Semantics for the Predicate Calculus: Part I

(Version 0.3, revised 6:15pm , April 14, 2005. Please report typos to hhalvors@princeton.edu.)

The study of formal logic is based on the fact that the validity of an argument depends only on its logical form; for the purposes of logical investigation, we can ignore that argument's content. In particular, if two arguments have the same logical form, then either they are both valid, or they are both invalid.

Our main goal in this course is to figure out what it means to say that an argument is valid; or, in other words, to define what a valid argument form is. As an aid in achieving this goal, we have developed two ways for representing argument forms: First we developed the propositional calculus, which is supposed to give a perspicuous representation of the role of (truth functional) sentence connectives in sentences and arguments. Then we developed the predicate calculus, which also represents the role of quantifiers such as "all" and "some."

Now, since the sentences of the propositional and predicate calculi have no content, they are not, by themselves, either true or false. But our definition of validity makes use of the notion of truth-preservation:

> *An argument form is valid just in case: In any situation where its premises are true, its conclusion is also true.*

We make precise this notion of "situation" by means of the concept of an *interpretation*. In the propositional calculus, an interpretation of a sentence is just a line in that sentence's truth table; or, it other words, it is an assignment of truth values to the elementary sentences that occur in that sentence.

We now need a notion of *interpretation* for sentences in the predicate calculus. But truth tables will not do the trick: in the predicate calculus, we cannot arbitrarily assign truth values to truth-functionally simple sentences. For example, the sentence $(\exists x)(Fx \,\&\, -Fx)$ is truth-functionally simple, but we do not want to say that this sentence could be true! So, an interpretation of a predicate calculus sentence must start at a more primitive level than the level of truth-functionally simple sentences. Rather, it must start at the level of names and predicates.

A predicate calculus interpretation includes three things: (a) a specification of the class of objects that we will be talking about; (b) an assignment of names to objects; (c) an assignment of predicate letters to predicates, or classes of objects.

1 Informal Counterexamples

Let's begin by looking at an example. You will share my intuition that:

$$\frac{(x)(Fx \vee Gx)}{(x)Fx \vee (x)Gx}$$

is not a valid argument form. And if asked to justify your claim, you would most likely think of examples of *predicates* to replace F and G so that the premise comes out as true, and the conclusion comes out as false. For example, you might present the following argument as a counterexample:

<u>All numbers are either even or odd</u>
Either all numbers are even, or all numbers are odd.

Since this argument has true premises and a false conclusion, it certainly cannot have a valid form.

What you have done here is to give an interpretation of the argument form. Here are the details of your interpretation:

(a.) The domain is the natural numbers: $1, 2, \ldots$ ("Domain" is just a fancy name for the collection of things that you are talking about.)

(b.) The predicate letter F is interpreted as the predicate "is even", and the predicate letter G is interpreted as the predicate "is odd."

2 Interlude: predicates and their extensions

Definition. The *extension* of a predicate is just the collection of those things to which the predicate truly applies.

Example. The extension of the predicate "is even" (relative to the domain of natural numbers) is just the collection of even numbers.

Example. Suppose that our domain consists of just the numbers 1,2 and 3. Then the extension of the predicate "is even" consists of just the number 2.

3 Interlude: elementary set theory

To be more precise about predicate calculus interpretations, we are going to have to learn a bit of elementary set theory. Set theory is the science of collections of objects; and a collection of objects is called a *set*. Elementary set theory does not require any knowledge that you do not already have — it only requires learning some notation to make your reasoning more perspicuous and efficient.

Here are three numbers: $2, 6, 17$. The set of those three numbers is $\{2, 6, 17\}$. That is, we use brackets around the objects to indicate the set that consists of those objects.

The number 2 is a member of the set $\{2, 6, 17\}$. We can indicate this fact by writing $2 \in \{2, 6, 17\}$, where "$\in$" is shorthand for "is a member of."

Every member of the set $\{2, 17\}$ is also a member of the set $\{2, 6, 17\}$. So, we say that $\{2, 17\}$ is a *subset* of $\{2, 6, 17\}$. And if we don't feel like writing out the words, we might just write $\{2, 17\} \subseteq \{2, 6, 17\}$ as a shorthand.

