The Syntax of Predicate Logic

LX 502 – Semantics I
October 11, 2008

1. Below the Sentence-Level

In Propositional Logic, atomic propositions correspond to simple sentences in the object language. Since atomic propositions are the smallest elements of the system, simple sentences are the smallest parts of the object language that we can represent in our metalanguage. In this respect, Propositional Logic is a blunt instrument. It is ill-equipped to capture the valid arguments in (1) or (2).

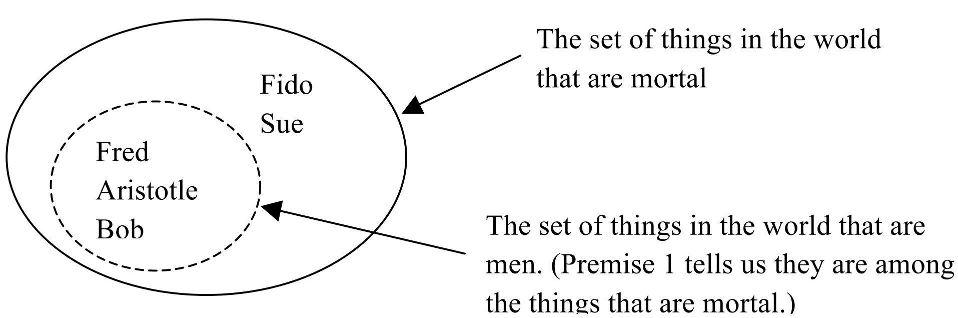
(1) Every man is mortal
Aristotle is a man
____
Therefore: Aristotle is mortal

(2) Aristotle is a man
____
Therefore: Someone is a man

Our intuitions tell us these arguments are valid, i.e., the premises entail the conclusion. The reasoning is straightforward but it draws on relationships in the object language formed below the sentence-level. These relationships are sensitive to things in the world, like *Aristotle*, and properties they have, like *being a man* and *being mortal*. Consider (1). Suppose the first premise is true: those things in the world that are mortal include those things in the world that are men. Put another way, every thing in the world that is a man is mortal. Now suppose Aristotle is a man. It follows Aristotle is necessarily mortal.

Set Theory gives us a way to represent this inference, illustrated in (3), which shows the relationship between being mortal and being a man: from the first premise, that being a man is a subset of being mortal. If Aristotle is in the set of men, he must also be in the set of mortal things. This follows from the subset relation.

(3)

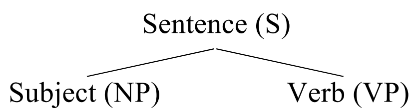


We need a more detailed metalanguage to capture this kind of inference, one which refers to things in the world. We need a metalanguage that translates the first premise in (1) into something like (4).

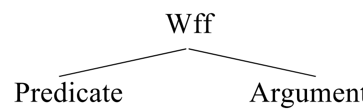
(4) “Every man is mortal” expresses the proposition
For every entity in the world (call it x), if x is a man then x is also mortal

The metalanguage needed to capture inference patterns like (1) and (2) is called *Predicate Logic*. Its basic elements (*entities*) correspond to things in the world and their properties (predicates). Predicate Logic is an extension of Propositional Logic not a replacement. It retains the central tenet of Propositional Logic: that sentences express propositions and propositions denote truth-conditions. The difference between these logics is that the basic building blocks of Predicate Logic are much like the building blocks of a sentence in a language like English. The smallest English sentence is formed by combining a verb with a subject. In Predicate Logic, the smallest proposition is formed by combining a predicate with an individual. Predicates correspond to verbs and individuals correspond to subjects and objects. The parallel is presented in (5).

(5) a. Object Language



b. Predicate Logic



2. The Basic Elements of Predicate Logic

To strengthen your understanding of Predicate Logic and its basic elements, let me first appeal to your intuitions. Consider the sentences in (6).

