

Lecture Notes on Propositional and Predicate Logic

Martin Pilát

Based on lecture by Petr Gregor

OCTOBER 6, 2017

Introduction

Generally, logic is a study of arguments and inferences. While it started as philosophical discipline in ancient times, it is now widely studied in mathematics and computer science. As such, logic provides the basic language and tools for most of mathematics. It studies different prove systems and discusses whether they are sound (everything they prove is valid) and complete (everything that is valid can be proven).

While logic provides rather low-level tools for mathematics and computer science, it still has rather wide applications. For example, in some areas of artificial intelligence, logic is used to represent the knowledge of the intelligent agents and reason about it. The agents than use logical reasoning (theorem proving) to decide what to do next, or prove that a certain action is safe in a given environment. Another important area is formal software verification, where logic (and, again, theorem proving) can be used to formally verify, that a program indeed does what it should according to a specification. This is essential while implementing e.g. cryptographic protocols. Formal verification is also used while designing digital circuits.

There are also attempts to formalize the whole mathematics in logic and use computers to check that all the proofs are correct. For example, Mizar¹ is a system that aims to re-create most of the mathematics with formal and verified proofs. The verification starts from the basic mathematical axioms – in the case of the Mizar system, authors of so called Mizar articles can use only axioms of set-theory and theorems from previously verified articles. Therefore, everything published in the Mizar mathematical library is verified to be a correct consequence of the base axioms.

¹ <http://www.mizar.org/library/>

Logic serves as the formal language of mathematics, and therefore logic also needs to formally specify the syntax of the language. The syntax defines, what is a valid logical formula and what is not, however the meaning and validity of a formula is given by the semantics of the language. Logic itself prescribes the meaning of only a handful of symbols – namely the logical connectives (\wedge , \vee , \rightarrow , \leftrightarrow , \neg) and the quantifiers (\forall , \exists). Additionally, in languages with equality, the meaning of “=” is also given. All other symbols used in logical formulas can have arbitrary meaning, which is given by the semantics. So, for example, if we write a formula $(\forall x)(\forall y)(x + y = y + x)$, we cannot discuss its validity before defining the meaning of “+”. The formula is valid if we are talking about the real numbers and “+” denotes their addition, however, the symbol “+” can also represent (quite un-

usually) the multiplication of square matrices, and in such a case, the formula is not valid.

There are different levels of the language of logic. In propositional logic, only propositional variables (those that are either true or false) and the logical connectives can be used. In first order logic, we can additionally use functions, relations and quantifiers for variables that range objects from some universe. In second order logic, there are additionally quantifiers for sets of objects in the universe (and, more specifically, for functions and relations). In higher order logic, we also have variables for sets of sets of objects. For example, a formula in propositional logic

$$(d \wedge c) \rightarrow s$$

can express that if it is dark and clear outside, the stars are visible. In a first order language, we can have a more complex formula

$$(\forall x)(\forall y)(S(x) \wedge E(y) \rightarrow (L(x, y) \rightarrow P(x, y)))$$

that expresses that if x is a student ($S(x)$) and y is an exam ($E(y)$), if x learns for y ($L(x, y)$) then x passes y ($P(x, y)$). As an example of second order language, we can write the axiom of induction:

$$(\forall P)(P(0) \wedge (\forall n)(P(n) \rightarrow P(n+1)) \rightarrow (\forall n)P(n)).$$

In the lecture, we will deal mostly with propositional and first-order logic, however there are also other extension of logic. For example, in multi-agent systems, so called modal logic is often used to represent the knowledge. In modal logic, there are special modalities, that can further qualify a statement. For example, there is a modality that says that a statement may be true, or must be true. Other types of modal logic contain modalities that express knowledge of other agents (e.g. “agent A knows that statement S is true” can be written as $K_A.S$, or even “agent A knows that agent B knows that statement S is true” ($K_A.K_B.S$)). Another interesting type of modal logic is temporal logic, which contain modalities about time and can express e.g. “statement S will be true sometimes in the future”.

