Lecture 9
Normal forms for first-order logic
Equivalences, prenex form, Skolem form

Print version of the lecture in *Introduction to Logic for Computer Science*

presented by Prof Hongseok Yang

These lecture notes are very minor variants of the ones made by Prof James Worrell and Prof Christoph Haase for their 'Logic and Proof' course at Oxford.

In this lecture we show how to transform an arbitrary formula of first-order logic to an equisatisfiable formula in *Skolem form*. This translation is in preparation for our subsequent treatment of deduction using unification and resolution.

1 Equivalence and substitution

Two first-order formulas F and G over a signature σ are **logically equivalent**, denoted $F \equiv G$, if for all σ-assignments $\mathcal{A}$, we have $\mathcal{A} \models F$ iff $\mathcal{A} \models G$.

It is easy to check that all the propositional equivalences carry over to the first-order setting, e.g., we still have De Morgan's law $\neg(F \wedge G) \equiv (\neg F \vee \neg G)$, etc. Moreover, logical equivalence remains a congruence with respect to the Boolean connectives $\wedge$, $\vee$ and $\neg$, that is, $(F_1 \wedge G_1) \equiv (F_2 \wedge G_2)$ if $F_1 \equiv G_1$ and $F_2 \equiv G_2$, etc. In addition we have that if $F \equiv G$, then $\forall x\, F \equiv \forall x\, G$ and $\exists x\, F \equiv \exists x\, G$.

The following equivalences will play an important role in transforming formulas into Skolem form.

Proposition 1. Let F and G be arbitrary formulas. Then, the following hold for them.[1]

(A) $\neg\forall x F \equiv \exists x \neg F$ and $\neg\exists x F \equiv \forall x \neg F$.

(B) If x does not occur free in G then:

$$(\forall x F \wedge G) \equiv \forall x(F \wedge G), \qquad (\forall x F \vee G) \equiv \forall x(F \vee G),$$
$$(\exists x F \wedge G) \equiv \exists x(F \wedge G), \qquad (\exists x F \vee G) \equiv \exists x(F \vee G).$$

(C) $(\forall x F \wedge \forall x G) \equiv \forall x(F \wedge G)$ and $(\exists x F \vee \exists x G) \equiv \exists x(F \vee G)$.

(D) $\forall x \forall y F \equiv \forall y \forall x F$ and $\exists x \exists y F \equiv \exists y \exists x F$.

Proof. As an example, we prove the first equivalences in (A) and (B). For the former, we have

$$\mathcal{A} \models \neg\forall x F \text{ iff } \mathcal{A} \not\models \forall x F$$
$$\text{iff } \mathcal{A}_{[x \mapsto a]} \not\models F \text{ for some } a \in U_{\mathcal{A}}$$
$$\text{iff } \mathcal{A}_{[x \mapsto a]} \models \neg F \text{ for some } a \in U_{\mathcal{A}}$$
$$\text{iff } \mathcal{A} \models \exists x \neg F.$$

---

[1] Recall that quantifiers bind more tightly than the Boolean connectives. Thus, $\forall x F \wedge G$ means $(\forall x F) \wedge G$, not $\forall x(F \wedge G)$.

For the first equivalence in (B) we have

$$\mathcal{A} \models (\forall x F \wedge G) \text{ iff } \mathcal{A} \models \forall x F \text{ and } \mathcal{A} \models G$$
$$\text{iff for all } a \in U_\mathcal{A}, \mathcal{A}_{[x \mapsto a]} \models F \text{ and } \mathcal{A} \models G$$
$$\text{iff for all } a \in U_\mathcal{A}, \mathcal{A}_{[x \mapsto a]} \models F \text{ and } \mathcal{A}_{[x \mapsto a]} \models G \text{ (by the Relevance Lemma)}$$
$$\text{iff for all } a \in U_\mathcal{A}, \mathcal{A}_{[x \mapsto a]} \models F \wedge G$$
$$\text{iff } \mathcal{A} \models \forall x (F \wedge G).$$

$\square$

A particularly useful and readable normal form is the prenex normal form.

