Part II

Logic

Chapter 5

Logic

5.1 Why Study Logic?

We have seen in the previous part of this script how formal language theory studies the *syntax* of languages. Yet, this formalism is not concerned with how sentences – i.e. elements of a language – are *related* to each other. In other words, formal language theory does not answer the question of how a sentence can be *derived* from already existing sentences.

This is where *logic* comes into play. As a formal science, logic investigates how valid statements can be formally derived or inferred from other statements, or from a set of axioms. In short, logic enables us to formalize the notion of a *proof*.

Traditionally studied as a branch of philosophy, logic has become since the nineteenth century the foundation of mathematics and computer science. You may have heard of some of the key figures of mathematical logic: G. Boole, B. Bolzano, G. Frege, D. Hilbert, A. Church, G. Peano, B. Russel, A. Tarski or A. Turing.

Interestingly, the study of logic not only provides a formalism about abstract objects, but also opens the way to concrete applications. By defining how theorems – i.e. valid formulas – can be derived in a purely syntactic way, computers can then be used to derive statements about the "real" world. For instance, predicate logic is often used in expert systems to represent the knowledge and to derive new facts from the knowledge base. Or, in linguistics, declarative programming languages are often used, the most prominent one being Prolog.

5.1.1 An Experiment

Imagine that you have been asked to perform two tasks[1].

In the first, you are presented with a stack of cards. One side of each card has either the letter A or the letter D. The opposite side of each card has either the number 4 or the number 7. The cards are now stacked with either side up, at random, and shuffled, so that thumbing through the deck you would see some A's, some D's, some 4s and some 7s. Your task is to determine whether or not the cards of this deck satisfy the rule "If the letter side of a card is an A, then the number side must be a 4." To make that determination, you are to imagine that you are going through the deck, looking at the turned-up side of each card, one at a time, and turning over whichever cards you must, but only those cards, in order to verify or contradict the rule that every A must be accompanied by a 4.

Think about the task for a moment. Would you turn over only those cards with A's showing? Or those showing A's and 4s? Or showing A's and 7s? Or perhaps those showing A's, 4s and 7s? Or did you choose some other combination? Write down your choice and proceed to task two. And don't feel discouraged. Task one is difficult, and most of the English college students upon whom the experiment was first performed failed to give the correct answer.

As task two, you are to imagine that your are the cashier at a supermarket and have the checks received that day stacked before you; some face up and some face down. Your supermarket has a rule. The checkout people are to accept checks for more the $50 only if approved on the back by the manager. Imaginge that you are to go through the checks, one at a time, and turn over only those checks necessary to establish if the approval rule has been followed.

Again, think about the task for a moment. Would you turn over only checks bigger than $50? Or those, pluse checks with their face down bearing the manager's approval? Or those for over $50 and those with no approval on the back? Or perhaps checks exceeding $50, plus all checks with their faces down? Or some other combination?

As before, jot down your answer. If you are typical of most subjects of this experiment, you did not find task two nearly as difficult as task one. You probably correctly answered two by turning over checks for more than $50 and those with no approval on their backs. You were more likely to miss on task one, for which the correct solution is to turn over A's and 7s only.

Why this pair of experiments? Because the two tasks are essentially identical.

[1] This experiment is taken from (Dreyfus and Dreyfus, 1986, p.18).

If you designate "over $50" as A, "not over $50" as D, "approved on back" as 4 and "unapproved" as 7, task two becomes task one. But while they are abstractly identical, the statement of task two draws, for many, on "knowing how," whereas task one is perceived as a logical puzzle requiring the application of logical rules, that is , requiring the reduction to "knowing that." All of you who did task two easily and correctly and had trouble with task one have learned from this experience that "knowing how" is quite distinct from "knowing that" and in no way requires using conscious abstract rules.

5.2 Definition of a Formal System

Definition 5.1. In logic, a *formal system* consists of:

(a) a formal language,

(b) a set of axioms,

(c) a set of inference (or transformation) rules.

The choice of these defines the power of the system (which assertions can be expressed) and its properties (completeness, decidability, etc.).

5.3 Propositional Logic

One of the most basic types of logic (and one of the simplest formal systems) is *propositional logic* or *propositional calculus*. Propositional calculus examines the syntax and semantics of expressions which are formed by connecting atomic formulas (i.e. variables that are either true or false) by logical connectives. Propositional calculus can only make atomic statements. Thus, a sentence from natural language such as "Socrates is human" becomes `socrates_is_human`.[2] This statement can then have either the value of true or false.

