First-Order Predicate Calculus

CS 475

April 29, 2003

1 Introduction

Consider a disjunctive logic program clause:

$$even(X) \leftarrow odd(Y), X > Y.$$

This rule is true when X is even ($even(X)$ is true), Y is odd ($odd(Y)$ is true) and $X > Y$. This could also means that "for every odd number Y ($odd(Y)$ is true), there exists an even number X ($even(X)$ is true), which is greater than Y".

In general, a disjunctive logic program clause of the form

$$l_1 \vee \ldots \vee l_n \leftarrow b_1 \wedge \ldots \wedge b_m \wedge \ not\ c_1 \wedge \ldots \ not\ \wedge c_k \tag{1}$$

with $X_1, \ldots, X_p$ are the variables occuring in the head and $Y_1, \ldots, Y_q$ are the variables occurring in the body expresses the fact that for every tuple of values $(y_1, \ldots, y_q)$ of $(Y_1, \ldots, Y_q)$ such that the body is true there exists a tuple of value $(x_1, \ldots, x_p)$ of $(X_1, \ldots, X_p)$ such that the head is true.

We will now studying a language, called *first-order predicate calculus*, that allows us to write quantifications (for all and exists, denotes by, $\forall$ and $\exists$, respectively) explicitely in the clauses.

2 Preliminary

A *first-order theory* consists of an alphabet, a first order language, a set of axioms and a set of inference rules.

Definition 1 *An* alphabet *consists of the following sets:*

1. *variables*
2. *constants*
3. *function symbols*
4. *predicate symbols*
5. *connectives:* $\{\wedge, \vee, \neg, \leftrightarrow, \rightarrow\}$
6. *quantifiers:*$\forall, \exists$
7. *punctuation symbols:* $'(',')',',','.'$

NOTE: • The last three sets are the same for every alphabet.

• For an alphabet, only the set of constants or the set of function symbols may be empty.

• Notation convention: Variables: u, v, w, x, y, and z (possibly with indexes); constants: a, b, and c (possibly with indexes); function symbols of arities > 0: f, g, and h (possibly with indexes); and predicate symbols of arities ≥ 0: p, q, and r (possibly with indexes).

The precedence among the connectives: $\neg, \forall, \exists, \wedge, \vee, \rightarrow, \leftrightarrow$

Given an alphabet, a *first order language* is defined by the set of *well-formed formula*(*wff* or *sentences*) of the theory.

Definition 2 *A* term *is either*

1. *a variable,*
2. *a constant, or*
3. $f(t_1, \ldots, t_n)$ *where* f *is an n-ary function symbol and* $t_1, \ldots, t_n$ *are terms.*

Definition 3 *A* (well-formed) formula *is defined inductively as follows.*

1. $p(t_1, \ldots, t_n)$ *where* p *is an n-ary predicate symbol and* $t_1, \ldots, t_n$ *are terms,*
2. *if* p *and* q *are formulas then* $(\neg p)$, $(p \wedge q)$, $(p \vee q)$, $(p \rightarrow q)$, $(p \leftrightarrow q)$ *are formulas*
3. *if* p *is a formula and* X *is a variable then* $(\forall X\ p)$ *and* $(\exists X\ p)$ *are formulas.*

Definition 4 *A first order language given by an alphabet consists of the set of all formulas constructed from the symbols of the alphabet.*

Example 1 $(\forall X(\exists Y(p(X,Y) \rightarrow q(X))))$ *and* $(\neg\exists X((p(X,a) \wedge q(f(X)))))$ *are formulas. We can simplify them to* $\forall X \exists Y(p(X,Y) \rightarrow q(X))$ *and* $\neg\exists X(p(X,a) \wedge q(f(X)))$.

Definition 5 *The scope of* $\forall X$ *(resp.* $\exists X$*) in* $\forall X p$ *(resp.* $\exists X p$*) is* p. *A* bound occurrence *of a variable in a formula is an occurrence immediately following a quantifier or an occurrence within the scope of the quantifier, which has the same variable immediately after the quantifier. Any other occurrence of a variable is* free.

