5.6. Construction Revisited: Quantifiers, Variables, and Binding

1. Bound and Free Variables. The Chapter Five construction rules, in their final form, were stated in terms of "formulas" as follows.

Atomic Formulas:

A1. Sentence letters are atomic formulas

A2. A predicate letter followed by a name letter *or* variable is an atomic formula.

Formulas:

1. Atomic formulas are formulas.
2. If ● is a formula, then ~● is a formula.
3. If ● and ▲ are formulas, then (● ∧ ▲) is a formula.
4. If ● and ▲ are formulas, then (● ∨ ▲) is a formula.
5. If ● and ▲ are formulas, then (● →▲) is a formula.
6. If ● and ▲ are formulas, then (● ↔▲) is a formula.
7. If ★ is a variable and ● is a formula, then

∃★ ●

and

∀★ ●

are both formulas.

And recall that we use "**formula**" as an umbrella term, covering any formal **sentence or quasi-sentence**. So "P," "GA," and "Gx" are all formulas.

Formulas

Formal Sentences	**Quasi-Sentences**
P, GA	Gx

Note that Construction Rule 7 is quite lax about what it attaches a quantifier to: as long as it's a formula, of whatever sort, we can attach a universal or existential quantifier to it.

> 7. If ★ is a variable and ● is a formula, then
>
> ∃★ ●
>
> and
>
> ∀★ ●
>
> are both formulas.

For instance, the formal language allows us to attach a universal quantifier to the formula "GA".

∀x GA *(7)*

|

GA *(A1, 1)*

Featuring as it does a name letter, "GA" is the formal counterpart to an English sentence such as "Benjamin Franklin is mortal".

A: Benjamin Franklin **G**: __ is a mortal

Benjamin Franklin is a mortal: **GA**

So the formula "∀x GA" translates the following English sentence.

For every object in the universe, the following holds true of it: Benjamin Franklin is mortal.

Of course that would be a thoroughly strange thing to say, since attaching the long quantifier phrase at the beginning adds nothing to the claim about Benjamin Franklin which follows. To say "For every object in the universe, the following holds true of it: Benjamin Franklin is a mortal" is, meaning- and communication-wise, just to say "Benjamin Franklin is mortal". The quantifying preamble is **semantically empty**.

And from what we already understand of quantifier semantics we can see that the same is true of the formula "∀x GA". To determine whether "∀x GA" is true in a model, we would (i) remove the quantifier "∀x," and then (ii) replace the variable in the remaining scope formula with each name used in that model – counting "∀x GA" true just in case we get a true sentence for each name substitution.

But the scope formula of "∀x GA" is "GA". Replacing 'the variable' in "GA" by a name letter in fact involves no change at all – since "GA" contains no variable. We judge the universal sentence true or false based on the result of "replacing the variable with each name letter" in the scope formula. But since that process is in this case empty, the outcome of such 'replacement' will in every case just be "GA". And that means: whether the universal sentence is true or false in the model is determined simply by whether "GA" is true or false in that model. Semantically, "∀x GA" is true in exactly the same models that make "GA" true. So adding the "∀x" winds up being semantically null.

In the jargon of formal logic we say that "∀x GA" is a case of **vacuous quantification**: adding a quantifier which makes no (semantic) change.

That is in sharp contrast to earlier examples of quantification such as "∀x Hx," which translates the English sentence "Everything is material" – or, more technically, "For every object in the universe, the following holds true of it: it is material". We might challenge the truth of that claim (reasoning that abstract things, such as numbers, aren't material); but it has none of the long-winded semantic oddness we found in vacuous quantification. Here the quantifier phrase really seems to be 'quantifying over' things.

Intuitively, that's because the quantifier phrase links up with the English mini-sentence "it is material" which follows – just as the quantifier in the formal language links up with the formula "Hx" following it.

In both cases the quantifier, and the (quasi-)claim that follows, contain the same variable term: "it" in English, the variable "x" in Formalese.

For every object in the universe, the following holds true of **it**:
it is mortal.

$$\forall \underline{\mathbf{x}}\ G\underline{\mathbf{x}}$$

To this notion – of the two parts 'linking up' thanks to a matching variable – we give the name "**variable binding**". Even before setting out the technical details of variable binding, we spot one crucial component: the quantifier phrase and the formula that follows must **contain the same variable**.

That's what left the quantifier useless in the earlier example "∀x GA": the "x" in "∀x" found no counterpart in "GA" to allow such a connection. (Likewise in English: the variable pronoun "it," in the quantifier phrase "For every object in the universe, the following holds true of it," had no counterpart in the English sentence "Benjamin Franklin is mortal".)

For every object in the universe, the following holds true of **it**:
Benjamin Franklin is mortal.

$$\forall \underline{\mathbf{x}}\ G\underline{\mathbf{A}}$$

It is lack of any opportunity for variable binding that leaves the quantifier vacuous in the formula "∀x GA".

