Chapter 2

Quantified Expressions

We learned in Chapter 1 that variables are unwelcome in propositions. Variables are, however, incredibly useful in math and programming, making their utility in logic no surprise. In this chapter we add variables (and their associated baggage) to propositions, as we continue on the trail to proofs.

2.1 Propositions to Predicates

Flashback: We said in Chapter 1 that we "consider variables to be 'illegal' within propositions." That's because adding variables to a proposition requires the introduction of several new concepts that explain and bind the variables. We had enough to talk about as it was.

Without the additional complications, predicates and propositions are very similar.

Definition 12: Predicate *predicate*

A statement that includes at least one variable and will evaluate to either true or false when all variables are assigned values is a *predicate* (a.k.a. a *propositional function*).

As we did with propositions, we will be giving predicates labels for ease of reference. To help distinguish predicates from propositions, we will use uppercase letters for predicates. Also, we will follow the predicate label with a list

of the variables used by the definition of the predicate, much as a program's invocation of a subprogram is followed by the parameters of the call.

Example 33:

Here are two examples of basic predicates:

$H(t) : (0 \leq t) \wedge (t < 24)$
$A(x, y) : x$ is on the y

The first predicate evaluates to true when the argument is in the range $[0 \ldots 23]$, which are the legal hour values in military (a.k.a. 24-hour) time. The second example demonstrates a two-variable predicate dealing with spacial orientation.

Something's missing from those examples: Declarations of the variables. Many programming languages require that variables be declared (added to the code's scope, assigned a type, etc.) before they are assigned values. Other languages permit variables to take on such characteristics dynamically during execution. In logic, we take the first approach: You must state the *domain* of a variable when you use it in a predicate.

domain

Definition 13: Domain

The *domain* (or *universe*) of a variable is the collection of values from which its value may be drawn.[1]

You can think of variables in a program as having domains, too. For example, in Java a variable declared to be of type `int` has a domain consisting of the integers from -2,147,483,648 through 2,147,483,647.

[1] More formally, the term is *domain of discourse* or *universe of discourse*, but who wants to say or write those?

 Copyright © Lester I. McCann Please do not distribute; thx!

Example 34:

Let's fix the predicates in Example 33:

$$H(t) : (0 \leq t) \wedge (t < 24),\ t \in \mathbb{Z}$$

We could have used $\mathbb{R}$ or even $\mathbb{Z}^*$ for t's domain, but $\mathbb{Z}$ is all we really need – we want to exclude non-integer hours, and the predicate's condition already deals with negative values.

$$A(x, y) : x \text{ is on the } y,\ x, y \in \text{Furniture}$$

x and y are taken from the same domain in the second example. That's not a requirement; they can be from separate domains if that suits your needs, as in:

$$A(x, y) : x \text{ is on the } y,\ x \in \text{Toys}, y \in \text{Furniture}$$

which allows us to consider the logical values of $A(\text{ball}, \text{table})$ and $A(\text{go-kart}, \text{piano})$.

To avoid misunderstandings, here are the rules for domains that we will follow in this book:

1. Each variable's domain is stated after the predicate.
2. Domains may be no more than one step more specific than "Things."
3. Operators may not be concealed within domains.

Like most rules, these have lots of room for loopholes. Example 35 tries to clear up some of the more common ones.

Example 35:

Consider this predicate:

$$G(x) : x \text{ contains gossip},\ x \in \text{supermarket tabloids}$$

The domain "supermarket tabloid" runs afoul of rules (2) and (3). This

 Copyright © Lester I. McCann Please do not distribute; thx!

domain is for a specific kind of publication, which suggests that the domain ought to be "magazines" or "periodicals." Further, it merges the characteristics of "sold in supermarkets" and "is a tabloid." As the last sentence suggests, the merger is best accomplished with AND. We should be clear about that by creating separate predicates for each of those characteristics:

$G(x)$: x contains gossip, $x \in$ magazines
$S(x)$: x is sold in supermarkets, $x \in$ magazines
$T(x)$: x is a tabloid, $x \in$ magazines

We can combine them with ANDs to recapture the original meaning: $S(x) \wedge T(x) \wedge G(x)$, $x \in$ magazines — a supermarket tabloid that contains gossip.

The quantification examples in sections 2.3 and 2.4 will show the value of these rules. You may be viewing them as causing extra clutter ("Now we need multiple predicates instead of just one!"), but that 'clutter' helps extract the meaning that is being expressed by the statement.

2.2 Evaluating Predicates

This may seem straight-forward; if you want to evaluate a predicate, plug in a value and produce a result. But that's not the only way to do it.

2.2.1 Evaluating Predicates on Specific Domain Elements

Much as we don't know the return value of a square root function until we pass it a value, until we assign the variables of a predicate appropriate values, we cannot determine its the truth value.

Example 36:

Consider this predicate:

$O(x)$: x is evenly divisible by 2, $x \in \mathbb{R}$

Supplying $O()$ with any multiple of two will make it true: $O(8)$, $O(-4)$,

 Copyright © Lester I. McCann Please do not distribute; thx!

and $O(0)$ are all true. All other real values cause $O()$ to be false, including 1, $\frac{5}{7}$, and π.

Notice that when we supply values for the variables, we are converting predicates to propositions. *x is evenly divisible by 2* is a fine predicate (especially with a domain!), but an unacceptable proposition. However, *7 is evenly divisible by 2* is a proposition, one that happens to be false.

2.2.2 Quantification: Evaluating Predicates on Sets of Domain Elements

We aren't restricted to assigning predicate variables individual values to treat the predicates as propositions. A more general approach that permits us to express some common situations compactly works by considering certain subsets of the variables' domains instead of specific individual values. The idea is called *quantification*, and we will consider two types: *quantification*

1. *Universal Quantification* (∀, LaTeX: `\forall`) requires that the entire domain of values be considered .
2. *Existential Quantification* (∃, LaTeX: `\exists`) considers one or more (which could include all) members of the domain.

When applying quantification to a predicate, we will use one type of quantifier or the other (but never both) with each variable of the predicate. We consider quantified variables to be *bound* by the quantifier. In some contexts, it's possible for some variables not to be quantified; such variables are considered to be *free* (a.k.a. *unbound*). Sorry; none of those contexts appear in this book. We'll be quantifying all of our predicates' variables.