5.3.1 Language

The language of propositional logic consists of:

- a set P of atomic formulas, consisting of e.g. symbols such as $p, q, r...$

[2]The statement of this archetypal example usually reads `human_socrates`.

- the following logical connectives:

  $\neg$ negation

  $\wedge$ logical "and"

  $\vee$ logical "or"

  $\rightarrow$ logical implication ("if ... then")

  $\leftrightarrow$ equivalence ("if and only if")

- auxiliary symbols: "(" and ")"

Definition 5.2. *Propositional formulas* can then be constructed from the symbols of the language by a recursive definition:

(i) Every atomic formula $p \in P$ is a propositonal formula.

(ii) If the expressions A and B are propositional formulas, then the expressions $\neg A, (A \wedge B), (A \vee B), (A \rightarrow B), (A \leftrightarrow B)$ are propositional formulas.

(iii) Every propositional formula arises from a finite number of application of (i) and (ii).

Example 5.1. If $P = \{p, q, r\}$ is a set of atomic formulas, then the expressions

- p, q and r are propositional formulas according to (i)
- $\neg p$ and $(q \wedge r)$ are propositional formulas according to (ii)
- $(\neg p \rightarrow (q \wedge r))$ is a propositional formula according to another application of (ii)

Alternatively, propositional (or well-formed) formulas can also be generated by means of a grammar:

$$
\begin{array}{rcl}
\langle \text{formula} \rangle & \rightarrow & \langle \text{atomic formula} \rangle \,|\langle \text{ propositional formula} \rangle \\
\langle \text{atomic formula} \rangle & \rightarrow & T \mid F \mid p \mid q \mid r \mid \ldots \\
\langle \text{propositional formula} \rangle & \rightarrow & (\ \langle \text{formula} \rangle\ ) \\
 & \mid & \langle \text{formula} \rangle \langle \text{connector} \rangle \langle \text{formula} \rangle \\
 & \mid & \neg \langle \text{formula} \rangle \\
\langle \text{connector} \rangle & \rightarrow & \wedge \mid \vee \mid \rightarrow \mid \leftrightarrow
\end{array}
$$

where T and F stand for the logical "true" and "false" values.

This grammar still contains some ambiguity as long as the priority of connectors is not defined. Priority is defined as follows:

1. The negation $\neg$ has the highest priority,
2. followed by the logical "and" ($\wedge$),
3. the logical "or" ($\vee$),
4. and the logical implications ($\rightarrow$, $\leftrightarrow$).

Alternatively, parenthesis can be used, which also makes a logical expression more readable. For instance, $A \wedge \neg B \vee C \rightarrow D$ can also be written as $((A \wedge (\neg B)) \vee C) \rightarrow D$.

Now we are able to construct propositional (or well-formed) formulas.

5.3.2 Semantics

The semantics of propositional logic is concerned with the truth or falsity of propositional formulas. The atomic formulas cannot be analysed further, their truth value is given, by means of a truth function. This process, assigning a value of true or false to every atomic formula, is called interpretation. The truth values of the atomic formulas can then be unambigously extended to all propositional formulas by defining the semantics of logical connectives. This can be done by means of a truth table:

P	Q	$\neg$P	$P \vee Q$	$P \wedge Q$	$P \rightarrow Q$	$P \leftrightarrow Q$
T	T	F	T	T	T	T
T	F	F	T	F	F	F
F	T	T	T	F	T	F
F	F	T	F	F	T	T

Note that the logical implication $P \rightarrow Q$ is simply defined by the above truth table. It does not contain any notion of causality relating events in the real world.

Definition 5.3. A propositional formula A is *satisfiable*, if there exists an interpretation of its atomic formulas (assigning truth values to all of them) such that A is true.

Definition 5.4. Let M be a set of formulas. If a fomula A is true in every interpretation in which all formulas of M are true, then A is a *tautologic consequence* of M and we write $M \vDash A$.

Definition 5.5. If A is true in all interpretations, it is called a *tautology*. This is written as $\vDash A$. In other words, $\vDash A$ implies that $M \vDash A$ is valid for any M. An example of a tautology is $P \vee \neg P$.

Definition 5.6. *Contradiction*: If a statement is always false for all interpretations, it is called a contradiction. An example of contradiction is $P \wedge \neg P$.

