# Logic for Computer Science - Lecture Notes

Alexandru Ioan Cuza University, Iași
Faculty of Computer Science
University Year 2023-2024

Ștefan Ciobâcă
Andrei Arusoaie
Rodica Condurache
Cristian Masalagiu

# Contents

**1 Introduction** **7**

**2 Informal Propositional Logic** **9**
2.1 Propositions . . . . . . . . . . . . . . . . . . . . . . . . . . . 9
2.2 Atomic Propositions . . . . . . . . . . . . . . . . . . . . . . . 10
2.3 Conjunctions . . . . . . . . . . . . . . . . . . . . . . . . . . 10
2.4 Disjunctions . . . . . . . . . . . . . . . . . . . . . . . . . . . 11
2.5 Implications . . . . . . . . . . . . . . . . . . . . . . . . . . . 12
2.6 Negations . . . . . . . . . . . . . . . . . . . . . . . . . . . . 13
2.7 Equivalences . . . . . . . . . . . . . . . . . . . . . . . . . . 14
2.8 Logical Connectives . . . . . . . . . . . . . . . . . . . . . . . 14
2.9 Ambiguities in Natural Language . . . . . . . . . . . . . . . . 15
2.10 Exercise Sheet . . . . . . . . . . . . . . . . . . . . . . . . . 15

**3 The Formal Syntax of Propositional Logic** **17**
3.1 Alphabets in Computer Science . . . . . . . . . . . . . . . . . 17
3.2 The Alphabet of Propositional Logic . . . . . . . . . . . . . . 18
3.3 Propositional Formulae . . . . . . . . . . . . . . . . . . . . . 18
3.4 Showing That a Word Is In $\mathbb{PL}$ . . . . . . . . . . . . . . . . . . 19
3.5 The Main Connective of a Formula . . . . . . . . . . . . . . . 20
3.6 Showing That a Word Is Not in $\mathbb{PL}$ . . . . . . . . . . . . . . . 20
3.7 Unique Readability . . . . . . . . . . . . . . . . . . . . . . . 21
3.8 Object-language and meta-language . . . . . . . . . . . . . . 22
3.9 Exercise Sheet . . . . . . . . . . . . . . . . . . . . . . . . . 23

**4 Functions Defined Recursively on $\mathbb{PL}$** **25**
4.1 The Abstract Syntax Tree of a Formula . . . . . . . . . . . . 26
4.2 Other Examples of Recursively Defined Functions . . . . . . . . 28
4.3 Proofs by Structural Induction . . . . . . . . . . . . . . . . . 29
4.4 Exercise Sheet . . . . . . . . . . . . . . . . . . . . . . . . . 31

**5 Semantics of Propositional Logic 33**
5.1 Boolean Algebra . . . 33
5.2 Assignments . . . 33
5.3 Truth Value of A Formula in An Assignment . . . 34
5.4 Satisfiability . . . 36
5.5 Valid Formulae . . . 37
5.6 Contingent Formulae . . . 37
5.7 Equivalence . . . 37
5.8 Semantical Consequence . . . 38
5.9 Consistent set of formulae . . . 39
5.10 Application 1 . . . 40
5.11 Exercise Sheet . . . 41

**6 Logical Connectives 45**
6.1 Several Propositional Logics . . . 45
6.2 The relation between implications and semantical consequence 46
6.3 *Translating* propositions from English into $\mathbb{PL}$ . . . 47
6.4 Application 2 . . . 49
6.5 Exercise sheet . . . 50

**7 Natural Deduction 51**
7.1 Sequents . . . 52
7.2 Inference Rules . . . 52
7.3 Proof system . . . 54
7.4 Formal Proofs . . . 54
7.5 Natural Deduction . . . 55
7.5.1 The Rules for Conjunction . . . 55
7.5.2 The Rules for Implications . . . 56
7.5.3 Rules for Disjunctions . . . 58
7.5.4 The Rules for Negations . . . 60
7.5.5 Alte reguli . . . 61
7.6 Natural Deduction . . . 63
7.7 Derived Rules . . . 63
7.8 Soundness and Completeness for Natural Deduction . . . 64
7.9 Exercise Sheet . . . 65

**8 Normal Forms 67**
8.1 Notations . . . 67
8.2 Replacement Theorem . . . 69
8.3 Literal . . . 69
8.4 Clauses . . . 69
8.5 Conjunctive Normal Form . . . 70
8.6 Bringing a Formula into CNF . . . 71

8.7 Disjunctive Normal Form . . . . . . . . . . . . . . . . . . . . . . . 72
8.8 The Link between DNF and CNF . . . . . . . . . . . . . . . . . . . 73
8.9 Exercise Sheet . . . . . . . . . . . . . . . . . . . . . . . . . . 74

**9 Resolution in Propositional Logic 75**
9.1 Clauses as Sets of Literals . . . . . . . . . . . . . . . . . . . . . 75
9.2 CNFs as Sets of Clauses . . . . . . . . . . . . . . . . . . . . . . 77
9.3 Resolution . . . . . . . . . . . . . . . . . . . . . . . . . . . . 77
9.4 Formal Proofs . . . . . . . . . . . . . . . . . . . . . . . . . . . 79
9.5 Soundness . . . . . . . . . . . . . . . . . . . . . . . . . . . . 80
9.6 Completeness . . . . . . . . . . . . . . . . . . . . . . . . . . . 81
9.7 Proving Validity and Logical Consequences by Resolution . . . 82
9.8 Exercise Sheet . . . . . . . . . . . . . . . . . . . . . . . . . . 84

# Chapter 1

# Introduction

If Arithmetic is the science that studies numbers and operations with numbers, then Logic is the science that studies *propositions* and operations with propositions.

For example, if in Arithmetic we notice that the sum of two even numbers is an even number, then in Logic we could notice that the disjunction of two true sentences is also true.

Logic sits at the intersection of philosophy, mathematics and computer science and has experienced its greatest development starting with the 1950s, because of its numerous applications in Computer Science.

In this course, we will study at an introductory level *propositional logic* and *first-order logic.*

Propositional logic is extremely simple, but the concepts that we study, the methods that we learn and the issues that we face in propositional logic generalize to other more complex logics. Moreover, propositional logic corresponds intimately to the internal organization of computers at an abstract level, in the sense that electronic circuits can be modeled as formulae in propositional logic. Propositional logic has a rich and mathematically interesting theory (examples: compactness theorem, Craig's interpolation theorem). The *satisfiability problem* for propositional logic has many applications in computer science. It is especially important both from a theoretical viewpoint (being the canonical NP-complete problem) and from a practical viewpoint as well (with applications in program verification, circuit verification, combinatorial optimization, and others).

First-order logic is an extension of propositional logic and also has numerous application in computer science, but also in mathematics. For example, all of the math that you have studied in highschool is based on a so-called first-order logic theory called **ZFC** (the **Z**ermelo–**F**raenkel set theory, together with the Axiom of **C**hoice). In Computer Science, applications of first-order logic appear in the fields of descriptive complexity, relational databases, software and hardware verification, and others. Additionally, several other logics (for example, higher-order logics) have applications in programming languages, the fundamentals of mathematics, type theory, etc.

# Chapter 2

# Informal Propositional Logic

*Propositional logic* is the logic of propositions, connected among themselves by *logical connectives* such as *or*, *and* and *not*. In this chapter, we study the basics of propositional logic.