About these lecture notes

In these lecture notes, logic is presented for students of the computer science programme. Therefore, focus is given to areas most needed for computer scientists. For example, we use the more intuitive tableau method instead of the Hilbert-style prove systems. We also explain the resolution method in logic as a background to Prolog and logical programming. The more advanced topics on decidability and incompleteness are explained in a more informal way.

You are currently reading the first version of the lecture notes, which can, and most probably will, contain some errors. If you find an error, or if something is not clear, do not hesitate to contact the author by e-mail², or, alternatively, create an issue in the GitHub repository of the book³.

² Martin.Pilat@mff.cuni.cz

³ <https://github.com/martinpilat/logic-book>

There are also other resources you may want to check. One of them are the presentations created by Petr Gregor for his version of the lecture⁴, that serve as a base for these lecture notes.

⁴ <http://ktiml.mff.cuni.cz/~gregor/logics/index.html>

Informal! The author of these notes sometimes likes to explain things in a more intuitive way with some not-so-formal examples and metaphores. While he believes these can help to get better understanding of the given concept, they sometimes (read “often”) are rather informal and have some limitations. Therefore, in these notes, they will be set in boxes like the one you are reading just now with a bold “**Informal!**” warning. The information contained in these boxes is always non-essential to the rest of the text and can be (some would even argue that should be) skipped.

Preliminaries

Like many mathematical texts, these lecture notes also assume that the reader has some basic knowledge. The most important concepts (many of which should sound familiar) are briefly introduced in this short section, both to provide a single place where these can be found and to introduce the notation used in these lecture notes.

We will start with the basic set-theoretic notions. The most basic of these is the *class*. Each property of sets $\varphi(x)$ defines a class $\{x \mid \varphi(x)\}$. Some classes are also sets, those that are not are called *proper classes*. The distinction between sets and classes is probably new for most of the readers. Why would we need any other collections of objects than sets? How is it possible, that there is a collection of objects, which is not a set? The reason to distinguish between these two is that if everything was considered a set, we could find paradoxes in the set theory. For example, if we had a set of all sets that do not contain themselves, does this set contain itself, or not? Let us assume, it does, but then, by definition, it does not. If we instead assume it does not contain itself, then, again, by definition, it does. This so called Russel’s paradox can be avoided by using a notion of classes, that cannot contain other classes.

Informal! A class can be understood as any collection of sets that can be described by the language of set theory. However, some of these collections do not make much sense and can lead to paradoxes. Therefore, any collection, that would lead to some paradox is denoted as a proper class instead of a set and the paradoxes can thus be avoided.

The other set-theoretic notions should be much more familiar. We use $x \in y$ to denote that x is a member of set y , $x \notin y$ and $x \neq y$ are shortcuts for $\neg(x \in y)$ and $\neg(x = y)$. A set containing exactly elements x_0, x_1, \dots, x_n is denoted as $\{x_0, x_1, \dots, x_n\}$. A set with only one element $\{x\}$ is called a *singleton* and a set with two elements $\{x_0, x_1\}$ is called an *unordered pair*. We will also use the common notation for set operations: \emptyset denotes an *empty set*, \cup and \cap denote *union* and *intersection* of sets. The \setminus is the *set difference* operator and \triangle

is the *symetric set difference operator*

$$x \triangle y = (x \setminus y) \cup (y \setminus x).$$

Two sets are *disjoint*, if their intersection is an empty set, and $x \subseteq y$ denotes that x is a subset of y (all elements of x are also elements of y). The set of all subsets of a set x – the *power set of x* – is denoted as $\mathcal{P}(x)$. The *union of set x* , $\bigcup x$, is the union of all sets contained in x . A *cover of a set x* is a set $y \subseteq \mathcal{P}(x) \setminus \emptyset$, such that $\bigcup y = x$, if all the sets in the cover y are mutually disjoint, than y is a *partition of x* .