Definition 14: Bound Variable *bound variable*

A quantified variable within a predicate is a *bound variable*.

 Copyright © Lester I. McCann Please do not distribute; thx!

free variable

Definition 15: Free Variable

An unquantified variable within a predicate is a *free variable*.

Table 11 shows the situations in which quantified expressions are true or false.

Table 11: Evaluating $\forall$ and $\exists$

This expression:	is True when:	is False when:
$\forall x\, P(x),\ x \in D$	every $d \in D$ makes $P()$ true	at least one $d \in D$ makes $P()$ false
$\exists x\, P(x),\ x \in D$	at least one $d \in D$ makes $P()$ true	every $d \in D$ makes $P()$ false

Quantified expressions are used frequently in human languages. For example, in English:[2]

- "Most politicians are slimy weasels"
- "None of those peaches are ripe"
- "Few students like taking discrete math exams"
- "That bag contains at least two pieces of candy"

There are quantifications that cannot be adequately expressed in first-order logic. For example, how many politicians can be considered to be "most," and how many students are there in a "few?" Many quantifications can be expressed in FOL, although it is sometimes a challenge. In particular, the second and fourth of those examples can be expressed; we'll see how before the end of this chapter.

The next two sections cover $\forall$ and $\exists$ in detail. For now, here's a simple example that should help distinguish them.

[2] A joke: "What do you call a person who speaks three languages? Trilingual. What do you call a person who speaks two languages? Bilingual. What do you call a person who speaks one language? American." If you want examples in other languages, you need a different author!

 Copyright © Lester I. McCann Please do not distribute; thx!

Example 37:

Let $S(x) : x^2 < 9,\ x \in \mathbb{Z}^+$.

Are there any members of the set $\mathbb{Z}^+$ that make $S(x)$ true? Yes; try $x = 2$. $S(2)$ is the proposition "$4 < 9$", which is true. Because at least one member of the domain makes $S()$ true, we can say that $\exists x\, S(x)$ is true. To read that aloud, we would say "There exists a value for x such that $S(x)$ is true."

In order for $\forall x\, S(x)$ to be true, *all* members of the domain must make $S()$ evaluate to true — no exceptions. Showing this can be a significant task, one often requiring that formal reasoning techniques be applied. However, to show that a universally quantified expression is false, all we need is one domain member that makes $S()$ false. For this predicate, that's easy: Any integer above 3 (or under -3) makes $S()$ false. Thus, $S()$ is not true for all domain elements; the expression $\forall x\, S(x)$ is false.

Let's take the example a step further by changing the domain from $\mathbb{Z}$ to $\{0, 1, 2\}$. Because every member of this new domain makes $x^2 < 9$ true, $\forall x\, S(x)$ is now true. Read aloud: "For all x in the domain, $S(x)$ is true." $\exists x\, S(x)$ is also true, because at least one element of the domain makes $S()$ true. If $\forall x\, S(x)$ is true, then $\exists x\, S(x)$ is also true.

2.3 More Detail on Universal Quantification

We already know (from Section 2.2) that $\forall x\, P(x)$ is true only when all members of the domain make $P()$ true. Otherwise, the quantification is false. Happily (?), universal quantification is more interesting than that.

2.3.1 Identifying Universal Quantification in English

There are some words and phrases in English that can be indications of universal quantification, including "All," "For all," "Every," and "Any." Unfortunately, it's not as simple as doing a text search for these phrases to location universal quantification.

 Copyright © Lester I. McCann Please do not distribute; thx!

Example 38:

Here's a sentence that uses "all" in a non-universal way:

Darling, I will come home with all possible speed!

Unless this person is planning to make the trip an uncountably infinite[3] number of times, 'with all possible speed' is just a fancy way of saying 'as soon as possible.'

Example 39:

Sometimes, none of those phrases are used, yet the sentence is still clearly universal in nature. For instance:

Car doors open outward.

The point: You have to read statements carefully to understand their intended meaning. You can't simply learn and follow a tidy set of rules.

2.3.2 Nesting Universal Quantifiers

Recall that it's possible for a predicate to have more than one variable, as in $L(a, b) : a < b$. When working with such predicates, we have to quantify all of their variables. But what does it mean for an expression to have two universal quantifiers?

Let's start with the notation. Authors have a small variety of ways to write nested quantifiers. We'll stick with the classic: $\forall a \forall b\, L(a, b)$. Other options include $(\forall a)(\forall b)L(a, b)$, $\forall a \forall b : L(a, b)$, and $\forall a, b\, L(a, b)$.

In English, assuming a domain of real numbers for both a and b, we'd express that nested quantification as "For all pairs of real numbers, the first is less than the second." This isn't true, of course, but that's OK – we need to be able to represent both true and false expressions.

[3] Will be covered in a future chapter. Really!

 Copyright © Lester I. McCann Please do not distribute; thx!

Example 40:

When we subtract one integer from another (e.g., $4-6$), the result is also an integer. Mathematicians say that $\mathbb{Z}$ is *closed* under subtraction. How can we express it as a quantified expression?

We need to say that subtracting any integer from any other integer produces an integer. We don't want to create an expression that is overly restrictive. For instance, it's possible for all three integers to be the same ($0-0=0$). We don't want to create an expression that excludes such possibilities.

Consider this attempt: $\forall x\,[(x-x)\in\mathbb{Z}],\ x\in\mathbb{Z}$. x represents integers, and so $x-x$ represents the subtraction of two integers, right? Close: $x-x$ represents the action of subtracting an integer from itself (as in $19-19$ or $(-4)-(-4)$). We need to say that the two integers have no other connection (besides both being integers). Here's the expression we need:

$$\forall x\forall y\,[(x-y)\in\mathbb{Z}],\ x,y\in\mathbb{Z}.$$

Adding y allows the expression to represent the subtraction of two integers that may or may not be the same integer.