## 2.1 Propositions

A *proposition* is a statement that is either true or false. Propositions are sometimes called *sentences*. Here are examples of propositions:

1. *I wear a blue shirt.*
2. *You own a laptop computer and a tablet computer, but no smartphone.*
3. $2 + 2 = 4$. (*Two plus two is four.*)
4. $1 + 1 = 1$. (*One plus one is one.*)
5. $1 + 1 \neq 1$. (*One plus one is not one.*)
6. *If* $1 + 1 = 1$, *then I'm a banana.*
7. *All natural numbers are integers.*
8. *All rational numbers are integers.*

Here are examples of things that are not propositions:

1. *Red and Black* (not a statement);

2. $\pi$ (not a statement);
3. *Is it raining?* (question, not a statement);
4. *Go fish!* (imperative);
5. *$x$ is greater than* 7 (we have here a *predicate of $x$*; once we set a value for $x$, the predicate becomes a proposition);
6. *This sentence is false.* (although a statement, it is not a proposition, since it is not either true or false: if it were true, it would need to be false and vice-versa).

Sometimes it is debatable whether something is truly a proposition. For example, we generally agree that *Snow is white* is true, but someone might argue that they have seen black snow, so the truth value of *Snow is white* is put in question. Arguing about whether something is a proposition or not is more a matter of philosophical logic than computer science logic and we will therefore not be too concerned about these sort of issues.

## 2.2 Atomic Propositions

Some propositions are atomic, in that they cannot be decomposed further into smaller propositions:

1. *I wear a blue shirt.*
2. *You own a laptop computer.*
3. $2 + 2 = 4$. (*Two plus two is four.*)

## 2.3 Conjunctions

Others however seem to be composed of smaller parts. For example, the proposition *I play games often and I study very well* is composed of two smaller propositions: *I play games often* and *I study very well*, joined together by *and*. When two propositions $\varphi$ and $\psi$ are joined by an *and*, the resulting proposition *$\varphi$ and $\psi$* is called a *conjunction* (*the conjunction of $\varphi$ and $\psi$*). The propositions $\varphi$ and $\psi$ are called the *conjuncts* of the proposition *$\varphi$ and $\psi$*.

A conjunction is true if both of its conjuncts are true. For example, the proposition *I play games often and I study very well.* is true if both *I play games often* and *I study very well* are true. In particular, as I do not play games often, this proposition is false (when I say it).

Note that a conjunction need not use explicitly the word *and*. For example, the proposition *It is raining outside, but I have an umbrella* is also a conjunction, and it conjuncts are *It is raining outside* and *I have an umbrella.* This particular conjunction uses the adversative conjunction *but.*

**Exercise 1.** *Find the conjuncts of* I play at home and I study at school..

**Exercise 2.** *Give an example of a conjunction that is false and an example of a conjunction that is true.*

## 2.4 Disjunctions

*Disjunctions* are propositions linked together by *or*. For example, *I can install the software on my smartphone or on my tablet* is a disjunction between *I can install the software on my smartphone* and *I can install the software on my tablet.* The two parts of the disjunction are called the *disjuncts.*

In the example above, note that the English grammar allows us to omit *I can install the software ...* in the second disjunct, as it is implicit in our understanding of the language. However, when we find the disjuncts, is helps to state them explicitly.

**Exercise 3.** *Find the disjuncts of* I will buy a laptop or a tablet. *Pay attention! The two disjuncts must be propositions (some words in them could be implicit and not appear in the text).*

A disjunction is true if at least one of the disjuncts is true. For example, *I am Darth Vader or I teach* is true because *I teach* is true (I do not have to worry about being Darth Vader). *I teach or I program* is also true (it happens that both disjuncts are true).

This meaning of disjunctions is called the *inclusive or.* It is standard in mathematics. Sometimes people use *or* in natural language to mean *exclusive or.* For example *Either white wins or black wins in a game of go* is an example where the *or* is exclusive. The meaning of the sentence is that *white wins* or *black wins*, but not both. Here is an example of a false proposition that uses *exclusive or*: *Either I program or I teach* (hint: false because I do both). When you see *either* in a sentence, it is a sign that you are dealing with an *exclusive or.*

In the following, by *disjunction* we will understand *inclusive or* (the standard interpretation in mathematics).

**Exercise 4.** *Give an example of a false disjunction and an example of a true disjunction.*

**Exercise 5.** *When is a disjunction $\varphi$* or *$\psi$ false (depending on the truth value of $\varphi$ and $\psi$)?*

## 2.5 Implications

*Implications* are propositions of the form *if $\varphi$ then $\psi$*. The proposition $\varphi$ is called the *antecedent* and the proposition $\psi$ is called the *conclusion* (or *consequent*) of the implication.

An example of an implication is *If I get a passing grade in Logic, I will buy everyone beer.* The antecedent is *I get a passing grade in Logic.* and the conclusion is *I will buy everyone beer.* When is an implication true? Actually, it is easier to say when it is false. An implication is false if and only if the antecedent is true, but the conclusion is false. Assume that I got a passing grade in Logic. Therefore, the proposition *I get a passing grade in Logic* is true. However, I will not buy beer for everyone (just a few select friends). Therefore the proposition *I will buy everyone beer* is false. Therefore the implication *If I get a passing grade in Logic, I will buy everyone beer* as a whole is false (antecedent is true, but conclusion is false).

The meaning of implications is worth a more detailed discussion as it is somewhat controversial. This is mostly because implication as we understand it in mathematics can sometimes be subtly different from implication as we understand it in everyday life. In everyday life, when we say *If I pass Logic, I buy beer*, we understand that there is a cause-and-effect relation between passing Logic and buying beer. This subtle cause-and-effect relation is evident in a number of *if-then* statements that we use in real life: *If I have money, I will buy a car*, *If you help me, I will help you*, etc. We would never connect two unrelated sentences with an implication: the proposition *If the Earth is round, then 2+2=4* would not be very helpful, even though it is true (both the antecedent and the conclusion are true).

This implication that we use in mathematics is called *material implication* or *truth functional implication*, because the truth value of the implication as a whole depends only on the truth values of the antecedent and the conclusion, not on the antecedent and the conclusion itself. This meaning of implication sometimes does not correspond to the meaning of natural language implications, but it turns out that it is the only sensible interpretation of implications in mathematics (and computer science).

In particular, we will take both the propositions *If the Earth is flat, then* $2+2=5$ and *If the Earth is flat, then* $2+2=4$ to be true, because the antecedent is false. Implications that are true because the antecedent is false are called *vacuously true.*

**Exercise 6.** *What are the truth values of* If $2+2=4$, then the Earth is flat *and* If $2+2=5$, then the Earth is flat*?*

The truth value of an implication *if $\varphi$ then $\psi$*, depending on the truth values of its antecedent $\varphi$ and its conclusion $\psi$, is summarized in the truth-table below:

| $\varphi$ | $\psi$ | if $\varphi$ then $\psi$ |
|---|---|---|
| *false* | *false* | *true* |
| *false* | *true* | *true* |
| *true* | *false* | *false* |
| *true* | *true* | *true* |

The following example aims at convincing you that the truth table above is the only reasonable one. You must agree that every natural number is also an integer. Otherwise put, the proposition *for any number $x$, if $x$ is a natural, then $x$ is an integer* is true. In particular, you will agree that the proposition above holds for $x = -10$, $x = 10$ and $x = 1.2$. In particular, the propositions *If* $-10$ *is a natural, then* $-10$ *is an integer*, *If* $10$ *is a natural, then* $10$ *is an integer* and *If* $1.2$ *is a natural, then* $1.2$ *is an integer* must all be true. This accounts for the first, second and fourth lines of the truth table above (typically, the second line is controversial). As for the third line, false is the only reasonable truth value for an implication *if $\varphi$ then $\psi$* where $\varphi$ is true but $\psi$ is false. Otherwise, we would have to accept propositions such as *If* $2 + 2 = 4$*, then* $2 + 2 = 5$ (antecedent $2 + 2 = 4$ true, conclusion $2 + 2 = 5$ false) as being true.

