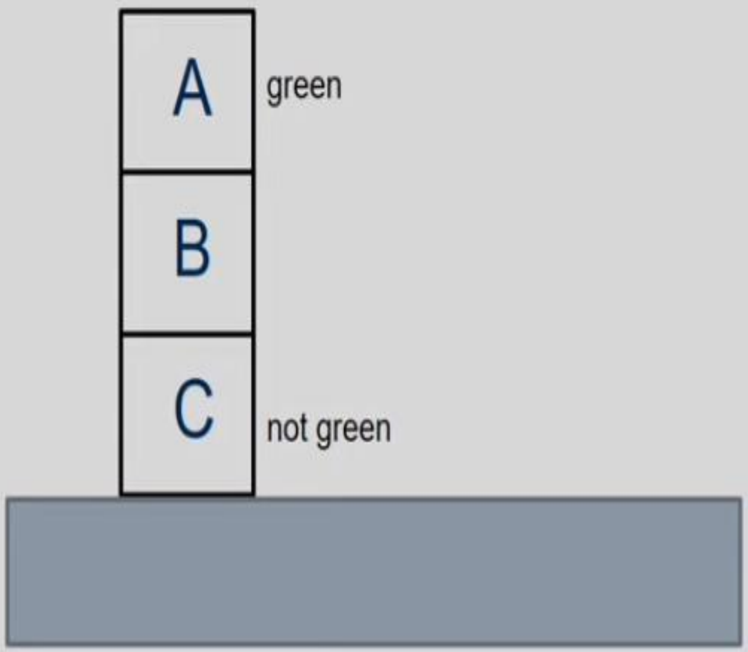
Resolution Trace

1.	$[\neg \text{father}(x_1,y_1) \vee \text{ancestor}(x_1,y_1)]$	C1	
2.	$[\neg \text{mother}(x_2,y_2) \vee \text{ancestor}(x_2,y_2)]$	C2	
3.	$[\neg \text{ancestor}(r,s) \vee \neg \text{ancestor}(s,t) \vee \text{ancestor}(r,t)]$	C3	
4.	father(Jesse, David)	C4	
5.	ancestor(David, Mary)	C5	
6.	mother(Mary, Jesus)	C6	
7.	$\neg$ ancestor(Jesse, Jesus).	C7	
8.	$\neg$ ancestor(Jesse,s) $\vee$ $\neg$ ancestor(s,Jesus)	3,7	{Jesse/r, Jesus/t}
9.	$\neg$ father(Jesse,s) $\vee$ $\neg$ ancestor(s,Jesus)	1,8	{Jesse/x_1, s/y_1}
10.	$\neg$ ancestor(David,Jesus)	4,9	{David/s}
11.	$\neg$ ancestor(David,s) $\vee$ $\neg$ ancestor(s,Jesus)	3,10	{David/r, Jesus/t}
12.	$\neg$ ancestor(Mary,Jesus)	5,11	{Mary/s}
13.	$\neg$ mother(Mary,Jesus)	2,12	{Mary/x_2, Jesus/y_2}
14.	$\square$	6,13	

Resolution Refutation Proofs

Example 3

Suppose there are three coloured blocks stacked as shown, where the top one is \`green' and the bottom is \`not green'. Is there a \`green' block on top of a \`non-green' block?

