Syntax of Predicate Calculus

The predicate calculus uses the following types of symbols:

Constants: A constant symbol denotes a particular entity. E.g. "john", "muriel" "1".

Functions: A function symbol denotes a mapping from a number of entities to a single entities: E.g. "father_of" is a function with one argument. "plus" is a function with two arguments. "father_of(john)" is some person. "plus(2,7)" is some number.

Predicates: A predicate denotes a relation on a number of entities. e.g. "married" is a predicate with two arguments. "odd" is a predicate with one argument. "married(john, sue)" is a sentence that is true if the relation of marriage holds between the people John and Sue. 'odd(plus(2,7))" is a true sentence.

Variables: These represent some undetermined entity. Examples: "X" "$S1$" …

Boolean operators: $\neg$, $\vee$, $\wedge$, $\Rightarrow$, $\Leftrightarrow$.

Quantifiers: The symbols $\forall$ (for all) and $\exists$ (there exists).

Grouping symbols: The open and close parentheses and the comma.

A *term* is either

1. A constant symbol; or
2. A variable symbol; or
3. A function symbol applied to terms.

Examples: "john", "X", "father_of(john)", "plus(X,plus(1,3))".

An *atomic formula* is a predicate symbol applied to terms.
Examples: "odd(X)" "odd(plus(2,2))", "married(sue,father_of(john))".

A *formula* is either

1. An atomic formula; or
2. The application of a Boolean operator to formulas; or
3. A quantifier followed by a variable followed by a formula.

Examples: "odd(X)," "odd(X) $\vee$ $\neg$odd(plus(X, X))," "$\exists_X$ odd(plus(X, Y))," "$\forall_X$ odd(X) $\Rightarrow$ $\neg$odd(plus(X,3))."

A *sentence* is a formula with no free variables. (That is, every occurrence of every variable is associated with some quantifier.)

Clausal Form and Skolemization

A *literal* is either an atomic formula or the negation of an atomic formula.

Examples: odd(3). ¬odd(plus(X,3)). married(sue,Y).

A *clause* is the disjunction of literals. Variables in a clause are interpreted as universally quantified with the largest possible scope.

Example: $\text{odd}(X) \vee \text{odd}(Y) \vee \neg\text{odd}(\text{plus}(X, Y))$ is interpreted as
$\forall_{X,Y}\ \text{odd}(X) \vee \text{odd}(Y) \vee \neg\text{odd}(\text{plus}(X, Y))$.

Converting a sentence to clausal form

1. Replace every occurrence of $\alpha \Leftrightarrow \beta$ by $(\alpha \Rightarrow \beta) \wedge (\beta \Rightarrow \alpha)$. When this is complete, the sentence will have no occurrence of $\Leftrightarrow$.
2. Replace every occurrence of $\alpha \Rightarrow \beta$ by $\neg\alpha \vee \beta$. When this is complete, the only Boolean operators will be $\vee$, $\neg$, and $\wedge$.
3. Replace every occurrence of $\neg(\alpha \vee \beta)$ by $\neg\alpha \wedge \neg\beta$; every occurrence of $\neg(\alpha \wedge \beta)$ by $\neg\alpha \vee \neg\beta$; and every occurrence of $\neg\neg\alpha$ by α.

   New step: Also, replace every occurrence of $\neg\exists_\mu \alpha$ by $\forall_\mu \neg\alpha$ and every occurrence of $\neg\forall_\mu \alpha$ by $\exists_\mu \neg\alpha$.

   Repeat as long as applicable. When this is done, all negations will be next to an atomic sentence.
4. (New Step: Skolemization). For every existential quantifier $\exists_\mu$ in the formula, do the following:
   If the existential quantifier is not inside the scope of any universal quantifiers, then

   i. Create a new constant symbol γ.
   ii. Replace every occurrence of the variable μ by γ.
   iii. Drop the existential quantifier.

   If the existential quantifier is inside the scope of universal quantifiers with variables $\Delta_1 \dots \Delta_k$,then

   i. Create a new function symbol γ.
   ii. Replace every occurrence of the variable μ by the term $\gamma(\Delta_1 \dots \Delta_k)$
   iii. Drop the existential quantifier.