Example 2 $\exists X p(X,Y) \rightarrow q(X)$ *– the first two occurrences of* X *are bound but the third one is free.*

$\exists X(p(X,Y) \rightarrow q(X))$ *– all occurrences of* X *are bound.*

Definition 6 *A* closed formula *is a formula with no free occurrences of any variable.*

Example 3 $\exists X p(X,Y) \rightarrow q(X)$ *is not a closed formula.*

$\forall Y\ \exists X(p(X,Y) \rightarrow q(X))$ *is a closed formula.*

Definition 7 *A* grounded term *is a term not containing a variable. A* grounded atom *is an atom not containing a variable.*

3 Interpretation

NOTE: When we say 'a first order language L' we understand that the alphabet of L is given.

Definition 8 *Let L be a first order language. An* interpretation *I of L consists of*

1. *a non-empty set D, called the* domain *of I,*
2. *for each constant in L, the assignment of an element of D, (i.e., a constant c is mapped into an element $I(c) \in D$),*
3. *for each n-ary function symbol in L, the assignment of a mapping from D^n to D, (i.e., a function symbol f is mapped into a function f^I)*
4. *for each n-ary predicate symbol in L, the assignment of a mapping from D^n to into $\{true, false\}$, (i.e., a predicate symbol p is mapped into a relation p^I).*

Let I be an interpretation. A *variable assignment (wrt. I)* is an assignment to each variable in L of an element in D.

Let I be an interpretation and V be a variable assignment (wrt. I). The term assignment (wrt. I and V) of the terms in L is defined as follows.

1. Each variable is given its assignment according to V,
2. Each constant is given its assignment according to I,
3. If $t'_1, \ldots, t'_n$ are the term assignments of $t_1, \ldots, t_n$ and f' is the assignment of the n-ary function symbol f, then $f'(t'_1, \ldots, t'_n)$ is the term assignment of $f(t_1, \ldots, t_n)$.

Let I be an interpretation and V be a variable assignment (wrt. I). Then, a formula L can be given a *truth value*, true or false, (wrt. I and V) as follows:

1. If $L = p(t_1, \ldots, t_n)$ and $t'_1, \ldots, t'_n$ are the term assignments of $t_1, \ldots, t_n$ (wrt. I and V), and p' be the mapping assigned to the n-ary predicate symbol p by I, then the truth value of L is obtained by calculating the truth value of $p'(t_1, \ldots, t'_n)$,
2. If the formula has the form $(\neg p)$, $(p \wedge q)$, $(p \vee q)$, $(p \rightarrow q)$, $(p \leftrightarrow q)$ then the truth value of the formula is given by the following table

p	q	¬ p	p ∧ q	p ∨ q	p → q	p ↔ q
t	t	f	t	t	t	t
t	f	f	f	t	f	f
f	t	t	f	t	f	f
f	f	t	f	t	f	t

3. if the formula has the form $\exists XP$, then the truth value of the formula is true if there exists $d \in D$ such that P has the truth value wrt. I and the variable assignment V in which X is assigned to d; Otherwise, its truth value is false.
4. if the formula has the form $\forall Xp$, then the truth value of the formula is true if for all $d \in D$, P has the truth value wrt. I and the variable assignment V in which X is assigned to d; Otherwise, its truth value is false.

From the above, the truth value of a closed formula does not depend on the variable assignment. Thus, we can speak about the truth value of a closed formula wrt. to an interpretation (without mentioning the variable assignment).

Definition 9 *Let I be an interpretation of a first order language L and let p be a closed formula of L. Then, I is a* model *of p if p is true wrt. I.*

Definition 10 *Let T be a first order theory and L be its first order language L and let p be a closed formula of L. Then, I is a* model *of p if p is true wrt. I.*

Let S be a closed formulas of a first order language L and I be an interpretation of L. I is a *model* of S if every formula $p \in S$ is true wrt. I.

Definition 11 *Let S be a closed formulas of a first order language L. We say*

1. *S is* satisfiable *if L has an interpretation which is a model of S,*
2. *S is* valid *if every interpretation of L is a model of S,*
3. *S is* unsatisfiable *if no interpretation of L is a model of S,*
4. *S is* nonvalid *if L has an interpretation which is not a model of S.*

Definition 12 *Let S be a closed formulas of a first order language L and p a formula in L. S entails F, denoted by $S \models p$, if p is true wrt. to every model of S.*

4 Unification

Definition 13 *A* substitution *η us a finite set of the form $\{V_1/t_1, \ldots, V_n/t_n\}$ where each V_i is a variable, each t_i is a term distinct from V_i and the variables $V_1, \ldots, V_n$ are distinct.*

Each V_i/t_i is called a binding *for V_i;*

η is called a grounded substitution if the t_i are all ground terms.