The definition of an unordered pair can be used to define the *ordered pair* $(a, b) = \{a, \{a, b\}\}$ and an *ordered n -tuple* $(x_0, \dots, x_{n-1}) = ((x_0, \dots, x_{n-2}), x_{n-1})$ for $n > 2$. A Cartesian product of two sets a and b is $a \times b = \{(x, y) | x \in a, y \in b\}$ and the Cartesian power of a set x is $x^0 = \{\emptyset\}$, $x^n = x^{n-1} \times x$. A *binary relation R* is a set of ordered pairs. The *domain of R* is defined as $\text{dom}(R) = \{x | (\exists y)(x, y) \in R\}$, the *range of R* is similarly $\text{rng}(R) = \{y | (\exists x)(x, y) \in R\}$. The *extension of x in R* is the set $R[x] = \{y | (x, y) \in R\}$. The symbol R^{-1} denotes the *inverse relation* $R^{-1} = \{(y, x) | (x, y) \in R\}$. The *restriction of R to a set z* is defined as $R \upharpoonright z = \{(x, y) \in R | x \in z\}$. Two relations can also be *composed* into one, $R \circ S = \{(x, z) | (\exists y)((x, y) \in R \wedge (y, z) \in S)\}$. The *identity relation* on set z , $\text{Id}_z = \{(x, x) | x \in z\}$.

A binary function f is a special type of binary relation where for every $x \in \text{dom}(f)$ there is exactly one y such that $(x, y) \in f$, then, y is the value of f in x denoted as $f(x)$. $f : X \rightarrow Y$ denotes a function f with $\text{dom}(f) = X$ and $\text{rng}(f) \subseteq Y$. The set of all such functions is ${}^Y X$. A function $f : X \rightarrow Y$ is a *surjection* (onto) if $\text{rng}(f) = Y$, and it is an *injection* (one-to-one) if for any $x, y \in \text{dom}(f)$, $x \neq y \rightarrow f(x) \neq f(y)$. A function that is both a surjection and injection is called a *bijection*. Similarly to relation, we can define the inverse function f^{-1} , and the composition of funtions $f : X \rightarrow Y$ and $g : Y \rightarrow Z$ as a function $f \circ g$ with $(f \circ g)(x) = g(f(x))$. The image of a set A , denoted as $f[A]$ is the set of function values for all elements of A , $f[A] = \{y | (x, y) \in f, x \in A\}$.

There are also two special types of relations which will be important later: equivalences and orders. An equivalence on a set X is relation that is reflexive ($R(x, x)$ for all $x \in X$), symetric ($R(x, y) \rightarrow R(y, x)$ for $x, y \in X$) and transitive ($(R(x, y) \wedge R(y, z)) \rightarrow R(x, z)$) for all $x, y, z \in X$. The extension of x in R is called the equivalence class of x and is also denoted as $[x]_R$. $X/R = \{R[x] | x \in X\}$ is the *quotient set of X by R* . The quotient set is always a partition of X and every partition of X also defines an equivalence on X (two elements are equivalent if they are in the same set in the partition).

The other imporant types of the relations are the orders, usually an order is denoted as \leq . Such a relation is a *partial order of a set X* , if it is reflexive ($x \leq x$ for $x \in X$), antisymmetric ($x \leq y \wedge y \leq x \rightarrow x = y$ for $x, y \in X$) and transitive ($x \leq y \wedge y \leq z \rightarrow x \leq z$ for $x, y, z \in X$). If, additionally, for every two elements $x, y \in X$ it holds that $x \leq y$ or $y \leq x$ (dichotomy) than \leq is a total (linear) order. It is a well-order if additionally every non-empty subset of X has a least element. Finally,

and order of X is dense, if X is not a singleton and for every two elements $x, y \in X$, there is another element $z \in X$ between these two ($x < y \rightarrow (\exists z)(x < z \wedge z < y)$), where $a < b$ means that $a \leq b \wedge a \neq b$.