Implications are sometimes subtle to spot and identify correctly. For example, in the proposition *I will pass Logic only if I study hard* (emphasis on *only if*), the antecedent is *I will pass Logic* and the conclusion is *I study hard.* In particular, the above proposition does not have the same meaning as *If I study hard, then I will pass Logic.*

Pay attention! In propositions of the form *I will pass Logic only if I study*, the antecedent is *I will pass Logic*, and the consequent is *I study.* This proposition does not have the same meaning as *if I study, then I will pass Logic.*

Implications can sometimes not make use of *if.* For example, take the proposition *I will pass Logic or I will drop school* (apparently a disjunction). This most likely meaning of this proposition is *If I do not pass Logic, then I will drop school.* Thankfully, both of these propositions are equivalent, in a sense that we shall study in the following lectures.

## 2.6 Negations

A proposition of the form *it is not the case that $\varphi$* (or simply *not $\psi$*) is the *negation* of $\varphi$. For example, *It is not raining* is the negation of *It is raining.*

The negation of a proposition takes the opposite truth value. For example, as I am writing this text, the proposition *It is raining* is false, and therefore the proposition *It is not raining* is true.

**Exercise 7.** *Give an example of a false proposition that uses both a negation and a conjunction.*

## 2.7 Equivalences

A proposition of the form $\varphi$ *if and only if* $\psi$ is called an *equivalence* or *double implication.* Such a proposition, as a whole, is true if $\varphi$ and $\psi$ have the same truth value (both false or both true).

For example, when I am writing this text, *It is raining if and only if it is snowing* is true. Why? Because both of the propositions *It is raining* and *It is snowing* are false.

**Exercise 8.** *What is the truth value of the proposition* The number 7 is odd if and only if 7 is a prime.*?*

Equivalences are, semantically speaking, conjunctions of two implications: $\varphi$ *if and only if* $\psi$ gives the same information as

$$\underbrace{\varphi \textit{ if } \psi}_{\textit{the reverse implication}} \qquad \textit{and} \qquad \underbrace{\varphi \textit{ only if } \psi}_{\textit{the direct implication}} \quad .$$

The proposition $\varphi$ *if* $\psi$ is the same as *if* $\psi$*, then* $\varphi$ (but it has a different topic). The proposition $\varphi$ *only if* $\psi$ has the same meaning as *if* $\varphi$*, then* $\psi$, as we dicussed in the previous section on implications.

## 2.8 Logical Connectives

The words *and*, *or*, *if-then*, *not*, *only if*, *if-and-only-if* (and other similar phrases) are called *logical connectives*, as they can be used to connect smaller propositions in order to obtain larger propositions.

Pay attention! A proposition is *atomic* in propositional logic only if it cannot be decomposed into smaller propositions separated by the connectives discussed above. For example, the proposition *every natural number is an integer* is an atomic proposition (in propositional logic).

The same proposition is not necessarily atomic in other logics. For example, in first-order logic (which we study in the second half of the semester), we have additional logical connectives called *quantifiers* that can be used to construct *every natural number is an integer* from smaller propositions. Therefore, *every natural number is an integer* is not atomic in first-order logic.

## 2.9 Ambiguities in Natural Language

We have described above the language of propositional logic: atomic propositions connected by *and*, *or*, *not*, etc. So far, our approach has used English. However, English (or any other natural language) is not suitable for our purposes because it exhibits imprecisions in the form of ambiguitites.

Here are a few examples of ambigious propositions:

1. *John and Mary are married* (meaning 1: John and Mary are married to each other; meaning 2: John and Mary are married, but not necessarily to each other).

2. *It is not true that John is tall and Jane is short* (meaning 1: John is not tall and Jane is short; meaning 2: the conjunction *John is tall and Jane is short* is false).

In the study of the laws of logic, such ambiguities can get in the way, just like a wrong computation could impact the resistance of buildings or bridges in civil engineering.

The state of the art in Logic for over 2.000 years, from Aristotle up to the development of Symbolic Logic in the 19th century, has been to use natural language. Symbolic logic (formal logic) has changed the game by introducing languages so precise that there is no risk of misunderstandings.

Such ambiguities impede the study of propositional logic. Therefore we will design a *formal language*, the language of propositional logic, where no ambiguity can occur. The first such language that we study will be the language of *propositional logic*.

Normally, when people say *formal*, they mean it in a bad way, such as having to dress or act formally to go to dinner.

However, in computer science (and in mathematics), formal is a good thing: it means making things so precise that there is no possibility of misunderstanding.

## 2.10 Exercise Sheet

**Exercise 9.** *Establish which of the following phrases are propositions:*

*1.* You own a laptop computer.

*2.* Snow is white.

*3.* Snow is not white.

*4.* My father goes to work and I go to school.

*5.* It is raining outside, but I have an umbrella.

*6.* Either it will rain tomorrow, or it won't rain.

*7.* If I get a passing grade in Logic, I will celebrate.

*8.* $2 + 2 = 4$. *(*Two plus two is four.*)*

*9.* Red and Black.

*10.* $\pi$.

*11.* Is it raining?

*12.* Let's go fishing!

*13.* $x$ is greater than 7.

*14.* This sentence is false.

**Exercise 10.** *For all propositions that you identified, establish whether they are atomic or not. If not, establish if they are conjunctions, disjunctions, implications, negations, etc.).*

**Exercise 11.** *Determine whether the sentences below are atomic or not. If they are not atomic, identify their types (conjunctions, disjunctions, implications, negations, equivalences). For each such sentence, discuss their truth values based on the truth values of their components.*

*1.* I ate so much that I felt sick.

*2.* I have neither a dress nor shoes.

*3.* I'm going home or to my friend's place.

*4.* I'll go outside if it's not raining.

*5.* I'll go to him only if he doesn't listen to me.

*6.* If I don't go fishing, then the sun isn't round.

*7.* If the sun isn't round, then I'll go fishing.

**Exercise 12.** *Reformulate the examples in Section 2.9 so that ambiguities are avoided. You can choose any interpretation.*

# Chapter 3

# The Formal Syntax of Propositional Logic

Next, we will study the formal language of propositional logic.

By *syntax* we generally understand a set of rules for writing correctly. As mentioned previously, in this chapter we develop an artificial language, that we call *formal propositional logic* (or simply *propositional logic*). The formal syntax of propositional logic refers to the rules for writing correctly in this language.

## 3.1 Alphabets in Computer Science

A set is called an *alphabet* in computer science if we use the elements of the set to make up words. The elements of an alphabet are called *symbols*. Most of the time, an alphabet is a finite set (but not in the case of propositional logic).

What is the difference between an alphabet and a set? A priori, none. The intention matters – what we plan on doing with the elements of the set/alphabet. We use the elements of an alphabet to create *words*. A *word* over an alphabet is a sequence of symbols in the alphabet.

**Example 13.** *Consider the alphabet $X = \{\mathtt{0}, \mathtt{1}\}$. The strings/sequences of symbols* `001010`*,* `101011` *and* `1` *are examples of words of the alphabet $X$.*

The *empty* word, formed of 0 symbols of the alphabet, is usually denoted by $\epsilon$.

## 3.2 The Alphabet of Propositional Logic

The language of propositional logic is made of *propositional formulae*, which model propositions, as discussed in the previous chapter.

*Propositional formulae* are strings (sequences of characters) over the *alphabet of propositional logic.*

The alphabet of propositional logic is the union of the following sets:

1. $A = \{\mathtt{p}, \mathtt{q}, \mathtt{r}, \mathtt{p}', \mathtt{q}_1, \ldots\}$ is an infinite set of *propositional variables* that we fix from the very beginning;

2. $\{\neg, \wedge, \vee\}$ is the set of *logical connectives*;

3. $\{(, )\}$ is the set of auxiliary symbols; in our case, it consists of two symbols called *brakets.*

The set $L = A \cup \{\neg, \wedge, \vee\} \cup \{(, )\}$ is the alphabet of propositional logic. Here are a few examples of words over $L$:

1. $))\mathtt{p} \vee \wedge$;
2. $\vee \vee \neg(\mathtt{p})$;
3. $\mathtt{p}$;
4. $\mathtt{ppp}$;
5. $\neg(\mathtt{p} \vee \mathtt{q})$.