Definition 2. A formula is in **prenex form** if it can be written

$$Q_1 y_1 \, Q_2 y_2 \ldots Q_n y_n \, F,$$

where $Q_i \in \{\exists, \forall\}$, $n \geq 0$, and F contains no quantifiers. In this case F is called the **matrix** of the formula.

Example 3. We use Proposition 1 to transform the formula

$$\neg(\exists x \, P(x,y) \vee \forall z \, Q(z)) \wedge \exists w \, Q(w) \tag{1}$$

to prenex form by the following chain of equivalences:

$$\begin{aligned}
\neg(\exists x \, P(x,y) \vee \forall z \, Q(z)) \wedge \exists w \, Q(w) &\equiv (\neg \exists x \, P(x,y) \wedge \neg \forall z \, Q(z)) \wedge \exists w \, Q(w) \\
&\equiv (\forall x \, \neg P(x,y) \wedge \exists z \, \neg Q(z)) \wedge \exists w \, Q(w) \\
&\equiv \forall x \, \exists z \, (\neg P(x,y) \wedge \neg Q(z)) \wedge \exists w \, Q(w) \\
&\equiv \forall x \, \exists z \, \exists w \, ((\neg P(x,y) \wedge \neg Q(z)) \wedge Q(w)).
\end{aligned}$$

Note that in the above equational reasoning, we use the fact that logical equivalence is a congruence with respect to the Boolean operators (i.e., the Substitution Theorem).

Let F be a formula, x a variable, and t a term. Then, $F[t/x]$ (read "F with t for x") denotes the formula with t substituted for every *free* occurrence of x in F. This is analogous to the substitution that we have already seen in propositional logic. For example,

$$(\forall x \, P(x,y) \wedge Q(x))[t/x] = \forall x \, P(x,y) \wedge Q(t).$$

Note that x is not replaced in $P(x,y)$ as it is bound by the $\forall$ quantifier.

Formally, we define $F[t/x]$ by induction on terms and formulas as follows. On terms we have:

$$\begin{aligned}
c[t/x] &= c \quad \text{for } c \text{ a constant symbol} \\
y[t/x] &= y \quad \text{for } y \neq x \text{ a variable} \\
x[t/x] &= t \\
f(t_1, \ldots, t_k)[t/x] &= f(t_1[t/x], \ldots, t_k[t/x]) \quad \text{for } f \text{ a } k\text{-ary function symbol.}
\end{aligned}$$

We then extend the definition of $[t/x]$ to formulas as follows:

$$\begin{aligned}
P(t_1, \ldots, t_k)[t/x] &= P(t_1[t/x], \ldots, t_k[t/x]) \\
(\neg F)[t/x] &= \neg(F[t/x]) \\
(F \wedge G)[t/x] &= F[t/x] \wedge G[t/x] \\
(F \vee G)[t/x] &= F[t/x] \vee G[t/x] \\
(Qy \, F)[t/x] &= Qy \, (F[t/x]) \quad y \neq x \text{ a variable, } Q \in \{\forall, \exists\} \\
(Qx \, F)[t/x] &= Qx \, F \quad Q \in \{\forall, \exists\}.
\end{aligned}$$

A key fact about substitution is the following. The proof is in Appendix A.

Lemma 4 (Translation Lemma). *If t is term and F is a formula such that no variable in t occurs bound in F, then $\mathcal{A} \models F[t/x]$ iff $\mathcal{A}_{[x \mapsto \mathcal{A}(t)]} \models F$.*

To illustrate the necessity of the side-condition in the Translation Lemma, let F be the formula $\forall y P(x)$ and let $\mathcal{A}$ be the assignment with $U_{\mathcal{A}} = \{1, 2\}$, $P_{\mathcal{A}} = \{1\}$, $x_{\mathcal{A}} = 2$, and $y_{\mathcal{A}} = 1$. Then, $F[y/x] = \forall y\, P(y)$ and so $\mathcal{A} \not\models F[y/x]$. But $\mathcal{A}(y) = 1$ and so $\mathcal{A}_{[x \mapsto \mathcal{A}(y)]} \models F$. The reason we cannot apply the Translation Lemma in this case is that the variable y in the term to be substituted becomes bound by the quantifier $\forall y$ in F. This phenomenon is called *variable capture*.