For example, the common ordering of natural numbers (\leq on \mathbb{N}) is a linear well-order (as every two natural numbers are comparable and every subset of natural numbers has a least element under this order), however, it is not a dense order, as for example there is no natural number between 0 and 1. On the other hand, the common ordering of rational numbers is a dense linear order (there is a rational number between any pair of distinct rational numbers), however it is not a well-order, as e.g. the set $\{x \in \mathbb{Q} \mid x \leq 0\}$ has no least element.

The natural numbers can be defined using the empty set in an inductive way $0 = \emptyset, 1 = \{0\} = \{\emptyset\}, 2 = \{0, 1\} = \{\emptyset, \{\emptyset\}\}, \dots, n = \{0, \dots, n-1\}, \dots$. The set of all natural numbers \mathbb{N} is the smallest set containing \emptyset and closed under the operation of successor $S(x) = x \cup \{x\}$. The other common sets of numbers are the integers, which can be defined as the $\mathbb{Z} = (\mathbb{N} \times \mathbb{N}) / \sim$, with $(a, b) \sim (c, d)$ if and only if $a + d = b + c$. Similarly, the set of rational numbers \mathbb{Q} can be defined as $\mathbb{Q} = (\mathbb{Z} \times (\mathbb{Z} \setminus \{0\})) / \sim$, with $(a, b) \sim (c, d)$ if and only if $ad = bc$. The definition of real numbers \mathbb{R} is more complex. These are usually defined as cuts of the rational numbers \mathbb{Q} , where a cut is a partition of \mathbb{Q} into two sets A and B , where all numbers in B are greater than all numbers of A , and A has no greatest element. For example, the cut corresponding to the irrational number $\sqrt{2}$ is $A = \{a \in \mathbb{Q} \mid a^2 < 2 \vee a < 0\}, B = \{b \in \mathbb{Q} \mid b^2 > 2 \wedge b > 0\}$.

Another important notion for the rest of the lecture deals with the cardinality ("size") of sets. A set X has a cardinality smaller or equal to set Y ($X \preceq Y$) if there is an injective function $f : X \rightarrow Y$. If there is a bijection $f : X \rightarrow Y$ then we say that X and Y have the same cardinality ($X \approx Y$), finally X has strictly smaller cardinality than Y ($X \prec Y$) if ($X \preceq Y \wedge \neg(X \approx Y)$). For each set x , there is a cardinal number $\kappa \approx x$, denoted as $|x| = \kappa$. A set X is *finite* if $|X| = n$ for some $n \in \mathbb{N}$. It is *countable*, if it is finite or if $|x| = |\mathbb{N}| = \omega$. Otherwise, it is *uncountable*. The cardinality of $\mathcal{P}(\mathbb{N})$ is called the continuum.

It is interesting to know the cardinality of the common sets of numbers. Obviously, the set of natural numbers \mathbb{N} is countable. A less obvious fact is that the sets of integers and rational numbers also have the same cardinality and are therefore also countable. For the integers, we can create an infinite sequence of integers $s = \langle 0, 1, -1, 2, -2, 3, -3, \dots \rangle$, then a function $f(i) = s_i$ is an injective function $\mathbb{N} \rightarrow \mathbb{Z}$, therefore $\mathbb{Z} \preceq \mathbb{N}$. The other inequality ($\mathbb{N} \preceq \mathbb{Z}$) is obvious (use identity as the injective function). In order to show that the set of rational numbers \mathbb{Q} is also countable, we can create a function $f(\frac{p}{q}) = 2^{|p|} 3^q 5^{\text{sign}(p)}$ (we consider only cases where $p \in \mathbb{Z}, q \in \mathbb{N} \setminus \{0\}$, which clearly covers all the rationals), this is again an injective mapping $\mathbb{Q} \rightarrow \mathbb{N}$ and therefore $\mathbb{Q} \preceq \mathbb{N}$. As before, the other inequality is trivial. Finally, we can show that the set of real numbers \mathbb{R} has bigger cardinality than the set of natural numbers \mathbb{N} . Obviously $\mathbb{N} \preceq \mathbb{R}$ as $\mathbb{N} \subseteq \mathbb{R}$. Let us assume both the sets have the