The full details of variable binding simply combine two elements encountered above: (1) construction attaches a quantifier to its **scope formula**, and (2) the quantifier binds the **matching variable** in that scope formula.

For a quantifier to **bind** a variable,

> (1) the variable being bound must appear in the *scope* formula of that quantifier,

and

> (2) the variable being bound must be the same variable used in the quantifier,

So in the previous example the “x” in “Gx” was bound by “∀x” because it met both these conditions.

$$\begin{array}{ll} \mathbf{\forall x\, Gx} & \textit{(7)} \\ \quad | & \\ \quad \mathbf{Gx} & \textit{(1, A2)} \end{array}$$

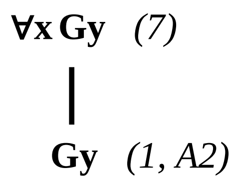
But in “∀x GA” the quantifier “∀x” finds no matching “x” in its scope to match up with – so “∀x” ends up being extra baggage.

$$\begin{array}{ll} \mathbf{\forall x\, GA} & \textit{(7)} \\ \quad | & \\ \quad \mathbf{GA} & \textit{(1, A1)} \end{array}$$

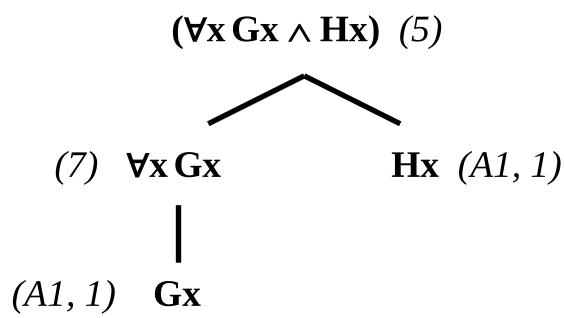
Indeed, that failure to bind variables forms our official definition of a vacuous quantifier.

> **Vacuous quantifier**: a quantifier which binds no variables

"∀x Gy" likewise sins against the matching variable requirement, leaving "∀x" nothing in its scope to bind. So here too "∀x" is a vacuous quantifier.



On the other hand, the formula "(∀x Gx ∧ Hx)" instead violates the scope requirement on binding: because the "x" in "Hx" is not in the scope formula of quantifier "∀x," the quantifier can't reach that variable to bind it.



A bit more jargon here helps head off potential confusion over variables.

While the "x" in "Gx" and the one in "Hx" are certainly two different *things* – they're in different locations – it seems odd to call them "different variables". On the contrary: they're both the *same* variable, namely "x".

The situation is like one where Rex is reading *Methods of Logic* in Las Vegas and discussing it on the phone with Suki, who's reading *Methods of Logic* in San Diego. Does *Methods of Logic* then count as one book (because Rex and Suki are reading *the same book*) or two (because the book Rex is reading is hundreds of miles away from the one Suki is reading)?

With a book we sort out the confusion by speaking of **copies** of the book: Rex and Suki have two different **copies** of **the same book**. And in formal logic we similarly speak of different **occurrences** of a variable: in the formula "(∀x Gx ∧ Hx)" the "x" in "Gx" and the one in "Hx" are two different **occurrences** of **the same variable**, "x".

That allows us to fold in another piece of jargon: a variable-occurrence which isn't bound by any quantifier is said to be **free**. Since in "($\forall$x Gx $\wedge$ Hx)" the formula "Hx" lay outside the scope of "$\forall$x," the "x" occurrence in "Hx" is free.

> A **free variable (**-occurrence) is a variable (-occurrence) not bound by any quantifier.[1]

We can now make good on an outstanding debt, by drawing the official border between sentences and quasi-sentences.

> A **sentence** is a formula with no free variables.

> A **quasi-sentence** is a formula with one or more free variables.[2]

Note that since the formal sentences of previous chapters lacked variables entirely, they have no *free* variables – and so still qualify as sentences.

As stated, the requirements for variable binding allow more than one quantifier to bind the same variable instance.

---

[1] A variable is free in a formula if even one of its occurrences is free; and a variable is bound in a formula if even one of its occurrences is bound. So, though it might seem odd, a variable can be both bound and free in the same formula. But any particular variable-**occurrence** is either bound or free, *not both*. By analogy: if a library with multiple copies of *Methods of Logic* is taking inventory, and recording whether a book has been written in by readers or is free of notes, *Methods of Logic* might be recorded as both – because one **copy** has notes written in the margins while another doesn't. But any particular copy of *Methods of Logic* is either free of readers' notes or else written in, *but not both*.