5.3.3 Formal System

We have already defined the language and propositional formulas. To complete the formal system of propositional logic we need a set of axioms and inference rules.

Why would we need a formal system? We are already able to construct well-formed formulas and decide on their truthfulness by means of a truth table. However, imagine we had a set of formulas M and we know that they are true – they represent our knowledge about a certain problem. We would then be interested in other formulas valid in this situation, i.e. in some A, such that $M \vDash A$. How would you find them? By means of a truth table, we would have to list all interpretations for which M is true and then randomly generate various formulas and check whether they are true in those interpretations. In complex situations, this would be a tedious job!

On the other hand, a formal system would allow to generate valid formulas in an automated and more effective manner. You can think of the formal system as syntax, as a complement of semantics.

Axioms

An important requirement we have on any formal system is that only valid (i.e. logically true) formulas can be derived. Such a system is then said to be *sound*. The basis of the formal system are the axioms. A logical choice thus is to choose axioms from tautologies – formulas valid in every interpretation.

A particularly compact and well-known axiom system for propositional logic is the following (after Jan Lukasiewicz):

$$
\begin{array}{rcll}
p & \rightarrow & (q \rightarrow p) & \text{(A1)} \\
(p \rightarrow (q \rightarrow r)) & \rightarrow & ((p \rightarrow q) \rightarrow (p \rightarrow r)) & \text{(A2)} \\
(\neg p \rightarrow \neg q) & \rightarrow & (q \rightarrow p) & \text{(A3)}
\end{array}
$$

Note that other axiom systems are also possible.

Inference rules

What requirements do we have on the rules of inference? They should be *correct*, i.e. from a formula that is valid in an interpretation, they can derive only a formula that is valid in the same interpretation. Propositional logic has a single inference rule: Modus ponens.

Modus Ponens

$$
\begin{array}{l}
p \rightarrow q \\
\underline{p} \\
q
\end{array}
$$

Example 5.2. Let us replace p with `bad_weather` and q with `I_stay_home`. If we assume the axioms `bad_weather` $\rightarrow$ `I_stay_home` as well as `bad_weather`, we can derive, using modus ponens, the formula `I_stay_home`:

$$
\begin{array}{l}
\texttt{bad\_weather} \rightarrow \texttt{I\_stay\_home} \\
\underline{\texttt{bad\_weather}} \\
\texttt{I\_stay\_home}
\end{array}
$$

All theorems of propositional logic can be derived from the three axioms and the single inference rule.

Proofs

Definition 5.7.

(i) A finite sequence of formulas $A_1, A_2, \ldots, A_n$ is the *proof* of a formula A, if A_n is the formula A and for any i, formula A_i is either an axiom or it is derived from previous formulas A_j $(j < i)$ by modus ponens.

(ii) If there exists a proof of formula A, we say that A is provable in propositional logic and we write $\vdash A$.

Example 5.3. To get an idea, how the formal system works, let us look at the formal proof of a seemingly trivial formula $A \to A$.

$\vdash A \to ((A \to A) \to A)$	a case of (A1) axiom
$\vdash (A \to ((A \to A) \to A)) \to$ $[(A \to (A \to A)) \to (A \to A)]$	a case of (A2) axiom
$\vdash (A \to (A \to A)) \to (A \to A)$	modus ponens
$\vdash (A \to (A \to A))$	a case of (A1) axiom
$\vdash A \to A$	modus ponens

Although this seems that it is uselessly complicated and you could get this trivially just by looking at it or from a truth table, there are also cases when the reverse is true.

Definition 5.8. *Proof from assumptions.* Let T be a set of formulas. We say that formula A is provable from T and write $T \vdash A$ if A can be proven from axioms and formulas in T, by using the inference rule. That is we have basically enriched the set of axioms by the formulas in T.

Theorems

The following are some theorems of propositional logic. They can be derived from axioms or you can check that they are tautologies by examining their truth tables.

de Morgan's Rules

1. $(\neg(P \vee Q)) \leftrightarrow (\neg P) \wedge (\neg Q)$

2. $(\neg(P \wedge Q)) \leftrightarrow (\neg P) \vee (\neg Q)$

Distributive Laws

1. $(P \vee (Q \wedge R)) \leftrightarrow ((P \vee Q) \wedge (P \vee R))$

2. $(P \wedge (Q \vee R)) \leftrightarrow ((P \wedge Q) \vee (P \wedge R))$

5.3.4 Completeness

Theorem 5.1 (Post)**.** *For every propositional formula A*

$$\vdash A \quad \textit{if and only if} \quad \models A$$

This means that, in propositional logic, tautologies are provable, and what is provable is a tautology. Thus, the formal system of propositional logic is not only sound (i.e. generates only valid formulas) but also generates all of them.

