1 Zero-order logic

1.1 Terms and predicates

1.1.1 Predicates

Zero-order logic adds predicates. Like propositional variables, these have truth values. Unlike propositional variable, predicates take terms as inputs.

For example using propositional logic we can write the statement "you are 25" as θ.

With preterites we can write this as $P(you, 25)$.

A propositional variable can be considered a special case of a predicate variable, where the number of inputs is 0.

1.2 Relations and equality

1.2.1 Relations

A special type of predicates is a relation. These take two terms and can be written differently: $P(x, y) \Leftrightarrow x \oplus y$

1.2.2 Equality

In preterite logic we define the relation for equality.

$a = b$

It is defined by the following:

- Reflexivity : $x = x$
- Symmetry: $x = y \leftrightarrow y = x$
- Transivity: $x = y \land y = z \to x = z$
- Substitution for functions: $x = y \to f(x) = f(y)$
- Substitution for formulae: $x = y \land P(x) \to P(y)$

1.3 Functions and brackets

1.3.1 Functions (or maps)

Functions take other terms, and are themselves terms. For example if we wanted to know if someone can legally drive in a specific country, we could use:

$P(you, age(UK))$

A function may not be able to produce an output for all inputs. For examples $age(green)$ has no interpretation.

Functions can also take different numbers of inputs. Constants, such as “you” and “UK” can be shown as functions with 0 inputs. As a result we could instead write:

$P(you(), age(UK()))$

We generally denote functions with a lower case letter, so would instead write:

$P(y(), a(b()))$

Functions are also called maps.

1.4 Signatures

1.4.1 Structures

A logical structure consists of:

- Domain
- Interpretation
- Signature

1.4.2 Domain

The domain is the set of variables in the structure.

We include an infinite number of variables.

1.4.3 Interpretation

The interpretation assigns values to propositional and predicate variables.

1.4.4 Signature

A logical signature describes the language of the logic which is used to construct statements. This includes:

- Functions
- Relations
- Operators

The language of a signature is all possible sentences, or formulae which can be constructed from this signature.

We include an infinite number of functions, relations and all operators.

1.5 Completeness of zero-order logic

A theory is complete if all true formulae are included.

Note that there are three types of formulae in a theory.

- Tautologies (always true)
- Refutable formulae (always false)
- Satisfiable formulae which are not tautologies (true in some, but not all, interpretations).

1.6 Injective, bijective and surjective functions

1.6.1 Injective functions

$f(a) = f(b) \rightarrow a = b$

1.6.2 Surjective functions

All points in codomain have at least one matching point in domain

Mapping info, details

1.6.3 Bijective

Both injective and surjective

1.6.4 Other

___Identity function___

The identity function maps a term to itself.

___Idempotent___

An idempotent function is a function which does not change the term if the function is used more than once. An example is multiplying by 0.

1.6.5 Inverse functions

An inverse function of a function is one which maps back onto the original value.

$g(x)$ is an inverse function of $f(x)$ if

$g(f(x)) = x$

1.6.6 Properties of binary functions

Binary functions can be written as:

$f(a, b) = a \oplus b$

A function is commutative if:

$x \oplus y = y \oplus x$

A function is associative if:

$(x \oplus y) \oplus z = x \oplus (y \oplus z)$

A function $\otimes$ is left distributive over $\oplus$ if:

$x \otimes (y \oplus z) = (x \otimes y) \oplus (x \otimes z)$

Alternatively, function $\otimes$ is right distributive over $\oplus$ if:

$(x \oplus y) \otimes z = (x \otimes z) \oplus (y \oplus z)$

A function is distributive over another function if it both left and right distributive over it.

1.7 Binary functions

1.7.1 Properties of binary functions

Binary functions can be written as:

$f(a, b) = a \oplus b$

A function is commutative if:

$x \oplus y = y \oplus x$

A function is associative if:

$(x \oplus y) \oplus z = x \oplus (y \oplus z)$

A function $\otimes$ is left distributive over $\oplus$ if:

$x \otimes (y \oplus z) = (x \otimes y) \oplus (x \otimes z)$

Alternatively, function $\otimes$ is right distributive over $\oplus$ if: $(x \oplus y) \otimes z = (x \otimes z) \oplus (y \oplus z)$

A function is distributive over another function if it both left and right distributive over it.

