First Order Predicate Logic

MATH230

Te Kura Pāngarau
Te Whare Wānanga o Waitaha

Outline

Motivation

Recall that we have introduced logic so that we can write mathematics in a precise enough manner to analyse proof and provability.

In defining propositional logic we suppressed a lot of detail into a single propositional variable P. It's time now to look at each proposition in more detail and parse out different components.

In this lecture we will approach the topic of first-order logic by determining which components are still missing from our language.

Fundamental Theorem of Arithmetic

For all integers $n > 1$ either n is a prime or n is a finite product of primes.

Details of Propositions

In the statement of the fundamental theorem of arithmetic was found the following details within' propositions:

T	:	"Universe of discourse" or "Type" or "Set".
n : T	:	Terms of that type.
is prime	:	"Predicate" or "Property" which makes propositions.
For all	:	Quantification over the Type.

If we consider "is prime" or "n $>$ 1", then we find even more structure. Each of these are making "existential" claims; that is, claims about whether or not certain other terms exist. Previously we put all of these details into one propositional variable. First Order Predicate Logic brings this information out into the open, hence allowing for more nuanced propositions.

Terms and their Properties

Mathematicians and computer scientists speak about terms of specific types:

$3 : \mathbb{N}$ $\qquad -4 : \mathbb{Z}$ $\qquad \pi : \mathbb{R}$

$\mathsf{True} : \mathsf{Bool}$ $\qquad [1,2,3] : \mathsf{List}\ \mathbb{N}$

Furthermore they are interested in proving that these terms have certain properties:

$\mathsf{even} : \mathbb{N} \to \mathsf{Prop}$	$2 \mapsto \mathsf{even}\ 2$
$\mathsf{prime} : \mathbb{N} \to \mathsf{Prop}$	$4 \mapsto \mathsf{prime}\ 4$
$\mathsf{null} : \mathsf{List}\ \mathbb{N} \to \mathsf{Prop}$	$[1,2,3] \mapsto \mathsf{null}\ [1,2,3]$
$\mathsf{equal} : \mathbb{N} \times \mathbb{N} \to \mathsf{Prop}$	$(a, b) \mapsto a = b$

These properties are called predicates. In first order predicate logic they are thought of as functions that return propositions.

Variables: $x, y, z...$

Some of these statements are written *generally*. That is to say, the values associated to the particular letters are left open to be anything within' the domain of discourse:

- If n is an integer...
- If a, b are integers...
- If p is some prime...
- There exists an integer n...

So our language will need the ability to express *variables* which can be interpreted in some specific *type*. We will also refer to a general type and general predicates on types; thus we will use variables for terms, types, and predicates. Lowercase variables will be used for terms, whereas uppercase variables will be used to denote types and predicates.

Quantification: $\forall$ and $\exists$

Quantification goes hand-in-hand with this expression of generality and use of variables. What we mean when we say:

$$\text{If } n \text{ is an integer, then even } (n) \vee \text{odd } (n)$$

Every integer satisfies the proposition even $(n) \vee$ odd (n).

Similarly claims like *there exists* an n such that $\phi(n)$, are quantifiying over some type. Of all the terms in that type, at least one of them satisfies the predicate $\phi(n)$.

Functions: $f, g, h, \ldots$

Sums, products, and exponentiation were used to state the above claims. These are examples of the fundamental notion of *function*. Mathematicians and computer scientists use functions to express all sorts of things: we would be lost with them.

The idea being that functions take in some number of inputs (from the domain) and return one element of the domain of discourse.

Note: this makes them different from predicates/relations as predicates do not return elements of the domain. Rather, they return propositions.

PL to First-Order Logic

In summary, we need to add the following to propositional logic:

- Domain type
- Constants
- Functions
- Variables
- Properties/Relations (Predicates)
- Quantifiers

These, with propositional logic, form the language of *first-order predicate logic*.

Alphabet of FoL

We use the following standard language of *first-order predicate logic* to build terms and well-formed formula.

- Type variables $P, Q, R, S, T, ...$
- Term variables $x, y, z, ...$ or $x_1, x_2, x_3, ...$
- Constants $a, b, c, ...$ or $c_1, c_2, c_3, ...$
- Predicate (relation) symbols $R_0, R_1, R_2, ...$
- Function symbols $f_0, f_1, f_2, ...$
- Logical connectives from propostional logic $\neg, \wedge, \vee, \rightarrow, \bot$
- Quantifiers $\forall$ and $\exists$

Note: For clarity we also use parentheses and commas when writing expressions. These can be included in the symbols of our language as well.