(6) a. Aristotle is a man
b. Socrates is a man
c. Bob is a man

These sentences express different propositions but they have the same syntactic form. Each one has a subject (*Aristotle*, *Socrates*, and *Bob*) and a verb phrase (*is a man*). Proper names like *Aristotle*, *Socrates*, and *Bob* are expressed in Predicate Logic using terms like *a*, *s*, *b*, which are called **individual constants** (or simply **individuals**).

(7) a. a = Aristotle
b. s = Socrates
c. b = Bob

Each individual constant refers to the **entity** in the world which bears that name. For example, the proper name *Aristotle* is expressed by the individual constant *a* which denotes the person who bears that name in the world. More succinctly, *Aristotle* denotes the person who bears that name in the world.

Now consider the sentences in (8), in which the subject is the same but the verb phrase changes.

(8) a. Aristotle is a man
b. Aristotle is pompous
c. Aristotle jogs

Verb phrases like *is a man*, *is pompous*, and *jogs*, express **predicate constants** (simply **predicates**), which are written using uppercase letters like *M*, *P*, *J*; alternatively, MAN, POMPOUS, JOG.[1] A predicate constant denotes a property of entities in the world. For instance, the verb phrase *is pompous* expresses the predicate constant P (or POMPOUS) and this predicate constant denotes the property of being pompous in the world, whose extension is the set of entities that are pompous.

(9) a. MAN = is a man
b. POMPOUS = is pompous
c. JOG = jogs

Predicate constants are not propositions, much like verbs are not sentences. Like a verb, a predicate constant has to combine with one or more elements to form a proposition. These elements are typically, but not exclusively, individual constants. Traditionally, the elements required by a verb are called its **arguments**. Simple predicate constants, like those in (9), need only combine with one argument to form a proposition. If a predicate constant only needs one argument, then it is called a 1-place predicate; if it requires two, it is called a 2-place predicate, and so on.

(10) a. Aristotle is a man
MAN(a)

b. Socrates is pompous
POMPOUS(s)

c. Bob jogs
JOG(b)

A sentence like *Aristotle is a man* is expressed in Predicate Logic by the proposition *M(a)*, which is obtained by combining the 1-place predicate *M* and the individual *a*. The proposition *M(a)* is true if and only if it correctly describes a situation in which the entity bearing the name Aristotle has the property of being a man in the world.

There are of course verb phrases that denote relationships between multiple entities, as in (11).

(11) a. Aristotle chased Socrates
b. Kermit kissed Ms. Piggy
c. Dorothy met the Wizard of Oz

The sentences in (11) contain verbs that require both a **subject** and an **objects**. Each verb corresponds to a relationship between the subject and its object. In this case, the predicate constant expressed by each verb needs two arguments to form a proposition, as in (12).

---

[1] You can choose to write predicate constants as single letters or as full words. In most of what I do, I use full words but sometimes I will stick with letters. At all times, use upper-case letters to avoid confusion with the object language and with the individual constants. In some textbooks, to distinguish the object language and the metalanguage, lower-case bold letters are used with an apostrophe ending the word, e.g., *is a man* translates into **man'**. I recommend against this for now. Just keep it simple and consistent.

(12) a. Aristotle chased Socrates
CHASE(a,s)

b. Kermit kissed Ms. Piggy
KISS(k,p)

c. Dorothy met the Wizard of Oz
MET(d,z)

This is where Predicate Logic gets complicated. Individual constants, like *a*, *s*, and *b*, denote specific entities in the world. For this reason, they are excellent translations of the proper names *Aristotle*, *Socrates*, and *Bob*. Unfortunately, they do not allow us to capture the meanings in (13).

(13) a. Every man is mortal
b. No man chased Socrates
c. Kermit kissed someone

It is intuitively clear from these examples that noun phrases like *every man* and *no man* do not refer to specific entities in the world. Translating them into individual constants would therefore be inappropriate. We need to abstract away from specific entities to deal with these meanings. Predicate Logic contains a set of special elements called **individual variables** (or simply **variables**), written x, y, z,…, that serve this purpose. An individual variable does not have a constant reference to a specific entity. You can think of a variable as a place-holder for the argument of a predicate. The examples in (14) illustrate this.

