Chapter 1

Introduction to Logic

The word *logic* is used in two senses:

- A set of principles that underlie the organization of elements within a system (e.g., a computer program, an electronic device, a communication protocol)
- Reasoning conducted according to strict rules that preserve validity

In computer science, these two meanings converge: first, we formally describe the given system, and then we *formally reason* about it (in practical applications, this is done automatically), i.e., we derive valid *inferences* about the system using some *(formal) proof system*.

Practical applications of logic in computer science include software verification, logic programming, SAT solving, automated reasoning, database theory, knowledge-based representation, and many others. Moreover, logic (to a greater extent than mathematics) is a fundamental tool for describing theoretical computer science.

1.1 Propositional Logic

Now let's demonstrate logic in action with two real-life examples (from the lives of a treasure hunter and a theoretical computer scientist):

1.1.1 Example: Treasure Hunting

Example 1.1.1. While searching for treasure in a dragon's lair, we encountered a fork in the path with two corridors. We know that at the end of each corridor, there is either treasure or a dragon, but not both. A dwarf we met at the fork told us, "At least one of these two corridors leads to the treasure," and after some further urging (and a small bribe), she also said, "The first corridor leads to a dragon." It is well known that dwarves you meet in a dragon's lair either always tell the truth or always lie. Which way should we go?

1.1.2 Formalization in Propositional Logic

We will begin by formalizing the situation and our knowledge in propositional logic. A *proposition* is a statement to which we can assign a truth value: *True* (1) or *False* (0).

Some propositions can be expressed using simpler propositions and logical connectives, e.g., "(The dwarf is lying) *if and only if* (the second corridor leads to a dragon)" or "(The first corridor leads to treasure) *or* (the first corridor leads to a dragon)." If a proposition cannot be decomposed in this way, it is called a *simple proposition*, *atomic proposition*, or *propositional variable*.

Thus, we will describe the entire situation using *propositional variables*. We can also think of these as simple yes/no questions we need to answer to know everything about the given situation. Let's choose "The treasure is in the first corridor" (denote as p_1), and "The treasure is in the second corridor" (p_2). Other propositional variables could be considered, such as "There is a dragon in the first corridor" (d_1) or "The dwarf is telling the truth" (t). However, these can be expressed using $\{p_1, p_2\}$, e.g., t holds if and only if p_1 does not hold. That is, if we know the truth values of p_1, p_2, the truth values of d_1, t are uniquely determined. A smaller number of propositional variables means a smaller search space.

Next, we will express all our knowledge as (*compound*) propositions and write them in formal notation in the *language* of propositional logic over the set of atomic propositions $\mathbb{P} = \{p_1, p_2\}$, using symbols representing logical connectives: $\neg$ ("not X", *negation*), $\wedge$ ("X and Y", *conjunction*), $\vee$ ("X or Y", *disjunction*), $\rightarrow$ ("if X, then Y", *implication*), $\leftrightarrow$ ("X if and only if Y", *equivalence*), and parentheses (,). It is worth mentioning that disjunction is not exclusive; that is, "X or Y" holds even if both X and Y hold, and implication is purely logical: "if X, then Y" holds whenever X does not hold or Y holds.

The information that the corridor contains either a treasure or a dragon, but not both, is already encoded in our choice of propositional variables: the presence of a dragon is the same as the absence of treasure. The dwarf's statement that "The first corridor leads to a dragon" is thus expressed as "It is not the case that the treasure is in the first corridor," formally $\neg p_1$. The statement that "At least one of these two corridors leads to the treasure" is expressed as "The treasure is in the first corridor or the treasure is in the second corridor," formally $p_1 \vee p_2$. The information that dwarves either always tell the truth or always lie can be understood to mean that either both our propositions hold or the negations of both our propositions hold, formally:

$$(\neg p_1 \wedge (p_1 \vee p_2)) \vee (\neg(\neg p_1) \wedge \neg(p_1 \vee p_2))$$

Let us denote this proposition as φ (from the word "formula"; propositions are sometimes also called *propositional formulas*). In our example, all information can be expressed with a single proposition. But in practice, we often need more propositions, sometimes even infinitely many (for example, if we want to describe the execution of a computer program and we do not know in advance how many steps it will take). We then describe the situation using a set of propositions, called a *theory*, here $T = \{\varphi\}$. The propositions in T are also called *axioms* of the theory T[1].