Notice that is does *not* make sense to write $2 \subseteq \{2, 6, 17\}$, because 2 is not itself a set. It does make sense to write $\{2\} \subseteq \{2, 6, 17\}$.

Two sets A and B are equal, written $A = B$, just in case they have the same members (and order does not matter). So, for example, $\{2, 6, 17\}$ and $\{17, 2, 6\}$ are the same set.

There is one (and only one) set that does not contain anything at all. This set is called the *empty set*, and it is sometimes denoted by $\emptyset$.

Question. Is the empty set a subset of $\{2, 6, 17\}$?

If A and B are sets, we use $A \cap B$ to denote the set that consists of those things that are in both A and B. We use $A \cup B$ to denote the set that consists of those things that are in either A or B.

4 Formal Counterexamples

Now we revisit the concept of counterexamples and interpretations, this time with an eye toward precision and rigor.

Definition. An *interpretation* of a sentence (or sentences) of the predicate calculus consists of:

(a.) A set X, called the *domain of quantification*.

(b.) An assignment Ref of names to elements of X.

(c.) An assignment Ext of predicate letters to subsets of X.

Definition. A *formal counterexample* to an argument in the predicate calculus is an interpretation where the premises of the argument are true, and the conclusion of the argument is false.

I have not yet given a precise definition of when a sentence is true relative to an interpretation. I have not done so because the definition turns out to be quite complicated, and your intuitions will be sufficient for many cases.

Example. Consider the interpretation with domain $X = \{1, 2, 3, 4\}$ and

$$\mathrm{Ref}(m) = 1, \quad \mathrm{Ref}(n) = 1, \quad \mathrm{Ref}(o) = 2,$$
$$\mathrm{Ext}(Fx) = \{1, 2, 3\} \quad \mathrm{Ext}(Gx) = \{3, 4\} \quad \mathrm{Ext}(Hx) = \emptyset.$$

Then, it follows that:

- $(\exists x)(Fx \,\&\, Gx)$ is true relative to this interpretation, since the element 3 is in both $\mathrm{Ext}(Fx)$ and $\mathrm{Ext}(Gx)$.
- $(x)(Fx \to Gx)$ is false relative to this interpretation, since the element 1 is in $\mathrm{Ext}(Fx)$ but not in $\mathrm{Ext}(Gx)$.

- $(x)(Fx \vee Gx)$ is true relative to this interpretation, since every element in the domain is in either $\mathrm{Ext}(Fx)$ or $\mathrm{Ext}(Gx)$.
- $(x)(Hx \rightarrow Gx)$ is true relative to this interpretation, since $\mathrm{Ext}(Hx)$ is the empty set.

Problem. Give a formal counterexample to the argument:

$$(x)(Fx \vee Gx) \vdash ((x)Fx \vee (x)Gx).$$

Solution: Let $X = \{1, 2\}$, let $\mathrm{Ext}(Fx) = \{1\}$ and $\mathrm{Ext}(Gx) = \{2\}$. Since all elements of X are either in $\mathrm{Ext}(Fx)$ or $\mathrm{Ext}(Fx)$, $(x)(Fx \vee Gx)$ is true relative to this interpretation. But since $\mathrm{Ext}(Fx)$ is not equal to X, $(x)Fx$ is not true relative to this interpretation. Similarly, $(x)Gx$ is not true relative to this interpretation, and so $(x)Fx \vee (x)Gx$ is not true relative to this interpretation.

Problem. Give a formal counterexample to the argument with no premises and conclusion $(x)(y)(Fx \rightarrow Fy)$.

4.1 Specifying interpretations with tables

It can be boring to write out the interpretations of a bunch of predicates. So, in order to spice up life, we give here an alternative way to specify the interpretation of predicate letters. (This method only works when the domain X is a finite set.)

First we list the predicate letters in the top row of the table, and we list the elements of the domain in the first column of the table. We then write "$+$" in cell (i, j) if the predicate in column j applies to the object on row i. Otherwise, we write "$-$" in cell (i, j).