[2] What we call a "sentence" some authors call a "closed sentence," and our "quasi-sentence" is instead an "open sentence" – for example, (Kleene 1967/2002: 105) and (Quine 1982: 134). Reaching back even earlier: our "sentence" is what Russell and Whitehead call a "proposition," while our "quasi-sentence" is called a "propositional function" (Russell and Whitehead 1910: 38). The different technical dialects can overlap: Quine equates "closed sentence" with "statement," while (Smullyan 1968/1995: 44) equates "closed formula" with "sentence". Mixing vocabularies, (Gamut 1982/ 1991 Vol. I: 74) contrasts "sentence" and "propositional function".

So in the following formula the "x" in "Hx" is, in the course of the construction, bound first by "∃x" and then by "∀x".

$$\mathbf{\forall x\,(Gx \land \exists xHx)}\ \ \mathit{(7)}$$

$$\mathbf{(Gx \land \exists xHx)}\ \mathit{(5)}$$

$$\mathit{(A2,\,1)}\ \ \mathbf{Gx} \qquad\qquad \mathbf{\exists x\,Hx}\ \ \mathit{(7)}$$

$$\mathbf{Hx}\ \ \mathit{(A2,1)}$$

In light of this we might wish to add a 'tie-breaking' clause to our variable binding rules – declaring that once a variable-occurrence is bound, it is ineligible for further binding higher up in the construction tree.[3]

But in fact no such addition is needed. For both the semantic and deduction rules introduced later will legislate on their own that the first quantifier in the tree to bind a variable occurrence is the only one that matters to that occurrence. And the reason for this is by now familiar: if a variable occurrence is already bound when a quantifier is added, the quantifier acts (for that variable occurrence) just like a vacuous quantifier.

So in this next formula "∃x" binds just the x-occurrence in "Hx," while "∀x" binds two x-occurrences: in "Gx" and in "Hx".

$$\mathbf{\underline{\forall x}\,(Gx \rightarrow \underline{\exists x}\,Hx)}$$

Vacuous quantification provided one example of something that's permitted by the construction rules, but still semantically odd. Free variables provide a second example. Indeed, such is the oddness of free variables,

[3] Kleene adopts this 'tie-breaking' policy on variable binding in (Kleene 1967/2002: 81). Our policy instead follows Kalish and Montague (1962/1980: xx).

communication-wise, that it's difficult even to give an English example. The following sort of exchange is perhaps the closest we come to illustrating free variables in ordinary language.

> **Rex**: Something is made of wood, and it is painted red.
> **Ace**: I assume that the "it" which is painted red is the previously-mentioned wooden object, right?
> **Rex**: Oh, no. I just mean "it" to point to some object (or objects) – not necessarily the wooden object. I'm not saying *what* object "it" points to in the sentence "It is painted red".
> **Ace**: That's an extremely odd way of talking. Who goes around using the word "it," but without giving any indication which object the word means?

We're inclined to side with Ace here: a use of "it" which is explicitly *not* bound by a quantifier phrase (such as "something") is communicatively perverse.[4] If there's no telling what "it" points to, its use seems *pointless*.

It's no coincidence that free variables strike us as semantically odd, just as vacuous quantifiers did earlier. For really they're two sides of the same coin: vacuous quantification involves a quantifier with no variables to bind, while free variables are variables with no quantifier to bind them. Semantically, quantifiers and variables are made for each other.

And for that reason, quasi-sentences – formulas with free variables – will never be of interest to us, except as stepping stones toward constructing genuine sentences without free variables. That provides a good general policy for translating from English to the formal language: **if the finished translation contains free variables, something has gone wrong**.

2. The Uses of Variable Binding. In expanding the mechanisms of our formal language, we expanded as well the ability to express complex claims. But with greater expressive power comes greater potential for confusion and

[4] As Quine puts it: "The analogue of a free variable in ordinary language is a pronoun for which no grammatical antecedent is expressed or understood, and the analogue of an open sentence [i.e., a quasi-sentence] is a clause containing such a dangling pronoun." (Quine 1982: 134)

error. Though it may not be obvious, a proper understanding of variable-binding proves essential for drawing subtle distinctions between different, but deceptively similar sentences.

So, returning to earlier examples, we had better not confuse the following pair of English sentences – since, despite similar wording, they make very different claims.

(1) Something is both a cat and not a cat.
(2) Something is a cat, and something is not a cat.

And the formal language should translate these sentences in a way that preserves that difference.

We rephrase (1) in 'technical English'.

(1) Something is both a cat and not a cat.

For some object, *x*, the following holds of *x*:
***x* is both a cat and not a cat**.

The mini-sentence "*x is both a cat and not a cat*" is a simple conjunction. A translation table yields the following formula.

G: ___ is a cat

(Gx ∧ ~Gx)

The first part of the sentence – "*For some object, x, the following holds of x*" – is translated by an existential quantifier.

∃x (Gx ∧ ~Gx)

Sentence (2), by contrast, is itself a conjunction of two existential claims – as technical rephrasing brings out.