In first-order logic, we can rename bound variables in a formula while preserving logical equivalence. For example, we have $\forall x\, P(x) \equiv \forall y\, P(y)$. This is similar to the fact that the definite integral $\int_0^{\infty} f(s)ds$ denotes the same value as $\int_0^{\infty} f(t)dt$. We make this idea formal as follows:

Proposition 5. Let $F = Qx\, G$ be a formula where $Q \in \{\forall, \exists\}$. Let y be a variable that does not occur in G. Then $F \equiv Qy\, (G[y/x])$.

Proof. We prove the proposition in the case of $\forall$; the case for $\exists$ is similar. Let $\mathcal{A}$ be an assignment. Then

$$
\begin{aligned}
\mathcal{A} \models \forall y\, (G[y/x]) \quad &\text{iff} \quad \mathcal{A}_{[y \mapsto a]} \models G[y/x] \text{ for all } a \in U_{\mathcal{A}} \\
&\text{iff} \quad \mathcal{A}_{[y \mapsto a][x \mapsto \mathcal{A}_{[y \mapsto a]}(y)]} \models G \text{ for all } a \in U_{\mathcal{A}} \text{ (Translation Lemma)} \\
&\text{iff} \quad \mathcal{A}_{[y \mapsto a][x \mapsto a]} \models G \text{ for all } a \in U_{\mathcal{A}} \\
&\text{iff} \quad \mathcal{A}_{[x \mapsto a][y \mapsto a]} \models G \text{ for all } a \in U_{\mathcal{A}} \\
&\text{iff} \quad \mathcal{A}_{[x \mapsto a]} \models G \text{ for all } a \in U_{\mathcal{A}} \text{ (Relevance Lemma)} \\
&\text{iff} \quad \mathcal{A} \models \forall x\, G.
\end{aligned}
$$

$\square$

2 Skolem form

A formula is **rectified** if no variable occurs both bound and free, and if all quantifiers in the formula refer to different variables. For example, the formula

$$\forall x\, \exists y\, P(x, f(y)) \wedge \forall y\, (Q(x, y) \vee R(x))$$

is not rectified since y is bound on two separate occasions and x occurs both free and bound. By renaming the bound variables, we obtain the following equivalent rectified formula:

$$\forall u\, \exists v\, P(u, f(v)) \wedge \forall y\, (Q(x, y) \vee R(x)).$$

In general, we can always obtain an equivalent rectified formula by renaming bound variables using Proposition 5.

Lemma 6. *Every formula is equivalent to a rectified formula.*

Given a rectified formula F, we can use the equivalences in Proposition 1 to convert F to an equivalent formula in rectified prenex form (RPF) by "pushing all quantifiers to the front" in the manner of Example 3.

Theorem 7. *Every formula is equivalent to a rectified formula in prenex form.*

We say that a formula in RPF is in *Skolem form* if it does not contain any occurrences of the existential quantifier. We can transform a formula in RPF to an equisatisfiable (though not necessarily logically equivalent) formula in Skolem form by using extra function symbols. For example, the formulas $\forall x \exists y P(x, y)$ and $\forall x P(x, f(x))$ are equisatisfiable. An assignment that satisfies the left-hand formula can be extended to an assignment satisfying the right-hand formula by interpreting f as a "selection function" that maps each x to some y such that $P(x, y)$ holds. More generally we have the following proposition.

