Formal Logic: Handout 4

Propositional Logic – Syntax

Yılmaz Kılıçaslan
March 15, 2023

Key Concepts: *(Un)certainty, Ambiguity, Vagueness, Preciseness, Formal(ization), System, Semiotic Triangle, Syntax, Model, Semantics, Basic Expressions, Logical Constants, Propositional Variables, Formation Rules, Well-Formed(ness), Construction Tree, Reverse Engineering, Derivation, Atomic (Component), Main Operator, Leaf of a Tree, Re-Write Rule, Context-Free Grammar, (Non-)Terminal Symbol.*

1 Introduction

Let us begin with a retrospective look at what we have done so far.

As we pointed out at the outset of the term, computers scientists have to be interested in logic for various reasons. Let us re-emphasize one of these reasons. Computer science has two major theories: computing theory and information theory. Computing theory is a theory of certainty while information theory deals with phenomena involving uncertainty. In the same vein, scientifically recognized logic splits into two branches: deductive logic and inductive logic. It is the former that is the logic of certainty and, hence, can serve as the ultimate basis for computing theory; whereas inductive logic can do the same thing for information with its capacity to handle uncertainty.

As you know, it is deductive logic that is the subject matter of this course. Deductive logic is the science of reasoning conducted through arguments and, most basically, arguments are a sequence of sentences. Up until now, we have constructed our arguments with sentences of a natural language, namely English. This means that our endeavor has involved a means-end conflict so far: trying to capture certainty with a very flexible instrument. Let us be a little bit more specific. (Deductive) logic is a branch of science that cannot tolerate ambiguities, vagueness, and inexactness. However, natural languages suffer from all these deficiencies: they are full of ambiguities, vague constructs, and unruly exceptions.

Today we will highlight a problematic facet of natural languages for doing logic; we will emphasize the need for formalization to overcome such problems; and to this effect, we will devise two formal languages for propositional logic, one in English and one in Backus-Naur Form (BNF).

2 Ambiguity in Natural Language

To illustrate just ambiguity in natural language, consider the following sentences:

(1) Every dog chased a cat.

(2) Every student did not pass the exam.

(3) The child saw her duck.

(4) The man saw the woman with binoculars.

(5) Mary wants to marry a millionaire.

(1) is ambiguous as to whether there was a single cat chased by every dog or there might be a separate cat for each dog to chase. (2) is ambiguous between a reading where all the students failed the exam and a reading where there is at least one student who did not pass the exam. (3) can receive an interpretation where the child saw a duck belonging to a female person, who might be the child herself or an interpretation where the child saw a female person duck, i.e., lower her head or body. In (4), the woman seen by the man could be carrying binoculars or it was with binoculars that he saw her. In (5), it might be the case that Mary wants the person whom she will marry to be a millionaire or that she wants to marry someone, who happens to be a millionaire even without Mary knowing this.

Consider now the following arguments:

(6) Every dog chased a cat.
Fido is a dog.
Rodrigo is a dog

There is a cat that was chased by both Fido and Rodrigo.

(7) Every student did not pass the exam.
John is a student.

John failed the exam.

(8) John saw her duck.
Ducks are animals.

John saw an animal.

(9) He saw the woman with binoculars.

It is not the case that he saw the woman with unaided eyes.

(10) Mary wants to marry a millionaire.

There is a millionaire.

Obviously, whether each of these arguments is valid or not depends on the interpretation to be assigned to its first premise. For example, (6) is valid on an interpretation of the first premise where there was a single cat chased by every dog but it is invalid if that premise is given an interpretation where there might be a separate cat for each dog to chase. (8) is much simpler to evaluate in this respect: it is valid if its first premise receives an interpretation where the child saw a duck belonging to a female person but invalid if that premise has an interpretation where the child saw a female person duck. Similar evaluations can be made for the remaining arguments.

What really matters here is this: natural language is not an appropriate medium of expression for logic. It is not tidy and precise. Therefore, it does not always allow for a clear expression of meaning elements which validity, the most important feature of an argument, rests on.

3 FORMALIZATION OF LOGIC

4.1 SEMIOTIC TRIANGLE

Formalization is the way of bringing virtues like tidiness and preciseness into logical study. It is only by using formal means of expression that logic can get rid of the 'mess' inherent in natural language.