Theorem 5.2 (completeness of propositional logic)**.** *Let T be a set of formulas and A a formula. Then*

$$T \vdash A \quad \textit{if and only if} \quad T \models A$$

This is a more general version of Post's theorem. In a sense, completeness implies that loosely speaking syntax and semantics are equivalent in this case. This is by no means true for any formalism (see below).

5.3.5 Normal Forms of Propositional Formulas

Every propositional formula can be expressed in two standard or *normal* forms.

Definition 5.9 (Conjunctive normal form – CNF)**.** A formula in CNF has the following form:

$$(A_1 \vee \ldots \vee A_N) \wedge (B_1 \vee \ldots \vee B_M) \wedge \ldots$$

where the A_i can be either atomic formulas or their negations. It is thus a conjuction of clauses, i.e. disjunctions of literals (variables or their negations).

CNF is used in machine proving of theorems. Resolution in Prolog is also based on a special form of CNF.

Definition 5.10 (Disjunctive normal form – DNF)**.** A formula in DNF has the following form:

$$(A_1 \wedge \ldots \wedge A_N) \vee (B_1 \wedge \ldots \wedge B_M) \vee \ldots$$

where A_1 etc. can be either atomic formulas or their negations. It is thus a disjuction of cojunctions of literals.

Applications of DNF include databases.

Example 5.4. This example shows how the following formula can be transformed into conjunctive normal form.

$$A \rightarrow \neg(B \rightarrow C)$$

1st step: Elimination of $\rightarrow$. Using the rule $P \rightarrow Q = \neg P \vee Q$:

$$(A \rightarrow \neg(B \rightarrow C)) = (\neg A \vee \neg(\neg B \vee C))$$

2nd step: Distribution of $\neg$ onto atomic expressions.

$$\begin{array}{lll} & (\neg A \vee \neg(\neg B \vee C)) & \\ = & (\neg A \vee (\neg\neg B \wedge \neg C)) & \text{using rule } \neg(P \vee Q) = \neg P \wedge \neg Q \\ = & (\neg A \vee (\neg\neg B \wedge \neg C)) & \text{using rule } \neg\neg P = P \\ = & (\neg A \vee (B \wedge \neg C)) & \end{array}$$

3rd step: Transforming into a conjuction of disjunctions by the distributive rule. Using rule $P \vee (Q \wedge R) = (P \vee Q) \wedge (P \vee R)$:

$$\begin{array}{ll} & (\neg A \vee (B \wedge \neg C)) \\ = & ((\neg A \vee B) \wedge (\neg A \vee \neg C)) \end{array}$$

5.3.6 A "Logical" Anecdote

This example shows the power of propositional calculus by illustrating how it allows us to easily solve a problem which doesn't offer an intuitive solution. The problem reads as follows:

1. If the weather is bad, I stay at home.
2. I die if and only if my house explodes because of a gas leak and I'm at home.
3. If my house explodes or I go out, and only then, my neighbors notice something strange.
4. Whenever my neighbors notice something strange, they call home the next day (they only do so if I'm alive). They never call home otherwise.

My wife (who works at home) tells you that she received a call from the neighbors last Tuesday. Can you say something about the weather of last Monday?

Simplification

Let's rewrite the above statements as logical expressions by setting:

- a: `bad_weather`
- b: `stay_home`
- c: `I_die`
- d: `gas_leak_explosion`
- e: `neighbors_notice_something_strange`
- f: `neighbors_call_home`

This gives:

1. $a \rightarrow b$
2. $c \leftrightarrow d \wedge b$
3. $d \vee \neg b \leftrightarrow e$
4. $e \wedge \neg c \leftrightarrow f$

From the last statement of the problem, we can derive:

$$\begin{aligned} f &\leftrightarrow e \wedge \neg c \\ &\leftrightarrow (d \vee \neg b) \wedge \neg(d \wedge b) \\ &\leftrightarrow (d \vee \neg b) \wedge (\neg d \vee \neg b) \\ &\leftrightarrow (d \wedge \neg d) \vee \neg b \\ &\leftrightarrow \neg b \\ &\rightarrow \neg a \end{aligned}$$

Therefore: you know that on Monday, the weather was good! □

5.4 Predicate Calculus (First Order Logic)

Propositional calculus has a limited expressive power. It works with atomic statements that are either true or false and studies the properties of logical connectives that connect the atomic formulas.