1.1.3 Models and Consequences

Is our information sufficient to determine whether there is treasure in one particular corridor? In other words, we are asking if one of the propositions p_1 or p_2 is a logical *consequence* of the proposition φ (or the theory T). What does this mean?

Let's imagine that there are several "worlds" differing in what is at the end of the first and second corridors. For example, in one of the worlds, there is treasure at the end of the first

[1] Terminology in logic often comes from its application in mathematics.

corridor and a dragon at the end of the second corridor. We can describe this world using a truth valuation of the propositional variables: $p_1 = 1, p_2 = 0$. Such a valuation is called a *model* of the language $\mathbb{P} = \{p_1, p_2\}$ and we write it succinctly as $v = (1, 0)$ (v from the word "valuation"). Thus, we have a total of four different worlds, described by the *models of the language*:

$$\mathrm{M}_{\mathbb{P}} = \{(0,0), (0,1), (1,0), (1,1)\}.$$

Is the world described by the model $v = (1, 0)$ consistent with the information we have, i.e., does the proposition φ (or the theory T) *hold* in it? We can easily determine the truth value of the (compound) proposition φ in the model v, denote it $v(\varphi)$: We know that $v(p_1) = 1$ and $v(p_2) = 0$, so $v(\neg p_1) = 0$, and also $v(\neg p_1 \wedge (p_1 \vee p_2)) = 0$ (it is a conjunction of two propositions, and the first conjunct is false in the model v). Similarly, $v(p_1 \vee p_2) = 1$ (because $v(p_1) = 1$), so $v(\neg(p_1 \vee p_2)) = 0$, and $v(\neg(\neg p_1) \wedge \neg(p_1 \vee p_2)) = 0$. The proposition φ is a disjunction of two propositions, neither of which holds in the model v, so $v(\varphi) = 0$.

A keen reader will surely see the tree structure of the proposition φ and the step-by-step evaluation of $v(\varphi)$ from the leaves to the root. We will present a formal definition in the next chapter.

Similarly, we determine the truth values of the proposition φ in other models. We find that the set of *models of the proposition* φ (or the *models of the theory* T), i.e., the set of all models of the language in which φ (or all axioms of the theory T) holds, is

$$\mathrm{M}_{\mathbb{P}}(\varphi) = \mathrm{M}_{\mathbb{P}}(T) = \{(0,1)\}.$$

We see that our information uniquely determines the model $(0, 1)$, the world in which there is a dragon in the first corridor and treasure in the second corridor. In general, there can be more models; we only need to know that in every model of φ (or of T) the proposition p_2 holds, to conclude that p_2 is a *consequence* of the theory T; we also say that p_2 *holds* in the theory T.

1.1.4 Proof Systems

The approach we have chosen is very inefficient. If we have n propositional variables[2], there are 2^n models of the language, and it is practically impossible to check the validity of the theory in each of them. This is where *proof systems* come into play. In a given proof system, a *proof* of a proposition ψ from a theory T is a precisely, formally defined syntactic object that includes an easily (mechanically) verifiable "proof" (reason) that ψ holds in T, and which can be searched for (using a computer) purely based on the structure of the proposition ψ and the axioms of the theory T ("syntax"), i.e., without having to deal with models ("semantics").

We want two properties from a proof system:

- *soundness*, i.e., if we have a proof of ψ from T, then ψ holds in T, and
- *completeness*, i.e., if ψ holds in T, then there exists a proof of ψ from T,

where soundness is a necessity (without it, searching for proofs makes no sense), and completeness is a desirable property, but an efficient proof system can be useful even if not everything that holds can be proven.

[2]In practice, we commonly have thousands of variables.