2 First-order logic

2.1 Writing first-order logic

2.1.1 Existential quantifier

We introduce a shorthand for “at least one term satisfies a predicate”, that is:

$P(x_0) \lor P(x_1) \lor P(x_2) \lor P(x_2) \lor P(x_3)\ldots$

The short hand is:

$\exists x P(x)$

2.1.2 niversal quantifier

We introduce another shorthand, this time for:

$P(x_0) \land P(x_1) \land P(x_2) \land P(x_2) \land P(x_3)\ldots$

The shorthand is

$\forall x P(x)$

2.1.3 Free and bound variables

A bound variable is one which is quantified in the formula. A free variable is one which is not. Consider:

$\forall x P(x, y)$

In this, x is bound while y is free.

Free variables can be interpreted differently, while bound variables cannot.

We can also bind a specific variable to a value. For example 0 can be defined to be bound.

2.1.4 Ground terms

A ground term does not contain any free variables. A ground formula is one which only includes ground terms.

$\forall x\ x$ is a ground term.

$\forall x P(x)$ is a ground formula.

2.2 Inference rules for first-order logic

2.2.1 Existential instantiation

If P is true for a specific input, then there exists an input for P where P is true.

$P(r) \Rightarrow \exists x P(x)$

2.2.2 Existential generalisation

$\exists x P(x) \Rightarrow P(r)$

Where r is a new symbol.

2.2.3 Universal instantiation

If P is true for all values of x, then P is true for any input to P.

$\forall x P(x) \Rightarrow P(a/x)$

Where a/x represents substituting a for x within P.

2.2.4 Universal generalisation

If there is a derivation for $P(x)$, then there is a derivation for $\forall x P(x)$.

$\vdash P(x) \Rightarrow \vdash \forall x P(x)$

2.3 Duality of first-order logic

The dual of:

$\exists x$ň$P(x)$

Is:

ň$\forall x P(x)$

2.4 Gödel's completeness theorem

2.4.1 Completeness of first-order logic

We previously showed that zero-order logic was complete. What about first-order logic?

Gödel's' completeness theorem says that for first order logic, a theory can include all tautologies, the first category.

If the completeness theorem is true and a formula is not in the theory, then the formula is either refutable or satisfiable under some, but not all interpretations.

That is, either the theory will contain θ, ňθ, or θ will be satisfiable in some but not all interpretations, and neither will be in the in theory.

To prove this we look for a proof that every formula is either refutable or true under some structure. So for an arbitrary formula θ we want to show it is either refutable or satisfiable under some interpretation.

2.4.2 Part 1: Converting the form of the formula

Remove free variables, functions

Note that if this is true, all valid formulae of the form below are provable:

ňθ

This means that there is no interpretation where the following is true:

θ

Conversely if ňθ is not in the theory, then θ must be true under some interpretation.

That is, if all valid formulae are provable, then all

Reformulating the question:

This is the most basic form of the completeness theorem. We immediately restate it in a form more convenient for our purposes:

Theorem 2. Every formula θ is either refutable or satisfiable in some structure.

"$\theta$is refutable" means by definition "ňθ is provable".

2.4.3 Decidability

Given a formula, can we find out if can be derived from the axioms? We can follow a process for doing so which would inform us if the formula was or was not a theorem. Alternative, the process could carry on forever.

If the process never carries on forever the system is decidable: there is a finite process to determine whether the formula is in or out. If the process halts for true formulas, but can carry on forever for false formulas, the system is semi-decidable. If the process takes a long time, we do not know if it is looping infinitely, or approaching its halt point.

Intuitively, use of axioms can make an existing formula shorter or longer, so finding all short formulas can require going forwards and backwards an infinite number of times.