Problem. Give an interpretation for a problem involving the predicate letters F, G, H, and the names m, n, o.

Solution:

$$X = \{1, 2, 3\}, \qquad \mathrm{Ref}(m) = 1, \quad \mathrm{Ref}(n) = 1, \quad \mathrm{Ref}(o) = 2.$$

	F	G	H
1	$+$	$-$	$-$
2	$+$	$+$	$-$
3	$+$	$-$	$-$

Instead of drawing the table, we could have just written:

$$\mathrm{Ext}(Fx) = \{1,2,3\}, \quad \mathrm{Ext}(Gx) = \{2\}, \quad \mathrm{Ext}(Hx) = \emptyset.$$

But don't underestimate how fun it can be to draw tables.

When giving an interpretation, you must be clear and explicit about which subsets are assigned to which predicate letters. But given that you satisfy the requirements of clarity and explicitness, you can use whatever method you want to specify your interpretation.

Note. An interpretation must assign the same subset of the domain to both Fx and Fy, even though they have different variables. In other words, the interpretation gives the extension of the predicate letter F, and doesn't care about what variable we put after F. We will deal later with the tricky case of sentences that involve the same predicate letter with different variables — e.g., $(x)(y)(Fx \to Fy)$ and $(\exists x)(Fx \to (y)Fy)$.

5 Semantic properties of sentences, relations between sentences

Recall that when we were doing the propositional calculus, we defined a bunch of special kinds of sentences (e.g., tautologies, inconsistencies), and a bunch of special logical relationships between sentences (e.g., implies, is subcontrary to). Actually, each of those concepts was defined in terms interpretations, and so they can be naturally extended to predicate calculus sentences. For example, a predicate calculus sentence is *tautologous* just in case it is *true relative to every interpretation*.

Definition.

- If ϕ is false relative to some interpretation, then ϕ is said to be *falsifiable*.

- If ϕ is false relative to every interpretation, then ϕ is said to be *inconsistent*.
- If ϕ is true relative to some interpretation, then ϕ is said to be *consistent*.
- Let Γ be a set of sentences. If there is an interpretation relative to which every sentence in Γ is true, then Γ is said to be *consistent*. Otherwise, Γ is said to be *inconsistent*.
- A set Γ of sentences *logically implies* a sentence ψ if there is no interpretation that makes all of Γ true while making ψ false. That is, there is no *formal counterexample*. In that case, the argument with premises Γ and conclusion ψ is *valid*.
- Two sentences ϕ, ψ are *logically equivalent* if they have the same truth value relative to every interpretation.

Example. $(x)(Fx \vee -Fx)$ is tautologous, because in any interpretation, each object in the domain is either in $\mathrm{Ext}(Fx)$, or is not in $\mathrm{Ext}(Fx)$.

Example. $(\exists x)(Fx \,\&\, -Fx) \rightarrow Gm$ is tautologous, because $(\exists x)(Fx \,\&\, -Fx)$ is inconsistent.

6 Semantic problems and answers

An answer to a problem requiring the presentation of an interpretation is best seen as having two parts, as follows: (1) state the domain of your proposed interpretation, and present (using one of the above methods) the interpretation of the names and predicate symbols; (2) state the truth values of the various sentences, defend your claim that they have those truth values, and (most importantly) be explicit as to how this information solves the problem you began with. Let's consider a couple of examples of solved problems.

Problem. Show that $(\exists x)Fx \rightarrow (x)Fx$ is not tautologous.

Solution:

1. Let $X = \{1, 2\}$, and let $\mathrm{Ext}(Fx) = \{1\}$.
2. The sentence is false relative to this interpretation: Since 1 is in $\mathrm{Ext}(Fx)$, $(\exists x)Fx$ is true relative to this interpretation. However, since 2 is not in $\mathrm{Ext}(Fx)$, $(x)Fx$ is false relative to this interpretation. Therefore (by truth

tables) $(\exists x)Fx \rightarrow (x)Fx$ is false relative to this interpretation. Since the sentence is false relative to *some* interpretation, it is not tautologous. $\square$

Problem. Show that the sentence from the previous problem is consistent (true relative to some interpretation).