It might be a good idea to have a quick look at the question of what a natural language is before delving into to the question of how to develop a formal system for logic. Like any system, a natural language is a combination of interacting components. Most simply and superficially, a natural language is a set of expressions used to speak about reality by human agents with sufficient mental capacity. Linguistics meaning arises out of the interaction of these three components; namely expressions, reality, and mind. Usually, the so-called semiotic triangle is used to describe such meaning creating interaction. Below is our version of the semiotic triangle:

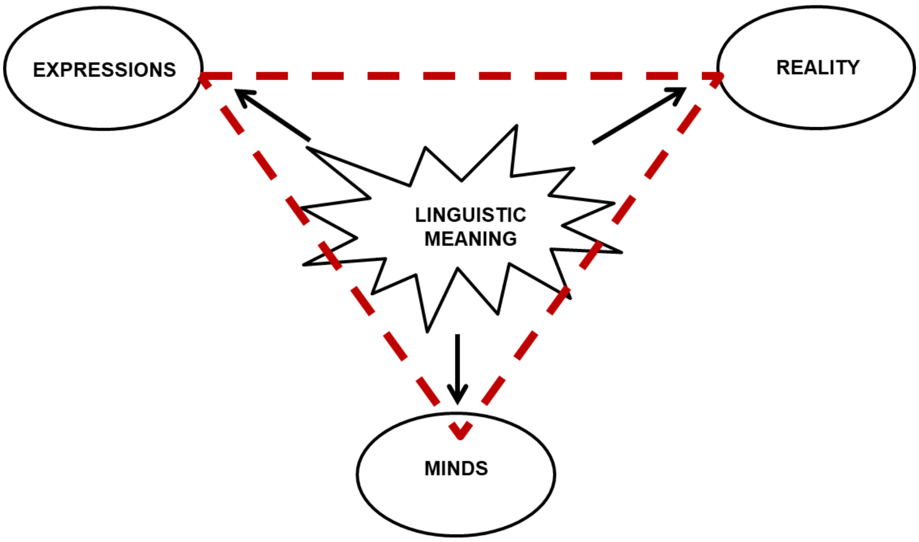


Figure 1. Semiotic triangle.

Each aspect of the semiotic interaction is studied in a particular scientific field. Below are some relevant fields of linguistic study:

- **Syntax:** the field where the question of how to construct the expressions of a language is addressed.
- **Model Theory:** the theory that defines the assumed structure of the reality that the well-formed expressions refer to.
- **Semantics:** the study that defines the mapping between the syntax and the model.
- **Pragmatics:** the endeavor to discover how a mentally capable agent contextualizes linguistic meaning.

Like many other branches of formal science, logic can make use of a formal language in order to get rid of the deficiencies of natural languages and benefit the advantages of formalization. Devising a formal language for logic, we will get to a formal system of logic where validity can be determined in a mechanical way. To this effect, we will sequentially develop several formal languages by formally defining their syntactic, model theoretic and semantic components. We will not have to be concerned with pragmatic questions because the expressions of all the formal languages that we will devise will be interpretable without taking the context into consideration.

As pointed above, a language, whether natural or formal, is a set of expressions. Each element of this set is called a well-formed expression. It is the syntax of a (formal) language that describes how to produce its well-formed expressions.[1] Today, we will define the syntax of a formal language for propositional logic, which we call L_1, in two different ways: first, in a logical tradition with descriptions in English and, then, in a tradition of computer science with rules in Backus-Naur Form (BNF). Next week, we will formally describe the other components of L_1; and, in the subsequent weeks, we will design and implement some other formal languages.

4.2 SYNTAX OF L_1 IN ENGLISH

4.2.1 Basic expressions

Like all languages, L_1 has a vocabulary, a set of basic expressions upon which to build larger expressions. Among the basic expressions of L_1 are logical constants and propositional variables:

1. Logical Constants: $\neg, \vee, \wedge, \rightarrow, \leftrightarrow$
2. Propositional Variables: p, q, r (with or without subscripts)

It is worth noting that the categories of basic expressions are not given above exhaustively. In fact, there is a third category of basic expressions, brackets, which will be introduced syncategorematically when specifying the formation rules.

4.2.2 Formation rules

The set of formation rules of L_1 includes four clauses:

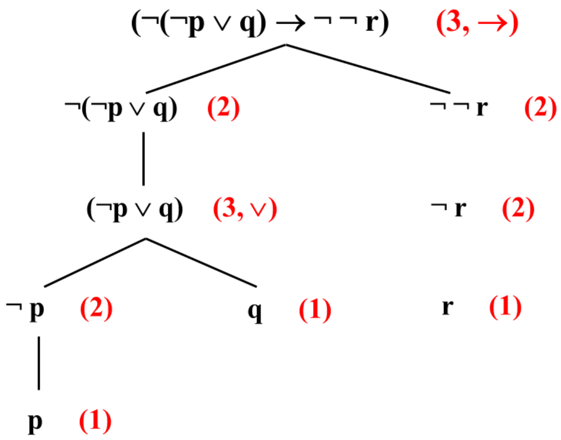
1. Any propositional variable of L_1 is a formula of L_1.
2. If φ is a formula of L_1, then so is $\neg\varphi$.
3. If φ and ψ are formulas of L_1, then so are $(\varphi \vee \psi)$, $(\varphi \wedge \psi)$, $(\varphi \rightarrow \psi)$, $(\varphi \leftrightarrow \psi)$.
4. Nothing else is a formula of L_1.