Predicate calculus introduces variables that do not have a value of true or false, but can take up any value, depending on the model, such as an element of a set (e.g. of the set of all inhabitants of Prague), or a natural number. There are also functions operating on the variables and predicates that have a truth value. By means of predicates, formulas can be constructed. In addition, the language also contains *quantifiers*. Such a language is called *first order language*.

5.4.1 Language

Definition 5.11. *First order language* contains:

(i) An unlimited number of symbols for variables: $x, y, z, \ldots$

(ii) Symbols for logical connectives: $\neg, \wedge, \vee, \rightarrow, \leftrightarrow$.

(iii) Symbols for quantifiers: the *universal* quantifier $\forall$ ("for all"), and the *existential* quantifier $\exists$ ("there exists").

(iv) Symbols for predicates: $p, q, ...$

Predicates are relations. The *arity* of a predicate symbol specifies the number of arguments of the predicate. For example, equality "=" is a binary predicate (its arity is 2).

(v) Symbols for functions: $f, g, \ldots$

The *arity* of a function symbol specifies the number of arguments of the function. Function symbols of arity 0 are constants.

(vi) Auxiliary symbols: "(", ")"

This language is called first order because one can only quantify over variables (for individuals), such as in $(\forall P)\texttt{inhabitant\_of\_Prague}(P) \rightarrow \texttt{mortal}(P)$. However, one cannot quantify over predicates, i.e. one cannot say "$\forall$ inhabitants of Prague".

Symbols for variables, connectives, quantifiers and equality (if present) are called *logical symbols* because they are present in every first order language. On

the other hand, symbols for predicates and functions are called special symbols and the choice of these symbols depends on what we want to study, i.e. on the particular laguage.

Examples of languages

a) Language of predicate logic. This simply is a first order language without any special symbols.

b) Language of group theory. This a first order language with equality with two special symbols: e, a constant for the unit element, and a binary function symbol '$\cdot$' for the group operation.

c) Language of set theory is a language with equality and a single special symbol $\in$ as a binary predicate symbol for belonging to a set.

d) Language of elementary number theory[3] (with equality) contains function symbols 0 (constant for zero), S (unary symbol for the consecutive natural number, e.g.), $+$ and $\times$ (binary symbols for addition and multiplication).

Example 5.5 (Colonel West)**.** The law says that it is a crime for an American to sell weapons to hostile nations. The country Nono, an enemy of America, has some missiles, and all of its missiles were sold to it by Colonel West, who is American.

What we wish to prove is that West is a criminal. We can represent these facts in first-order logic, and then show the proof as a sequence of applications of the inference rules.

Yet, expecting the proof to be found by a computer requires a pretty smart program. The reasons not only include the problem of formalizing common-sense knowledge (e.g. that a missile is a weapon, that an enemy of America counts as a "hostile", etc.), but also a large branching factor, and hence a potentially explosive search problem. Thus, even a simple problem for a human to solve can lead to serious difficulty when needed to be formally proven.

[3]Number theory is the branch of pure mathematics concerned with the properties of numbers in general, and integers in particular, as well as the wider classes of problems that arise from their study. Some also refers to it as "arithmetic".

Knowledge base (or axis of a special theory – the world of Colone West)

"it is a crime for an American to sell weapons to hostile nations"

$$(\forall X)\ (\forall Y)\ (\forall Z) \quad (\ \text{american}(X) \wedge \text{weapon}(Y) \wedge \text{nation}(Z) \wedge \text{hostile}(Z) \wedge \text{sells}(X, Z, Y) \rightarrow \text{criminal}(X)\ ) \tag{5.1}$$

"the country Nono"

$$\text{nation}(\text{Nono}) \tag{5.2}$$

"Nono, an enemy of America"

$$\text{enemy}(\text{Nono}, \text{America}) \tag{5.3}$$

"Nono has some missiles"

$$(\exists X)(\text{own}(\text{Nono}, X) \wedge \text{missile}(X)) \tag{5.4}$$

"All its missiles were sold to it by Colonel West"

$$(\forall X)(\text{own}(\text{Nono}, X) \wedge \text{missile}(X) \rightarrow \text{sells}(\text{West}, \text{Nono}, X)) \tag{5.5}$$