Proposition 8. Let $F = \forall y_1 \forall y_2 \ldots \forall y_n \exists z\, G$ be a rectified formula. Given a function symbol f of arity n that does not occur in F,[2] write

$$F' = \forall y_1 \forall y_2 \ldots \forall y_n\, G[f(y_1, \ldots, y_n)/z].$$

Then, F and F' are equisatisfiable.

Proof. We prove that if F is satisfiable, then so is F'. The reverse direction is left as an exercise.

Suppose that $\mathcal{A} \models F$ for some assignment $\mathcal{A}$. We define an assignment $\mathcal{A}'$ that extends $\mathcal{A}$ with an interpretation of the function symbol f such that $\mathcal{A}' \models F'$.

Given $a_1, \ldots, a_n \in U_{\mathcal{A}}$, pick $a \in U_{\mathcal{A}}$ such that $\mathcal{A}_{[y_1 \mapsto a_n] \ldots [y_n \mapsto a_n][z \mapsto a]} \models G$ and define $f_{\mathcal{A}'}(a_1, \ldots, a_n) = a$. Since the function symbol f does not occur in G, we have

$$\mathcal{A}'_{[y_1 \mapsto a_n] \ldots [y_n \mapsto a_n][z \mapsto f_{\mathcal{A}'}(a_1, \ldots, a_n)]} \models G\,,$$

and so, by the Translation Lemma,

$$\mathcal{A}'_{[y_1 \mapsto a_n] \ldots [y_n \mapsto a_n]} \models G[f(y_1, \ldots, y_n)/z].$$

Since the above holds for all $a_1, \ldots, a_n \in U_{\mathcal{A}}$, we conclude that

$$\mathcal{A}' \models \forall y_1 \forall y_2 \ldots \forall y_n\, G[f(y_1, \ldots, y_n)/z].$$

$\square$

Example 9. We find an equisatifiable Skolem form of the formula

$$\forall x\, \exists y\, \forall z\, \exists w\, (\neg P(a, w) \vee Q(f(x), y)).$$

We apply Proposition 8, eliminating $\exists y$ and introducing a new function symbol g, yielding

$$\forall x\, \forall z\, \exists w\, (\neg P(a, w) \vee Q(f(x), g(x))).$$

Then, we eliminate $\exists w$ by introducing a new function symbol h, yielding

$$\forall x\, \forall z\, (\neg P(a, h(x, z)) \vee Q(f(x), g(x))).$$

Conversion to Skolem form: Summary

We convert an arbitrary first-order formula F to an equisatisfiable formula in Skolem form as follows:

1. Rectify F by systematically renaming its bound variables, yielding a logically equivalent formula F_1.
2. Using the equivalences in Proposition 1, move all the quantifiers in F_1 to the outside, yielding an equivalent formula F_2 in prenex form.
3. Repeatedly eliminate the outermost existential quantifier in F_2 until an equisatisfiable formula F_3 in Skolem form is obtained. (This process is called *Skolemisation*.)

A   Proof of The Translation Lemma

In this section, we give the proof of the Translation Lemma. The proof is quite technical and can be regarded as optional.

---

[2]In the case $n = 0$ we consider f as a constant symbol.

Given an assignment $\mathcal{A}$, we first show by induction on terms s that $\mathcal{A}(s[t/x]) = \mathcal{A}_{[x \mapsto \mathcal{A}(t)]}(s)$. The base cases are as follows:

$$\mathcal{A}(c[t/x]) = \mathcal{A}(c) = \mathcal{A}_{[x \mapsto \mathcal{A}(t)]}(c) \quad c \text{ a constant symbol}$$
$$\mathcal{A}(y[t/x]) = \mathcal{A}(y) = \mathcal{A}_{[x \mapsto \mathcal{A}(t)]}(y) \quad y \neq x \text{ a variable}$$
$$\mathcal{A}(x[t/x]) = \mathcal{A}(t) = \mathcal{A}_{[x \mapsto \mathcal{A}(t)]}(x).$$