(2) Something is a cat, and something is not a cat.

For some object, *x*, the following holds of *x*:
***x* is a cat.**

and

For some object, *x*, the following holds of *x*:
***x* is not a cat.**

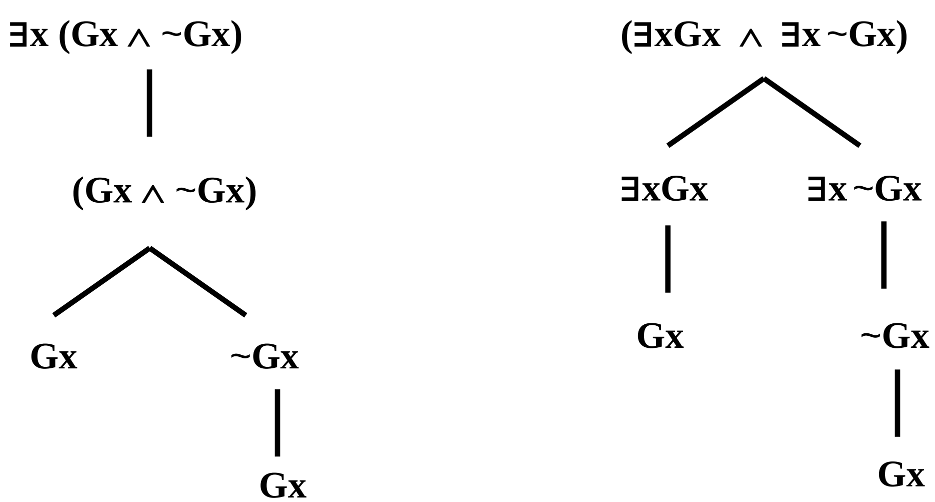
Using the same translation table, the first existential claim translates as "$\exists x\, Gx$," while the second is "$\exists x\, {\sim}Gx$" – yielding this conjunction.

$$(\exists x\, Gx \wedge \exists x\, {\sim}Gx)$$

Construction trees show how variable binding draws the necessary distinctions here between the two sentences.

(1) "***Something is both a cat and not a cat***"

(2) "***Something is a cat, and something is not a cat***."



In (1) one existential quantifier binds both the instances of "x" that follow; whereas in (2) "Gx" and "~Gx" are not bound by the same quantifier.

English Sentence (2) shows that there's nothing absurd about having cathood and lack-of-cathood together in the world (or mentioned in the same sentence).

(1) Something is both a cat and not a cat. $\mathbf{\exists x\ (Gx \wedge \sim Gx)}$

(2) Something is a cat, and something is not a cat. $\mathbf{(\exists x\ Gx \wedge \exists x\ \sim Gx\ )}$

What's absurd about English Sentence (1) is its claim that *being a cat* and *not being a cat* are found **in the same object**. Sentence (2) escaped absurdity because it **didn't** claim that the cat object and the non-cat object are **one and the same object**.
Variable binding draws that same distinction in formal language. In the formal translation of (1), the single quantifier "∃x" binds both the "x" in "Gx" and the one in "~Gx". Binding both variable occurrences by the same quantifier is the formal way of claiming that the object that's G and the one that's ~G are the **same object**.

By contrast, in the formal translation of (2) the "x" in "Gx" is bound by one quantifier, the "x" in "~Gx" by another. Since the different variable occurrences aren't here bound by the same quantifier, Sentence (2) *doesn't* claim that that the G object and the ~G object are the *same* object.

(1) "***Something is both a cat and not a cat***"

$$\mathbf{\underline{\exists x}\ (Gx \wedge \sim Gx)}$$

(2) "***Something is a cat, and something is not a cat***."

$$\mathbf{(\underline{\exists x}Gx\ \wedge \underline{\exists x}\sim Gx)}$$

We noted earlier that adding a quantifier – and hence binding otherwise free variables – transforms an incomplete quasi-sentence into a full-fledged formal sentence, by forcing those variables to point to **some** object(s). Now we see something more: by having a *single* quantifier bind several different variable occurrences, we force all those occurrences to point to the **same** object(s) throughout – and so we force the formulas containing those variable-occurrences to **make claims** about the same object(s).

Summary: Sentences, Formulas, and Binding

- For a **quantifier** to **bind** an **occurrence of a variable**:

  (1) the variable-occurrence being bound must be the same variable appearing in that quantifier,

  and

  (2) the variable-occurrence being bound must appear within the *scope* of that quantifier.

- A **free** variable-occurrence is one that is not bound.

- A **variable** is **free** if it has a free occurrence; a **variable** is **bound** if it has a bound occurrence. (So a variable can be both bound and free. Each variable-occurrence, however, is bound or free, but not both.)

- A **quasi-sentence** is a formula with at least one free variable.

- A **sentence** is a formula with no free variables.