(14)	a.	MAN(x)	*x is a man*
	b.	CHASE(x,s)	*x chased Socrates*
	c.	KISS(k,x)	*Kermit chased x*
	d.	LOVE(x,y)	*x loves y*

Since the variables in (14) do not refer to specific entities, the expressions in (14) have no truth values. I.e., you cannot check to see whether *x* is actually a man in the world without knowing what *x* refers to. Consequently, the expressions in (14) are not propositions. But there is a sense in which they are well-formed and almost propositions: if in each case, the variables were replaced by an individual constant, the resulting expression would be a proposition. For example, by replacing the variable *x* with the individual constant *a* and the variable *y* with *s*, we get the propositions in (15).

(15)	a.	MAN(a)	*Aristotle is a man*
	b.	CHASE(a,s)	*Aristotle chased Socrates*
	c.	KISS(k,a)	*Kermit chased Aristotle*
	d.	LOVE(a,s)	*Aristotle loves Socrates*

So far, I haven't told you how this helps us with phrases like *every man*. It is crucial that you first understand what a variable is. Unlike individual constants, variables do not correspond to the any thing in the outside world. That makes it quite difficult to describe what variables are. The closest thing to a variable in natural language is a pronoun, like *he*, which also do not have constant reference to the outside world.

Consider the examples in (16). The pronoun *he* does not refer to a specific entity. What it refers to is dependent on the context in which the pronoun is spoken. The context determines the value of the pronoun.

(16) a. He is a man
b. He chased Socrates
c. Socrates kissed him

In (17), the interpretation of the pronoun *he* can also come from the preceding noun phrase *Aristotle*. In this instance, the interpretation of the pronoun is bound by the interpretation of the noun phrase *Aristotle*. We say that they are coreferential. This further suggests that the interpretation of the pronoun is not fixed in the way that the interpretation of the proper name *Aristotle* is.

(17) a. Aristotle said that he is a man
b. Aristotle said that he chased Socrates
c. Aristotle said that Socrates kissed him

Variables behave much like pronouns. They are meaningful when they are combined with another element in the language. In Predicate Logic, each variable combines with and is bound by a single **quantifier**. Predicate Logic has two such quantifiers: ∀ (the universal quantifier) and ∃ (the existential quantifier). Since a predicate can combine with more than one variable, it is necessary to write the variable immediately after the quantifier to indicate which variable the quantifier interacts with. In other words, we write ∀x to indicate that the universal quantifier ∀ is interacting with the variable x and not, say, with the variable y.

The logical expressions in (14) are wffs but not propositions. When they combine with a quantifier (one for each variable), the result is (18) - (20), which are all propositions.

(18)	a.	∀x MAN(x)	*Everything is a man*
	b.	∀x CHASE(x,s)	*Everything chased Socrates*
	c.	∀x KISS(k,x)	*Kermit chased everything*
	d.	∀x ∀y LOVE(x,y)	*Everything loves everything*

(19)	a.	∃x MAN(x)	*Something is a man*
	b.	∃x CHASE(x,s)	*Something chased Socrates*
	c.	∃x KISS(k,x)	*Kermit chased something*
	d.	∃x ∃y LOVE(x,y)	*Something loves something*

When both ∀ and ∃ are used in a single expression, their order is relevant. This is illustrated in (20). The paraphrases are lengthy to avoid the ambiguity present in the English sentences. The meaning of each arrangement is different. This is also why we need to keep track of the variable that each quantifier interacts with.

(20)	a.	∀x ∃y LOVE(x,y)	*For every thing (x) there is a thing (y) such that x loves y*
	b.	∃y ∀x LOVE(x,y)	*There is a thing (y) such that for every thing (x) x loves y*

The sentence in (21) is ambiguous: it expresses these two propositions.