Universal and Existential Quantifiers

The symbol $\forall$ will be interpreted as "for all..."

The symbol $\exists$ will be interpreted as "there exists..."

Example: For all real numbers $\epsilon > 0$ there exists another real number $\delta > 0$

For us, in these notes, quantifiers will always be stated with a domain of quantification; so called bounded quantification. This departs from most of the literature on logic which considers unbounded quantification.

Grammar of FoL

Terms of a domain T are defined inductively as follows:

- Variables and constants are terms
- If f is an n-ary function and $t_1, ..., t_n$ are terms, then $f(t_1, ..., t_n)$ is a term.

Propositions are defined inductively as follows:

- $\perp$: Prop
- An n-ary predicate applied to n-terms $R(t_1, ..., t_n)$ is a Prop.
- If α, β is a Prop, then $\neg\alpha$, $(\alpha \vee \beta)$, $(\alpha \wedge \beta)$, and $(\alpha \rightarrow \beta)$ are Prop.
- If α is a Prop and x is a variable, then $(\forall x : T,\ \alpha)$ and $(\exists x : T,\ \alpha)$ are Prop.

Examples

Let T be a type, R be a two-place (binary) predicate, P a one-place (unary) predicate, f a binary function, and a, b be constants. Determine which of the following are wff and which are not.

1. Pa
2. a
3. Pab
4. $\exists x : T,\ Rxb \rightarrow Px$
5. $\exists x(Px \vee Rbx)$
6. $\forall x : T,\ (\exists y : T,\ (R(x, y)))$
7. $\exists x : T,\ (\forall y : T,\ (R(x, y)P(y)P(x)))$
8. $f(y, b)$

Binding Conventions

If you want to drop parentheses, then your expressions will be interpreted using the following binding conventions:

- $\forall, \exists, \neg$ bind most tightly;
- $\wedge$ and $\vee$ bind more tightly than $\rightarrow$;
- $\rightarrow$ binds more tightly than $\leftrightarrow$.

With this convention one can unambiguously interpret the string:

$$\neg \exists x : T,\ Rx \rightarrow \neg Q \wedge \forall x : T,\ \neg Px$$

as the following wff:

Comments

The last few slides have said how we can build strings that are part of the language of first order logic. So far, these are just strings of symbols that fit together into terms and wff with the grammar defined.

They have no meaning, *yet*. We don't work this abstractly. Instead we choose construct specific base types with constants, functions, and predicates to express ideas and data precisely. It's this choice of "signature" that determines the meaning of the strings in this language.

First Order Languages

In practice we decide on a few constants, predicates, and functions depending on the part of mathematics (or computer science) we want to study. Thus specifying a *first order language* by defining the *signature* of the language.

Example

Let us say we have the constant 0, unary function s, binary functions $+$ and $\times$, and $=$ as the only relation.

We collect these together into a first order language, which we can denote $\mathcal{L}$, with signature $\{0, s, +, \times, =\}$

WFF in a First Order Language

Given a first order language $\mathcal{L}$ with signature $\{0, s, +, \times, =\}$ we see that the constants and function symbols generate the base type of the first order language:

Prop in a First Order Language

Given a first order language $\mathcal{L}$ with signature $\{0, s, +, \times, =\}$ whose terms generate type $\mathbb{N}$ we may write the following Propositions:

- $\forall x : \mathbb{N},\ \forall y : \mathbb{N},\ (=(x, y) \vee \neg(=(x, y)))$
- $\exists x : \mathbb{N},\ (\forall y : \mathbb{N},\ \neg(=(x, s(y))))$
- $=(+(0, s(0)), 0)$

Prefix, Infix, Postfix

Example

In the textbook Logic and Proof[1] they pick the following first order language for the entire chapter on first order logic:

$$\mathcal{L} : \{1, 2, 3, \mathit{add}, \mathit{mul}, \mathit{square}, \mathit{even}, \mathit{odd}, \mathit{prime}, \mathit{lt}, =\}$$

- Every natural number is even or odd, but not both.
- $\mathit{Even}(n) \leftrightarrow \exists x : \mathbb{N}^+,\ (n = \mathit{mul}(2, x))$.
- If x is even, then x^2 is even.
- There exists a prime number that is even.

[1] Note: this uses an outdated version of Lean, but is still a good resource for the theory of this course.