Example. Change "$\exists_X$ blue(X)" to "blue(sk1)".
Change $\forall_X \exists_Y$ odd(plus(X,Y)) to $\forall_X$ odd(plus(X,sk2(X)).
Change $\forall_{X,Y} \exists_Z \forall_A \exists_B$ p(X,Y,Z,A,B) to p(X,Y,sk3(X,Y),A,sk4(X,Y,A)).

5. New step: Elimination of universal quantifiers:
   Part 1. Make sure that each universal quantifier in the formula uses a variable with a different name, by changing variable names if necessary.
   Part 2. Drop all univeral quantifiers.

   Example. Change [$\forall_X$ p(X)] $\vee$ [$\forall_X$ q(X)] to p(X) $\vee$ q($X1$).

6. (Same as step 4 of CNF conversion.) Replace every occurrence of $(\alpha \wedge \beta) \vee \gamma$ by $(\alpha \vee \gamma) \wedge (\beta \vee \gamma)$, and every occurrence of $\alpha \vee (\beta \wedge \gamma)$ by $(\alpha \vee \beta) \wedge (\alpha \vee \gamma)$. Repeat as long as applicable. When this is done, all conjunctions will be at top level.

7. (Same as step 5 of CNF conversion.) Break up the top-level conjunctions into separate sentences. That is, replace $\alpha \wedge \beta$ by the two sentences α and β. When this is done, the set will be in CNF.

Example:

Start. $\forall_X$ [even(X) $\Leftrightarrow$ [$\forall_Y$ even(times(X,Y))]]

After Step 1: $\forall_X$ [[even(X) $\Rightarrow$ [$\forall_Y$ even(times(X,Y))]] $\wedge$
[[$\forall_Y$ even(times(X,Y))] $\Rightarrow$ even(X)]].

After step 2: $\forall_X$ [[$\neg$even(X) $\vee$ [$\forall_Y$ even(times(X,Y))]] $\wedge$
[$\neg$[$\forall_Y$ even(times(X,Y))] $\vee$ even(X)]].

After step 3: $\forall_X$ [[$\neg$even(X) $\vee$ [$\forall_Y$ even(times(X,Y))]] $\wedge$
[[$\exists_Y$ $\neg$even(times(X,Y))] $\vee$ even(X)]].

After step 4: $\forall_X$ [[$\neg$even(X) $\vee$ [$\forall_Y$ even(times(X,Y))]] $\wedge$
[$\neg$even(times(X,sk1(X))) $\vee$ even(X)]].

After step 5: [$\neg$even(X) $\vee$ even(times(X,Y))] $\wedge$
[$\neg$even(times(X,sk1(X))) $\vee$ even(X)].

Step 6 has no effect.

After step 7: $\neg$even(X) $\vee$ even(times(X,Y)).
$\neg$even(times(X,sk1(X))) $\vee$ even(X).

Resolution

A *substitution* is an association of variables with terms;

Example: $\sigma = \{ \text{X} \rightarrow \text{a}, \text{Y} \rightarrow \text{f}(Z) \}$ is a substitution.

The *application* of a substitution σ to a clause ϕ, written $\phi\sigma$, is the clause that is obtained when each occurrence in ϕ of a variable in σ is replaced by the associated term.

Example: If ϕ is the clause $\text{p}(X, Y) \vee \neg\text{q}(Y, Z)$, and σ is the substution above, then $\phi\sigma$ is $\text{p(a,f}(Z)) \vee \text{q}(\text{f}(Z),Z)$.

Fact: If ϕ is true, then $\phi\sigma$ is true.

Let α and β be atomic formulas. α and β are *unifiable* if there are substutions σ_A and σ_B such that $\alpha\sigma_A = \beta\sigma_B$.

Examples. "p(a,b)" is unifiable with "$\text{p}(X, Y)$" under the substitution $\sigma_B = \{ X \rightarrow \text{a}, Y \rightarrow \text{b} \}$

"p(a,b)" is not unifiable with "$\text{p}(X, X)$".

"$\text{p(a,}Z)$" is unifiable with "$\text{p}(Z\text{,b})$" under the substitutions $\sigma_A = \{Z \rightarrow \text{b} \}$, $\sigma_B = \{Z \rightarrow \text{a} \}$.

"$\text{p(f}(X),W)$" is unifiable with "$\text{p}(Z, Z)$" under the substitutions $\sigma_A = \{W \rightarrow \text{f}(X) \}$, $\sigma_B = \{Z \rightarrow \text{f}(X) \}$.