(21) Everything loves something

Predicate Logic is a more detailed logical language than Propositional Logic but we are still interested in propositions and truth-conditions. Our goals have not changed. The difference is that, in Predicate Logic, propositions are built up by combining individuals and predicates, which denote entities and sets of entities, respectively. To translate expressions like *every man* and *some man*, Predicate Logic includes variables and quantifiers to bind the variables. What remains is to give a formal definition of Predicate Logic, its syntax and its semantics, and to reformulate what we mean by a model to fit this new logic.

3. The Syntax

Predicate Logic is defined in the same way as Propositional Logic. The major difference you will find is that the lexicon for Predicate Logic is substantially bigger. There are three primitives in this logic, plus quantifiers. The rules are more extensive but their purpose is the same: to determine the well-formed formulas (wffs) in the language. There is one additional difference I want to emphasize. While the syntax below determines the formulas that are well-formed, it does not determine which wffs are propositions. This is a big departure from Propositional Logic, in which every wff was a proposition. In Predicate Logic, not all wffs are propositions. I will elaborate on this shortly. For now focus on the wffs.

(22) THE SYNTAX OF PREDICATE LOGIC

A. Lexicon

i. Individual constants: j, m, s, …
ii. Individual variables: x, y, z, … (also $x_1, x_2, \ldots, x_n$)
iii. Predicate constants: P, Q, R, …
(each with a fixed finite number of argument places)
iv. A binary identity predicate: =
v. Logical connectives: $\neg, \wedge, \vee, \rightarrow, \leftrightarrow$
vi. Quantifiers: $\forall, \exists$
vii. Brackets: (,), [,]

B. The Syntactic Rules

(Individual terms are just the individual constants and individual variables)

i. If t_1 and t_2 are individual terms, then $t_1 = t_2$ is a wff
ii. If P is an n-place predicate, and $t_1, t_2, \ldots, t_n$ are terms, then $P(t_1, t_2, \ldots, t_n)$ is a wff
iii. If φ and ψ are wff, then $\neg\varphi$, $(\varphi \wedge \psi)$, $(\varphi \vee \psi)$, $(\varphi \rightarrow \psi)$, $(\varphi \leftrightarrow \psi)$ are wffs
iv. If φ is a wff and x is an individual variable, then $\forall x\ \varphi$ and $\exists x\ \varphi$ are wffs
v. Nothing else is a wff

Last section I described the lexicon of Predicate Logic so this section I focus primarily on the rules, excluding the rules in iii., which you should already be familiar with from Propositional Logic. The remaining three rules are new to you. In particular, rule iv. requires more explanation.

Rule i. is straightforward but its use may be unclear. It is used to equate two individuals. This allows us to translate sentences like (23).[2]

(23) a. Aristotle is not Socrates
$\neg a=s$
Aristotle is not equal to Socrates

b. Some man is Aristotle
$\exists x$ (MAN(x) $\wedge$ x=a)
There is an entity x such that x is a man and x is equal to Aristotle

c. Every wizard who is not Voldemorte is mortal
$\forall x$ ((WIZARD(x) $\wedge$ $\neg$x=v) $\rightarrow$ MORTAL(x))
For every entity x, if x is a wizard and x is not equal to Voldemorte, then x is mortal

Rule ii. is the main way of forming well-formed formulas. It states that if you combine an n-place predicate with n terms (individuals and variables) then the result is a wff. This rule is mainly responsible for forming propositions. In fact, if the n terms are all individuals constants then the result is a proposition. Examples are given in (24) and (25).

(24)	a.	CAT(b)	The 1-place predicate CAT and the individual b combine to form a wff. *Bob is a cat*
	b.	KISS(f,s)	The 2-place predicate KISS combines with the individuals f and s to form a wff. *Fred kissed Sue*
(25)	a.	CAT(x)	The 1-place predicate CAT and the variable x combine to form a wff. *x is a cat*
	b.	KISS(x,s)	The 2-place predicate KISS combines with the variable x and individual s to form a wff. *x kissed Sue*
	c.	KISS(x,y)	The 2-place predicate KISS combines with the variables x and y to form a wff. *x kissed y*

Rule iv. takes a wff φ and forms a complex wff, called a quantified expression, by combining φ with a quantifier (for a given variable x). The wff φ is called the **scope** of the quantifier $\forall x$ or $\exists x$. The purpose of this rule will be discussed below.