Example 41:

Integers are closed under subtraction, but not under division. $4/3$, for instance, doesn't produce an integer result.[4] Real numbers are closed under division, but we have to be careful to exclude division by 0. How can we add that exception to the expression? Here's how:

$$\forall a\forall b\,[(b\neq 0)\rightarrow((a/b)\in\mathbb{R})],\ a,b\in\mathbb{R}.$$

This can be read as, "For any real numbers a and b, if b is not zero, the division of a by b produces a real number," or, more conversationally, "Dividing a real by a non-zero real produces a real." The implication 'protects' the division from the case that $b=0$, much as we'd use an `if` statement in a program for the same purpose.

 Copyright © Lester I. McCann Please do not distribute; thx!

Example 41 demonstrates the difference between *logical English* and *conversational English*. If someone looks over your shoulder, sees you writing a quantified expression, and asks, "What's that mean?", you'd want to provide a conversational answer. In many examples in this chapter, we will provide both logical and conversational versions, because it's reasonably easy to create the former from a quantified expression, but not the latter, at least not without practice. Moving from logic to conversational English via logical English usually makes the task easier (while cutting down on errors).

If you are still having trouble wrapping your mind around this nested quantification idea, viewing it from the perspective of nested loops in a programming language may help. Consider this nested loop pseudocode example that returns `True` only if all nine products are negative:

```
for i in {1,2,3}:
    for j in {-2,-4,-6}:
        if (i * j >= 0)
            return False
        endif
    endfor
endfor
return True
```

The loops are a short-cut that help us avoid having to write nine separate *if* statements. They systematically pair every member of the first set (`i`'s domain, if you will) with all members of the second set. This is also what nested quantification represents. Here's the logical notation version:

$$\forall i \forall j \, (i * j < 0), \, i \in \{1, 2, 3\}, j \in \{-2, -4, -6\}.$$

Each quantifier can be thought of as representing a loop that covers the corresponding variable's domain. The condition that we want to establish is the negation of the code's condition, because we want the code to stop checking as soon as we know what the result will be — remember, to show that a universal quantification is false, all we need is one exception.

[4] Integers are closed under *integer* division (4\3), but this example is using real division.

 Copyright © Lester I. McCann Please do not distribute; thx!

2.3.3 An Extended Universal Quantification Example

The next example brings together many of the quantification ideas we have covered so far.

Example 42:

Problem: Express the sentence "*Everyone who reads this book smells fresh*" in logical notation.

Solution: We will (intentionally!) make several errors as we work toward the correct answer to this problem, so that you can avoid making the same errors yourself.

The first word ("everyone") suggests that this is a universal quantification situation, and thus that we should be using predicates instead of propositions. But, what are the predicates? It's tempting to create a single predicate:

$P(x)$: x is a fresh-smelling reader of this book, $x \in$ People

The problem: There are two ideas covered by that predicate — readers of this book, and people who smell fresh. They need to be separated.[5] Attempt #2:

$R(x)$: x reads this book, $x \in$ People who smell fresh

Nice try, but as we talked about earlier, we want the domain to be no more than one step removed from "Things." "People" is a fine domain, but in isolation. This means that we need a second predicate for fresh-smellers. Attempt #3:

$R(x)$: x reads this book, $x \in$ People
$F(x)$: x smells fresh, $x \in$ People

Not bad, but $R()$ needs some help. "This book" is pretty vague. We can tidy this up by making $R()$ into a binary predicate. Attempt #4:

$R(x, y)$: x reads y, $x \in$ People, $y \in$ Books

 Copyright © Lester I. McCann Please do not distribute; thx!

$F(x)$: x smells fresh, $x \in$ People

That looks good. We have the quantification, we have the predicates, and when we use $R()$ in our final expression, we'll use the name of this book in place of y. One last step: We just need to combine the predicates with a suitable logical operator. The given sentence combines these two characteristics, which makes AND seem like a reasonable operator choice:

$\forall x\,[R(x, \text{“Discrete Structures for Computer Science”}) \wedge F(x)]$,
$x \in$ People

But what does that mean? In English, we would read this as, "Everyone reads this book and smells fresh." Everyone? All ∼7 billion of us? That's overstepping the meaning of the sentence. We just want to say that those people who read this book smell fresh. Put another way: If a person reads this book, they smell fresh. Looks like a job for . . . implication!

$\forall x\,[R(x, \text{“Discrete Structures for Computer Science”}) \rightarrow F(x)]$,
$x \in$ People

In not-very-conversational English: "Considering all people, if a person reads this book, that person smells fresh." Stated conversationally: "Everyone who reads this book smells fresh," which is exactly what we were given.

Example 42 showed that implication was the appropriate operator. That is often the case: In universally-quantified expressions, when you appear to have a choice between AND and implication, the correct choice is usually implication.

2.4 More Detail on Existential Quantification

Recall from Section 2.2.2 that existentially quantified expressions are true so long as a member of the domain makes it true. More than one is fine, but we need at least one. To make such existential quantifications false, none of the domain members can make them true.

[5] No, not because fresh-smelling people want nothing to do with readers of this book . . .

 Copyright © Lester I. McCann Please do not distribute; thx!

2.4.1 Identifying Existential Quantification in English

Like universal quantification, there are some key words and phrases in English that can suggest existential quantification as the appropriate choice. These include "Some," "A few," "More than one," and "Several." These words and phrases could also have nothing to do with quantification, and there are many other ways to suggest 'there exists' — always read the English statement carefully to understand its intended meaning.

Example 43:

You know, quite a few people need food to survive . . .

"Quite a few" is clearly existential, stopping short of universal. But looking at the complete sentence, universal certainly seems more appropriate, under the reasonable assumption that all people need food to live. The tone of the sentence is one of sarcasm, as though the speaker is chiding another person for an obvious error in reasoning. Such undercurrents of meaning are a key complication when converting English to logic.

Example 44:

That was some game!

This is the sort of remark one friend might say to another at the end of a 8-7 (that is, unusually high-scoring) soccer game. "Some" suggests existential quantification, but after more thought it's clear that this sentence is devoid of definite content. It's just not worth trying to express logically, because we don't know the information supporting the remark. We can imagine converting "The total number of goals scored in today's game exceeds the 98th percentile for soccer matches" to logic, but it's much harder to imagine it being used to break an awkward silence in casual post-game conversation.