Definition 14 *Let $\eta = \{V_1/t_1, \ldots, V_n/t_n\}$ be a substitution and E be an expression (term, literal, conjunction of literals, or disjunction of literals). Then, $E\eta$, the instance of E by η, is the expression obtained from E by simultaneously replacing each occurrence of the variables V_i in E by the term t_i.*

If $E\eta$ is ground, then it is called a ground instance *of E.*

Example 4 *Let $E = p(X, Y, f(a))$ and $\eta = \{X/b, Y/X\}$ then $E\eta = p(b, X, f(a))$.*

Definition 15 *Let $\sigma = \{V_1/t_1, \ldots, V_n/t_n\}$ and $\eta = \{U_1/s_1, \ldots, U_m/s_n\}$ be two substitutions. Then the* composition *$\eta\sigma$ of η and σ is the substitution obtained from the set $\{U_1/s_1\sigma, \ldots, U_m/s_n\sigma, V_1/t_1, \ldots, V_n/t_n\}$ by deleting any binding V_j/t_j for which $V_j \in \{U_1, \ldots, U_m\}$.*

Example 5 *Let $\sigma = \{X/a, Y/b, Z/Y\}$ and $\eta = \{X/f(Y), Y/Z\}$ then $\eta\sigma = \{X/f(b), Z/Y\}$ since $X = \{X/f(Y)\sigma, Y/Z\sigma\} = \{X/f(b), Y/Y, X/a, Y/b, Z/Y\}$ and $\eta\sigma$ is obtained from X by deleting $Y/Y, X/a, Y/b$ because X and Y are variables occurring in η.*

Two expressions E and F are called *variants* if there exists a substitution η such that $E = F\eta$ and $F = E\eta$. (We also say that E is a variant of F and vice versa!)

A set of expressions $S = \{E_1, \ldots, E_n\}$ is unifiable if there exists a substitution η such that $E_1\eta = E_2\eta = \ldots = E_n\eta$. In that case, η is a *unifier* of S.

A unifier η of S is called a *most general unifier* (or *mgu*) of S if for each unifier σ of S there exists a substitution γ such that $\sigma = \eta\gamma$.

Example 6 *The set* $S = \{p(f(X), a), p(Y, f(W))\}$ *is not unifiable because we can not unify the constant* a *with* $f(W)$.

The set $S = \{p(f(X,Y), g(Z), a), p(f(Y,X), g(U), a)\}$ *is unifiable since for* $\eta = \{X/Y, Y/X, Z/U\}$, $S\eta = \{p(f(X,X), g(U), a)\}$. *Here,* η *is a mgu of* S.

Definition 16 *Let* S *be a set of simple expressions (a simple expression is a term or an atom). The* disagreement set *of* S *is defined as follows. Locate the leftmost symbol position at which not all expressions in* S *have the same symbol and extract from each expression expression in* S *the subexpression beginning at that symbol position. The set of all such subexpressions is the disagreement set.*

Example 7 *Let* $S = \{p(f(X), h(Y), a), p(f(X), z, a), p(f(X), h(Y), b)\}$. *Then the disagreement set is* $\{h(Y), Z)\}$.

Example 8 *Let* $S = \{p(f(X), h(Y), a), p(f(X), Z, a), p(f(X), W, b)\}$. *Then the disagreement set is ?.*

5 Unification Algorithm

Let $S = \{P_1, \ldots, P_m\}$ be a set of simple expressions.

- S1 Put $k = 0$ and $\sigma_0 = \{\}$.
- S2 If $S\sigma_k$ is a singleton ($P_i\sigma_k = P_j\sigma_k$ for every $i \neq j$), then stop; σ_k is an mgu (most general unifier) of S; Otherwise, find the disagreement set D_k of $S\sigma_k$.
- S3 If there exist v and t in D_k such that v is a variable that does not occur in t, then put $\sigma_{k+1} = \sigma_k\{v/t\}$, increment k and go to S2. Otherwise, stop; S is not unifiable.