---

[2] The principal use of rule ii. is in equating a variable with an individual constant to translate expressions that pick out or exclude one entity from a range of entities.

(26) a. ∀x CAT(b) The wff CAT(b) is turned into a quantified expression for the variable x. Since CAT(b) has not variable x in it, this application of rule iv. does nothing.
For every entity x, Bob is a cat.

b. ∀x KISS(x,s) The wff KISS(x,s) is turned into a quantified expression for the variable x.
For every entity y, y kissed Sue; in other words, everything kissed Sue

c. ∃y KISS(x,y) The wff KISS(x,y) is turned into a quantified expression for the variable x.
There is an entity y such that x kissed y; in other words, x kissed something

4. Propositions

Now that we have defined what it means to be a well-formed formula in Predicate Logic, it is time to define what it means to be a proposition in Predicate Logic. In simplest terms, a proposition is a wff which contains no variable that is not interacting with a quantifier. There is of course a more elegant (and ultimately clearer) way of stating this but it requires two definitions first.

(27) In the expressions ∀x φ, the wff φ is called the **scope** of the quantifier ∀x
In the expressions ∃x φ, the wff φ is called the **scope** of the quantifier ∃x

Examples

a. ∃x (MAN(x) ∧ x=a)
The wff (MAN(x) ∧ x=a) is the scope of the quantifier ∃x

b. ∀x ((WIZARD(x) ∧ ¬x=v) → MORTAL(x))
The wff ((WIZARD(x) ∧ ¬x=v) → MORTAL(x)) is the scope of the quantifier ∀x

c. (∀x (WITCH(x) ∨ WIZARD(x)) → ¬∃y MUGGLE(y))
The wff (WITCH(x) ∨ WIZARD(x)) is the scope of the quantifier ∀x.
The wff MUGGLE(y) is the scope of the quantifier ∃y.

(28) A variable *x* is **bound** in a wff φ if and only if it is in the scope of a quantifier ∀x or ∃x in φ; otherwise it is **free**.

Examples

a. ∃y KISS(x,y)
In this expression, the variable y is bound because it is in the scope of ∃y but the variable x is free because it is not in the scope of a quantifier.

b. (∀x (MAN(x) ∨ WOMAN(x)) → ¬MUGGLE(y))
In this expression, the variable x is bound because it is in the scope of ∀x but the variable y is free because it is not in the scope of a quantifier.

(29) A wff φ is a **proposition** iff it has no free variables in it.

The primary use of rule iv. is to bind variables so that none are free in a wff. In essence, this rule forms propositions out of wffs with free variables in them.

5. Translation

Translating sentences without variables or quantifiers is straightforward. Just remember that proper names translate into individual constants and verbs, common nouns, and adjectives into predicates. Be careful to combine the right number of arguments with each n-place predicate. You combine two wffs together using the connectives just like you did in Propositional Logic. Here are two examples to get you started.

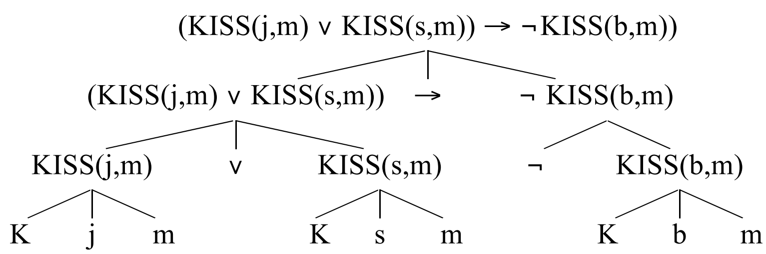
(30) If John will kiss Mary or Sue will, then Bob won't