"West, who is American"

$$\text{american}(\text{West}) \tag{5.6}$$

Frame problem (i.e. common-sense knowledge):

$$\text{nation}(\text{America}) \tag{5.7}$$

$$(\forall X) \quad (\text{missile}(X) \rightarrow \text{weapon}(X)) \tag{5.8}$$

$$(\forall X) \quad (\text{enemy}(X, \text{America}) \rightarrow \text{hostile}(X)) \tag{5.9}$$

Goal (or theorem to be proved)

$$\text{criminal}(\text{West})$$

Proof

Using 5.4 and existential elimination[4]:

$$\text{own}(\text{Nono}, \text{M1}) \wedge \text{missile}(\text{M1}) \tag{5.10}$$

Using 5.10 and "and" elimination[5]:

$$\text{own}(\text{Nono}, \text{M1}) \tag{5.11}$$

and

$$\text{missile}(\text{M1}) \tag{5.12}$$

Using 5.8 and universal elimination[6]:

$$\text{missile}(\text{M1}) \rightarrow \text{weapon}(\text{M1}) \tag{5.13}$$

Using 5.12 and 5.13 and modus ponens:

$$\text{weapon}(\text{M1}) \tag{5.14}$$

Using 5.5 and universal elimination:

$$\text{own}(\text{Nono}, \text{M1}) \wedge \text{missile}(\text{M1}) \rightarrow \text{sells}(\text{West}, \text{Nono}, \text{M1}) \tag{5.15}$$

Using 5.10 and 5.15 and modus ponens:

$$\text{sells}(\text{West}, \text{Nono}, \text{M1}) \tag{5.16}$$

Using 5.1 and universal elimination (3 times):

$$\begin{aligned} &\text{american}(\text{West}) \wedge \text{weapon}(\text{M1}) \wedge \text{nation}(\text{Nono}) \wedge \text{hostile}(\text{Nono}) \\ &\wedge \text{sells}(\text{West}, \text{Nono}, \text{M1}) \rightarrow \text{criminal}(\text{West}) \end{aligned} \tag{5.17}$$

[4]The existential elimination is a rule where the $\exists$ quantifier can be instantiated with a particular variable that does not appear elsewhere, for instance:

$$(\exists X)(\text{likes}(X, \text{IceCream})) \Rightarrow \text{likes}(\text{Person1}, \text{IceCream})$$

[5]The "and" elimination is another rule where from $X \wedge Y$ follows X (and Y).

[6]The universal elimination is a rule where the $\forall$ quantifier can be instantiated with any variable:

$$(\forall X)(\text{human}(X) \rightarrow \text{mortal}(X)) \Rightarrow \text{human}(\text{Socrates}) \rightarrow \text{mortal}(\text{Socrates})$$

Using 5.9 and universal elimination:

$$\text{enemy}(\text{Nono}, \text{America}) \rightarrow \text{hostile}(\text{Nono}) \tag{5.18}$$

Using 5.3 and 5.18 and modus ponens:

$$\text{hostile}(\text{Nono}) \tag{5.19}$$

Using 5.6, 5.2, 5.14, 5.16, 5.19 and "and" introduction[7]:

$$\begin{aligned} &\text{american}(\text{West}) \wedge \text{weapon}(\text{M1}) \wedge \text{nation}(\text{Nono}) \\ &\wedge \text{hostile}(\text{Nono}) \wedge \text{sells}(\text{West}, \text{Nono}, \text{M1}) \end{aligned} \tag{5.20}$$

Using 5.17 and 5.20 and modus ponens:

$$\text{criminal}(\text{West}) \tag{5.21}$$

□

Definition 5.12 (Expression)**.** An *expression* is any sequence of symbols of a particular language.

Definition 5.13 (Term)**.** An *term* is an expression defined recursively as follows:

(i) Every variable is a term.

(ii) If the expressions $t_1, \ldots, t_n$ are terms and f is an n-ary function symbol, then $f(t_1, \ldots, t_n)$ is a term.

(iii) Every term arises from a finite number of applications of (i) and (ii).

Example 5.6. In number theory $x, x + y$, are terms. The latter term could also be written as $+(x, y)$, where + is our f (but it is conventional to write these binary predicates in infix notation).

Definition 5.14 (Formula)**.** A *formula* is defined recursively as follows:

(i) If p is an n-ary predicate symbol and the expressions $t_1, \ldots, t_n$ are terms, then the expression $p(t_1, \ldots, t_n)$ is an atomic formula.