"$\text{p(f}(X),X)$" is not unifiable with "$\text{p}(Z, Z)$".

There may be more than one set of substitutions that unifies two formulas. For example "$\text{p(a,f(a),}X)$" can be unified with "$\text{p(a,f(a),}Y)$" by substituting X for Y, or by substituting "a" for both X and Y, or by substituting f(a) for both X and Y, or by substituting "$\text{f}(W)$" for both X and Y ... However, the *best* way to unify them is to substitute X for Y (or vice versa), because all the other substitutions can be derived by further substitutions from it. It is called the *most general unifier* (mgu).

Resolution: Rules of Inference

1. (Factoring) Let ϕ be the clause $\alpha_1 \vee \alpha_2 \vee ... \vee \alpha_k$. Let α_i and α_j be two literals that are either both positive or both negative, and let σ be a single substitution that unifies α_i and α_j. Then infer $(\phi - \alpha_j)\sigma$.

Example: From "$\text{p(a,}X) \vee \text{p}(Y\text{,b}) \vee \text{q}(X, Y\text{,c})$" infer "$\text{p(a,b)} \vee \text{q(b,a,c)}$".

2. (Resolution) Let ϕ be the clause $\alpha_1 \vee \alpha_2 \vee ... \vee \alpha_k$, and let ψ be the clause $\beta_1 \vee \beta_2 \vee ... \vee \beta_m$. Suppose that $\alpha_i = \gamma$ and $\beta_j = \neg\delta$, where γ and δ are atomic and where γ unifies with δ under the substitutions σ_A and σ_B Then infer $(\phi - \alpha_i)\sigma_A \vee (\psi - \beta_j)\sigma_B$.

Examples: From “p(a,b) $\vee$ q(b,c)” and “$\neg$p(X,Y) $\vee$ r(X,Y)” infer “q(b,c) $\vee$ r(a,b)”.

From “man(socrates)” and “$\neg$man(X) $\vee$ mortal(X)”, infer “mortal(socrates)”.

From “man(socrates)” and “$\neg$man(X)” infer the empty clause.

Fact: Δ is an inconsistent set of clauses if and only if there is a derivation of the empty clause from Δ using the rules of resolution and of factoring.

Resolution: Proof Technique

To prove sentence ϕ from a set of axioms Γ:

Step 1. Set $\Delta = \Gamma \cup \{\neg\phi\}$;

Step 2. Skolemize Δ;

Step 3. Keep applying rules 1 and 2 to derive new sentences. If you succeed in deriving the empty clause, then ϕ is provable from Γ. If there is no way to derive the empty clause, then ϕ is not provable.

Example:

Given: 1. $\forall_{S1,S2}$ subset$(S1,S2) \Leftrightarrow [\forall_X$ member$(X,S1) \Rightarrow$ member$(X,S2)]$.
Prove: H. $\forall_{S1,S2,S3}$ [subset$(S1,S2) \wedge$ subset$(S2,S3)] \Rightarrow$ subset$(S1,S3)$.

Negation of H: 2. $\neg[\forall_{S1,S2,S3}$ [subset$(S1,S2) \wedge$ subset$(S2,S3)] \Rightarrow$ subset$(S1,S3)]$.

Converted to clausal form:
1a. $\neg$subset$(S1,S2)$ $\vee$ $\neg$member$(X,S1)$ $\vee$ member$(X,S2)$.
1b. member(sk0$(S1,S2)$,$S1$) $\vee$ subset$(S1,S2)$.
1c. $\neg$member(sk0$(S1,S2)$,$S2$) $\vee$ subset$(S1,S2)$.
2a. subset(sk1,sk2).
2b. subset(sk2,sk3).
2c. $\neg$subset(sk1,sk3).

From 2a and 1a, infer	3. $\neg$member(X,sk1) $\vee$ member(X,sk2).
From 2b and 1a, infer	4. $\neg$member(X,sk2) $\vee$ member(X,sk3).
From 3 and 4, infer	5. $\neg$member(X,sk1) $\vee$ member(X,sk3).
From 2c and 1b infer	6. member(sk0(sk1,sk3),sk1).
From 2c and 1c infer	7. $\neg$member(sk0(sk1,sk3),sk3).
From 6 and 5 infer	8. member(sk0(sk1,sk3), sk3).
From 7 and 8 infer	9. The empty clause.