Example 9 *Let* $S = \{p(f(a), g(X)), p(Y, Y)\}$.

- *S1 Put* $k = 0$ *and* $\sigma_0 = \{\}$.
- *S2* $S\sigma_0 = S$ *is not a singleton. So, we need to find the disagreement set* D_0 *of* $S\sigma_0 = S$. *We have:* $D_0 = \{f(a), Y\}$.
- *S3 Here,* Y *is a variable which does not occur in* $f(a)$. *So, we let* $\sigma_1 = \sigma_0\{Y/f(a)\} = \{Y/f(a)\}$ *and go to S2.*
- *S2* $S\sigma_1 = \{p(f(a), g(X)), p(f(a), f(a))\}$ *is not a singleton. So, we need to find the disagreement set* D_1 *of* $S\sigma_1 = S$. *We have:* $D_1 = \{g(X), f(a)\}$.
- *S3 Here, there is no variable in* D_1. *So, we stop;* S *is not unifiable.*

Example 10 *Let* $S = \{p(a, X, h(g(Z))), p(Z, h(Y), h(Y))\}$.

S1 *Put* $k = 0$ *and* $\sigma_0 = \{\}$.

S2 $S\sigma_0 = S$ *is not a singleton. So, we need to find the disagreement set* D_0 *of* $S\sigma_0 = S$. *We have:* $D_0 = \{a, z\}$.

S3 *Here,* z *is a variable which does not occur in* a. *So, we let* $\sigma_1 = \sigma_0\{Z/a\} = \{Z/a\}$ *and go to S2.*

S2 $S\sigma_1 = \{p(a, X, h(g(a))), p(a, h(Y), h(Y))\}$ *is not a singleton. So, we need to the disagreement set* D_1 *of* $S\sigma_1$. *We have:* $D_1 = \{X, h(Y)\}$.

S3 *Here,* X *is a variable which does not occur in* $h(Y)$. *So, we let* $\sigma_2 = \sigma_1\{X/h(Y)\} = \{Z/a, X/h(Y)\}$ *and go to S2.*

S2 $S\sigma_2 = \{p(a, h(Y), h(g(a))), p(a, h(Y), h(Y))\}$ *is not a singleton. So, we need to the disagreement set* D_2 *of* $S\sigma_2$. *We have:* $D_1 = \{Y, g(a)\}$.

S3 *Here,* Y *is a variable which does not occur in* $g(a)$. *So, we let* $\sigma_3 = \sigma_2\{Y/g(a)\} = \{Z/a, X/h(g(a)), Y/g(a)\}$ *and go to S2.*

S2 $S\sigma_3 = \{p(a, h(g(a)), h(g(a)))\}$ *is a singleton. So we stop and one mgu of* S *is* $\sigma_3 = \{Z/a, X/h(g(a)), Y/g(a)\}$.

Theorem 1 *Let* S *be a finite of simple expressions. If* S *is unifiable then the algorithm terminates and gives an mgu for* S. *If* S *is not unifiable then the algorithm terminates and reports this fact.*

6 Resolution

Definition 17 *A* literal *is either an atom* p *or its negation* $\neg p$.

A clause *is a disjunction of literals. (sometime it is written as* $p_1 \vee \ldots \vee p_n$ *or* $\{p_1, \ldots, p_n\}$*)*

A formula q *is said to be in* conjunctive normal form *(or* CNF*) if* q *is a conjunction of clauses.*

A formula q *is said to be in* implicative normal form *if* q *is a conjunction of implication of the form* $p_1 \wedge \ldots \wedge p_n \rightarrow q_1 \vee \ldots \vee q_m$ *where each* p_i, q_j *is an atom.*

$(p \vee q \vee \neg s) \wedge (\neg p \vee q \vee s) \wedge (\neg p \vee \neg r \vee \neg s) \wedge (p \vee t \vee \neg s)$ is a CNF.

Theorem 2 *For every formula* ϕ *there exists a formula* ψ *in CNF form such that* ϕ *and* ψ *is equivalent, i.e.,* $\forall(\phi \leftrightarrow \psi)$ *is a valid formula.*

Algorithm to convert a formula into CNF form.