 Copyright © Lester I. McCann Please do not distribute; thx!

2.4.2 Nesting Existential Quantifiers

Having covered nested universal quantifiers in section 2.3.2, this section should be easier to follow, because the ideas are much the same. We'll even use the same starting example: $L(a, b) : a < b,\ a, b \in \mathbb{R}$. Using nested existential quantification, we create $\exists a \exists b\, L(a, b),\ a, b \in \mathbb{R}$, which can be expressed in English as "There exist two real numbers a and b such that $a < b$," or more conversationally as "There are two reals such that the first is less than the second." We need just one true example to verify the truth of that expression, such as $2.15 < 6.736$.

Example 45:

Problem: Convert $\exists c \exists d\ \neg D(c, d),\ c \in$ Plants, $d \in$ Defenses to conversational English, where $D(c, d)$ represents "c is protected by d."

Solution: A good place to start is to remember how to handle negations. The negation of "c is protected by d" is "c is not protected by d."

From there, we can construct a formal English version: "There exists a plant and there exists a defense such that the plant is not protected by that defense." Conversationally, "Some plants don't have some defenses" means the same thing.

In section 2.6, we will learn how to correctly relocate negations within quantified expressions. Doing so is often helpful when interpreting the meaning of a complex expression.

Example 46:

Problem: Express *There are three integers such that the sum of the squares of the first two equals the square of the third* in logic.

Solution: Let's let r, s, and t represent the three integers. The sentence says that we need to add together the squares of r and s $(r^2 + s^2)$ to get a result that equals the square of other integer (t^2). That is, $\exists r \exists s \exists t\, [r^2 + s^2 = t^2],\ r, s, t \in \mathbb{Z}$. If you remember your basic geometry, you'll know that this expression is true — it's the Pythagorean Theorem.

 Copyright © Lester I. McCann Please do not distribute; thx!

There's no limit on the quantity of variables (and therefore on the quantity of quantifiers) we can have in an expression. Use as many as you need to say what needs to be said.

2.4.3 An Extended Existential Quantification Example

Ready to see a more involved example of nested existential quantification? Sure, you say that now ...

Example 47:

Problem: Express this sentence in logic: *There are some tow trucks in some of the downtown parking garages.*

Solution: Compared to our examples so far, this sentence has a lot of detail. So that the structure of the answer doesn't get obscured by that detail, we will temporarily suspend our "one concept per predicate" rule. Once we have the structure in place, we'll reinstate that rule and produce the final answer.

We have three basic ideas for which predicates are needed: Tow trucks, downtown parking garages, and storing the trucks in the garages.

$T(x)$: x is a tow truck, $x \in$ Vehicles
$G(x)$: x is a downtown parking garage, $x \in$ Buildings
$I(x, y)$: x is inside of y, $x \in$ Vehicles, $y \in$ Buildings

We need to say that some trucks are in some garages. Maybe this will work:

$\exists v \exists b \; I(T(v), G(b)), v \in$ Vehicles, $b \in$ Buildings

No, it won't. $I()$ accepts a vehicle and a building, but $T()$ and $G()$ return boolean values. It's like trying to pass a character string to a square root function. OK, let's separate the predicates and try combining them with implications (because that's what we ended up using in

 Copyright © Lester I. McCann Please do not distribute; thx!

the extended universal quantification example of section 42):

$$\exists v\ [T(v) \rightarrow \exists b\ (G(b) \rightarrow I(v,b))], v \in \text{Vehicles}, b \in \text{Buildings}$$

This is a WFF, but still isn't correct. In logical English, it reads as "There exists a vehicle such that, if the vehicle is a tow truck, then there exists a building such that, if the building is a downtown parking garage, then the vehicle is in the garage."[6] The problem is that the conditionals don't help here. They were important for the universal quantification example, because we needed to cover every member of a subset of the domain. Here, with existential quantification, we don't need to say *if* a tow truck exists; rather, we merely need to say that one does. For that purpose, AND is all we need to express our desired meaning:

$$\exists v\ [T(v) \wedge \exists b\ (G(b) \wedge I(v,b))], v \in \text{Vehicles}, b \in \text{Buildings}$$

In logical English, and playing fast and loose with plurals to keep it from being an even bigger mess, we have "There exists at least one vehicle such that the vehicles are tow trucks and there exists at least one building such that the buildings are downtown parking garages and the vehicles are in the garages," or, conversationally, "Some tow trucks are parked in a few downtown parking garages." That's not exactly the same wording as we were given, but it's logically equivalent, which is fine.

Now that we have the structure the way we want it, we need to fix the logical expression so that each predicate covers just one detail. We need several more predicates; here's the complete set:

$T(x) : x$ is a truck, $x \in$ Vehicles
$W(x) : x$ is for towing, $x \in$ Vehicles
$D(x) : x$ is located downtown, $x \in$ Buildings
$G(x) : x$ is a garage, $x \in$ Buildings
$P(x) : x$ is a vehicle parking location, $x \in$ Buildings
$I(x,y) : x$ is inside of y, $x \in$ Vehicles, $y \in$ Buildings

Adding them to the expression is easy; they are placed in groups based on the part of the sentence they are describing:

$$\exists v\ [T(v) \wedge W(v) \wedge \exists b\ (D(b) \wedge G(b) \wedge P(b) \wedge I(v,b))],$$

 Copyright © Lester I. McCann Please do not distribute; thx!

$v \in$ Vehicles, $b \in$ Buildings

And, finally, we're done!