Words, or strings, are simply sequences of symbols of the alphabet $L$. Some of these words will be formulae or, equivalently, well-formed formulae (wff). Some authors prefer to use the terminology *wff*, but we will simply use *formula* by default in these lecture notes.

As an example, the last word above, $\neg(\mathtt{p} \vee \mathtt{q})$ is a formula of propositional logic, but $\vee \vee \neg(\mathtt{p})$ is not. The following definition captures exactly the set of propositional formulae.

## 3.3 Propositional Formulae

**Definition 14** (The Set of Propositional Formulae ($\mathbb{PL}$))**.** *The set of formulae of propositional logic, denoted $\mathbb{PL}$ from hereon, is the smallest set of words over $L$ satisfying the following conditions:*

- *Base Case. Any propositional variable, seen as a 1-symbol word, is in $\mathbb{PL}$ (equivalently, $A \subseteq \mathbb{PL}$);*

- *Inductive Step i. If $\varphi \in \mathbb{PL}$, then $\neg\varphi \in \mathbb{PL}$ (equivalently, if the word $\varphi$ is a propositional formula, then so is the word starting with the symbol $\neg$ and continuing with the symbols in $\varphi$);*
- *Inductive Step ii. If $\varphi_1, \varphi_2 \in \mathbb{PL}$, then $(\varphi_1 \land \varphi_2) \in \mathbb{PL}$ (equivalently, exercise);*
- *Inductive Step iii. If $\varphi_1, \varphi_2 \in \mathbb{PL}$, then $(\varphi_1 \lor \varphi_2) \in \mathbb{PL}$ (equivalently, exercise);*

Here are examples of elements of $\mathbb{PL}$:

$$\mathtt{p} \qquad \mathtt{q} \qquad \neg\mathtt{p} \qquad \neg\mathtt{q} \qquad \neg\mathtt{p}' \qquad \neg\neg\mathtt{p}_1 \qquad (\mathtt{p} \lor \mathtt{q}) \qquad (\mathtt{p} \land \mathtt{q})$$

$$\neg(\mathtt{p} \lor \mathtt{q}) \qquad (\neg\mathtt{p} \land \neg\mathtt{q}) \qquad \neg(\neg\neg\mathtt{p} \lor \mathtt{p}) \qquad ((\mathtt{p} \lor \mathtt{q}) \land \mathtt{r}) \qquad (\mathtt{p} \lor (\mathtt{q} \land \mathtt{r}))$$

Here are examples of words not in $\mathbb{PL}$:

$$\mathtt{pp} \qquad \mathtt{q}\neg\mathtt{q} \qquad \mathtt{q} \land \neg\mathtt{p} \qquad \mathtt{p} + \mathtt{q}$$

The definition of the set $\mathbb{PL}$ is an example of an *inductive definition*. Such definitions are really important in computer science and it is a must to understand them very well. In inductive definitions of sets, there are usually some base cases, which say what *base* elements are part of the set and some inductive cases, which explain how to obtain new elements of the set from old elements. Another important part of an inductive definition is the minimality constraint, which says that nothing other than what is provable by the base case(s) and the inductive case(s) belongs to the set. It can be shown that the above set $\mathbb{PL}$ exists and is unique, but the proof is beyond the scope of this course.

## 3.4 Showing That a Word Is In $\mathbb{PL}$

We can show that a word belongs to $\mathbb{PL}$ by explaining how the rules of the inductive definition were applied. Here is an example of a proof that $\neg(\mathtt{p} \lor \mathtt{q}) \in \mathbb{PL}$:

1. $\mathtt{p} \in \mathbb{PL}$ (by the Base Case, because $\mathtt{p} \in A$);
2. $\mathtt{q} \in \mathbb{PL}$ (by the Base Case, as $\mathtt{q} \in A$);
3. $(\mathtt{p} \lor \mathtt{q}) \in \mathbb{PL}$ (by the Inductive Case iii, with $\varphi_1 = \mathtt{p}$ and $\varphi_2 = \mathtt{q}$);
4. $\neg(\mathtt{p} \lor \mathtt{q}) \in \mathbb{PL}$ (by the Inductive Case i, with $\varphi = (\mathtt{p} \lor \mathtt{q})$).

We can rearrange the above into an equivalent *annotated construction tree* for the formula $\neg(\mathtt{p} \vee \mathtt{q})$:

$$\cfrac{\cfrac{\cfrac{}{\mathtt{p} \in \mathbb{PL}}\ \textit{Base Case} \qquad \cfrac{}{\mathtt{q} \in \mathbb{PL}}\ \textit{Base Case}}{(\mathtt{p} \vee \mathtt{q}) \in \mathbb{PL}}\ \textit{Inductive Case iii}}{\neg(\mathtt{p} \vee \mathtt{q}) \in \mathbb{PL}}\ \textit{Inductive Case i}$$

You will often see this notation in Computer Science and it is worth getting to know it. Each line is called an inference; below each line is the conclusion of the inference and above the lines are the hypotheses (0, 1 or more). Besides each line is the name of the rule that was applied.

If we drop all annotation, we obtain the following bare *construction tree* for the formula:

$$\cfrac{\cfrac{\cfrac{}{\mathtt{p}} \qquad \cfrac{}{\mathtt{q}}}{(\mathtt{p} \vee \mathtt{q})}}{\neg(\mathtt{p} \vee \mathtt{q})}$$

It is easy to see that a word belongs to $\mathbb{PL}$ iff there is a construction tree for it.

## 3.5 The Main Connective of a Formula

A formula that consists of a single propositional variable, such as $\mathtt{p}$ or $\mathtt{q}$, is called an *atomic formula*. This explains why the set $A$ of propositional variables is called $A$ ($A$ stands for *atomic*).

More complex formulae, such as $\neg\mathtt{p}$ or $\mathtt{p} \vee \mathtt{q}$, are called *molecular* (to distinguish them from atomic formulae).

Each molecular formula has a *main connective*, which is given by the last inference in its construction tree. For example, the main connective of the formula $\neg(\mathtt{p} \vee \mathtt{q})$ is $\neg$ (the negation), while the main connective of the fomrula $(\neg\mathtt{p} \vee \mathtt{q})$, is $\vee$ (the disjunction). We call formulae whose main connective is $\wedge$ *conjunctions*. Similarly, if the main connective of a formula is a $\vee$, it is a *disjunction* and if the main connective is $\neg$, then it is a *negation*.

## 3.6 Showing That a Word Is Not in $\mathbb{PL}$

It is somewhat more difficult (or rather more verbose) to show that a word is not in $\mathbb{PL}$.

If the word is not over the right alphabet, such as the three-symbol word $\mathtt{p} + \mathtt{q}$, which uses a foreign symbol $+$, then we can easily dismiss it as $\mathbb{PL}$ only contains words over $L$.

However, if the word is over the right alphabet and we want to show that it is not part of $\mathbb{PL}$, we must make use of the minimality condition. Here is an example showing that $(\mathtt{p}\neg\mathtt{q}) \notin \mathbb{PL}$:

Let's assume (by contradiction) that $(\mathtt{p}\neg\mathtt{q}) \in \mathbb{PL}$. Then, by the minimality condition, it follows that $(\mathtt{p}\neg\mathtt{q}) \in \mathbb{PL}$ must be explained by one of the rules Base Case, Inductive Case i – iii.