L∃∀N Example

$$\mathcal{L} : \{1, 2, 3, \mathit{add}, \mathit{mul}, \mathit{square}, \mathit{even}, \mathit{odd}, \mathit{prime}, \mathit{lt}, =\}$$

$$\mathcal{L}_{\mathbb{B}} : \{\mathsf{True}, \mathsf{False}, \wedge, \vee, \neg, \rightarrow, =\}$$

Arguments and Proofs

Recall that our intention is to understand when an argument written in our formal language is valid; when one wff is said to follow from a set of other wff.

To this end we now write down rules of inference for the new structure of first-order predicate logic. Namely, introduction and elimination rules for the quantifiers.

Bounded Quantification

Thus far we have considered the terms generated by the signature of a first-order language and collecting them together into a type. We have taken the quantifiers to range over this type.

This explicit mention of the type appears atypical of presentations of first-order logic. **For better, or for worse, we will present the rules of inference without explicit reference to a type of terms.**

BHK Interpretation

We turn, again, to the BHK for some idea of what it means to introduce/eliminate quantifiers:

$\forall$	a proof of $\forall x\ P(x)$ is an algorithm that, applied to *any* object t returns a proof of $P(t)$.
$\exists$	to prove $\exists x\ P(x)$ one must construct an object t and a proof of $P(t)$.

This presentation is taken from the Standford Encyclopedia of Philosophy article by Bridges, Palmgren, and Ishihara.

Universal Quantification

Suppose we had, in the course of a proof, deduced the statement $\forall x \alpha$, where α is a well-formed formula that (may) contain the variable x. What can we conclude from that?

$$\dfrac{\begin{array}{c}\Sigma \\ \mathcal{D} \\ \forall x\ \alpha\end{array}}{?}\ \forall E$$

Following the BHK interpretation of $\forall$ we should then be able to *substitute* any term, t, in place of each instance of the variable x in the well formed formula α.

Substitution Rules

If we don't take care when substituting for variables, then we can go astray.

Definition

If a variable, x, is used to quantify over α — as in $\gamma = \forall x \alpha$ — then we say any instance of x in α is bound in γ by the quantifer. Similarly for the existential quantifier.

Definition

If a variable, x, is not bound in γ, then we say that it is free in γ.

Bound and free variables act differently when interpreted; they play different roles.

Free vs. Bound Variables

We can translate the first-order expression: $\forall x(x < y)$ as "every x is less than _____". The interpretation of this statement depends on the interpretation of y; because y is free. It's truth will depend on the interpretation of y.

On the other hand the interpretation of $\exists y(\forall x(x < y))$ doesn't depend on y. As there either exists something that satisfies this sentence, or there doesn't.

For this reason we should not make free variables bound, or vice versa, when substituting.

Substitution Notation

If α is a well-formed formula in which x is a free variable and y is free to be substituted in for x, then we denote

$$\alpha[x/y] \qquad \alpha \text{ with } \underline{\text{all}} \text{ x replaced by y}$$

Example

$$\alpha \equiv x+y=s(x) \qquad \beta \equiv \forall x \forall y\ x+y=y+x \qquad \gamma \equiv \forall x\ x+y=x$$

Examples

x_2 is free for x_0 in $\exists x_3(P(x_0, x_3))$,

$f(x_0, x_1)$ is not free for x_0 in $\exists x_1(P(x_0, x_3))$,

x_5 is free for x_1 in $P(x_1, x_3) \rightarrow \exists x_1(Q(x_1, x_2))$.

What Can Go Wrong?

Consider the formula over the type of natural numbers:

$$\forall x \exists y (y > x).$$

This asserts that there is always a larger natural number. Notice that x is free in the formula $\exists y (y > x)$. So if we aren't careful and substitute $x \mapsto y + 1$, then we obtain the statement

$$\exists y (y > y + 1)$$

Which is not true. **This problem arose because y is bound in the formula, so we can't substitute $t = y + 1$ for x.**

Example from Section 7.3 of Logic and Proof

Universal Elimination

If $\overset{\Sigma}{\forall x \alpha}\mathcal{D}$ is a derivation of $\forall x\ \alpha$ from hypotheses Σ, where y is free for x in α, then

$$\dfrac{\begin{array}{c}\Sigma \\ \mathcal{D} \\ \forall x\ \alpha\end{array}}{\alpha[x/y]}\ \forall E$$

is a derivation of $\alpha[x/y]$ from hypotheses Σ.

Aside: Empty Types?