Tying up some loose ends from Example 47:

- *That last expression is a little overwhelming. Can you give me a breakdown to help me make sense of it?* Sure; take a look at Figure 2.1.
- *Seems that $\rightarrow$ goes with $\forall$ and $\wedge$ goes with $\exists$, right?* Basically, yes. This isn't to say that we can't mix and match when the situation warrants, but in many circumstances this rule of thumb works.
- *Why did you move $\exists b$ to the middle of the expression?* You've probably written a counter-controlled WHILE loop as part of a program. Where did you declare the variable, at the start of the subprogram or just ahead of the loop? Probably the latter, because that's where you needed the variable – why declare it until you need it? The same idea applies here. We didn't need b right away, so the quantifier wasn't needed right away, either.
- *So ... we can put all of the quantifiers at the front of the expression?* Yes, you can, but it's best that you don't. Readers will expect you to hold off on inserting quantifiers until they are needed. Why make them remember more variables before those variables are necessary?[7] Also, delaying the quantifiers helps the expression 'read' a little more naturally.
- *I really like compact expressions. Is that 'one concept per predicate' rule really necessary?* In this book it is! Here's one way to look at it: If you could dump everything into a single predicate, how would you ever learn how to use logical operators? It's like a child 'cleaning' her room by stuffing all of her toys into her closet.
- *Do we have to write the domains after every ... single ... expression?* Nope! You can instead begin by saying something like "In the following expressions, assume $x \in$ Vehicles and $y \in$ Buildings." But if you get in the habit of putting domains after every expression, you (and the reader) will be less likely to overlook them.

[6] Yuck!

[7] 'Cognitive load' is the phrase that describes this. You could also use it as an insult: *My friend, no one can say that you're a cognitive load.*

 Copyright © Lester I. McCann Please do not distribute; thx!

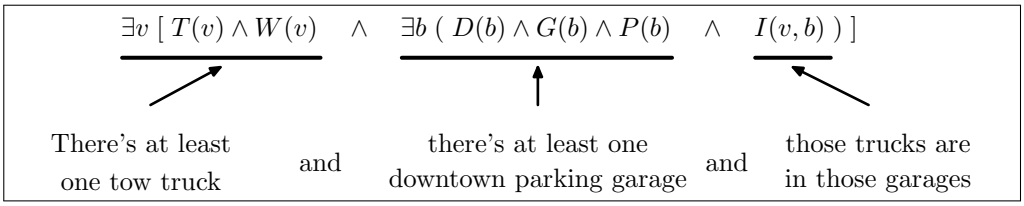

Figure 2.1: A breakdown of the final expression from Example 47.

2.5 Mixing Universal and Existential Quantification

Nesting matching quantifiers (e.g., $\exists a \exists b$) is relatively straight-forward and creates expressions whose meanings aren't too hard to understand. Nesting mismatched quantifiers (e.g., $\forall a \exists b$) is also often necessary. Unfortunately, making sense of such mixed nestings usually isn't as easy, even though the process is the same.

Example 48:

Problem: Express both $\forall p \exists q\ L(p, q)$ and $\exists p \forall q\ L(p, q)$ in conversational English, where $L(p, q)$ represents "p laughs at q," and p and q share the domain of people.

Solution: $\forall p \exists q\ L(p, q)$ means, in logical English, "Considering all people, there exists at least one person that all people laugh at." We can condense that to the more conversational, "There's someone that we all laugh at."[8] Notice that this means a universally-laughed-at person also laughs at himself or herself.

$\exists p \forall q\ L(p, q)$ means "there exists a person such that, no matter which person you name, the first person will laugh at them." Conversationally, "There's someone who laughs at everyone" (again, including himself or herself).

[8] Maybe one of those hapless burglars who gets stuck upside-down in a restaurant's kitchen exhaust ductwork. They must laugh at themselves . . . eventually.

 Copyright © Lester I. McCann Please do not distribute; thx!

A common question: *Does the order of the quantifiers' variables have to match the order of variables' appearances in the expression? That is, could I write "$\forall b \exists a\ Q(a,b)$" instead of "$\exists a \forall b\ Q(a,b)$"?* Technically, 'yes,' but practically 'no.' It's very rare for people to change the order. Why? Because it makes interpretation of mixed-quantifier expressions unnecessarily difficult. That's reason enough for us not to do it. By the way, the order of the variables in nested *matching*-quantifier expressions (such as $\forall x \forall y$ vs. $\forall y \forall x$), by the way, doesn't matter logically, but again, it's still best to quantify the variables in the order in which they appear in the expression being quantified.

Example 49:

Problem: Express this sentence in logic: *The tow trucks are in the downtown parking garages.*

Solution: This is the sentence of Example 47 with some small changes. Perhaps the most intimidating change is that the linguistic clues of quantification are missing — words such as 'some' and 'all' aren't provided. We need to deduce the intended quantifiers using our knowledge of English.

The subject "tow trucks" includes no limitations, indicating that the meaning is *all* tow trucks. There are also no limitations on the parking garages. It's tempting to assume that the intended meaning is again "all"; that is, *all* of the tow trucks are in *all* of the parking garages. But ... must all of the parking garages contain tow trucks? The statement doesn't quite say that; all we know is that, if the statement is true, the tow trucks can be found in the garages — *some* of the garages, possibly all of them. Thus, we need to universally quantify the vehicles and existentially quantify the buildings. The only real difference between this sentence and the one of Example 47 is the change in the vehicle quantification.

Using the same predicates as used in Example 47 and our deciphering of the quantifiers, we can create a corresponding logical version of the statement. Not surprisingly, it's just the solution to Example 47 with the swap of quantifiers and of one AND for an implication (as we've seen, $\rightarrow$ usually goes with $\forall$):

 Copyright © Lester I. McCann Please do not distribute; thx!

$\forall v\ [(T(v) \wedge W(v)) \rightarrow \exists b\ (D(b) \wedge G(b) \wedge P(b) \wedge I(v,b))]$,
$v \in$ Vehicles, $b \in$ Buildings

In somewhat condensed logical English: "For all vehicles, if the vehicle is a tow truck, then there exists a downtown parking garage and the tow truck is in it."

You may be wondering if the intended meaning of this statement could be exactly that of Example 47. Perhaps, but that seems unlikely, given how people use English. Based on the wording of the sentence, the all-trucks/some-garages interpretation is the most likely. Try to keep this issue of interpretation in mind the next time you have to write down a recipe, create a software specification, or read a legal document.

For reference, Table 12 summarizes when mixed quantifications are true and false.

Table 12: When Are Mixed Quantifications True/False?