But we cannot apply the Base Case to show that $(\mathtt{p}\neg\mathtt{q}) \notin \mathbb{PL}$, since $(\mathtt{p}\neg\mathtt{q}) \notin A$.

But we cannot apply the Inductive Case i either, because there is no formula $\varphi$ such that $(\mathtt{p}\neg\mathtt{q}) = \neg\varphi$ (no matter how we would choose $\varphi \in \mathbb{PL}$, the first symbol of the word on the lhs would be $($, while the first symbol of the word on the rhs would be $\neg$).

But we cannot apply the Inductive Case ii either, because there are no formulae $\varphi_1, \varphi_2 \in \mathbb{PL}$ such that $(\mathtt{p}\neg\mathtt{q}) = (\varphi_1 \vee \varphi_2)$ (no matter how we would choose $\varphi_1, \varphi_2 \in \mathbb{PL}$, the word on the lhs would not contain the symbol $\neg$, while while the the word on the rhs would contain it).

By similar reasons, we cannot apply Inductive Case iii either.

But this contradicts our assumption and therefore it must be the case that $(\mathtt{p}\neg\mathtt{q}) \notin \mathbb{PL}$.

## 3.7 Unique Readability

The definition of $\mathbb{PL}$ has an important property called *unique readability*:

**Theorem 15** (Unique Readability of Propositional Formulae)**.** *For any word $\varphi \in \mathbb{PL}$, there is a unique construction tree.*

The property above is also sometimes called unambiguity of the grammar. Proving the Unique Readability Theorem is beyond the scope of the present lecture notes. Instead we will try to understand the importance of the property above by showing how an alternative, fictive, definition of the set $\mathbb{PL}$ would be ambiguous:

**Beginning of Wrong Definition of $\mathbb{PL}$.**

Imagine that the Inductive Case ii would read:

$\vdots$

3. (Inductive Step ii) If $\varphi_1, \varphi_2 \in \mathbb{PL}$, then $\varphi_1 \vee \varphi_2 \in \mathbb{PL}$.

$\vdots$

The only difference would be that we do not place parantheses around disjunctions. With this alternative, fictive, definition of $\mathbb{PL}$, we have that the word $\neg p \lor q \in \mathbb{PL}$. However, this word has two different construction trees:

$$\dfrac{\dfrac{\dfrac{}{p} \qquad \dfrac{}{q}}{p \lor q}}{\neg p \lor q} \qquad \dfrac{\dfrac{\dfrac{}{p}}{\neg p} \qquad \dfrac{}{q}}{\neg p \lor q}$$

Such an ambiguity would be very troubling, because we would not know if $\neg p \lor q$ is a disjunction (if we would use the construction tree on the right) or a negation (the construction tree on the left). In fact, avoiding such syntactic ambiguities was the main reason why we left natural language and began studying formal logic.

**End of Wrong Definition of $\mathbb{PL}$.**

I hope that you can now see that the Unique Readability Theorem is non-trivial (as an exercise to the more mathematically inclined students, I suggest that you try to prove it) and that it is a very important property of our definition of formulae, since it essential says that any propositional formula can be read unambiguously.

## 3.8 Object-language and meta-language

A peculiarity in logic is that we study propositions (we analyse, for example, their truth values), but in order to perform such study, we use a form of reasoning that also uses propositions. For example, in our study, we could make the following argument:

> *The proposition I do not go to school is false if the proposition I go to school is true.*

Note that the sentences that are the object of our study (*I do not go to school* and *I go to school*) are propositions, but so is the entire statement *The proposition I do not go to school is false if the proposition I go to school is true.* In this way, it seems that we are performing a circular reasoning, in which we study our own method of study.

This apparent difficulty does not occur in the case of arithmetic, for example. In arithmetic, we study numbers and operations over numbers, and we reason about numbers using propositions.

The difficulty can be solved by strictly separating *the object language* from the *meta-language*. The object language is the language that we study ($\mathbb{PL}$), and the meta-language is the language that we use to perform the study (English).

The object language is the language that represents the object of our study (in our case, propositional logic). The meta-language is the language that we use to communicate about the object of our study (in our case, English).

In this course, to more easily differentiate between the two, we make the following convention: all elements in the object language are written in sans-serif blue font (for example, $(\mathsf{p} \wedge \mathsf{q})$), and the statement in the meta-language are written using *a regular black font* (for example, *any atomic formulae is a formula*).

It is extremly important to differentiate correctly between object language and meta-language. To this effect, the lecture notes use a typographic convention. The elements of the meta-language are written in regular black font. The elements of the object-language are writte as follows: $(\mathsf{p} \wedge \mathsf{q})$. For this reason, if you print out the lecture notes, please print them in color.

## 3.9 Exercise Sheet

**Exercise 16.** *Show that the following words are* propositional formulae *(i.e., element of* $\mathbb{PL}$*), by explaining which are the construction steps (base case, respectively one of the three inductive steps):*

1. $\neg\mathsf{q}$;
2. $(\mathsf{p}_1 \wedge \mathsf{q})$;
3. $\neg(\mathsf{p} \vee \mathsf{q})$;
4. $(\neg\mathsf{p} \vee \neg\mathsf{q})$;
5. $\neg(\neg\mathsf{p} \vee (\mathsf{q} \wedge \neg\mathsf{q}))$.

**Exercise 17.** *Show that the following words over the alphabet $L$ are not elements of $\mathbb{PL}$ (hint: show that none of the four construction rules applies):*

1. $((\neg)\mathtt{q})$;
2. $\mathtt{q} \wedge \neg$;
3. $\mathtt{pq}$;
4. $\mathtt{p} \wedge \mathtt{q}$;
5. $(\mathtt{p}\neg\mathtt{q})$;
6. $(\mathtt{p}) \wedge (\mathtt{q})$.

**Exercise 18.** *Which of the following are formulae (in $\mathbb{PL}$) and which are not:*

1. $\mathtt{p}_1$;
2. $\mathtt{p}_1 \vee \mathtt{q}_1$;
3. $(\mathtt{p}_1 \vee \mathtt{q}_1)$;
4. $(\neg\mathtt{p}_1 \vee \mathtt{q}_1)$;
5. $\neg(\mathtt{p}_1 \vee \mathtt{q}_1)$;
6. $((\neg\mathtt{p}_1) \vee \mathtt{q}_1)$;
7. $(\neg\mathtt{p})$?

**Exercise 19.** *Please provide* 5 *examples of interesting formulae (with many variables/operators, etc.).*

# Chapter 4

# Functions Defined Recursively on $\mathbb{PL}$

**Reminder.** *Recall that when $X$ is a set, by $2^X$ we denote the* powerset of X, *that is, the set of all subsets of $X$. For example, if $X = \{1,2,3\}$, then $2^X = \{\emptyset, \{1\}, \{2\}, \{3\}, \{1,2\}, \{1,3\}, \{2,3\}, \{1,2,3\}\}$. When $X$ is finite, note that $|2^X| = 2^{|X|}$, which partly explains this notation. You might have used the notation $\mathcal{P}(X)$ instead of $2^X$ in highschool.*

Whenever a set is inductively defined, we can define recursive functions that operate on the elements of the set, function which make use of the *structure* of the elements.

