CISC 204   Class 15

Quantifiers, Variable Binding, and Variable Substitution

Text Correspondence: pp. 102–107

Main Concepts:

- *Scope: the range of a quantifier*
- *Bound variable: within the scope of some quantifier*
- *Free variable: not within the scope of any quantifier*

There are open questions about the variable mentioned in a quantifier and its presence or absence in the formula within its *scope*.

15.1 Scope of a Quantifier

Consider the formula

$$\forall x\,((P(x) \rightarrow Q(x)) \wedge S(x, y))$$

We can see that the variable x is *bound* because it is within the scope of the quantifier; here, it is within the parentheses. The variable x is *free* because it is not within the scope of a quantifier in this formula.

Definition: Scope of a Quantifier

Let ϕ be a formula in predicate logic, Q and S be quantifiers, x be the variable that is quantified, and ψ be a sub-formula of ϕ. The *scope* of the quantifier $(Qx\,\phi)$ is the formula ϕ minus every formula ψ in ϕ that quantifies the variable x; that is, minus every formula of the form $\psi = (Sx\,\cdot)$.

The qualification in the definition of scope is familiar to us as computer scientists. It means that a variable can be "re-declared" within a formula. This is legitimate, because it is allowed by the syntax, but because it is confusing we will try to avoid re-using a quantified variable.

15.2 Free Variables and Bound Variables

Definitions: Free and Bound Variables

Let ϕ be a formula in predicate logic. An occurrence of x in ϕ is *bound* if it is within the scope of ϕ.

Let ϕ be a formula in predicate logic. An occurrence of x in ϕ is *free* if it is not within the scope of ϕ.

Consider the more complicated formula

$$(\forall x\,(P(x) \to Q(x))) \to (\neg\, P(x) \vee Q(y)) \tag{15.1}$$

The parse tree for this formula is shown in Figure 15.1, taken from the textbook. Before turning to this diagram, a student should be sure to know which terms of this formula have bound variables and which terms have free variables. These are labeled in the parse tree.

The text has alternative definitions of free and bound variables, dome with reference to the parse tree. We can see that the parse tree is a useful representation but is not strictly necessary to understand and define the terms.

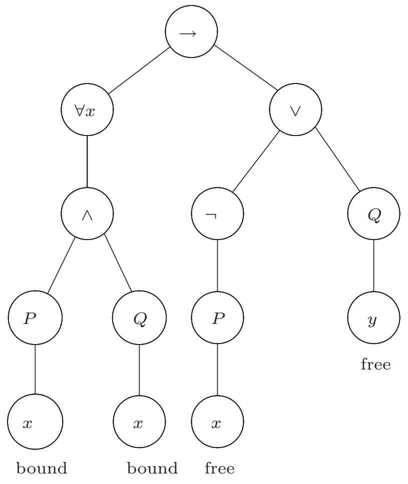

Figure 15.1: Parse tree for $(\forall x\,(P(x) \to Q(x))) \to (\neg\, P(x) \vee Q(y))$. From *Huth & Ryan, 2004*.

We may sometimes need to substitute the symbol for a variable with another symbol. This can arise, for example, when we want to instantiate a formula with definite values for the purposes of evaluating a model; this arose in predicate logic when we constructed a truth table.

We may also want to "clean up" a formula so that one symbol is not used in contexts that might cause confusion. This potential confusion arises in Equation 15.1, where the symbol x is bound in the antecedent part of the formula and free in the consequent part of the formula.

We can substitute a *variable* with a *term*, but cannot do the reverse. This allows us to take a simple formula and make it clearer – by re-naming variables – or more complex, by introducing a predicate in the place of a variable.

Definition: Substitution of Variables

Let ϕ be a formula, x be a variable, and t be a term. The ***substitution*** of t for x in ϕ is the result of the process of replacing every free occurrence of x in ϕ with t, written as $\phi[t/x]$.

For example, consider the formula

$$(\forall x\,((P(x) \to Q(x)) \land S(x,y))) \tag{15.2}$$

The parse tree for this formula is shown in Figure 15.2.