Quantification	Is True When ...	Is False When ...
$\forall m\, \exists n\, P(m,n)$	no matter the m, you can find an n that makes $P()$ true	there's an m for which no n exists to make $P()$ true
$\exists m\, \forall n\, P(m,n)$	some m can be paired with any n yet $P()$ is true	we try every m, but no n exists that makes $P()$ true

The 'true' case of $\forall m\, \exists n\, P(m,n)$ is worth an additional mention. You don't have to find a single n that works with all possible values of m – that is, each of the m values can have its own n value. For example, no matter which non-zero integer you give me ($\forall i$), I can find a rational number ($\exists j$) such that their product is 3. That rational number is based on i ($j = 3/i$), and is therefore different every time, but that's all we need to make $\forall i \exists j\ (i \cdot j = 3)$ true.

 Copyright © Lester I. McCann Please do not distribute; thx!

2.6 Generalized De Morgan's Laws

In Chapter 1 we learned a pair of equivalences known as DeMorgan's Laws: $\neg(p \wedge q) \equiv (\neg p \vee \neg q)$ and $\neg(p \vee q) \equiv (\neg p \wedge \neg q)$. They (as well as the rest of the equivalences) apply to predicates as well as they do to propositions.

Example 50:

Consider $\exists x[P(x) \wedge Q(x)]$ (it doesn't matter what $P()$ and $Q()$ represent). $P(x) \wedge Q(x)$, using an x from its domain, is just a compound proposition. We can apply Double Negation and De Morgan to it: $P(x) \wedge Q(x) \equiv \overline{\overline{P(x)} \wedge Q(X)} \equiv \overline{\overline{P(x)} \vee \overline{Q(x)}}$. Thus, $\exists x[P(x) \wedge Q(x)] \equiv \exists x[\overline{\overline{P(x)} \vee \overline{Q(x)}}]$.

But what if we want to move negations into (or out of) quantifiers rather than just into or out of parentheses? De Morgan's Laws don't work across quantifiers, but we can generalize those laws to handle quantification. The result is sometimes called *Generalized De Morgan's Laws*.

generalized de morgan

The generalization comes from a pair of observations, of which we will discuss just one. Consider the expression $\forall x\ \neg P(x), x \in \{1, 2, \ldots, n\}$. $\forall x\ \neg P(x)$ is a notation for $\neg P(1) \wedge \neg P(2) \wedge \ldots \wedge \neg P(n)$. If we were to apply De Morgan's to that, we would produce the equivalent expression $\neg(P(1) \vee P(2) \vee \ldots \vee P(n))$. Ignoring the leading negation for a moment, the disjunction can be represented by $\exists x\ P(x)$. Thus, $\forall x\ \neg P(x)$ is logically equivalent to $\neg \exists x\ P(x)$. Notice that the moving the negation from inside to outside 'flipped' the quantifier from universal to existential.

Both Generalized De Morgan's Laws are given in Table 13.

Table 13: Generalized De Morgan's Laws

1. $\neg \forall x\ P(x) \equiv \exists x\ \neg P(x)$
2. $\neg \exists x\ P(x) \equiv \forall x\ \neg P(x)$

A common use for Generalized De Morgan's Laws is to move negations inside of quantifiers to create a result that is easier to express in English.

 Copyright © Lester I. McCann Please do not distribute; thx!

Example 51:

Problem: Construct an expression that is equivalent to $\neg\forall c\forall d\, D(c,d)$ by moving the negation inside of the quantifiers.

Solution: First, let's think about what $\neg\forall c\forall d$ means. $\neg\forall c$ means 'not all' – that is, some or none. The negation does not also apply to $\forall d$. (If you need both to be negated, you need a second negation, as in $\neg\forall c\,\neg\forall d$.) In English, we could try to say, "For not all c and for all d, ..." Or, we could try, "For some c or no c, but for all d, ..." Either way, it's pretty awkward.

Applying the first Generalized De Morgan's Law (GDM #1) twice (once per quantifier), we can move the negation inside:

$$\begin{aligned} \neg\forall c\forall d\, D(c,d) &\equiv \exists c\neg\forall d\, D(c,d) \quad \text{[GDM \#1]} \\ &\equiv \exists c\exists d\,\neg D(c,d) \quad \text{[GDM \#1]} \end{aligned}$$

This is easy to express in English; in fact, we already did, back in Example 45: "Some plants don't have some defenses."

2.7 "Exactly n" Expressions

uniqueness quantifier

If you've studied quantifiers previously, you may have encountered the *uniqueness quantifier*, $\exists!x$. It was created to represent the concept of "exactly one." This is a useful concept, of course, but we don't need a new quantifier to express that idea – we can express it using the quantifiers and logical operators we already have. Learning how to do it is a worthwhile activity, not only as additional practice with quantification, but also as a starting point for extensions of the idea.

2.7.1 Exactly One

To represent this idea, we need a way to exclude "or more" from the "there exists one or more" meaning of $\exists$. One way to do that is to think of what "one or more" means. For the sake of illustration, consider integers. If x represents a quantity of items and is an integer, then $x \geq 1$ represents "one or more" of those items. Similarly, $x = 1$ represents "exactly one." To reduce $x \geq 1$ to

 Copyright © Lester I. McCann Please do not distribute; thx!

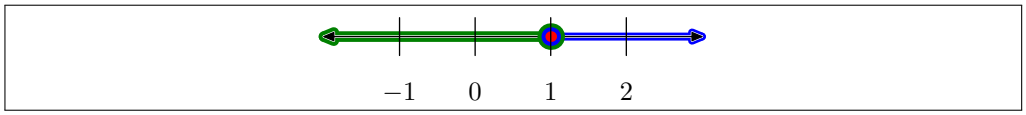

Figure 2.2: $(x \leq 1) \ \wedge \ (x \geq 1) \ \equiv \ (x = 1)$

$x = 1$, we can AND $x \geq 1$ with $x \leq 1$ – their intersection is $x = 1$. Figure 2.2 illustrates this on a number line.

Now all we have to do is figure out how to express $x \leq 1$ (in English, "no more than one") logically. Imagine that you thought you had two cans of beans in the cupboard, but you can find only one. The following conversation with your roommate 'Slash' results:

You: "Seriously, your name is 'Slash?' "
Slash: "Deal with it, OK?"
You: "Whatever. Look, I can't find the second can of beans. Do you see it?"
Slash: (takes a look) "Idiot, it's sitting right in front!"
You: "That's the first can, genius!" [9]

It may be hard to believe, but there *is* a point to that touching story: If you've think you've found two or more items, but they all turn out to be the same item, then you have exactly one item.