Here is an example of a function computing the set of *subformulae* of a formula:

$\mathit{subf}\colon \mathbb{PL} \to 2^{\mathbb{PL}}$, defined by:

$$\mathit{subf}\Big(\varphi\Big) = \begin{cases} \{\varphi\}, & \text{if } \varphi \in A; \\ \{\varphi\} \cup \mathit{subf}\Big(\varphi'\Big), & \text{if } \varphi = \neg\varphi' \text{ and } \varphi' \in \mathbb{PL}; \\ \{\varphi\} \cup \mathit{subf}\Big(\varphi_1\Big) \cup \mathit{subf}\Big(\varphi_2\Big), & \\ \qquad \text{if } \varphi = (\varphi_1 \land \varphi_2) \text{ and } \varphi_1, \varphi_2 \in \mathbb{PL}; & \\ \{\varphi\} \cup \mathit{subf}\Big(\varphi_1\Big) \cup \mathit{subf}\Big(\varphi_2\Big), & \\ \qquad \text{if } \varphi = (\varphi_1 \lor \varphi_2) \text{ and } \varphi_1, \varphi_2 \in \mathbb{PL}. & \end{cases}$$

Here is how we can compute $subf\Big(\varphi\Big)$ when $\varphi = \neg(\mathrm{p} \vee \mathrm{q})$:

$$\begin{aligned}
subf\Big(\neg(\mathrm{p} \vee \mathrm{q})\Big) &= \{\neg(\mathrm{p} \vee \mathrm{q})\} \cup subf\Big((\mathrm{p} \vee \mathrm{q})\Big)\\
&= \{\neg(\mathrm{p} \vee \mathrm{q})\} \cup \{(\mathrm{p} \vee \mathrm{q})\} \cup subf\Big(\mathrm{p}\Big) \cup subf\Big(\mathrm{q}\Big)\\
&= \{\neg(\mathrm{p} \vee \mathrm{q})\} \cup \{(\mathrm{p} \vee \mathrm{q})\} \cup \{\mathrm{p}\} \cup \{\mathrm{q}\}\\
&= \{\neg(\mathrm{p} \vee \mathrm{q}), (\mathrm{p} \vee \mathrm{q}), \mathrm{p}, \mathrm{q}\}.
\end{aligned}$$

Notice that we use two different types of brackets in the computation above. On one hand, we have the brackets that surround the argument to the function *subf*, and on the other hand we have the usual brackets that are part of the alphabet $L$. The former brackets are the usual brakets in mathematics, while the later brackets are simply symbols in our alphabet. In order to differentiate between them, we adopt the convention to use bigger brakets for the function call and the usual brackets in blue for the auxiliary symbols.

Note that both are important. For example, it would be an error to write $subf\Big(\mathrm{p} \vee \mathrm{q}\Big)$ instead of $subf\Big((\mathrm{p} \vee \mathrm{q})\Big)$, as the word $\mathrm{p} \vee \mathrm{q} \notin \mathbb{PL}$ is not a formula.

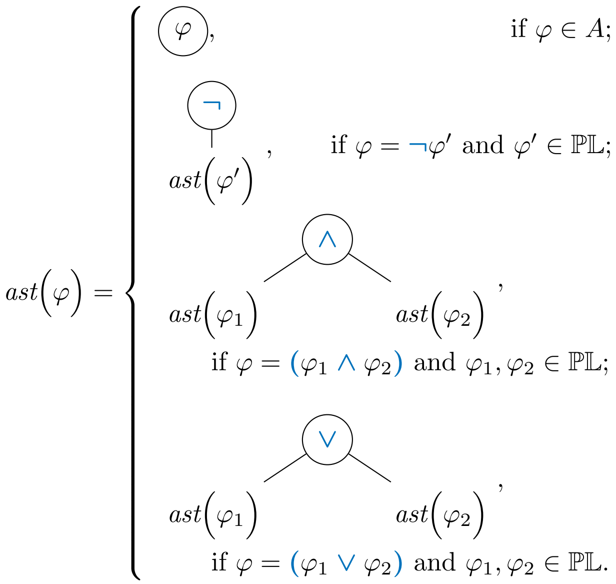
It is an error to write $subf\Big(\mathrm{p} \vee \mathrm{q}\Big)$ instead of $subf\Big((\mathrm{p} \vee \mathrm{q})\Big)$, as the word $\mathrm{p} \vee \mathrm{q} \notin \mathbb{PL}$ is not a formula.

The function *subf* is the first in a long line of functions defined recursively on the structure of a formula $\varphi \in \mathbb{PL}$. These functions are called *structurally recursive*. It is very important to understand how such functions operate in order to understand the remainder of this course. In the following sections we provide more examples of such functions.

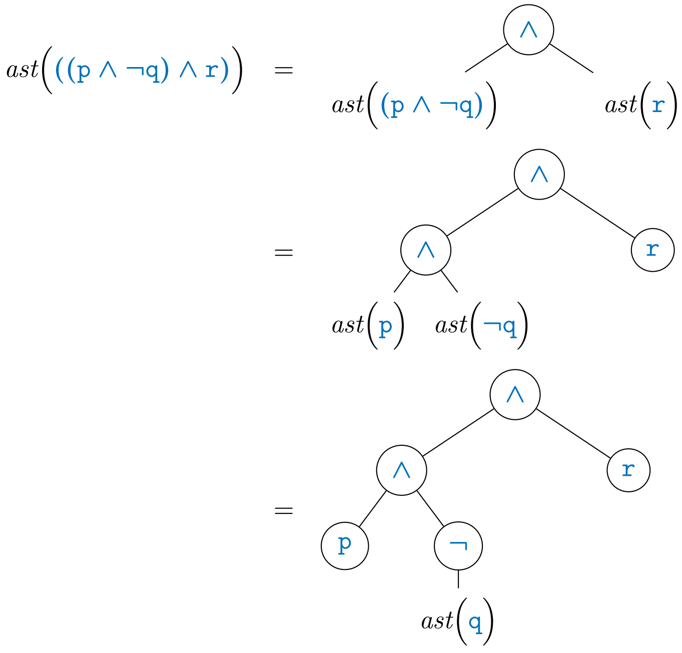
## 4.1 The Abstract Syntax Tree of a Formula

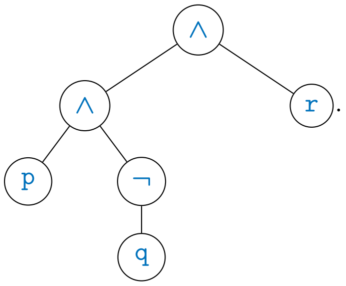
Here is an example of a function which computes the *abstract syntax tree* of a formula:

$ast : \mathbb{PL} \to Trees$, defined by:


$ast(\varphi) =$
$\varphi$, if $\varphi \in A$;
$\neg$, $ast(\varphi')$, if $\varphi = \neg\varphi'$ and $\varphi' \in \mathbb{PL}$;
$\wedge$, $ast(\varphi_1)$, $ast(\varphi_2)$, if $\varphi = (\varphi_1 \wedge \varphi_2)$ and $\varphi_1, \varphi_2 \in \mathbb{PL}$;
$\vee$, $ast(\varphi_1)$, $ast(\varphi_2)$, if $\varphi = (\varphi_1 \vee \varphi_2)$ and $\varphi_1, \varphi_2 \in \mathbb{PL}$.

Here is the computation of the abstract syntax tree of the formula $((\mathtt{p} \wedge \neg\mathtt{q}) \wedge \mathtt{r})$.



$=$ 

Note that *Trees* is the set of labeled, rooted trees.

The abstract syntax trees are very important because, conceptually, a propositional formula *is* its abstract syntax tree. Because writing trees is cumbersome for humans, we use the more conventional left-to-right notation. However, when implementing various algorithms as computer programs we prefer the tree representation, which is way more convenient.

Conceptually, a formula *is* its abstract syntax tree.

## 4.2 Other Examples of Recursively Defined Functions

Here are other examples of functions defined by structural recursion on formulae.