(ii) If the expressions A and B are formulas, then the expressions $\neg A, (A \wedge B), (A \vee B), (A \rightarrow B), (A \leftrightarrow B)$ are formulas.

[7]The "and" introduction is a rule where from X and Y follows $X \wedge Y$.

(iii) If x is a variable and A a formula, then $(\forall x)A, (\exists x)A$ are formulas.

(iv) Every formula arises from a finite number of applications of (i) to (iii).

Definition 5.15 (Theorem)**.** A *theorem* is a formula that is valid, i.e. a formula that is logically true in the given formal system.

5.4.2 Semantics

Now that we have learned the basics of syntax of predicate logic, we can have a look at the semantics. This is brought about by a relational structure $\mathfrak{M}$, which realizes (or instantiates) the symbols of our language. Moreover $\mathfrak{M}$ tells us which formulas are valid. To start with, we have to provide some values to our variables. The range of the values of our variables will be a nonempty set M, called universe of discourse $\mathfrak{M}$, and its members are individuals. On this universe of discourse, the function and predicate symbols are also realized on this universe of discourse.

Example 5.7. The realization of the language of number theory (arithmetics, see previous section) can be as follows: the universe of discourse is ω (set of all natural numbers), constant 0 is realized by an empty set $\emptyset$, the successor function is realized by a function that assigns the successive natural number to every number $n \in \omega$, and the function symbols $+$ and $\cdot$ are realized by conventional addition and multiplication.

Similarly to propositional calculus, we can investigate whether a certain formula is satisfiable or whether it is valid in every interpretation. However, in predicate logic, things get a bit more complicated. First, a relational structure $\mathfrak{M}$ realizing the language has to be chosen. This specifies how the function and predicate symbols are realized and also gives the universe of discourse M, from which we can choose the values for our variables. Once we have chosen $\mathfrak{M}$, we can assign various values to our variables – an interpretation of variables in predicate logic. An analogy to satisfiability in propositional logic would be to find an interpretation of the variables for which a formula is true.

For instance, suppose we have a standard realization (also called *model*) of number theory and the formula $x > y$. Obviously, we can find values for x and y such that the formula is true.

A stronger assertion is that a formula is *valid* in a realization $\mathfrak{M}$. That means that it is valid for every interpretation. This is analogous to a tautology. However, in predicate logic, it is with respect to a chosen realization. Obviously the formula

from the previous example is not valid – we can easily find values for x and y such that it is not true.

Suppose we had a formula $(\forall x)(\forall y)x > y$. In this case, whenever we find one interpretation giving a value of true, we automatically know that it is valid. This is because all free variables in the formula are universally quantified – we have to check all possible interpretations.

Scope of a Quantifier

The definition of the scope of a quantifier is illustrated in the following example.

Example 5.8. For every human x there exists a human y that loves x. Stated formally:

$$\underbrace{\forall x, (\text{human}(x) \rightarrow \exists y \underbrace{(\text{human}(y) \wedge \text{loves}(x, y))}_{\text{scope of } y})}_{\text{scope of } x}$$

Definition 5.16.

(i) A given occurrence of a variable x in a formula A is *bounded*, if it is part of a subformula of A (i.e. a substring of A that is also a formula) of the form $(\exists x)B$ or $(\forall x)B$. If an occurrence is not bounded, it is *free*.

(ii) A variable is *free* in A, if it has a free occurrence there.

A variable is *bounded* in A, if it has a bounded occurence there.

(iii) Formula A is *open*, if it does not contain any bounded variable.

Formula A is *closed*, if it does not contain any free variable.

Example 5.9. Formula A:

$$(\forall x)(x \rightarrow y)$$

In formula A, x has a bounded occurrence by the quantifier $\forall$, and hence it is bounded in A, whereas y is not quantified and hence it has a free occurrence and thus is free in A. Formula A is neither open nor closed.

Example 5.10. Formula B:

$$(\forall x)(\forall y)(x \rightarrow y)$$

In formula B both are variables are bounded and hence this is a closed formula.

5.4.3 Formal system

For the definition of the formal system, we will use a reduced form of the language – with logical connectives $\neg$ and $\rightarrow$ only and with only a universal quantifier $\forall$. You should be able to work out, why we can do this with the connectives. In case of the quantifiers, we use the fact that for a formula A, $(\exists x)A$ is equivalent to $\neg((\forall x)\neg A)$. The following is a formal system of predicate logic without equality.