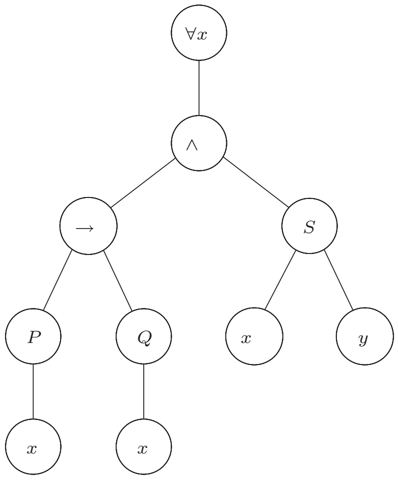

Figure 15.2: Parse tree for $(\forall x\,((P(x) \to Q(x)) \land S(x,y)))$. From *Huth & Ryan, 2004*.

What happens if we try to substitute the term $f(w,z)$ for x? We can see, from the formula and from the parse tree, that the variable x is bound in all occurrences. This means that the substitution process $\phi[f(w,z)/x]$ does not change the formula ϕ.

Being computer scientists, we are already aware of difficulties that can arise from re-naming variables; these especially occur from "global" replacements in a text editor. We need to prevent this from happening in variable substitution. A common, somewhat difficult, definition is used to address this situation.

15.3 Variable Substitution

Definition: Freedom to Substitute Variables

Let ϕ be a formula, x be a variable in ϕ, t be a term, and y be any variable in t. We say that t is *free for* x in ϕ if no free occurrence of y in t becomes a bound occurrence in ϕ as the result of the process of substitution.

The intention of this definition is straightforward, despite its complicated presentation. Before we perform a substitution process, we need to be sure that there are no "side effects" that arise. The specific side effect that we want to avoid is accidentally creating a bound variable from a free one.

Consider the simple formula

$$S(x) \wedge (\forall y\,(P(x) \rightarrow Q(y))) \tag{15.3}$$

The parse tree for this formula is shown in Figure 15.3.

For this example, we should be able to determine:

- Is $f(w, z)$ free for x in ϕ?
- Is $f(w, y)$ free for x in ϕ?

The answer to the first question is positive. There are two variables in this t, being w and z. Neither variable becomes bound in ϕ as a result of the substitution process.

The answer to the second question is negative. There are two variables in this t, being w and y. When we substitute $f(w, y)$ in the left conjunct, there is no problem because both w and y are free. However, when we substitute $f(w, y)$ in the right conjunct, the variable w remains free but the variable y becomes bound by the $\forall y$ quantifier. This formula is not free for $f(w, y)$ in y.

The essential concepts for substitution are that we cannot substitute a bound variable, and that we cannot substitute a term in a way that binds a variable in the term. These are familiar to us as computer scientists; predicate logic crisply states these concepts in a way that can later be automatically checked.

A useful problem for students to consider is this formula ϕ:

$$\exists x\,(P(y, x) \wedge (\forall x\,\neg Q(y, x) \vee P(y, z)))$$

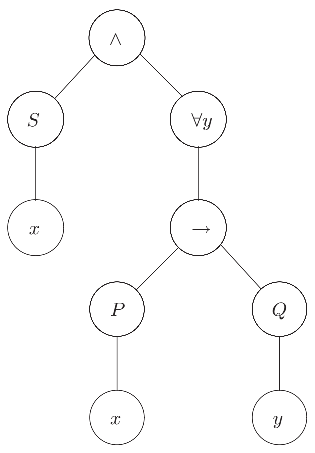

Figure 15.3: Parse tree for $S(x) \wedge (\forall y\, (P(x) \rightarrow Q(y)))$. From *Huth & Ryan, 2004*.

Students should be able to:

- Draw the parse tree for ϕ
- Identify all bound and free variable leaves in ϕ
- Compute $\phi[g(y,z)/z]$
- Determine the scope of $\exists x$ in the formula

Extra Notes

Practice Problems for Predicates

Let ϕ be the formula

$$\exists x\, (P(y,x) \wedge (\forall x\, \neg\, Q(y,x) \vee P(y,z)))$$

For this formula:

- Draw the parse tree for ϕ
- Identify every bound variable leaf and free variable leaf in ϕ
- Compute $\phi[g(y,z)/z]$
- State the scope of $\exists x$ in the formula ϕ

End of Extra Notes