same cardinality, in such a case there is bijection $f : \mathbb{N} \rightarrow \mathbb{R}$. We will now define a new real number r in the following way. The integer part of the number is 0, the first digit after the decimal point is different from the first digit after decimal point in $f(0)$, the second digit is different from the second digit in $f(1)$, and so on⁵. This real number is different from all the numbers in $\{f(0), f(1), \dots\}$ as it differs from the number $f(i)$ in the i -th digit after the decimal point. This is a contradiction with the assumption that f is a bijection between \mathbb{N} and \mathbb{R} and therefore $\mathbb{N} \prec \mathbb{R}$.

We will conclude the discussion of cardinalities by showing the Cantor's theorem.

Theorem 1 (Cantor). *For every set x , $x \prec \mathcal{P}(x)$.*

Proof. First, $f(x) = \{x\}$ is an injection $X \rightarrow \mathcal{P}(x)$ and therefore $x \preceq \mathcal{P}(x)$. Suppose there is also an injective $g : \mathcal{P}(x) \rightarrow x$. We can define a set $y = \{g(z) | z \subseteq x \wedge g(z) \notin z\}$. Now, similarly to the Russel's paradox, $g(y) \in y$ if and only if $g(y) \notin y$, which is a contradiction, and therefore there cannot be any such injective g and so $x \prec \mathcal{P}(x)$. \square

As the tableau method used in this lecture relies on trees, we will conclude this preliminary section by a brief discussion on trees. Most of the readers are probably familiar with finite trees, however, we will sometimes need to work with infinite trees and therefore we define a *tree* as a set T partial order $<_T$ (called the tree order) with a unique least element (*the root*) and in which the set of predecessors of any element is well-ordered by $<_T$. In this definition a branch is a maximal linearly ordered subset of T . Apart from this difference in definition, we will use the common terminology on trees from the graph theory. For simplicity, we will only consider finitely branching trees, where each node except the root has an immediate predecessor⁶. In such trees we can define the *levels of the tree*. The root is on the level 0, the sons of nodes on the $n - 1$ -th level are on level n . The depth of tree is maximal $n \in \mathbb{N}$ of a non-empty level. In case the tree has an infinite branch it has an infinite depth ω . In an n -ary tree, each node has at most n sons and a tree is finitely branching if each node has a finite number of sons.

Lemma 1 (König). *Every infinite, finitely branching tree contains an infinite branch.*

Proof. The root of the tree has only finitely many sons, therefore there is a son of the root that is infinite. We choose this son and continue in the same way with his sons, thus constructing an infinite branch. \square

Apart from the tree order $<_T$ we sometimes need to work with *ordered trees* where the sons of each node are additionally ordered from left to right with a *left-to-right order* $<_L$. In a *labeled tree* each node also contains an additional information. For example, the formula

$$(p \wedge q) \rightarrow q$$

can be represented as the labeled ordered tree on the left.

⁵ If we write the decimal value of the number r as $r = 0.r_0r_1r_2\dots$, where r_i is the i -th decimal digit, we can define $r_i = (f(i)_i + 1) \bmod 10$, where $f(i)_i$ is the i -th decimal digit of $f(i)$.

⁶ This means, we will not deal, for example, with trees where the nodes would be set of rational numbers \mathbb{Q} and the tree order $<_T$ would be the common order on \mathbb{Q} .

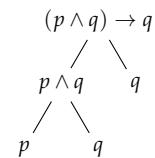


Figure 1: The labeled ordered tree representing the formula $(p \wedge q) \rightarrow q$.

Contents

<i>Introduction</i>	3
<i>About these lecture notes</i>	4
<i>Preliminaries</i>	5

List of Figures

- 1 The labeled ordered tree representing the formula $(p \wedge q) \rightarrow q$. 8

List of Tables

List of Algorithms

Todo list