For the induction step we have

$$
\begin{aligned}
\mathcal{A}(f(t_1, \ldots, t_k)[t/x]) &= \mathcal{A}(f(t_1[t/x], \ldots, t_k[t/x])) \\
&= f_{\mathcal{A}}(\mathcal{A}(t_1[t/x]), \ldots, \mathcal{A}(t_k[t/x])) \\
&= f_{\mathcal{A}}(\mathcal{A}_{[x \mapsto \mathcal{A}(t)]}(t_1), \ldots, \mathcal{A}_{[x \mapsto \mathcal{A}(t)]}(t_k)) \quad \text{(by induction hypothesis)} \\
&= f_{\mathcal{A}_{[x \mapsto \mathcal{A}(t)]}}(\mathcal{A}_{[x \mapsto \mathcal{A}(t)]}(t_1), \ldots, \mathcal{A}_{[x \mapsto \mathcal{A}(t)]}(t_k)) \\
&= \mathcal{A}_{[x \mapsto \mathcal{A}(t)]}(f(t_1, \ldots, t_k)).
\end{aligned}
$$

Next we use induction on formulas to show that for all formulas F, $\mathcal{A} \models F[t/x]$ iff $\mathcal{A}_{[x \mapsto \mathcal{A}(t)]} \models F$. The base case is that F is an atomic formula $P(t_1, \ldots, t_k)$ for a k-ary predicate symbol P. Then

$$
\begin{aligned}
\mathcal{A} \models P(t_1, \ldots, t_k)[t/x] \quad &\text{iff} \quad \mathcal{A} \models P(t_1[t/x], \ldots, t_k[t/x]) \\
&\text{iff} \quad (\mathcal{A}(t_1[t/x]), \ldots, \mathcal{A}(t_k[t/x])) \in P_{\mathcal{A}} \\
&\text{iff} \quad (\mathcal{A}_{[x \mapsto \mathcal{A}(t)]}(t_1), \ldots, \mathcal{A}_{[x \mapsto \mathcal{A}(t)]}(t_k)) \in P_{\mathcal{A}} \\
&\text{iff} \quad (\mathcal{A}_{[x \mapsto \mathcal{A}(t)]}(t_1), \ldots, \mathcal{A}_{[x \mapsto \mathcal{A}(t)]}(t_k)) \in P_{\mathcal{A}_{[x \mapsto \mathcal{A}(t)]}} \\
&\text{iff} \quad \mathcal{A}_{[x \mapsto \mathcal{A}(t)]} \models P(t_1, \ldots, t_k).
\end{aligned}
$$

The inductive cases for the propositional connectives are routine. The case for the universal quantifier $\forall y$, where $y \neq x$, is given below.

$$
\begin{aligned}
\mathcal{A} \models (\forall y F)[t/x] \quad &\text{iff} \quad \mathcal{A} \models \forall y (F[t/x]) \\
&\text{iff} \quad \mathcal{A}_{[y \mapsto d]} \models F[t/x] \text{ for all } d \in U_{\mathcal{A}} \\
&\text{iff} \quad \mathcal{A}_{[y \mapsto d][x \mapsto \mathcal{A}_{[y \mapsto d]}(t)]} \models F \text{ for all } d \in U_{\mathcal{A}} \quad \text{(induction hypothesis)} \\
&\text{iff} \quad \mathcal{A}_{[y \mapsto d][x \mapsto \mathcal{A}(t)]} \models F \text{ for all } d \in U_{\mathcal{A}} \quad (y \text{ does not occur in } t) \\
&\text{iff} \quad \mathcal{A}_{[x \mapsto \mathcal{A}(t)][y \mapsto d]} \models F \text{ for all } d \in U_{\mathcal{A}} \quad (y \neq x) \\
&\text{iff} \quad \mathcal{A}_{[x \mapsto \mathcal{A}(t)]} \models \forall y F.
\end{aligned}
$$

The case for the existential quantifier is similar to the above. This concludes the proof.