An interpretation closer to the BHK would have two branches. First we use a proof that any term $x : T$ necessarily has property α and then provide a term $t : T$ to substitute into the predicate.

$$\dfrac{\begin{array}{cc} \Sigma_1 & \Sigma_2 \\ \mathcal{D}_1 & \mathcal{D}_2 \\ \forall x : T,\ \alpha[x] & t : T \end{array}}{\alpha[x/t]}\ \forall E$$

In practice these are the same *under the assumption that T is non-empty.* Since logicians[2] do not concern themselves with empty types (models), they (and we) will just use the definition from the previous slide.

[2] Gentzen, for example, did not consider this in Section 2.21 of Investigations Into Logical Deduction.

Example

Every human is mortal. Sherlock is Human. Therefore, Sherlock is mortal.

Proof of For All

What does it mean to prove something "for all x"?

A proof of the irrationality of $\sqrt{2}$ starts as follows:

> Assume there are some integers A, B such that $\sqrt{2} = A/B$....
> Eventually deriving a contradiction from this assumption.

The proof itself, in the end, assumes *nothing* of the integers A, B apart from their (possible) existence. Thus it shows that no matter which integers are in place of x, y, the equation

$$(x^2 = 2y^2)$$

can have no integer solutions and hence we may conclude

$$\forall x, y \ \neg(x^2 = 2y^2)$$

Universal Introduction

If $\overset{\Sigma}{\underset{\alpha[x/y]}{\mathcal{D}}}$ is a derivation of $\alpha[x/y]$ from hypotheses Σ, where y does not appear free in Σ nor α, then

$$\dfrac{\begin{array}{c}\Sigma \\ \mathcal{D} \\ \alpha[x/y]\end{array}}{\forall x \alpha}\ \forall I$$

is a derivation of $\forall x \alpha$ from hypotheses Σ.

Properties of y can't play a role in the proof of $\alpha[x/y]$. They can't appear in the hypotheses Σ of the proof.
Nor can any y be free in alpha. You should be replacing all instances of y with an x when using the $\forall$ introduction.

Example

$\forall x(Px \to Qx),\ \forall xPx \ \vdash \ \forall xQx$

Example

$$\forall x(Px \to Qx),\ \forall x(Qx \to Rx) \vdash \forall x(Px \to Rx)$$

Example: When Things Go Wrong

Sherlock is a person. Gladstone is not a person. Therefore, Moriarty is the Queen.

Show $Ps, \neg Pg \vdash Q$

$$\dfrac{\dfrac{\neg Pg \qquad \dfrac{\dfrac{Ps}{\forall x Px}\,\forall I}{Pg}\,\forall E}{\bot}\,MP}{Q}\,\bot$$

Example: When Things Go Wrong

Sherlock is a person. Gladstone is not a person. Therefore, Moriarty is the Queen.

Show $Ps, \neg Pg \vdash Q$

$$\dfrac{\dfrac{\neg Pg \qquad \dfrac{\dfrac{Ps}{\forall x Px}\,\forall I}{Pg}\,\forall E}{\bot}\,MP}{Q}\,\bot$$

From one instance (Sherlock's Personhood) we cannot assume all objects are people.

Example: When Things Go Wrong

Let $\alpha \equiv (x = y)$ and hence $\alpha[x/y] \equiv (y = y)$.

$$\dfrac{\begin{array}{c}\vdots \\ \mathcal{D} \\ y = y\end{array}}{\forall x\ (x = y)}\ \forall I$$

This claims that if something is equal to itself, then everything is equal to that one thing. Obviously not something one would want in their logic!

Note: This is an example of an error resulting from not replacing all ys with xs in the $\forall$ introduction step i.e. not making sure y does not appear free after the substitution.

Existential Introduction

If $\overset{\Sigma}{\underset{\alpha[x/y]}{\mathcal{D}}}$ is a derivation of $\alpha[x/y]$ from hypotheses Σ, where y is free for x in α, then

$$\dfrac{\begin{array}{c}\Sigma \\ \mathcal{D} \\ \alpha[x/y]\end{array}}{\exists x\alpha}\ \exists I$$

is a derivation of $\exists x\alpha$ from hypotheses Σ.

Example(!!!)

$\forall x\ Px \vdash \exists x\ Px$

Example

$\forall x(Px \to Qx), \neg Qa \vdash \exists x\, \neg Px$

Existential Elimination

Under the BHK interpretation a proof of $\exists x\ P(x)$ has the form:

$$(t : T, p : P(t)) : (\exists x : T, p(x))$$

If we have as a hypothesis $\exists x\ P(x)$, then how are we to make use of this in a proof?