Solution:

1. Let $X = \{1\}$, and let $\mathrm{Ext}(Fx) = \{1\}$.
2. Since $\mathrm{Ext}(Fx) = X$, it follows that $(x)Fx$ is true relative to this interpretation, and therefore (by truth tables) $(\exists x)Fx \rightarrow (x)Fx$ is true relative to this interpretation. Since the sentence is true relative to some interpretation, it is consistent.

6.1 Is there an algorithm for finding interpretations?

Solving a semantic problem (e.g., "is the argument with premises Γ and conclusion ϕ valid?") requires one to check all possible interpretations. But there are infinitely many interpretations. So, it seems that your next homework assignment will take a long time to finish!

Joking aside, it is generally quite difficult to figure out if a predicate calculus argument is valid. Moreover, the difficulty cannot be fully resolved — as it is in the propositional calculus, where truth tables providing a completely reliable test for validity. Indeed, logicians have proven that *the task of deciding if a predicate calculus argument is valid cannot be reduced to a routine algorithm*. (This interesting result would be discussed in an advanced logic course, such as PHI 312 or PHI 321.)

However, in the special case where our sentences contain only monadic predicates (i.e., no "relation symbols"), we can transform this hard task into a routine algorithm. That will be our next topic.

7 Decision procedures for monadic predicate calculus

7.1 Algorithm A

The truth table method is an algorithm for testing the consistency of sentences. In the spirit of inflationary terminology, we shall henceforth call the truth table method "Algorithm A."

7.2 Algorithm B

Definition. Suppose that we put together some predicate letters with the same variable using truth functional connectives. For example:

$$(Fx \,\&\, Gx) \to -Hx.$$

Suppose that we then put a quantifier in front, yielding a sentence. For example:

$$(x)((Fx \,\&\, Gx) \to -Hx).$$

The result is called a *simple monadic sentence*. So, a simple monadic sentence is a sentence with only one quantifier at the very front.

Definition. If ϕ is a simple monadic sentence, we let ϕ^a denote the instance of ϕ that is obtained by taking off the initial quantifier and replacing all instances of the variable with the name a.

We now show how to determine if a collection $\phi_1, \dots, \phi_n$ of simple monadic sentences is consistent. We assume that the first m sentences are existential, and the remaining sentences are universal.

1. If $m = 0$, then choose one arbitrary name a. Perform a truth table test for consistency on $\phi_1^a, \ldots, \phi_n^a$. If the truth table test says consistent, then output Yes. If the truth table test says inconsistent, then output No.
2. If $m \geq 1$, then choose m arbitrary names $a_1, \ldots, a_m$. Perform a truth table test for consistency on:

$$\begin{array}{c} \phi_1^{a_1}, \ldots, \phi_m^{a_m} \\ \phi_{m+1}^{a_1}, \ldots, \phi_{m+1}^{a_m} \\ \vdots \\ \phi_n^{a_1}, \ldots, \phi_n^{a_m} \end{array}$$

   If the truth table test says consistent, then output Yes. If the truth table test says inconsistent, then output No.

7.3 Algorithm C

Before you read this section, you need to know how to transform a sentence into a "disjunctive normal form" equivalent. Please read Appendix A of Lemmon's book.

Definition. A *pure monadic sentence* is a sentence that is a truth-functional combination of simple monadic sentences. For example:

$$(\exists x)Fx \,\&\, (y)(Gy \to Fy),$$

is a pure monadic sentence.

The following algorithm will determine if a collection $\phi_1, \ldots, \phi_n$ of pure monadic sentences is consistent.

1. Transform $\phi_1, \ldots, \phi_n$ into disjunctive normal form. The result will be of the form $\psi_1 \vee \cdots \vee \psi_m$, where each ψ_i is a conjunction of simple monadic sentences and negated simple monadic sentences.
2. Within each disjunct $\psi_1, \ldots, \psi_n$, change each negated simple monadic sentence into a simple monadic sentence using the quantifier-negation equivalences.
3. Test each disjunct using Algorithm B. If Algorithm B answers Yes for one of the disjuncts, then output Yes. If Algorithm B answers No for each of the disjuncts, then output No.