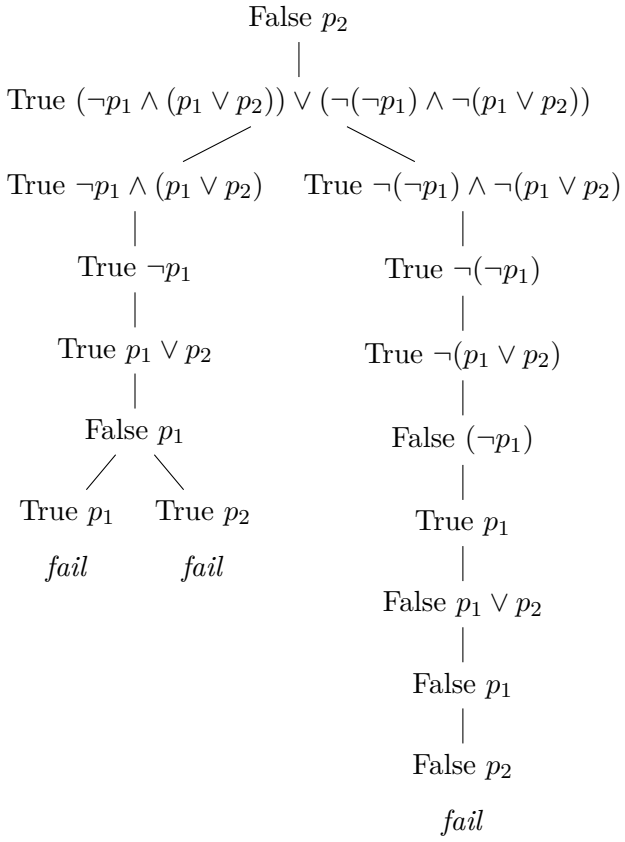

Figure 1.1: Tableau proof of the proposition p_2 from the theory T

Here we briefly outline two proof systems: the *method of analytic tableaux* and the *resolution method*. They will be formally introduced later, and we will prove soundness and completeness for both of them. Both of these proof systems are based on *proof by contradiction*, i.e., they assume the validity of the axioms from T and the negation of the proposition ψ, and try to reach a contradiction.

1.1.5 Tableau Method

In the method of analytic tableaux, a proof is a *tableau*: a tree whose nodes are labeled with assumptions about the validity of propositions. Let's look at an example of a tableau in Figure 1.1.

We start with the assumption that the proposition p_2 does not hold (because we are proving by contradiction). Then we add the validity of all axioms of the theory T (in our case, there is only one: the proposition φ constructed above). We then build the tableau by simplifying the propositions in the assumptions according to certain rules that ensure the following invariant:

> *Every model of the theory T in which p_2 does not hold must* agree *with one of the branches of the tableau (i.e., satisfy all assumptions on that branch).*

The proposition φ is a disjunction of two propositions, $\varphi = \varphi_1 \vee \varphi_2$. If it holds in some model, then either φ_1 holds in that model or φ_2 holds in it. We branch the tree according to these two possibilities. In the next step, we have the assumption about the truth of the proposition $\neg p_1 \wedge (p_1 \vee p_2)$. In that case, both $\neg p_1$ and $p_1 \vee p_2$ must hold, so we add both of these assumptions to the end of the branch. The truth of $\neg p_1$ means the falsity of p_1, and so on.

We proceed in this manner until it is no longer possible to simplify the propositions in the assumptions, i.e., until they are just propositional variables. If we find a pair of opposite assumptions about some proposition ψ on one branch, i.e., that it both holds and does not hold, we know that no model can agree with this branch. Such a branch is called *contradictory*. Since we are proving by contradiction, a proof is a tableau in which every branch is contradictory. This ensures that there is no model of T in which p_2 does not hold. From this, it follows that p_2 holds in every model of T, in other words, it is a consequence of T, which is what we wanted to prove.

For now, we will be satisfied with understanding the basic idea of this method; the details will be presented later in Chapter **??**.