Let KISS = will kiss, j = John, m = Mary, s = Sue

Translation: $(KISS(j,m) \vee KISS(s,m)) \rightarrow \neg\, KISS(b,m))$

John will kiss Mary	$KISS(j,m)$
Sue will kiss Mary	$KISS(s,m)$
Bob will kiss Mary	$KISS(b,m)$
Bob won't kiss Mary	$\neg KISS(b,m)$
John will kiss Mary or Sue will	$(KISS(j,m) \vee KISS(s,m))$
If John will kiss Mary or Sue will, then Bob won't	$(KISS(j,m) \vee KISS(s,m)) \rightarrow \neg KISS(b,m))$

Structure



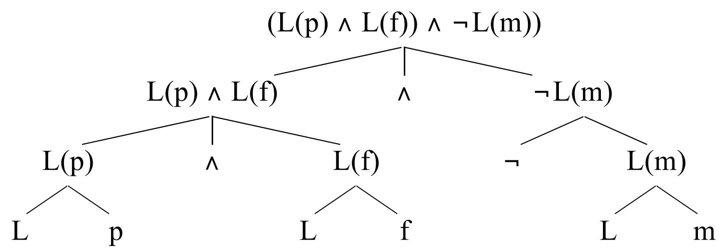
(31) Paul and Fred left home but Mary did not

Let L = left home, p = Paul, f = Fred, m = Mary

Translation: $((L(p) \wedge L(f)) \wedge \neg L(m))$

Paul left home	$L(p)$
Fred left home	$L(f)$
Mary left home	$L(m)$
Mary didn't leave home	$\neg L(m)$
Paul and Fred left home	$(L(p) \wedge L(f))$
Paul and Fred left home but Mary did not	$((L(p) \wedge L(f)) \wedge \neg L(m))$

Structure



Translating sentences with quantified expressions like *every man*, *someone*, and *nobody*, is far more treacherous. To do it, you will need at least an intuitive understanding of what the quantifiers ∃ and ∀ mean. You will also need to know how to restrict their meaning. We begin by translating simple sentences with quantified expressions like (32) into propositions.

(32) a. Someone is sleeping
  b. No one is sleeping
  c. Everyone is sleeping
  d. Not everyone is sleeping

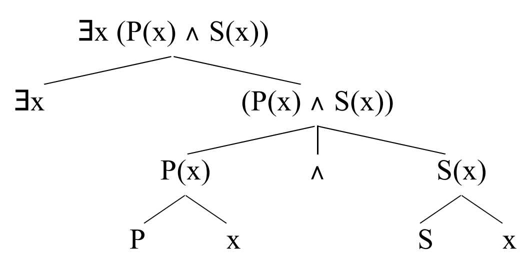
Here's the idea. A sentence like (30a) is true if and only if it describes a state of affairs in the outside world in which there is some object in the world such that it is a person and it is sleeping. We translate 'it is a person and it is sleeping' into the wff (P(x) ∧ S(x)), where P and S are 1-place predicates and *x* is a variable. The variable *x* stands in for the object that is both a person and sleeping. Finally, we translate 'there is some object' as ∃. Because the variable *x* is standing in for that object, we write ∃x. The resulting proposition has the form ∃x (P(x) ∧ S(x)).

(33) Someone is sleeping

Let S = is sleeping, P = person
∃x (P(x) ∧ S(x))

There is some entity x such that x is a person and x is sleeping

Structure



A sentence like (32b) is true if and only if it describes a state of affairs in which there is no object in the world such that it is a person and it is sleeping. This is the negation of the previous proposition. In other words, it is not the case that there is some object in the outside world such that it is a person and it is sleeping.