Existential Elimination

If $\overset{\Sigma_1}{\exists x \alpha} \mathcal{D}_1$ and $\overset{\Sigma_2}{\alpha[x/y] \to \gamma} \mathcal{D}_2$ are derivations, and y is neither free in γ nor $\Sigma_1 \cup \Sigma_2$, then

$$\dfrac{\begin{array}{cc} \Sigma_1 & \Sigma_2 \\ \mathcal{D}_1 & \mathcal{D}_2 \\ \exists x \alpha & \alpha[x/y] \to \gamma \end{array}}{\gamma} \exists E$$

is a derivation of γ from hypotheses $\Sigma_1 \cup \Sigma_2$.

Example

$\forall x(Px \to Qx),\ \exists xPx \ \vdash\ \exists xQx$

Example: When Things Go Wrong I

Sherlock is a detective. Therefore, everyone is a detective.

$$\exists x Dx \vdash \forall x Dx$$

$$\dfrac{\exists x Dx \quad \dfrac{\dfrac{\overline{Da}^{\,1}}{Da \to Da}\to I,1}{Da}\exists E}{\forall x Dx}\forall I$$

Example: When Things Go Wrong II

Sherlock is a human. Gladstone is a dog. Therefore, there exists something that is both a human and a dog.

$$Hs,\ \exists x Dx\ \vdash\ \exists x(Hx \wedge Dx)$$

$$\dfrac{\exists x Dx \qquad \dfrac{\dfrac{\dfrac{Hs \qquad \overline{Ds}^{\,1}}{Hs \wedge Ds}\wedge I}{\exists x(Hx \wedge Dx)}\exists I}{Ds \to \exists x(Hx \wedge Dx)}\to I, 1}{\exists x(Hx \wedge Dx)}\exists E$$

Proof Strategy

$$\exists x Px \vee \exists x Qx \vdash \exists x(Px \vee Qx)$$

Note: Our hypothesis is a disjunction. So we will need the disjunction elimination rule to make use of it.

This breaks the proof into two steps:

1. $\exists x Px \rightarrow \exists x(Px \vee Qx)$
2. $\exists x Qx \rightarrow \exists x(Px \vee Qx)$

Both of these steps will have the same form, so let's focus on the first.

Proof Strategy

We have reduced the problem to showing:

$$\vdash \exists x Px \rightarrow \exists x(Px \vee Qx)$$

Proof Strategy

We have reduced the problem to showing:

$$\vdash \exists x Px \rightarrow \exists x(Px \vee Qx)$$

The deduction theorem ($\rightarrow I$) implies this is equivalent to showing:

$$\exists x Px \vdash \exists x(Px \vee Qx)$$

Proof Strategy

We have reduced the problem to showing:

$$\vdash \ \exists x Px \rightarrow \exists x(Px \vee Qx)$$

The deduction theorem ($\rightarrow I$) implies this is equivalent to showing:

$$\exists x Px \ \vdash \ \exists x(Px \vee Qx)$$

Now our hypothesis has an existential quantifier in it, so we need to use the corresponding elimination rule. Our proof will have a line of the form:

$$\dfrac{\exists x Px \qquad \overline{Pa \rightarrow \exists x(Px \vee Qx)}\ ?}{\exists x(Px \vee Qx)}\ \exists E$$

How can we complete the proof?

Proof Strategy

We want to show $\vdash \exists x Px \to \exists x(Px \vee Qx)$ and so far our proof looks like this. What can we do to finish this proof?

$$\dfrac{\dfrac{\overline{\exists x Px}^{\,2} \qquad \dfrac{\dfrac{\dfrac{\overline{Pa}^{\,1}}{Pa \vee Qa}\vee I}{\exists x(Px \vee Qx)}\exists I}{Pa \to \exists x(Px \vee Qx)}\to I,1}{\exists x(Px \vee Qx)}\exists E$$

Proof Strategy

We have proven the first of our subgoals. Go through the same process to prove the second subgoal.

1. $\exists x Px \rightarrow \exists x(Px \vee Qx)$
2. $\exists x Qx \rightarrow \exists x(Px \vee Qx)$

Use the disjunction elimination to tie these goals together into a proof of the original statement

$$\exists x Px \vee \exists x Qx \vdash \exists x(Px \vee Qx)$$

Example

$\neg\exists x\ Px \vdash \forall x\ (\neg Px)$

Further Reading

Dirk van Dalen, Logic and Structure.

Simon Thompson, Type Theory and Functional Programming.

Chapters 1 - 4.

Jeremy Avigad et al, Logic and Proof[3].

Chapters 7, 8, and 9.

[3] Note: this uses an outdated version of Lean, but is still a good resource for the theory of this course.