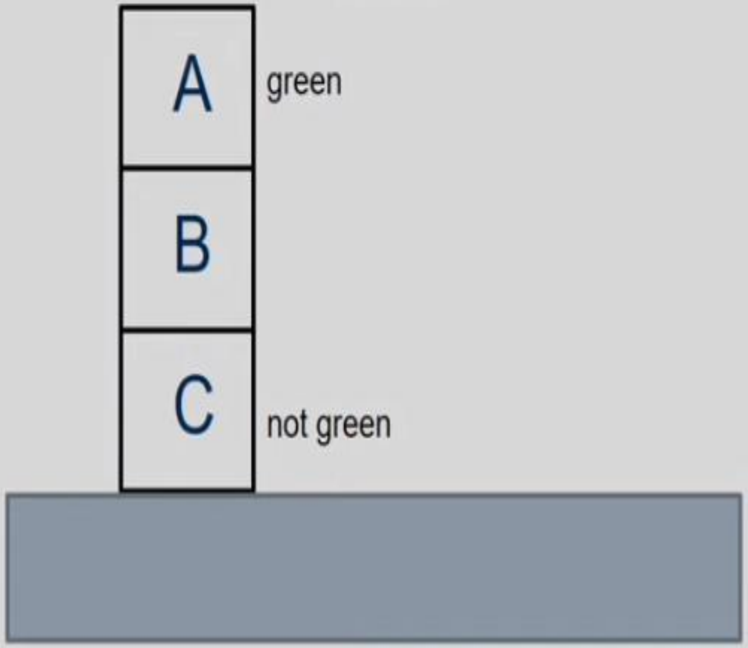


Resolution Refutation Proofs

Example 3

Suppose there are three coloured blocks stacked as shown, where the top one is `green' and the bottom is `not green'. Is there a `green' block on top of a `non-green' block?

on: holds if and only if block is immediately above the other.



Predicates

on(x,y): holds if and only if block `x' is immediately above `y'

green(x): block `x' is green.

Query

$$\exists x\, \exists y\ on(x, y) \land green(x) \land \neg green(y)$$

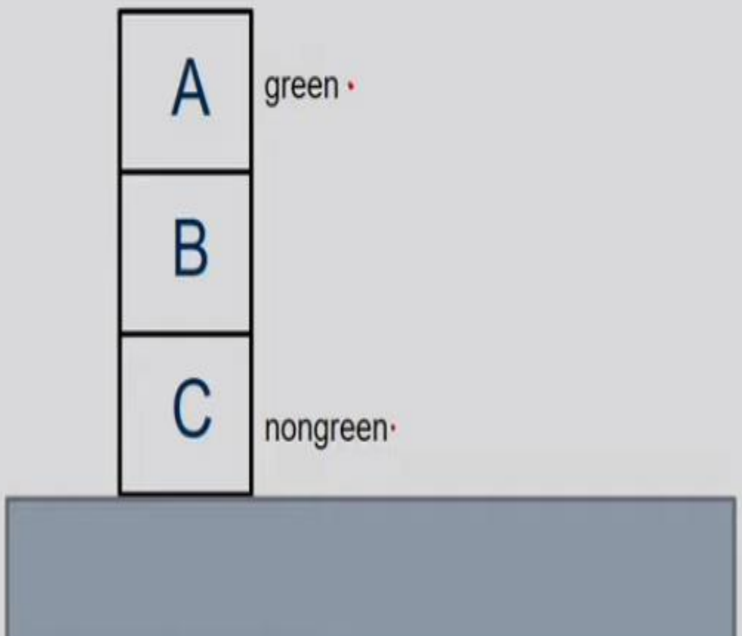
Resolution Refutation Proofs

Example 3

Negation of the Query

$$\neg\exists x\ \exists y\ (on(x,y) \land green(x) \land \neg green(y))$$

C5. $\neg\, on(x,y) \lor \neg\, green(x) \lor green(y)$



Refutation Trace

1.	$on(A,B)$	C1
2.	$on(B,C)$	C2
3.	$green(A)$	C3
4.	$\neg green(C)$	C4
5.	$\neg\, on(x,y) \lor \neg\, green(x) \lor green(y)$	C5
6.	$\neg\, green(A) \lor green(B)$	1,5
7.	$\neg\, green(B) \lor green(C)$	2,5
8.	$green(B)$	3,6
9.	$\neg\, green(B)$	4,7
10.	$\square$.	8,9

In this problem the KB entails that there is some block which must be green and on top of a nongreen block.