(34) No one is sleeping

Let S = is sleeping, P = person
$\neg\exists x\ (P(x) \wedge S(x))$

It is not the case that there is some entity x such that x is a person and x is sleeping

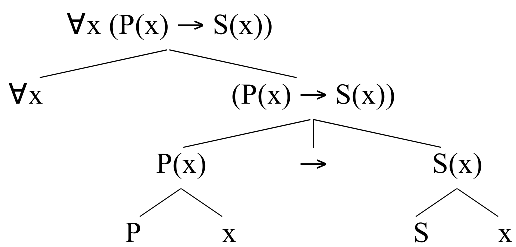
A sentence like (32c) is true if and only if it describes a state of affairs in the outside world in which for every object in the world, if it is a person then it is sleeping. We translate 'if it is a person, it is sleeping' into the wff $(P(x) \rightarrow S(x))$, where P and S are 1-place predicates and *x* is a variable. The variable *x* stands in for the object that is sleeping on the condition that it is a person. Finally, we translate 'for every object' as $\forall$. Because the variable *x* stands in for that object, we write $\forall x$. The resulting proposition is $\forall x\ (P(x) \rightarrow S(x))$.

(35) Everyone is sleeping

Let S = is sleeping, P = person
$\forall x\ (P(x) \rightarrow S(x))$

For every entity x, if x is a person, then x slept

<u>Structure</u>



A sentence like (32d) is true if and only if it describes a situation in which it is not the case that for every object in the world, if it is a person, then it is sleeping. This is just the negation of the previous proposition. It means there are some objects in the world that are people and not sleeping.

(36) Not everyone is sleeping

Let S = is sleeping, P = person: $\neg\forall x\ (P(x) \rightarrow S(x))$
It is not the case that for every entity x, if x is a person, then x slept

It is easy to confuse the translation for *everyone is sleeping* with the translation for *someone is sleeping*. In particular, it is easy to write (37) instead of (34).

(37) Everyone is sleeping

Let S = is sleeping, P = person
INCORRECT TRANSLATION: $\forall x\ (P(x) \wedge S(x))$

This is a proposition but it is not the correct translation of *everyone is sleeping*. The mistake is using conjunction $\wedge$ instead of $\rightarrow$. Think about what the proposition in (37) means: *for every entity x, x is a person* *and x slept*. For this proposition to be true, it must be the case that every entity in the universe is both a person and is sleeping. This is not what the sentence *everyone is sleeping* means. Using → guarantees that only those entities that are people are also sleeping.

We can now go from the sentences in (32) to slightly more elaborate cases in (38).

(38) a. Some dragon is sleeping
b. No dragon is sleeping
c. Every dragon is sleeping
d. Not every dragon is sleeping

The difference is minimal. The English sentences are a little less compact than with *someone* and *everyone* but the logical representations are identical. Instead of using a predicate P = person, you simply use a predicate D = dragon.

(39) a. Some dragon is sleeping
$\exists x\ (D(x) \wedge S(x))$

b. No dragon is sleeping
$\neg\exists x\ (D(x) \wedge S(x))$

c. Every dragon is sleeping
$\forall x\ (D(x) \rightarrow S(x))$

d. Not every dragon is sleeping
$\neg\forall x\ (D(x) \rightarrow S(x))$

We can get even more elaborate by adding the adjective, such as *green*, to the noun phrase. Simply let G = green. This is illustrated in (40).[3]

(40) a. Some green dragon is sleeping
$\exists x\ ((G(x) \wedge D(x)) \wedge S(x))$

b. No green dragon is sleeping
$\neg\exists x\ ((G(x) \wedge D(x)) \wedge S(x))$

c. Every green dragon is sleeping
$\forall x\ ((G(x) \wedge D(x)) \rightarrow S(x))$

d. Not every green dragon is sleeping
$\neg\forall x\ ((G(x) \wedge D(x)) \rightarrow S(x))$

---

[3] You'll notice that these equations contain a mess of parentheses. I assure you they are all in the right place but it's a mess to read. Parentheses are necessary in determining the scope of a quantifier in compound wffs. For example, the ill-formed formula $\exists x\ G(x) \wedge D(x) \wedge S(x)$ has no parentheses. It is impossible to tell the scope of the quantifier $\exists x$ is in this expression. Its scope could be G(x) alone, $(G(x) \wedge D(x))$, or $((G(x) \wedge D(x)) \wedge S(x))$. Each scope corresponds to a distinct meaning. Parentheses help disambiguate wffs in Predicate Logic. So don't neglect them.