1. **Eliminate implications**: Replace $p \rightarrow q$ with $\neg p \vee q$
2. **Move** $\neg$ **inward**: do the following
   (a) $\neg(p \vee q)$ is replaced by $\neg p \wedge \neg q$
   (b) $\neg(p \wedge q)$ is replaced by $\neg p \vee \neg q$
   (c) $\neg \forall X p$ is replaced by $\exists X \neg p$

(d) $\neg\exists X p$ is replaced by $\forall X \neg p$

(e) $\neg\neg p$ is replaced by p

3. **Standardize variable**: For sentences like $(\forall X p(X)) \vee (\exists X q(X))$ that use the same variable name twice, change the name of one of the variable.

4. **Move quantifier left**: Move all quantifiers in the formula to the left, for example, $p \vee \forall X q$ is equivalent to $\forall X q \vee p$ etc.

5. **Skolemize**: Remove the existential quantifier by elimination – this includes: (1) defines a Skolem function, one for a variable occurred immediately after an existential quantification and

   - either (2) introduces a new constant, one for a variable occurred immediately after an existential quantification, (3) removes the existential quantification and substitutes X for $f^x(a^x)$ in the formula;
   - or (2) introduces a new constant, one for a variable occurred immediately after an existential quantification, (3) removes the existential quantification and substitutes X for $f^x(Y, Z, ..)$ in the formula where $Y, Z, ...$ are the variables which are universally quantified outside the existential quantifier in question;

6. **Distribute $\wedge$ over $\vee$**: $(a \wedge b) \vee c$ becomes $(a \vee c) \wedge (b \vee c)$.

7. **Flatten nested conjunction and disjunction**: $(a \wedge b) \wedge c$ becomes $(a \wedge b \wedge c)$ and $(a \vee b) \vee c$ becomes $(a \vee b \vee c)$.

Example 11 *Convert* $((\neg\forall X a(X)) \vee (\forall Y b(Y))) \rightarrow (\neg(\forall Z q(Z, f(Z))))$ *to CNF.*

1. $\neg((\neg\forall X a(X)) \vee (\forall Y b(Y))) \vee (\neg(\forall Z q(Z, f(Z))))$ *(Eliminate implication)*

2. $((\neg\neg\forall X a(X)) \wedge (\neg\forall Y b(Y))) \vee (\neg(\forall Z q(Z, f(Z))))$ *(Move ¬ ...)*

3. $(\forall X a(X) \wedge \exists Y \neg b(Y)) \vee ((\exists Z \neg q(Z, f(Z))))$ *(Move ¬ ...)*

4. $\forall X \exists Y \exists Z((a(X) \wedge \neg b(Y)) \vee \neg q(Z, f(Z)))$ *(Move quantifier left)*

5. $\forall X((a(X) \wedge \neg b(fy(cy))) \vee \neg q(fz(cz), f(fz(cz))))$ *(Removing existential quantifier - fy, fz are two new functions and cy, cz are two new constants, correspond to the variable $Y and Z$ respectively)*

6. $(a(X) \wedge \neg b(fy(cy))) \vee \neg q(fz(cz), f(fz(cz)))$ *(Drop universal quantifier)*

7. $(a(X) \vee \neg q(fz(cz), f(fz(cz)))) \wedge (\neg b(fy(cy)) \vee \neg q(fz(cz), f(fz(cz))))$ *(Distribute $\wedge$ over $\vee$)*

NOTE: Implicative normal form is often used to. A formula of the form $\neg p_1 \vee \neg p_2 \vee \ldots \vee \neg p_n \vee q_1 \vee q_2 \ldots \vee q_m$ is equivalent to $p_1 \wedge p_2 \ldots \wedge p_n \rightarrow q_1 \vee q_2 \ldots \vee q_m$

It is easy to see that if q is in CNF then we can convert it into implicative normal form using the above conversion.

