

# Prenex and Skolem Normal Forms\*

Graeme Taylor

January 20, 2004

When working with sentences in predicate logic it is generally useful to rewrite in a normal form which ensures a standardisation of meaning. Prenex and Skolem form provide a means of dealing with sentences that feature quantifiers (for all and there exists) and they pave the way for more powerful (although arguably less elegant) normal forms which find application in logic programming systems such as Prolog.

First then some definitions:

- **A formula is in prenex normal form if the outermost operators are quantifiers (i.e. all the quantifiers are at the front).**  
So  $Q_1Q_2\dots Q_jM$ , where each  $Q_i$  is  $(\exists X_i)$  or  $(\forall X_i)$ ,  $X_i$  a variable and  $M$  quantifier-free is prenex normal form
- **A formula is in skolem form if it is in prenex normal form and only features universal  $(\forall)$  quantifiers.**

Of course, simply pulling quantifiers through to the front can cause problems if variables of the same name were quantified in different ways, and it is extremely unlikely that flipping all quantifiers to universal would preserve the original meaning. So it is implicit that when we seek a prenex/skolem form, we seek one which is logically equivalent to the sentence we had (this is not entirely true for skolem forms as shall be discussed).

## Prenex Normal form algorithm

Given a formula  $S$ , we desire a prenex normal form logically equivalent to  $S$ .

1. Replace all  $(A \Leftrightarrow B)$ s in subformulae with  $(A \Rightarrow B) \wedge (B \Rightarrow A)$ .
2. Then replace all  $\Rightarrow$  subformulae with  $((\neg A) \wedge B)$ .

---

\*First appeared on Everything2, at [http://www.everything2.com/index.pl?node\\_id=1514357](http://www.everything2.com/index.pl?node_id=1514357)

3. Rename bound variables so that no variable occurs in the scope of different quantifiers nor is bound and free in any given subformulae. This is a conversion; it is important to choose a previously used variable name as this could dramatically alter the meaning.
4. Move negations inside quantifiers-  $\neg(\forall X)(A)$  is  $(\exists X)(\neg A)$  and  $\neg(\exists X)(A)$  is  $(\forall X)(\neg A)$ .
5. Move all  $\wedge$ s and  $\vee$ s inside quantifiers-  $((QX)(A)opB)$  or  $(Aop(QX)(B))$  becomes  $(QX)(AopB)$  with  $op$  an operator,  $Q$  a quantifier.

Before looking at skolemisation, we will consider an example. Consider the statement "If any train is late, all trains are late". Then with  $t(x)$  meaning "x is a train",  $l(x)$  meaning "x is late" and  $x$  drawn from a domain of all objects in the universe we can convert this into a formula in a first order language as-

$$(\exists x)(t(x) \wedge l(x)) \Rightarrow (\forall x)(t(x) \Rightarrow l(x))$$

That is, if there is some  $x$  which is both a train and late, then it is true that for all objects, if the object is a train then it is late. So now we apply the rules to get PNF-

- $\neg(\exists x)(t(x) \wedge l(x)) \vee (\forall x)(\neg t(x) \vee l(x))$   
Replacement of implies with the equivalent or form.
- $\neg(\exists x)(t(x) \wedge l(x)) \vee (\forall y)(\neg t(y) \vee l(y))$   
Renaming the free variable in the second statement do draw a distinction between the  $x$  which we test for train- and lateness, and all other  $y$ 's for which we wish to imply lateness if they are trains.
- $((\forall x)(\neg t(x) \vee \neg l(x))) \vee ((\forall y)(\neg t(y) \vee l(y)))$   
Pulling negation into the first quantifier.
- $(\forall x)(\forall y)(\neg t(x) \vee \neg l(x) \vee (t(y) \Rightarrow l(y)))$   
Bringing quantifiers to the front is now allowed; for clarity I have rebuilt the implication at the end rather than leave it in terms of ORs, and in fact we could have kept it in this form throughout.