The first function, *height* : $\mathbb{PL} \rightarrow \mathbb{N}$, computes the *height* of the ast o formula:

$$
height(\varphi) = \begin{cases}
1, & \text{if } \varphi \in A;\\
1 + height(\varphi'), & \text{if } \varphi = \neg\varphi' \text{ and } \varphi' \in \mathbb{PL};\\
1 + max\big(height(\varphi_1), height(\varphi_2)\big), & \\
\quad \text{if } \varphi = (\varphi_1 \wedge \varphi_2) \text{ and } \varphi_1, \varphi_2 \in \mathbb{PL}; & \\
1 + max\big(height(\varphi_1), height(\varphi_2)\big), & \\
\quad \text{if } \varphi = (\varphi_1 \vee \varphi_2) \text{ and } \varphi_1, \varphi_2 \in \mathbb{PL}. &
\end{cases}
$$

The function $max : \mathbb{N} \times \mathbb{N} \rightarrow \mathbb{N}$, which occurs in the definition of *height*, is the well-known function that computes the maximum between two natural numbers.

The next function, $size : \mathbb{PL} \to \mathbb{N}$, computes the *size* of the abstract syntax tree of a formula (i.e., the number of nodes):

$$size\Big(\varphi\Big) = \begin{cases} 1, & \text{if } \varphi \in A; \\ 1 + size\Big(\varphi'\Big), & \text{if } \varphi = \neg\varphi' \text{ and } \varphi' \in \mathbb{PL}; \\ 1 + size\Big(\varphi_1\Big) + size\Big(\varphi_2\Big), & \\ \qquad \text{if } \varphi = (\varphi_1 \wedge \varphi_2) \text{ and } \varphi_1, \varphi_2 \in \mathbb{PL}; & \\ 1 + size\Big(\varphi_1\Big) + size\Big(\varphi_2\Big), & \\ \qquad \text{if } \varphi = (\varphi_1 \vee \varphi_2) \text{ and } \varphi_1, \varphi_2 \in \mathbb{PL}. & \end{cases}$$

The final example function, $prop : \mathbb{PL} \to 2^A$, compute the set of propositional variables occurring in a formula:

$$prop = \begin{cases} \{\varphi\}, & \text{if } \varphi \in A; \\ prop\Big(\varphi'\Big), & \text{if } \varphi = \neg\varphi' \text{ and } \varphi' \in \mathbb{PL}; \\ prop\Big(\varphi_1\Big) \cup prop\Big(\varphi_2\Big), & \\ \qquad \text{if } \varphi = (\varphi_1 \wedge \varphi_2) \text{ and } \varphi_1, \varphi_2 \in \mathbb{PL}; & \\ prop\Big(\varphi_1\Big) \cup prop\Big(\varphi_2\Big), & \\ \qquad \text{if } \varphi = (\varphi_1 \vee \varphi_2) \text{ and } \varphi_1, \varphi_2 \in \mathbb{PL}. & \end{cases}$$

Make sure that you are proficient at understanding, defining and computing with functions such as those given above.

## 4.3 Proofs by Structural Induction

We will sometimes do proofs by *structural induction*. You are already familiar with proofs by induction on the naturals.

In order to prove a theorem of the form

For all $n \in \mathbb{N}$, it is true that $P(n)$,

the mathematical induction principle postulates that it is sufficient to show that:

1. (Base Case) $P(0)$ holds;

2. (Induction Case) $P(k)$ implies $P(k+1)$, for all $k \in \mathbb{N}$.

Proofs by structural induction generalize the principle above to any inductively defined set such as $\mathbb{PL}$.

For the case of $\mathbb{PL}$, the structural induction principle is that, in order to show a theorem of the form

$$\text{For any propositional formula } \varphi \in \mathbb{PL}, \text{ it is true that } P\Big(\varphi\Big),$$

it is sufficient to prove that:

1. (Base Case) $P\Big(\varphi'\Big)$ holds for any atomic formula $\varphi' \in A$ (otherwise put, the property $P$ holds of every atomic formula);
2. (Inductive Case i) $P\Big(\neg\varphi'\Big)$ holds whenever $P\Big(\varphi'\Big)$ holds;
3. (Inductive Case ii) $P\Big((\varphi_1 \wedge \varphi_2)\Big)$ holds whenever $P\Big(\varphi_1\Big)$ and $P\Big(\varphi_2\Big)$ hold;
4. (Inductive Case iii) $P\Big((\varphi_1 \vee \varphi_2)\Big)$ holds whenever $P\Big(\varphi_1\Big)$ and $P\Big(\varphi_2\Big)$ hold.

Here is an example of a theorem and its proof by structural induction:

**Theorem 20** (Example of Theorem Provable by Structural Induction)**.** *For any propositional formula $\varphi$, we have that $height\Big(\varphi\Big) \leq size\Big(\varphi\Big)$.*

**Proof:** We proceed by structural induction (the property $P(\varphi)$ is simply that $height\Big(\varphi\Big) \leq size\Big(\varphi\Big)$). It is sufficient to show that:

1. (Base Case) $height\Big(\varphi'\Big) \leq size\Big(\varphi'\Big)$ for any propositional variable $\varphi' \in A$.

   But by definition, $height\Big(\varphi'\Big) = 1$ and $size\Big(\varphi'\Big) = 1$ when $\varphi' \in A$. And $1 \leq 1$, which is what we had to show.

2. (Inductive Case i) Assuming that $height\Big(\varphi'\Big) \leq size\Big(\varphi'\Big)$, we show that $height\Big(\neg\varphi'\Big) \leq size\Big(\neg\varphi'\Big)$.

   But by definition, $height\Big(\neg\varphi'\Big) = 1 + height\Big(\varphi'\Big)$ and $size\Big(\neg\varphi'\Big) = 1 + size\Big(\varphi'\Big)$. And $1 + height\Big(\varphi'\Big) \leq 1 + size\Big(\varphi'\Big)$ (what we had to show), as we already know $height\Big(\varphi'\Big) \leq size\Big(\varphi'\Big)$.

3. (Inductive Case ii) Assuming that $height\Big(\varphi_1\Big) \leq size\Big(\varphi_1\Big)$ and $height\Big(\varphi_2\Big) \leq size\Big(\varphi_2\Big)$, we show $height\Big((\varphi_1 \vee \varphi_2)\Big) \leq size\Big((\varphi_1 \vee \varphi_2)\Big)$.

   But, by definition, $height\Big((\varphi_1 \vee \varphi_2)\Big) = 1 + max\Big(height\Big(\varphi_1\Big), height\Big(\varphi_2\Big)\Big)$ and, as $max(a,b) \leq a+b$ (for any naturals $a, b \in \mathbb{N}$), we have that $height\Big((\varphi_1 \vee \varphi_2)\Big) \leq 1 + height\Big(\varphi_1\Big) + height\Big(\varphi_2\Big)$. But $height\Big(\varphi_i\Big) \leq size\Big(\varphi_i\Big)$ $(1 \leq i \leq 2)$ by our hypothesis, and therefore $height\Big((\varphi_1 \vee \varphi_2)\Big) \leq 1 + size\Big(\varphi_1\Big) + size\Big(\varphi_2\Big)$. But, by definition, $size\Big((\varphi_1 \vee \varphi_2)\Big) = 1 + size\Big(\varphi_1\Big) + size\Big(\varphi_2\Big)$, and therefore $height\Big((\varphi_1 \vee \varphi_2)\Big) \leq size\Big((\varphi_1 \vee \varphi_2)\Big)$, what we had to prove.

4. (Inductive Case iii) Similar to Inductive Case ii.

q.e.d.

## 4.4 Exercise Sheet

**Exercise 21.** *Compute, using the function subf, the set of subformulae of the following formulae:*

*1.* $((\mathtt{p} \wedge \neg\mathtt{q}) \wedge \mathtt{r})$; *2.* $((\mathtt{p} \vee \neg\mathtt{q}) \wedge \mathtt{r})$; *3.* $\neg((\mathtt{p} \vee \neg\mathtt{q}) \wedge \mathtt{r})$.