The *resolution inference rule* If β_1 and β_2 are unifiable and η is a mgu of β_1 and β_2, then

$$\frac{\alpha \vee \beta_1, \neg\beta_2 \vee \gamma}{\alpha\eta \vee \gamma\eta} \tag{2}$$

or

$$\frac{\neg\alpha \to \beta_1, \beta_2 \to \gamma}{\neg\alpha\eta \to \gamma\eta} \tag{3}$$

We can

Given a set of formulas S and a formula q, we would like to determines if $S \models q$.

We can use (2) (or (3)) to determine whether $S \vdash q$ holds or not.

We make the following assumptions:

1. Each formula in S is a clause (**why?**)
2. q is a literal (**why?**)

Example 12 *Let Δ be the set consisting of the following clauses:*

1. $\neg p(W) \vee q(W)$,
2. $p(X) \vee r(X)$,
3. $\neg q(Y) \vee s(Y)$, *and*
4. $\neg r(Z) \vee s(Z)$.

Question: $\Delta \vdash s(a)$*?*

Proof.

1. $\frac{\neg p(W) \vee q(Y), p(X) \vee r(X)}{\neg p(W) \vee s(W)}$ *Where* $\eta = \{Y/W\}$
2. $\frac{\neg p(W) \vee S(W), p(X) \vee r(X)}{s(X) \vee r(X)}$ *with* $\{W/X\}$
3. $\frac{s(X) \vee r(X), \neg r(Z) \vee s(Z)}{s(a)}$ *with* $\{X/a, Z/a\}$*! DONE!*

6.1 Refutation proof procedure

Given a set of clauses S and a literal q. The refutation proof procedure uses resolution to determine whether $S \models q$ holds or not.

1. **Idea:** If $S \models q$ then $S \cup \{\neg q\}$ is unsatisfiable, i.e., there is no model for $S \cup \{\neg q\}$. So, we will assume that $\neg q$ holds and try to derive a contradiction out of $S \cup \{\neg q\}$.
2. **Algorithm:** We try to derive a proof that derives a contradiction from $S \cup \{\neg q\}$. The algorithm can be described as follows.

   A1 Let $k = 0$, $G_k = \neg q$.

   A2 If $G_k = false$ then step and answer 'yes'; Otherwise, find a clause C in S that contains a literal L which is contradictory with some L' of G_k and η is a mgu of L and L'. Go to step [A3]!

   A3 Let $G_{k+1} = ((C \setminus \{L\}) \cup (G_k \setminus \{L'\}))\eta$, $k = k + 1$, and go to step [A2]!

Example 13 *We have the following English description:*

- *Everyone who loves all animals is loved by someone.*
- *Anyone who kills an animal is loved by no one*
- *Jack loved all animals.*
- *Either Jack or Curiosity kills the cat, who is named Tuna.*
- *Did Curiosity kill the cat?*

First, we need to represent the above sentences into formulae:

$$\forall X(\forall Y animal(Y) \rightarrow loves(X,Y)) \rightarrow (\exists Y loves(X,Y)) \tag{4}$$
$$\forall X(\exists Y animal(Y) \wedge kills(X,Y)) \rightarrow (\forall Z \neg loves(Z,X)) \tag{5}$$
$$\forall X animal(X) \rightarrow loves(jack, X) \tag{6}$$
$$kills(jack, tuna) \vee kills(curiosity, tuna) \tag{7}$$
$$cat(tuna) \tag{8}$$
$$\forall X cat(X) \rightarrow animal(X) \tag{9}$$

Convert to clausal form

$$animal(f(X)) \vee loves(g(X), X) \tag{10}$$
$$\neg loves(X, f(X)) \vee loves(g(X), X) \tag{11}$$
$$\neg animal(Y) \vee \neg kills(X,Y) \vee \neg loves(Z,X) \tag{12}$$
$$\neg animal(X) \vee loves(jack, X) \tag{13}$$
$$kills(jack, tuna) \vee kills(curiosity, tuna) \tag{14}$$
$$cat(tuna) \tag{15}$$
$$\neg cat(X) \vee animal(X) \tag{16}$$

Proving $kills(curiosity, tuna)$

$G_0 = \neg kills(curiosity, tuna)$, $\eta = \{\}$, *Clause (14)*

$G_1 = kills(jack, tuna)$, $\eta = \{X/jack, Y/tuna\}$, *Clause (13)*

$G_2 = animal(tuna)$, $\eta = \{X/tuna\}$, *Clause (15) and (16).*

$G_3 = \neg loves(Y, X) \vee \neg kills(X, tuna)$, $\eta = \{X/tuna\}$, *Clause (12) and* G_2