So we have our prenex normal form- for every  $x$  and  $y$  in our universe, either  $x$  wasn't a train,  $x$  wasn't late or  $y$  a train implies  $y$  late- so if  $x$  is a late train we have to conclude any train  $y$  is a late train  $y$  so our PNF is indeed logically equivalent. Now for skolem form.

The idea of skolemisation can be illustrated by the fact that whenever a  $\exists$  quantifier is invoked, there must indeed exist an object in our domain for which the statement is true. So we can give that object a name, which introduces a *skolem constant*. However, if existence was asserted after some for all... statements, then it is probable that the object meeting the criteria changes each time.

Nonetheless, knowing the values of the universally quantified variables allows us to determine which object in the universe is the one which we desire- so we can construct a skolem function which takes as inputs the universally quantified variables and returns that object. Finally, if we have a formulae rather than a sentence, that is to say there are free variables, then the truth of the existence of the object depends also on those free variables so we would extend our skolem function to incorporate those inputs too.

As an example, consider the statement "for all integers there exists a greater integer". Then letting  $g(x, y)$  mean  $y$  is greater than  $x$ , we could write this as  $(\forall x)(\exists y)g(x, y)$ . Now the  $y$  that needs to be supplied for any given  $x$  varies- we can't use any particular integer  $c$  as a skolem constant as setting  $x$  to  $c$  would yield falsehood ( $c > c$  is impossible). But a skolem function of  $x$ , e.g.  $f(x)$  where  $f(x)$  is interpreted as  $x+1$ , would work. Then our sentence becomes  $(\forall x)g(x, f(x))$  which is interpreted as  $(\forall x)g(x, x+1)$  i.e.  $x+1$  is greater than  $x$  - which is a true statement. So more rigorously let us define:

## Skolem Normal Form Algorithm

Given a formula  $S$  in PNF, we seek one in Skolem form

1. If  $S$  is already in skolem form, we are done.
2. If not, then  $S$  is of the form  $(\forall x_1)(\forall x_2)\dots(\forall x_n)(\exists y)M(W_1, \dots, W_k, x_1, \dots, x_n, y)$  - where the  $W$ 's are free variables and  $M$  is another formula in PNF. We wish to introduce a skolem function to eliminate  $(\exists y)$ . So pick a new function symbol, say  $f$  which will have arity  $n+k$  (universally quantified variables + free variables). Then rewrite as  $(\forall x_1)(\forall x_2)\dots(\forall x_n)M(W_1, \dots, W_k, x_1, \dots, x_n, f(W_1, \dots, W_k, x_1, \dots, x_n))$
3. Now, either  $M$  is in skolem form or it is not. So repeat at step 1 with  $M$  and hence continue with the introduction of skolem functions until all existential quantifiers have been removed.

It is worth noting that skolem form is not precisely logically equivalent to the original statement  $S$ . Indeed, the skolem form does not even make sense under the original interpretation of  $S$ - you need to extend the interpretation to give definitions of the skolem constants/functions. This is however possible for any interpretation in which  $S$  was true. Fortunately, as skolem form is more restrictive than the original formulation,  $S$  is a logical consequence of its skolem form (whenever the skolem form is true, so is  $S$ ). A formula is satisfiable iff its skolem form is satisfiable- whatever objects you use to satisfy  $S$  can be defined as the values of the skolem functions. Furthermore, in higher order languages logical equivalence can be maintained (if you allowed quantification of the functions).

Why use PNF and Skolem form? Having obtained the skolem form, the next step is to work towards a clausal form. A clause is a very simple rule- if a list of conditions is met, then one of another list must hold true. Thus a clausal form is a list of such clauses, which although unwieldy due to often escalating to exponential length is ideal for computer logic (see the headed horn clauses

of prolog). To obtain a clausal form, you take the skolem form, drop the universal quantifiers (hence the desire to only have universal quantifiers) to get a formula  $M$ , and then put  $M$  into conjunctive normal form, from which the list of clauses can be obtained.