However, it does not make any commitment to any specific one;

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:

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

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“:

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

West, who is American …

$American(West)$

The country Nono, an enemy of America …

$Enemy(Nono,America)$

Forward chaining

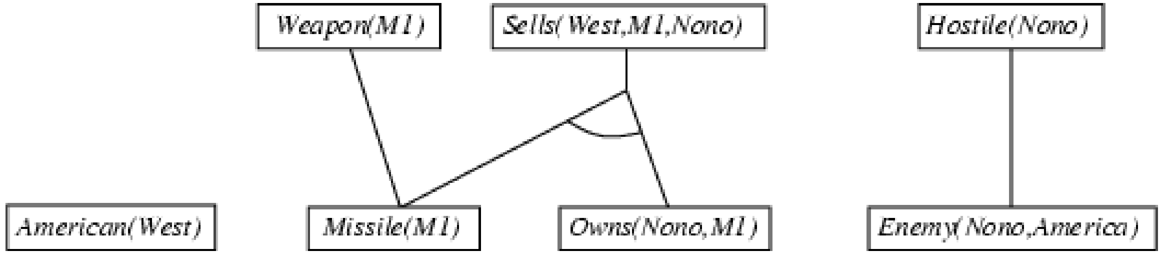
function FOL-FC-ASK(KB, α) **returns** a substitution or *false*

  repeat until *new* is empty
    $new \leftarrow \{\}$
    for each sentence r **in** KB **do**
      $(p_1 \wedge \ldots \wedge p_n \Rightarrow q) \leftarrow$ STANDARDIZE-APART(r)
      for each θ 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
        $q' \leftarrow$ SUBST(θ, q)
        if q' is not a renaming of a sentence already in KB or *new* **then do**
          add q' to *new*
          $\phi \leftarrow$ UNIFY(q', α)
          if ϕ is not *fail* **then return** ϕ
    add *new* to KB
  return *false*