$G_4 = \neg loves(Y, jack)$, $\eta = \{X/jack\}$, G_3 *and* G_1

$G_5 = \neg animal(f(jack)) \vee loves(g(jack), jack)$, $\eta = \{X/jack\}$, *Clause (11) and (13)*

$G_6 = loves(g(jack), jack)$, $\eta = \{X/jack\}$, *Clause (10) and* G_5

$G_7 = \square$ *(or* $G_7 = false$*)*, $\eta = \{Y/g(jack)\}$, G_4 *and* G_6*! DONE*

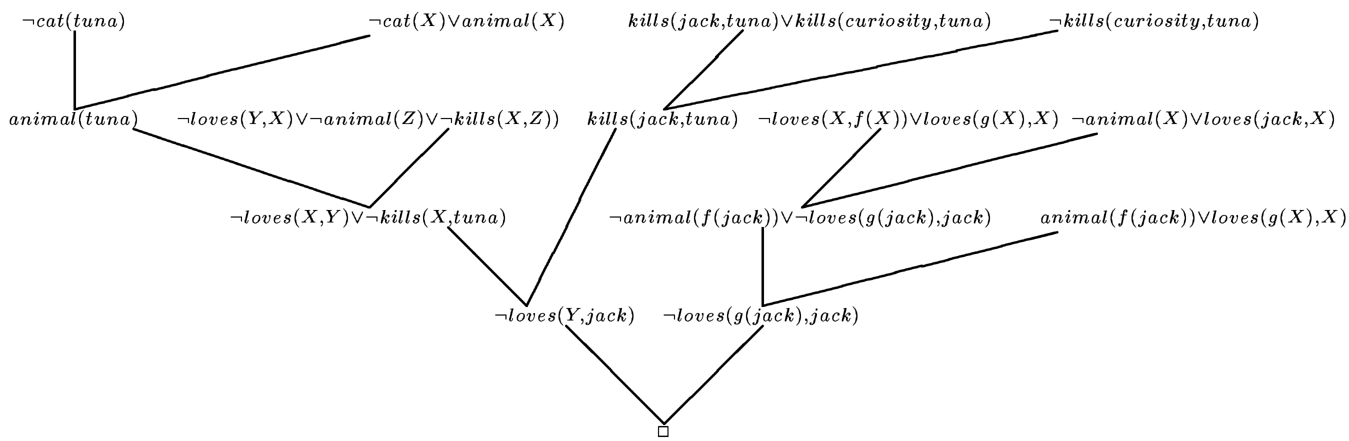

Figure 1: A refutation tree

Remark. The steps can be drawn into a *refutation tree* as follows:

Example 14 *Let us try to prove*

$$(\exists Y\ \forall X\ p(X,Y)) \rightarrow (\forall X\ \exists Y p(X,Y))$$

is valid.

To prove that this is a valid formula, we assume that the left hand side of the implication is true and prove that the right hand side is one of its consequence. That it, we are given $(\exists Y\ \forall X\ p(X,Y))$ *and we want to prove* $(\forall X\ \exists Y p(X,Y))$.

Using resolution: we need to show that we can get a refutation proof from

$$\begin{array}{l} (\exists Y\ \forall X\ p(X,Y)) \quad \text{and} \\ \neg(\forall X\ \exists Y p(X,Y)) \end{array}$$

Convert it to a CNF: for the first formula, we only need to do the Skolemization and for the second formula, we need to push ¬ *inward. This gives,*

$$\begin{array}{l} p(X, f^y(a^y))) \quad \text{and} \\ (\exists X\ \neg\exists Y p(X,Y)) \end{array}$$

One more pushing of ¬*, we get:*

$$\begin{array}{l} p(X, f^y(a^y))) \quad \text{and} \\ (\exists X\ \forall Y \neg p(X,Y)) \end{array}$$

Skolemization gives us

$$\begin{array}{l} p(X, f^y(a^y))) \quad \text{and} \\ \neg p(f^x(a^x), Y)) \end{array}$$