7.4 The small domain method

Although Algorithm C always gives the correct answer, it has the drawback that it does not match the way that we normally reason when we try to decide if some sentences are consistent. So, here I give you another method to test for consistency.

The small domain method is based on the following fact that has been proven by paid logicians:

> **Fact:** If a pure monadic sentence is consistent, then it is true relative to some "small" domain. (In fact, the size of the domain needed is a function of the number of predicates and variables in the sentences.)

Thus, to test a pure monadic sentence for consistency, do the following: Find a quantifier-free sentence that is equivalent to the original sentence relative to a domain with one object; test this resulting sentence for consistency using ordinary truth tables. If the resulting sentence is consistent, you are done — the original sentence is true in that domain. If the resulting sentence is inconsistent, then start over again with a domain with two individuals. If the resulting sentence is consistent, you are done — the original sentence is true in that domain. If the resulting sentence is inconsistent, then repeat the procedure in a domain with three individuals, etc., until either you find an interpretation relative to which the sentence is true, or you conclude that there is no such interpretation. For when you are entitled to draw the latter conclusion, see the last section.

7.4.1 Finding equivalent quantifier-free sentences

Suppose that the domain X has only three objects $1, 2, 3$. In this case, the universal statement "$(x)Fx$" is equivalent to a conjunction: "1 is an F, 2 is an F, and 3 is an F." Similarly, the existential statement "Something is a F" is equivalent to the disjunction "Either 1 is an F, or 2 is an F, or 3 is an F." In sum, when there are only finitely many things, we can (by naming each object) translate every simple monadic sentence into a sentence without quantifiers.

The same sort of equivalences also hold for truth-functional combinations of simple monadic sentences (i.e., pure monadic sentences). For example, if we take the standard association of names with numbers:

$$\mathrm{Ref}(a) = 1 \quad \mathrm{Ref}(b) = 2 \quad \mathrm{Ref}(c) = 3,$$

then $(x)Fx \to (\exists x)Fx$ is equivalent to:

$$(Fa \,\&\, Fb \,\&\, Fc) \to (Fa \vee Fb \vee Fc).$$

Generally, in order to obtain a quantifier-free sentence that is equivalent (relative to some finite domain) to a sentence ϕ, you should:

1. Disassemble ϕ into truth-functionally simple components;
2. If any of these components are quantified statements, expand them into equivalent conjunctions or disjunctions;
3. Put the expanded statements back together again using the original truth-functional connectives.

Problem. Find a quantifier-free sentence that is equivalent in a domain with three objects to $(x)(Fx \to Gx) \vee (\exists x)(Hx \,\&\, Mx)$.

Solution: Since the main operator is a disjunction "$\vee$", we separate the original sentence into $(x)(Fx \to Gx)$ and $(\exists x)(Hx \,\&\, Mx)$. Since these two statements are quantified statements, we expand them as follows:

$$\begin{array}{lcl} (x)(Fx \to Gx) & \equiv & (Fa \to Ga) \,\&\, (Fb \to Gb) \,\&\, (Fc \to Gc) \\ (\exists x)(Hx \,\&\, Mx) & \equiv & (Ha \,\&\, Ma) \vee (Hb \,\&\, Mb) \vee (Hc \,\&\, Mc) \end{array}$$

Finally, we put these back together again using $\vee$ to obtain:

$$\begin{aligned}&[(Fa \rightarrow Ga) \,\&\, (Fb \rightarrow Gb) \,\&\, (Fc \rightarrow Gc)]\\ &\quad\vee \quad [(Ha \,\&\, Ma) \vee (Hb \,\&\, Mb) \vee (Hc \,\&\, Mc)].\end{aligned}$$

$\square$

Problem. For each of the sentences $(x)Fx$, $(\exists x)Fx \rightarrow (\exists x)Gx$, $(x)(Gx \vee Gx)$, find a quantifier-free sentence that is equivalent relative to a domain containing two individuals.