This "no more than one" idea can be turned into logic fairly easily. The following description takes several liberties, but should help clarify the structure of the final logical expression. Let y be you and s be Slash. You search through all of the cans ($\forall y$) and find one of beans ($B(y)$ is true). Slash does the same ($\forall s$) and also finds one ($B(s)$ is true). But if there is only one can of beans, then the can you found is the same can he found ($y = s$). That is, any time more than one can seems to have been found, just one was found. As a quantified expression: $\forall y \forall s \, [(B(y) \wedge B(s)) \rightarrow (y = s)]$, where the domain of y and of s is the content of the cupboard (that is, the things we're trying to count). In English: "Whenever two cans of beans are found, they're really the same can." Or, "There is at most one can of beans in the cupboard." ('No cans' is a viable option; this is a conditional expression.)

Almost done. All we have left to do is to AND that expression with one for "at least one" and we'll have the structure of an expression for "exactly

[9] Tune in next week, when you and Slash try to find your marbles, and hilarity ensues!

 Copyright © Lester I. McCann Please do not distribute; thx!

one." And that's easy:

$$\exists a\ P(a)\ \wedge\ \forall b \forall c\,[(P(b) \wedge P(c)) \rightarrow (b = c)],\ a, b, c \in \text{The Domain}$$

$\exists a\ P(a)$ gives the "at least one" meaning, and the rest gives "at most one." Together, we have "exactly one." The two halves are independent expressions glued together.

There are several ways to express "exactly one" in first-order logic. One manages to shorten the version above by reusing the $\exists$ variable within the $\forall$ half:

$$\exists a\ (P(a)\ \wedge\ \forall b\,[P(b) \rightarrow (b = a)]),\ a, b \in \text{The Domain}$$

In logical English: "There exists a domain element that has the property and, for all domain elements, if an element has the property, it's the same element as the first one." Conversationally: "Exactly one domain element has the property." You can think of a as representing the "exactly one" candidate, and b as representing all of the challengers.

Now that we have two general structures that we can use, converting "exactly one" expressions to logic is fairly easy: Identify the domain and necessary predicates for the given situation, and use them instead of the placeholders in either of the two general structures, with adjustments as needed for the characteristics of the problem.

Example 52:

Problem: Express "Earth has exactly one moon" as a quantified logical expression.

Solution: We are trying to count moons, so we will start by using a domain of "moons," but will show how to change it to the more general "heavenly bodies" later.[10] We also need a way to say that a moon belongs to Earth. "$O(x)$: x orbits Earth" will do the job. Notice that our choice of domain for x prevents consideration of artificial satellites. All that remains is to insert these pieces into one of the two general expressions developed above. Let's use the second. The domain becomes 'Moons' and $O()$ goes in place of $P()$:

 Copyright © Lester I. McCann Please do not distribute; thx!

$\exists m\ (O(m)\ \wedge\ \forall n\ [O(n) \rightarrow (n = m)]),\ m, n \in \text{Moons}$

To use the more general "heavenly bodies" as the domain, we need to add a predicate that describes moons. Easy: "$M(x) : x$ is a moon." But where does this predicate go within the expression? Also easy: Alongside each occurrence of $O()$. The reason is that we are talking about Earth moons. $M()$ handles moons, and $O()$ gives them to Earth; together, they describe moons of Earth. The expression:

$\exists h\ (O(h) \wedge M(h) \wedge\ \forall i\ [(O(i) \wedge M(i)) \rightarrow (i = h)]),$
$h, i \in \text{Heavenly Bodies}$

One more adjustment: Some of you are probably not happy with $O()$ including Earth as a constant, and would prefer that $O()$ be more general so that our Earth restriction is more obvious. No problem: We can change $O()$ to have two arguments – "$O(x, y) : x$ orbits y" – and can use 'Earth' for y in the expression:

$\exists h\ (O(h, \text{Earth}) \wedge M(h) \wedge\ \forall i\ [(O(i, \text{Earth}) \wedge M(i)) \rightarrow (i = h)]),$
$h, i \in \text{Heavenly Bodies}$

Example 53:

Problem: Express this "exactly one" situation in logic: For all positive integers p, there's exactly one non-negative integer i such that $p \cdot i = 0$.

Solution: You're probably thinking, "Well, that's a long-winded way of saying that anything times zero is zero!" And it is. But, the point of the example is not long-windedness.[11]

At first glance, the structure of the statement makes this exactly one situation appear to be different that the previous examples. This statement begins with a $\forall$ and the "exactly one" is buried inside of it. Thanks to the flexibility of English, we can rewrite it to match the structure we expect to see: *Exactly one non-negative integer i makes $p \cdot i = 0$, where*

[10] If you're expecting a joke about swimsuit models, you'll have to make your own. I'm not looking for trouble!

 Copyright © Lester I. McCann Please do not distribute; thx!

p is any positive integer.

Assuming that we'll again use the shorter "exactly one" structure, we can see that i should be the existentially quantified variable and can guess that p will be universally quantified (one quantifier seeks one variable for a logical marriage, right?). But what should the predicate(s) be? We're only dealing with integers, but both of the variables have certain restrictions – i is non-negative and p is positive. Perhaps they can be our predicates:

$$\exists i\ ((i \geq 0)\ \wedge\ \forall p\ [(p > 0) \rightarrow (p \cdot i = 0)]),\ i, p \in \mathbb{Z}$$

Something's missing – nothing here says that there's exactly one i. There's at least one, but there can be more. (If this doesn't make sense, try converting the expression to English.)