With $\{X/f^x(a^x), Y/f^y(a^y)\}$ *we get the refutation, and hence, a proof for the formula.*

Second proof: *An alternative to prove the validity of original formula is show next. The idea is to deriving true from the original formula through equivalent transformation:*

$(\exists Y\ \forall X\ p(X,Y)) \rightarrow (\forall X\ \exists Y p(X,Y))$

$\leftrightarrow \neg(\exists Y\ \forall X\ p(X,Y)) \vee (\forall X\ \exists Y p(X,Y))$

$\leftrightarrow (\forall Y\ \neg\forall X\ p(X,Y)) \vee (\forall X\ \exists Y p(X,Y))$

$\leftrightarrow (\forall X\ \exists Y\ \neg p(Y,X)) \vee (\forall X\ \exists Y p(X,Y))$

$\leftrightarrow (\forall X\ \exists Y\ (\neg p(Y,X)) \vee p(X,Y))$

$\leftrightarrow$ *true for* $X = Y$.

7 Exercises (From Russel and Norvig's book) – Models, Interpretations, Validity, Satisfiability

1. Consider an alphabet with four propositions a, b, c, and d. How many models are there for the following formulae?
   (a) $(a \vee b) \vee (b \vee c)$
   (b) $a \vee b$
   (c) $a \leftrightarrow b \leftrightarrow c$

2. Decide whether each of the following formulae is valid, unsatisfiable, or neither. Verify your decision using truth tables.
   (a) $Smoke \rightarrow Smoke$
   (b) $Smoke \rightarrow Fire$
   (c) $(Smoke \rightarrow Fire) \rightarrow (\neg Smoke \rightarrow \neg Fire)$
   (d) $Smoke \vee Fire \vee \neg Fire$
   (e) $((Smoke \wedge Heat) \rightarrow Fire) \leftrightarrow ((Smoke \rightarrow Fire) \vee (Heat \vee Fire))$
   (f) $(Smoke \rightarrow Fire) \rightarrow ((Smoke \wedge Heat) \rightarrow Fire)$
   (g) $Big \vee Dumb \vee (Big \rightarrow Dumb)$
   (h) $(Big \wedge Dumb) \vee \neg Dumb$

3. Give the logical representations for the folowing sentences:
   (a) Horses, cows, and pigs are mammals.
   (b) An offspring of a horse is a horse.
   (c) Bluebeard is a horse.
   (d) Bluebeard is Charlie's parent.
   (e) Offsprings and parents are inverse relations.
   (f) Every mammal has a parent.

   Use your representations in the above exercise to answer the query: $\exists H horse(H)$?

Homework:

1. Here are two formulae in the language of first-order logic:

   (A): $\forall X \exists Y (X \geq Y)$

   (B): $\exists Y \forall X (X \geq Y)$

   (a) Assume that the variables range over all the natural numbers 0, 1, ..., ∞ and that the "$\geq$" predicate means "is greater than or equals to." Under this interpretation, translates (A) and (B) into English.

   (b) Is (A) true under this interpretation?

   (c) How about (B)?

   (d) Does (A) logically entail (B)?

   (e) Does (B) logically entail (A)?

   (f) Using resolution, try to prove that (A) follows from (B).

   (g) Using resolution, try to prove that (B) follows from (A).

2. Consider the robot in the delivery robot world.

   (a) Assume that in the initial situation, the door *door*1 is locked. What do we need to change in its situation calculus theory representation?

   (b) Assume that there are some renovation in the hallway along the rooms $r117, r115, r113$ which make these three rooms inaccessible to the robot (the robot cannot move there) and block the connection between the location $o109$ and *storage* (the robot cannot move from $o109$ to *storage*. Let us use the fluent $accessible(R)$ and $block(X, Y)$ to represent the fact that the room R is accessible to the robot and the path between X and Y is blocked. Modify the situation calculus theory of the delivery robot world to take into consideration of these new fluents. In particular, we assume that in the initial situation, the room $r117, r115, r113$ are inaccessible to the robot and the path between any $o109$ and *storage* is blocked. Assume also that we have the CWA. It would be helpful if you do this in the two steps:

      i. List all the changes that need to be made.

      ii. Make the changes.