The first clause says that any propositional variable is a well-formed expression of this language. The second clause states that any well-formed expression can be preceded by the negation operator. The third clause specifies how to combine two well-formed expressions into a larger one using connectives. Notice that it is via this clause that the brackets are introduced. The brackets serve to remove any possible ambiguities due to priority clashes.

[1] The form of a natural language expression also bears phonological and morphological features. As formal language expressions do not have sound contents (i.e., they are unspeakable) and its terms are atomic (i.e., without an internal structure), they lack the phonological and morphological dimensions.

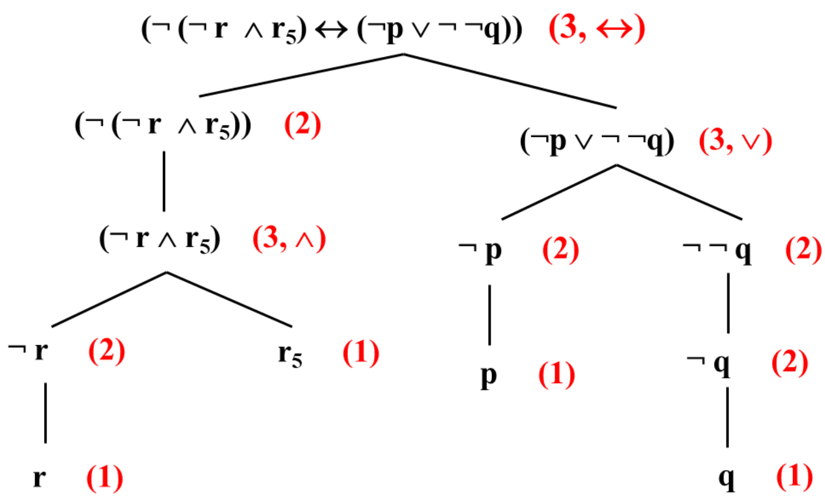
4.2.3 Construction trees

Each well-formed string of L_1 is associated with a unique *construction tree,* which is formed in accordance with the formation rules. Below is the construction tree of $(\neg(\neg p \vee q) \rightarrow \neg\neg r)$:



The construction of a tree like that is a reverse engineering process: we analyze the formula till we reach its atomic components. At each step of this analysis, we detect the *main* operator of the formula, which would be used last in its production, and determine the operand or operands as its child or children in the tree. We also indicate the number of the formation rule (in red font) according to which the formula would be produced from the main operator and its operands. The analysis ends with each leaf being a propositional letter. The existence of its construction tree demonstrates the well-formedness of a formula.

Here is another construction tree, that for $(\neg(\neg r \wedge r_5) \leftrightarrow (\neg p \vee \neg\neg q))$:



4.3 SYNTAX OF L_1 IN BNF

4.3.1 A set of BNF rules for the syntax of L_1

Below is a set of grammar rules specified in the Backus-Naur Form (BNF) that generate all well-formed expressions of L_1:

Formula → p

| q

| r

| p_1

| q_1

| r_1

| …

Formula → ¬ Formula

| (Formula ∧ Formula)

| (Formula ∨ Formula)

| (Formula → Formula)

| (Formula ↔ Formula)

Each of such rules is called a re-write rule. As the set of rules above defines a context-free grammar, there has to be a single non-terminal (i.e., variable) symbol to the left of the re-write operator, →. A string consisting of terminal (i.e., a logical constant or a propositional letter) and non-terminal symbols occurs to the right of the re-write operator. Formula is a special non-terminal symbol. It is the 'start' symbol.

4.3.2 Derivation of Formulas

Each well-formed string of L_1 can be derived from the symbol Formula and applying one single BNF rule at each step to move to the next string. Each move is represented by the symbol =>. Below is the derivation process of (¬(¬p ∨ q) → ¬ ¬ r):

Formula => (Formula → Formula) => (¬Formula → Formula) => (¬(Formula ∨ Formula) → Formula) => (¬(¬Formula ∨ Formula) → Formula) => (¬(¬p ∨ Formula) → Formula) => (¬(¬p ∨ q) → Formula) => (¬(¬p ∨ q) → ¬Formula) => (¬(¬p ∨ q) → ¬¬Formula) => (¬(¬p ∨ q) → ¬¬r)

REFERENCE

- Introduction to Logic: Logic, Language, and Meaning (Volume 1), Gamut, L.T.F. Chicago: The University of Chicago Press, 1991.