The problem is that we allowed p to fill the role of the challenger to i, but p is really just a bystander who got caught up in the excitement. We erred by trying to blindly fit our problem to the exactly-one structure we wanted to use, forgetting to think about what the parts of that structure represent. To do "exactly one," we need candidate and challenger variables that come from the same domain and represent the same kind of items. Compare the above attempt to this:

$$\exists i\ ((i \geq 0 \wedge p \cdot i = 0)\ \wedge\ \forall j\ [(j \geq 0 \wedge p \cdot j = 0) \rightarrow (j = i)]),\ i, j, p \in \mathbb{Z}$$

This looks like progress; if nothing else, the structure looks better. In English: "There's an integer that's non-negative that we can multiply with p to produce zero and, for all integers, if we find such an integer, it's the same as the first one." But we're not finished – p has two issues. First, we gave p a domain of $\mathbb{Z}$, but we need it to be a positive integer. Second, we didn't quantify p, which means it's free and we don't allow free variables (see Section 2.2.2). Because p can be any positive integer, p should be universally quantified.

As long as we're fixing those problems, let's also consider the $i \geq 0$ conditions. We can eliminate those by using a domain of $\mathbb{Z}^*$ for i and j. You can argue that by using such a domain we're ignoring our 'one step above things' rule for domains, but as $\mathbb{Z}^*$, $\mathbb{Z}^+$, and the like are well-

 Copyright © Lester I. McCann Please do not distribute; thx!

accepted and commonly-used in mathematics, we can make an exception without feeling too guilty. And, allowing the use of $\mathbb{Z}^*$ means we're free to use $\mathbb{Z}^+$ to restrict p to be a positive integer. After making all of these adjustments, our final expression is:

$$\forall p\, \exists i\, ((p \cdot i = 0)\ \wedge\ \forall j\, [(p \cdot j = 0) \rightarrow (j = i)]),\ i, j, \in \mathbb{Z}^*,\ p \in \mathbb{Z}^+$$

It's no wonder mathematicians call $p \cdot 0 = 0$ the *zero product property*; giving it a name is so much easier.

2.7.2 Exactly Two

Understanding how to express "exactly one" is a challenge. Happily, with "exactly one" covered, "exactly two" won't be as difficult. In fact, we'll approach it the same way, starting with almost the same equivalence: $(x \leq 2) \wedge (x \geq 2) \equiv (x = 2)$.

Let's start with $x \geq 2$. We don't have a quantifier for that, like we do for $x \geq 1$. We need to construct our own expression possessing the 'more than one' meaning. $\exists$ means 'at least one,' as we know. A pair of $\exists$s means 'at least two' so long as they are representing two distinct items. In symbols: $\exists a\, \exists b\ (P(a) \wedge P(b) \wedge (a \neq b))$. Because a and b are different items both possessing the property of interest ($P()$), we have two. Because there's no limit on more such items existing, we also still have the 'or more' meaning.

Now consider $x \leq 2$. To get $x \leq 1$, we created an expression that means, essentially, "if we think we found two, we were wrong." We'll just change 'two' to 'three:' $\forall c \forall d \forall e\, [(P(c) \wedge P(d) \wedge P(e)) \rightarrow ((c = d) \vee (c = e) \vee (d = e))]$. This says that if c, d, and e all have the property, we're at least double-counting (two of the three are the same), and maybe even triple-counting (all three are the same). Double-counting means there are two items with the property, triple-counting means there is just one, and the implication allows for there to be none. In short, we have captured the meaning 'at most two.'

The simple way to get "exactly two" is to AND these two expressions together, just as we did to create the first version of our "exactly one" structure:

$$\exists a\, \exists b\, (P(a) \wedge P(b) \wedge (a \neq b)) \wedge$$
$$\forall c \forall d \forall e\, [(P(c) \wedge P(d) \wedge P(e)) \rightarrow ((c = d) \vee (c = e) \vee (d = e))],$$

[11] Surprised? ☺

 Copyright © Lester I. McCann Please do not distribute; thx!

$$a, b, c, d, e \in \text{The Domain}$$

Also just like "exactly one," we can create a shorter version that reuses the $\exists$ variables:

$$\exists a\, \exists b\, (P(a) \wedge P(b) \wedge (a \neq b)\ \wedge\ \forall c\, [P(c) \rightarrow ((c = a) \vee (c = b))]),$$
$$a, b, c \in \text{The Domain}$$

Example 54:

Problem: Express in logical notation: "My bicycle has exactly two wheels."

Solution: Because we're counting bicycle wheels, that will be our domain.[12] We need a predicate to say that the wheels belong to my bike: "$B(x)$: x is part of my bike" will suffice. We're ready to put them in place:

$$\exists w\, \exists x\, (B(w) \wedge B(x) \wedge (w \neq x)\ \wedge\ \forall y\, [B(y) \rightarrow ((y = w) \vee (y = x))]),$$
$$w, x, y \in \text{bicycle wheels}$$

That's it! No intentional errors, no new twists. Many English $\rightarrow$ Logic problems are straight-forward. But, there is one lingering 'thing' thing ...

"Bicycle wheels" is more than a single step above 'Things.' "Wheels" would be better, with the addition of a predicate that's true when the given wheel is used on bicycles. If you've followed the previous examples where we've simplified domains (examples 42 and 52, say), you know how to adjust this expression. Do it! And, don't be afraid to adopt this approach yourself on future problems – use a complex domain until you get the logical structure in place, then simplify the domain and add the corresponding predicate(s).

2.7.3 Exactly n

We can use this $(x \leq n) \wedge (x \geq n)\ \equiv\ (x = n)$ approach to handle any $n \geq 1$ we need. Unfortunately, the logic gets polynomially messier as n increases. The

[12] I know what you're thinking; patience ...

 Copyright © Lester I. McCann Please do not distribute; thx!

number of comparisons in the $\forall$ half of the uncombined expressions follows a sequence known as the *Triangular numbers*: $0, 1, 3, 6, 10, 15, \ldots$ The n-th Triangular number equals $\frac{1}{2}(n^2 + n)$. For instance, if you had to express 'exactly four,' you'd have five variables in the $\forall$ section, and would need to express that at least two of them are the same. There are 10 ways to pair up those five variables: $\{(a, b), (a, c), (a, d), (a, e), (b, c), (b, d), (b, e), (c, d), (c, e), (d, e)\}$.[13]

On the bright side, it's unusual to need to express situations above "exactly two," even in a discrete structures class. But now you know how to do it if you need to.

[13]This set is an example of a *binary relation*. We cover a lot of material on such sets in Chapter 8.

 Copyright © Lester I. McCann Please do not distribute; thx!