Solution:

$$\begin{array}{ccc} (x)Fx & \equiv & Fa \,\&\, Fb \\ (\exists x)Fx \rightarrow (\exists x)Gx & \equiv & (Fa \vee Fb) \rightarrow (Ga \vee Gb) \\ (x)(Gx \vee Fx) & \equiv & (Ga \vee Fa) \,\&\, (Gb \vee Fb) \end{array}$$

7.4.2 Testing for consistency

Relative to domains with a finite number of individuals, we can test the consistency of a sentence of monadic PC using ordinary truth tables: Just translate the sentence into an equivalent quantifier-free sentence.

Problem. Could $(\exists x)Fx \,\&\, (\exists x) - Fx$ be true relative to a domain with only one object?

Solution: If there were only one thing in the universe, then $(\exists x)Fx \,\&\, (\exists x) - Fx$ would be equivalent to $Fa \,\&\, - Fa$, which is a contradiction. So, no; it couldn't be true if there were only only object in the domain. $\square$

Problem. Could $(\exists x)Fx \,\&\, (\exists x) - Fx$ be true if there were exactly two objects in the domain?

Solution: In this case $(\exists x)Fx \,\&\, (\exists x) - Fx$ would be equivalent to:

$$(Fa \vee Fb) \,\&\, (-Fa \vee -Fb).$$

A truth table test shows that this sentence is consistent — e.g., choose $v(Fa) = \mathrm{T}$ and $v(Fb) = \mathrm{F}$. So, yes; $(\exists x)Fx \,\&\, (\exists x) - Fx$ could be true if there were two things in the domain. $\square$

7.4.3 Putting it all together

Problem. Use the small domain method to show that $(x)(Fx \to -Fx)$ is consistent.

Solution: In a domain with one individual, the original statement becomes $Fa \to -Fa$. This is true when Fa is false. So, $(x)(Fx \to -Fx)$ can be true in a domain with one individual. In particular, here is an interpretation which shows explicitly that $(x)(Fx \to -Fx)$ is consistent:

$$X = \{1\}, \qquad \mathrm{Ext}(Fx) = \emptyset.$$

$\square$

Problem. Use the small domain method to determine if the following argument is valid:

1. $(\exists x)Hx \to (x)(Fx \to Gx)$
2. $(\exists x)Fx \qquad // \; (\exists x)Hx \to (x)Gx$

Solution: Relative to a domain with one member, we get the argument:

1. $Ha \to (Fa \to Ga)$
2. $Fa \qquad // \; Ha \to Ga$

A truth table test shows that if the two premises are true, then the conclusion must also be true. So, we now try a domain with two members. Relative to a domain with two members, we get the argument:

1. $(Ha \vee Hb) \to ((Fa \to Ga) \,\&\, (Fb \to Gb))$
2. $Fa \vee Fb \qquad // \; (Ha \vee Hb) \to (Ga \,\&\, Gb)$

A truth table test shows that the premises can be true while the conclusion is false. For example, we could choose the truth-assignment:

$$v(Ha) = \mathrm{T} \quad v(Hb) = \mathrm{F}$$

$$v(Ga) = \mathrm{T} \quad v(Gb) = \mathrm{F}$$

$$v(Fa) = \mathrm{T} \quad v(Fb) = \mathrm{F}$$

Therefore, the original argument is invalid. $\square$

7.4.4 Decision procedures

Suppose that you need to determine whether a sentence ϕ is consistent. You check a domain with one individual, and ϕ is false. You check a domain with two individuals and ϕ is false. You check a domain with three individuals and ϕ is false. Surely you cannot check domains of all sizes! When, if ever, are you entitled to conclude that there is *no* interpretation relative to which ϕ is true? Amazingly, it has been shown that:

> *Suppose that ϕ is a pure monadic sentence with n predicate letters. If ϕ is consistent, then there is an interpretation $\mathscr{I}$ whose domain has less than or equal to 2^n elements, and ϕ is true relative to $\mathscr{I}$.*[1]

So, to take a specific case, if ϕ has two predicate letters, then for ϕ to be consistent, it must be true in some domain with at most 4 objects! It follows that the small domain method is a "decision procedure" for the consistency of pure monadic sentences: it will answer any question you have about consistency in a finite amount of time.

[1]Compare with Boolos and Jeffrey, *Computability and Logic*, p. 250.