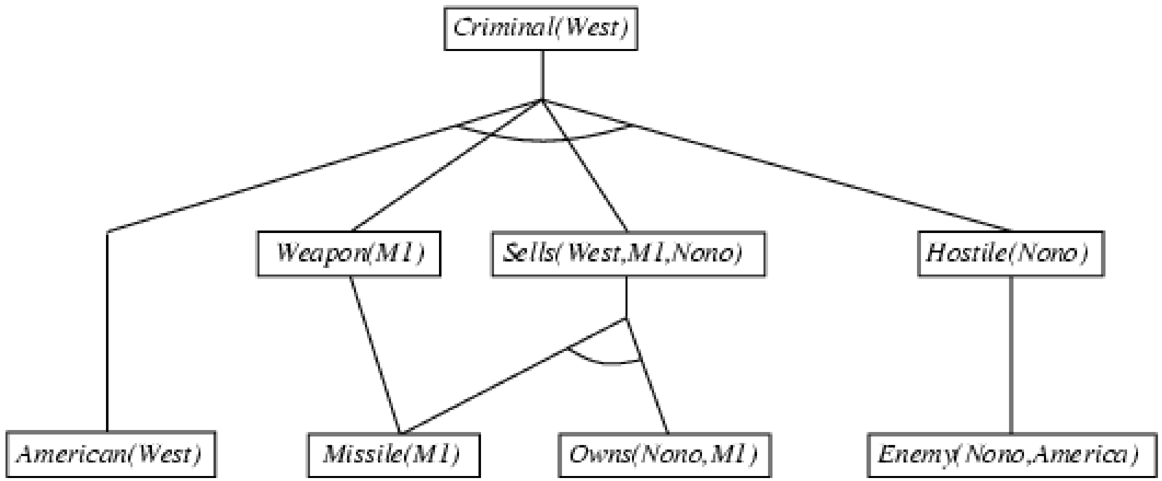
Forward chaining proof

American(West) *Missile(M1)* *Owns(Nono,M1)* *Enemy(Nono,America)*

Forward chaining proof



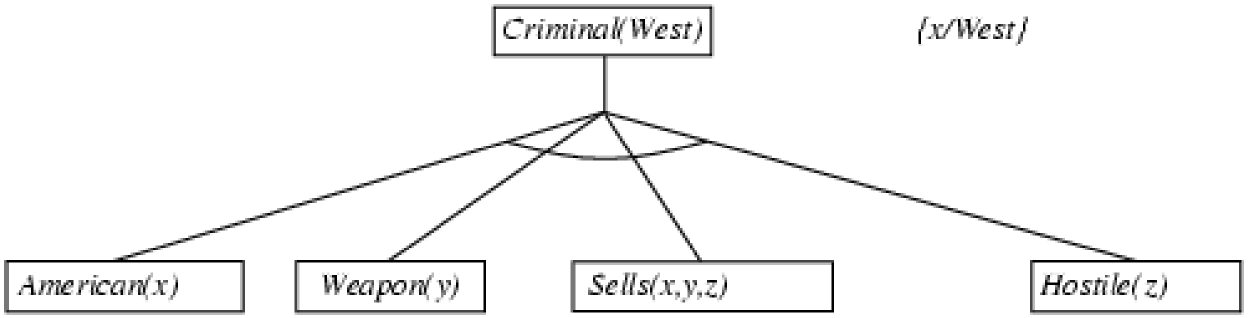
Forward chaining proof