1a) Axioms for logical connectives

(A1) – (A3) from propositional calculus

Thus, the whole propositional logic becomes a 'subset' of predicate logic. Tautologies of propositional calculus are automatically theorems of predicate calculus.

1b) Inference rule: Modus ponens

2) Axioms for quantifiers

2a) Specification scheme: Let A be a formula, x a variable and t a term that can be substituted for x into A

$$(\forall x)A \rightarrow A_x[t]$$

2b) "Jump scheme:" A, B are formulas, x a variable which is not free in A, then

$$(\forall x)(A \rightarrow B) \rightarrow (A \rightarrow (\forall x)B)$$

Comment: This is a rather technical axiom, to be used in prenex operations.

3) Inference rule: Universal generalization For an arbitrary variable x, from a formula A, derive $(\forall x)A$.

Comment: This shows the role of free variables in theorems. Whenever you can prove a formula A with a free variable x, then you can prove also a formula $(\forall x)A$. This is because, from a semantic point of view, for free variables you would have to check all possible interpretations anyway.

Rules of Manipulation

Permutation

$$\forall x(\forall y(P(x,y))) \leftrightarrow \forall y(\forall x(P(x,y)))$$

A similar rule can be shown for the existential quantifier.

Negation

$$\neg(\forall x(P(x))) \leftrightarrow \exists x(\neg P(x))$$

For the negation of the universal quanitifer it suffices to show that there exists one case for which $\neg P(x)$.

Nesting/Applicability

$$(\forall x : P(x)) \wedge Q \leftrightarrow \forall x : (P(x) \wedge Q)$$

Here, x appears in P, but not in Q. Therefore it does not affect the truth value of Q when it is grouped with P with respect to x. Similar argumentation holds true for the existential quantifier.

Prenex normal form

Just normal forms are useful for propositional calculus (conjunctive normal form, disjunctive normal form), there is a normal form for predicate calculus. Because of the higher complexity of predicate calculus – we have to take care of the quantifiers – are somewhat more involved. The goal is to move all the quantifiers to the beginning of the formula. This makes the formulas more transparent and comparable, and it makes them more accessible to automated processing.

Definition 5.17. We say that formula A is in *prenex form*, if it has the following form:

$$(Q_1 x_1) \ldots (Q_n x_n) B$$

where

1. Q_i are either $\forall$ or $\exists$
2. B is an open formula (i.e. all variables are free in it)
3. $x_1 \ldots x_n$ are all different

B is called an open core of A and the sequence of quantifiers is called prefix.

Replacement (renaming) of bounded variables

Suppose we have a formula A which contains a subformula of the form $(Qx)B$ (where Q is either $\forall$ or $\exists$). Then it is possible to replace x by y (in the prefix as well as in the formula B) and we obtain an equivalent formula A', a variation of A. However, we have to take care – the original formula B could not contain free occurences of y as these would then become bounded by our replacement. The safest way is to take a completely new symbol to name our variable.

Theorem 5.3. *For every formula A, it is possible to construct an equivalent formula A' in prenex form, such that $\vdash A \leftrightarrow A'$.*

Proof. Formula A' is constructed by using prenex operations. These replace subformulas of A according to one of the following schemes (where Q is either $\forall$ or $\exists$ and $\bar{Q}$ is the other quantifier than Q).

(a) replace subformula B by a variation of it B'

(b) replace subformula $\neg(Qx)B$ by $(\bar{Q}x)\neg B$

(c) if x is not free in B, replace subformula $B \rightarrow (Qx)C$ by $(Qx)(B \rightarrow C)$

(d) if x is not free in C, replace subformula $(Qx)B \rightarrow C$ by $(\bar{Q}x)(B \rightarrow C)$

(e) if the symbol $\star$ represents either $\wedge$ or $\vee$ and x is not free in C, then replace the subformula

$$(Qx)B \star C \quad \text{or} \quad C \star (Qx)B \quad \text{by} \quad (Qx)(B \star C)$$

□

5.5 Extensions

Although FOPC (First Order Predicate Calculus) has proved extremely useful and has broad applicability in virtually all areas of mathematics, and is used widely in computer science, etc., it does have some serious limitations. For example, we cannot express ideas like “this should be the case”, “I believe this to be the case”, or “this is almost correct”. Moreover, there is no notion of time. New forms of logic such as modal logic, fuzzy logic, and temporal logic have been developed to deal with these issues.