9 Quantifiers

Logical statements usually include variables, which range over sets of possible instances, often referred to as universes. We use quantifiers to specify that something holds for all possible instances or for some but possibly not all instances.

**Universal and existential quantifiers.** We introduce the concept by taking an in-depth look at a result we have discussed in Chapter II.

**Euclid’s Division Theorem.** Letting $n$ be a positive integer, for every integer $m$ there are unique integers $q$ and $r$, with $0 \leq r < n$, such that $m = nq + r$.

In this statement, we have $n, m, q, r$ as variables. They are integers, so $\mathbb{Z}$ is the universe, except that some of the variables are constrained further, that is, $n \geq 1$ and $0 \leq r < n$. The claim is “for all” $m” “there exist” $q$ and $r$. These are quantifiers expressed in English language. The first is called the universal quantifier:

$$\forall x \ [p(x)]: \text{for all instantiations of the variable } x, \text{the statement } p(x) \text{ is true.}$$

For example, if $x$ varies over the integers then this is equivalent to

$$\ldots \land p(-1) \land p(0) \land p(1) \land p(2) \land \ldots$$

The second is the existential quantifier:

$$\exists x \ [q(x)]: \text{there exists an instantiation of the variable } x \text{ such that the statement } q(x) \text{ is true.}$$

For the integers, this is equivalent to

$$\ldots \lor q(-1) \lor q(0) \lor q(1) \lor q(2) \lor \ldots$$

With these quantifiers, we can restate Euclid’s Division Theorem more formally:

$$\forall n \geq 1 \forall m \exists q \ [0 \leq r < n \ [m = nq + r]].$$

**Negating quantified statements.** Recall de Morgan’s Law for negating a conjunction or a disjunction:

$$\neg (p \land q) \iff \neg p \lor \neg q;$$

$$\neg (p \lor q) \iff \neg p \land \neg q.$$  

The corresponding rules for quantified statements are

$$\neg (\forall x \ [p(x)]) \iff \exists x \ [\neg p(x)];$$

$$\neg (\exists x \ [q(x)]) \iff \forall x \ [\neg q(x)].$$

We get the first line by applying de Morgan’s first Law to the conjunction that corresponds to the expression on the left hand side. Similarly, we get the second line by applying de Morgan’s second Law. Alternatively, we can derive the second line from the first. Since both sides of the first line are equivalent, so are its negations. Now, all we need to do it to substitute $\neg q(x)$ for $p(x)$ and exchange the two sides, which we can because $\iff$ is commutative.

To summarize: negating a quantified statement reverses the quantifier and pulls the negation into the inner statement. If this inner statement is again quantified, we can repeat the operation, again reversing the quantifier and pulling the negation one level deeper. In the end, we reverse all quantifiers in sequence and pull the negation into the innermost, unquantified statement.

**Big-Oh notation.** We practice the manipulation of quantified statements by discussing the big-Oh notation for functions. It is commonly used in statements about the convergence of an iteration or the running time of an algorithm. We write $\mathbb{R}^+$ for the set of positive real numbers.

**Definition.** Let $f$ and $g$ be functions from $\mathbb{R}^+$ to $\mathbb{R}^+$. Then $f = O(g)$ if there are positive constants $c$ and $n_0$ such that $f(x) \leq cg(x)$ whenever $x > n_0$.

This notation is useful in comparing the asymptotic behavior of the functions $f$ and $g$, that is, beyond a constant $n_0$. If $f = O(g)$ then $f$ can grow at most a constant times as fast as $g$. For example, we do not have $f = O(g)$ if $f(x) = x^2$ and $g(x) = x$. Indeed, $f(x) = xg(x)$ so there is no constant $c$ such that $f(x) \leq cg(x)$, because we can always choose $x$ larger than $c$ and $n_0$ and get a contradiction. We rewrite the definition in more formal notation. The statement $f = O(g)$ is equivalent to

$$\exists c > 0 \exists n_0 > 0 \forall x \in \mathbb{R} \ [x > n_0 \Rightarrow f(x) \leq cg(x)].$$

We can simplify by absorbing the constraint of $x$ being larger than the constant $n_0$ into the last quantifying statement:

$$\exists c > 0 \exists n_0 > 0 \forall x > n_0 \ [f(x) \leq cg(x)].$$
We have seen above that negating a quantified statement reverses all quantifiers and pulls the negation into the unquantified statement. Recall that \( \neg(p \Rightarrow q) \) is equivalent to \( p \land \neg q \). Hence, the statement \( f \neq O(g) \) is equivalent to
\[
\forall c > 0 \forall n_0 > 0 \exists x \in \mathbb{R} \ [x > n_0 \land f(x) > c g(x)].
\]
We can again simplify by absorbing the constraint on \( x \) into the quantifying statement:
\[
\forall c > 0 \forall n_0 > 0 \exists x > n_0 \ [f(x) > c g(x)].
\]

**Big-Theta notation.** Recall that the big-Oh notation is used to express that one function grows asymptotically at most as fast as another, allowing for a constant factor of difference. The big-Theta notation is stronger and expresses that two functions grow asymptotically at the same speed, again allowing for a constant difference.