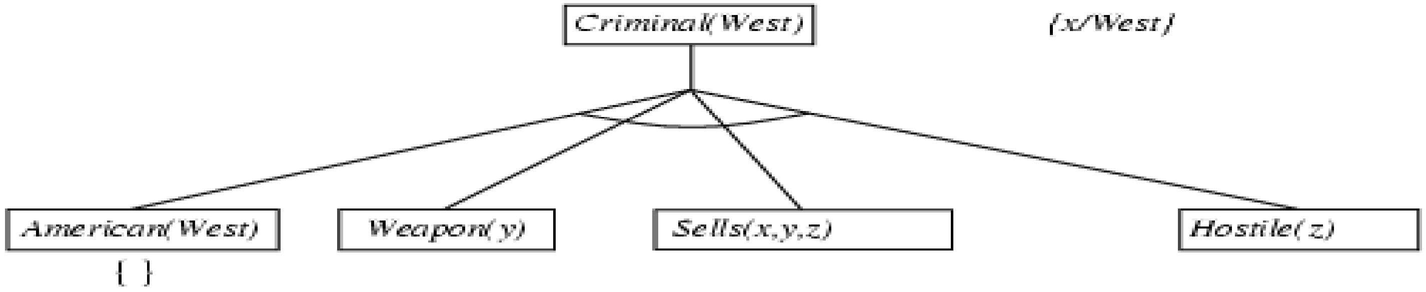
Backward chaining example

$\boxed{Criminal(West)}$

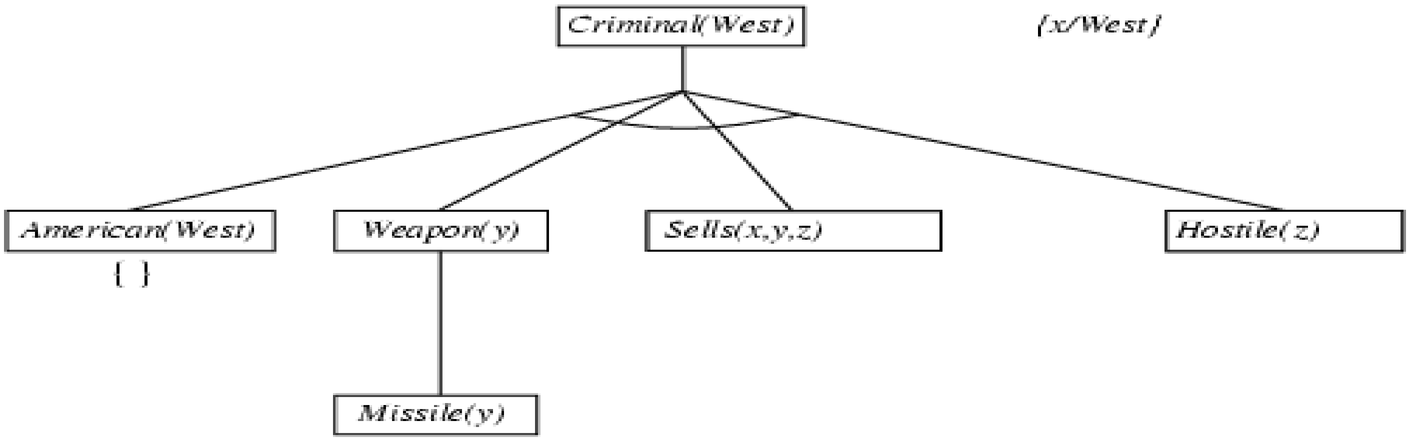
Backward chaining example



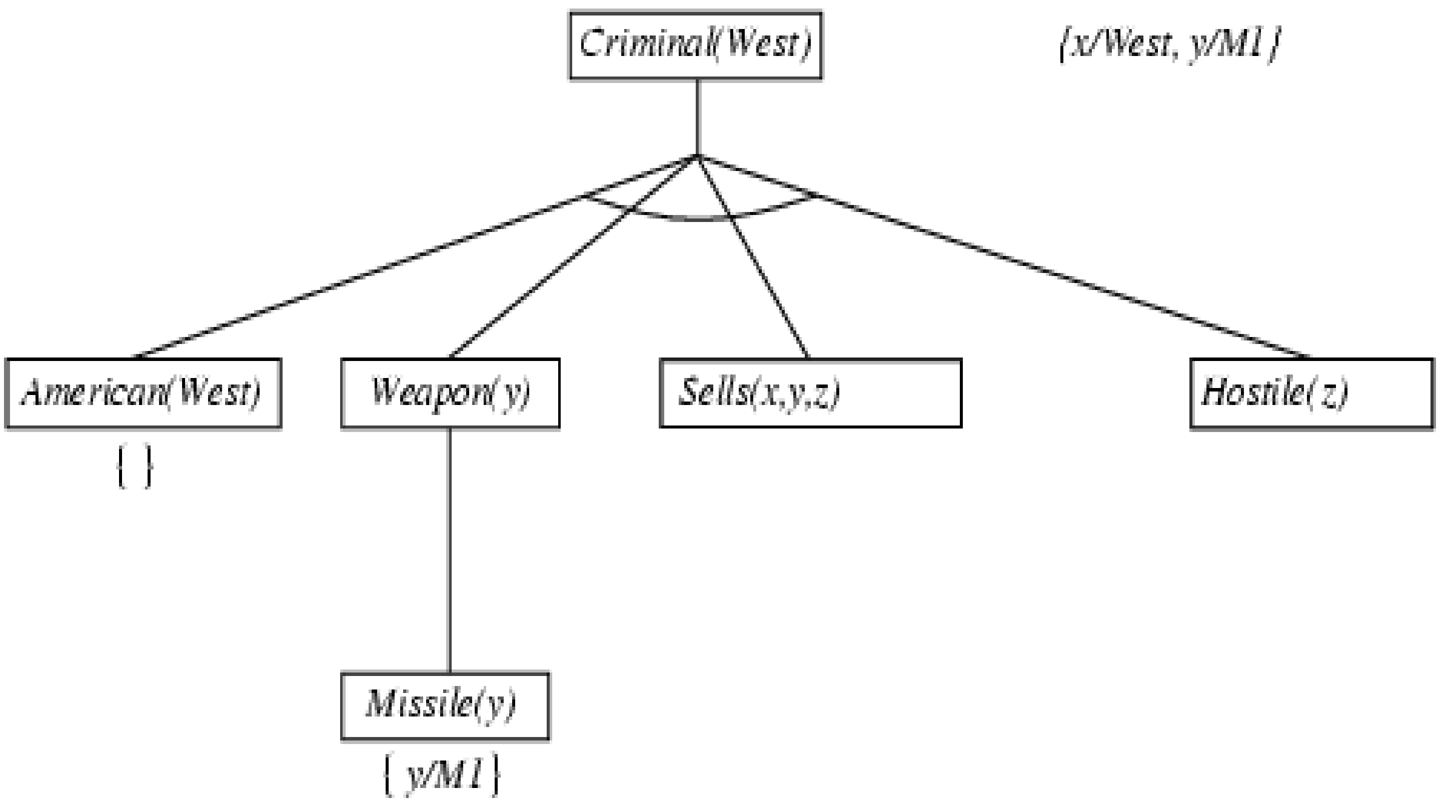
Backward chaining example