Take note as to where the predicate G surfaces in these wffs, close to the predicate D. Intuitively, its meaning—green things—intersects with the meaning of the noun—dragons. The result says something about things that are both green and dragons. (Not all adjectives behave like this but for now let's assume they do.)

In a similar fashion, if we conjoin another verb phrase to the original verb phrase their meanings also intersects. This is illustrated in (41), where T = is twitching.

(41) a. Some dragon is sleeping and twitching
$\exists x\ [D(x) \wedge (S(x) \wedge T(x))]$

b. No dragon is sleeping and twitching
$\neg\exists x\ [D(x) \wedge (S(x) \wedge T(x))]$

c. Every dragon is sleeping and twitching
$\forall x\ [D(x) \rightarrow (S(x) \wedge T(x))]$

d. Not every dragon is sleeping and twitching
$\neg\forall x\ [D(x) \rightarrow (S(x) \wedge T(x))]$

In (41), the verb phrase *is sleeping and twitching* is translated as the wff $(S(x) \wedge T(x))$. We can also coordinate verb phrases using disjunction—something we cannot do with adjectives. The result is illustrated in (42), where the verb phrase *is sleeping or twitching* translates into the wff $(S(x) \vee T(x))$.

(42) a. Some dragon is sleeping or twitching
$\exists x\ [D(x) \wedge (S(x) \vee T(x))]$

b. No dragon is sleeping or twitching
$\neg\exists x\ [D(x) \wedge (S(x) \vee T(x))]$

c. Every dragon is sleeping or twitching
$\forall x\ [D(x) \rightarrow (S(x) \vee T(x))]$

d. Not every dragon is sleeping or twitching
$\neg\forall x\ [D(x) \rightarrow (S(x) \vee T(x))]$

Things are identical with 2-place predicates and one quantificational expression, as in (43) and (44).

(43) a. Some man kissed Sue
$\exists x\ (MAN(x) \wedge KISS(x,s))$

b. Every man kissed Sue
$\forall x\ (MAN(x) \rightarrow KISS(x,s))$

(44) a. Sue kissed some man
$\exists x\ (MAN(x) \wedge KISS(s,x))$

b. Sue kissed every man
$\forall x\ (MAN(x) \rightarrow KISS(x,s))$

Complications arise when an n-place predicate has two quantificational expressions. Cases where both quantificational expressions are the same, e.g., *some-some* or *every-every*, are straightforward.

(45) a. Some girl kissed some boy

$\exists x\ \exists y\ [\ (GIRL(x) \wedge BOY(y)) \wedge KISS(x,y)\ ]$

There is an entity x and there is an entity y such that x is a girl and y is a boy and x kissed y; in other words, there is a boy and there is a girl such that she kissed him

b. Every girl kissed every boy

$\forall x\ \forall y\ [\ (GIRL(x) \wedge BOY(y)) \rightarrow KISS(x,y)\ ]$

For every entity x and for every entity y, if x is a girl and y is a boy, then x kissed y; in other words, for every girl and for every boy, she kissed him

Cases where two quantificational expressions differ, e.g., *some-every* or *every-some*, are not as clear because the quantificational expressions are doing different things. So there interaction produces ambiguity. We are not equipped to handle these yet. Here is an example to whet your appetite but to get to the bottom of (46) will have to wait.

(46) Every girl kissed some boy

Meaning 1: $\forall x\ [\ GIRL(x) \rightarrow \exists y\ (BOY(y) \wedge KISS(x,y))\ ]$

For every entity x, if x is a girl then there is some entity y, such that y is a boy and x kissed y; in other words, for every girl, there is a boy, such that she kissed him.

Meaning 2: $\exists y\ [\ BOY(y) \wedge \forall x\ (GIRL(x) \rightarrow KISS(x,y))\ ]$

There is some entity y, such that y is a boy and for every entity x, if x is a girl then x kissed y; in other words, there is a boy, such that for every girl, she kissed him.