**Definition.** Let \( f \) and \( g \) be functions from \( \mathbb{R}^+ \) to \( \mathbb{R}^+ \). Then \( f = \Theta(g) \) if \( f = O(g) \) and \( g = O(f) \).

Note that in big-Oh notation, we can always increase the constants \( c \) and \( n_0 \) without changing the truth value of the statement. We can therefore rewrite the big-Theta statement using the larger of the two constants \( c \) and the larger of the two constants \( n_0 \). Hence, \( f = \Theta(g) \) is equivalent to
\[
\exists c > 0 \exists n_0 > 0 \forall x > n_0 \ [f(x) \leq c g(x) \land g(x) \leq c f(x)].
\]
Here we can further simplify by rewriting the two inequalities by a single one: \( \frac{1}{c}g(x) \leq f(x) \leq c g(x) \). Just for practice, we also write the negation in formal notation.
The statement \( f \neq \Theta(f) \) is equivalent to
\[
\forall c > 0 \forall n_0 > 0 \exists x > n_0 \ [f(x) > c g(x) \lor g(x) > c f(x)].
\]
Because the two inequalities are connected by a logical or, we cannot simply combine them. We could by negating it first, \( \neg(\frac{1}{c}g(x) \leq f(x) \leq c g(x)) \), but this is hardly easier to read.

**Big-Omega notation.** Complementary to the big-Oh notation, we have

**Definition.** Let \( f \) and \( g \) be functions from \( \mathbb{R}^+ \) to \( \mathbb{R}^+ \). Then \( f = \Omega(g) \) if \( g = O(f) \).

In formal notation, \( f = \Omega(g) \) is equivalent to
\[
\exists c > 0 \exists n_0 > 0 \forall x > n_0 \ [f(x) \geq c g(x)].
\]
We may think of big-Oh like a less-than-or-equal-to for functions, and big-Omega as the complementary greater-than-or-equal-to. Just as we have \( x = y \) iff \( x \leq y \) and \( x \geq y \), we have \( f = \Theta(g) \) iff \( f = O(g) \) and \( f = \Omega(g) \).

**Little-oh and little-omega notation.** For completeness, we add notation that corresponds to the strict less-than and greater-than relations.

**Definition.** Let \( f \) and \( g \) be functions from \( \mathbb{R}^+ \) to \( \mathbb{R}^+ \). Then \( f = o(g) \) if for all constants \( c > 0 \) there exists a constant \( n_0 \) such that \( f(x) < cg(x) \) whenever \( x > n_0 \). Furthermore, \( f = \omega(g) \) if \( g = o(f) \).

This is not equivalent to \( f = O(g) \) and \( f \neq \Omega(g) \). The reason for this is the existence of functions that cannot be compared at all. Consider for example \( f(x) = x^2(\cos x + 1) \). For \( x = 2k\pi \), \( k \) a non-negative integer, we have \( f(x) = 2x^2 \), while for \( x = (2k + 1)\pi \), we have \( f(x) = 0 \). Let \( g(x) = x \). For even multiples of \( \pi \), \( f \) grows much faster than \( g \), while for odd multiples of \( \pi \), \( f \) grows much slower than \( g \), namely not at all. We rewrite the little-Oh notation in formal notation. Specifically, \( f = o(g) \) is equivalent to
\[
\forall c > 0 \exists n_0 > 0 \forall x > n_0 \ [f(x) < c g(x)].
\]
Similarly, \( f = \omega(g) \) is equivalent to
\[
\forall c > 0 \exists n_0 > 0 \forall x > n_0 \ [f(x) > \frac{1}{c} g(x)].
\]
In words, no matter how small our positive constant \( c \) is, there always exists a constant \( n_0 \) such that beyond that constant, \( f(x) \) is larger than \( g(x) \) over \( c \). Equivalently, no matter how big our constant \( c \) is, there always exists a constant \( n_0 \) such that beyond that constant, \( f(x) \) is larger than \( c \) times \( g(x) \). We can thus simplify the formal statement by substituting \( [f(x) > c g(x)] \) for the inequality.

**Summary.** We have introduced quantifiers in logical statements and discussed how to generalize de Morgan’s Law to negate quantified statements. Finally, we have practiced this operation on formal definitions of the big-Oh and related notations for functions.