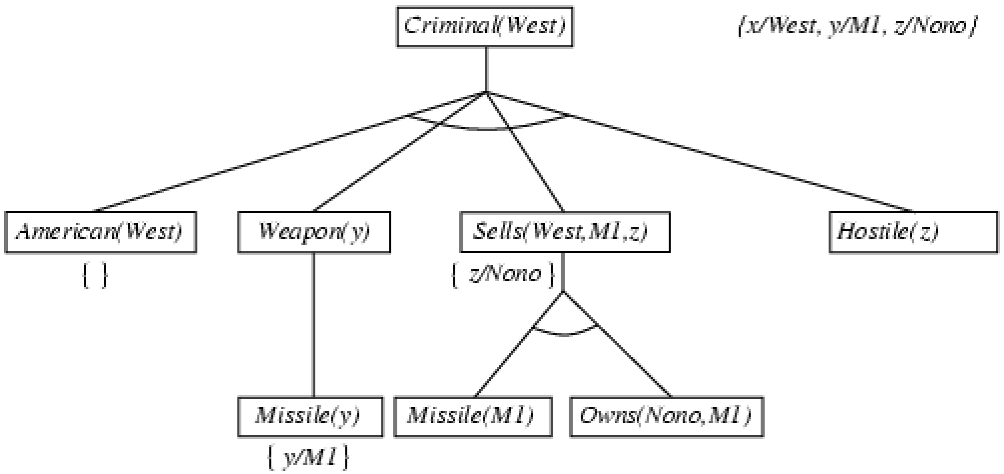
Backward chaining example



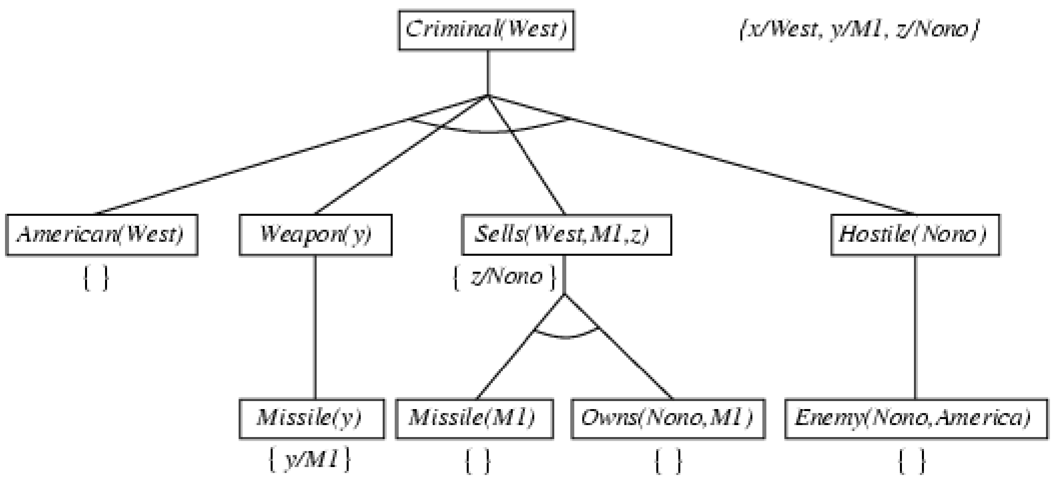
Backward chaining example



Backward chaining example



Backward chaining example



Backward chaining example