**Exercise 22.** *Compute the abstract syntax trees of the following formulae:*

*1.* $((\mathtt{p} \wedge \neg\mathtt{q}) \wedge \mathtt{r})$;

*2.* $((\mathtt{p} \vee \neg\mathtt{q}) \wedge \mathtt{r})$;

*3.* $\neg((\mathtt{p} \vee \neg\mathtt{q}) \wedge \mathtt{r})$;

*4.* $(\neg(\mathtt{p} \vee \neg\mathtt{q}) \wedge \mathtt{r})$.

**Exercise 23.** *Recall the recursive definition of the function height* $: \mathbb{PL} \to \mathbb{N}$, *which computes, given a formula, the height of its abstract syntax tree. Compute the height of the formulae shown in Exercise 22.*

**Exercise 24.** *Recall the recursive definition of the function size* $: \mathbb{PL} \to \mathbb{N}$, *which computes the number of nodes in abstract syntax tree of a formula. Compute the size of the formulae shown in Exercise 22.*

**Exercise 25.** *Recall the recursive definition of the function prop* $: \mathbb{PL} \to 2^A$, *which computes, given a formula, the set of propositional variables occuring in the formula. Compute the set of propositional variables occuring in the formulae shown in Exercise 22.*

**Exercise 26.** *Show by structural induction that* $height(\varphi) < size(\varphi) + 1$ *for any formula* $\varphi \in \mathbb{PL}$.

# Chapter 5

# Semantics of Propositional Logic

## 5.1 Boolean Algebra

The set $B = \{0, 1\}$ is called the set of boolean values (also called truth values). The value 0 denotes falsehood and the value 1 denotes truth.

The function $\bar{\ } : B \to B$ is called *logical negation* and is defined as follows: $\overline{0} = 1$ and $\overline{1} = 0$.

The function $+ : B \times B \to B$ is called *logical disjunction* and is defined as follows: $0 + 0 = 0, 0 + 1 = 1, 1 + 0 = 1, 1 + 1 = 1$.

The function $\cdot : B \times B \to B$ is called *logical conjunction* and is defined as follows: $0 \cdot 0 = 0, 0 \cdot 1 = 0, 1 \cdot 0 = 1, 1 \cdot 1 = 1$.

The tuple $(B, +, \cdot, \bar{\ })$ is a *boolean algebra*.

## 5.2 Assignments

A *truth assignment* (or simply *assignment* from hereon) is any function $\tau : A \to B$. In other words, an assignment is a function that associates to any propositional variable a truth value.

**Example 27.** *Let $\tau_1 : A \to B$ be a function defined as follows:*

1. $\tau_1(\mathtt{p}) = 1$;
2. $\tau_1(\mathtt{q}) = 0$;
3. $\tau_1(\mathtt{r}) = 1$;

4. $\tau_1(a) = 0$ *for all* $a \in A \setminus \{\mathtt{p}, \mathtt{q}, \mathtt{r}\}$.

*As it is a function from* $A$ *to* $B$, $\tau_1$ *is a truth assignment.*

**Example 28.** *Let* $\tau_2 : A \to B$ *be a function defined as follows:*

1. $\tau_2(\mathtt{p}) = 0$;
2. $\tau_2(\mathtt{q}) = 0$;
3. $\tau_2(\mathtt{r}) = 1$;
4. $\tau_2(a) = 1$ *for all* $a \in A \setminus \{\mathtt{p}, \mathtt{q}, \mathtt{r}\}$.

*As it is a function from* $A$ *to* $B$, $\tau_2$ *is also a truth assignment.*

**Example 29.** *Let* $\tau_3 : A \to B$ *be a function defined as follows:*

1. $\tau_3(a) = 0$ *for all* $a \in A$.

*As it is a function from* $A$ *to* $B$, $\tau_3$ *is also a truth assignment.*

## 5.3 Truth Value of A Formula in An Assignment

The truth value of a formula $\varphi$ in an assignment $\tau$ is denoted by $\hat{\tau}\left(\varphi\right)$ and is defined recursively as follows:

$$\hat{\tau}\left(\varphi\right) = \begin{cases} \tau(\varphi), & \text{if } \varphi \in A; \\ \overline{\hat{\tau}\left(\varphi'\right)}, & \text{if } \varphi = \neg\varphi' \text{ and } \varphi' \in \mathbb{PL}; \\ \hat{\tau}\left(\varphi_1\right) \cdot \hat{\tau}\left(\varphi_2\right), & \\ \quad \text{if } \varphi = (\varphi_1 \wedge \varphi_2) \text{ and } \varphi_1, \varphi_2 \in \mathbb{PL}; & \\ \hat{\tau}\left(\varphi_1\right) + \hat{\tau}\left(\varphi_2\right), & \\ \quad \text{if } \varphi = (\varphi_1 \vee \varphi_2) \text{ and } \varphi_1, \varphi_2 \in \mathbb{PL}. & \end{cases}$$

In fact $\hat{\tau} : \mathbb{PL} \to B$ is a function called *the homomorphic extension* of the assignment $\tau : A \to B$ to the entire set of formulae $\mathbb{PL}$, but do not feel obligated to recall this name.

Here is an example of how to compute the truth value of the formula $(\mathtt{p} \vee \mathtt{q})$ in the truth assignment $\tau_1$:

$\hat{\tau}_1\Big((p \vee q)\Big) = \hat{\tau}_1\Big(p\Big) + \hat{\tau}_1\Big(q\Big) = \tau_1(p) + \tau_1(q) = 1 + 0 = 1.$
We conclude that the truth value of $(p \vee q)$ in $\hat{\tau}_1$ is 1.

Here is another example, where we compute the truth value of the formula $\neg(p \wedge q)$ in the truth assignment $\tau_1$:

$$\hat{\tau}_1\Big(\neg(p \wedge q)\Big) = \overline{\hat{\tau}_1\Big((p \wedge q)\Big)} = \overline{\hat{\tau}_1\Big(p\Big) \cdot \hat{\tau}_1\Big(q\Big)} = \overline{\tau_1(p) \cdot \tau_1(q)} = \overline{1 \cdot 0} = \overline{0} = 1.$$

We conclude that the truth value of the formula $\neg(p \wedge q)$ in $\tau_1$ is 1.

Here is yet another example, where we compute the truth value of the formula $\neg\neg q$ in the truth assignment $\tau_2$:
$\hat{\tau}_2\Big(\neg\neg q\Big) = \overline{\hat{\tau}_2\Big(\neg q\Big)} = \overline{\overline{\hat{\tau}_2\Big(q\Big)}} = \overline{\overline{\tau_2(q)}} = \overline{\overline{0}} = \overline{1} = 0$. So the truth value of $\neg\neg q$ in $\tau_2$ is 0.

**Remark.** *It does not make sense to say "the truth value of a formula". It makes sense to say "the truth value of a formula in an assignment".*

*Also, it does not make sense to say "the formula is true" or "the formula is false". Instead, it makes sense to say "this formula is true/false in this assignment".*

*This is because a formula could be true in an assignment but false in another. For example, the formula $\neg\neg p$ is true in $\tau_1$, but false in $\tau_2$.*

In general, it does not make sense to say *the truth value of a formula* because a formula can be true for an assignment and false for another assignment. It only makes sense to say *the truth value of a formula in an assignment.*

**Definition 30** (Satisfaction)**.** *An* assignment $\tau$ satisfies $\varphi$ *if* $\hat{\tau}\Big(\varphi\Big) = 1$.

Instead of $\tau$ *satisfies* $\varphi$, we may equivalently say any of the following:

1. $\tau$ is a model of $\varphi$;

2. $\tau$ is true of $\varphi$;

3. $\varphi$ holds in/at $\tau$;

4. $\tau$ makes $\varphi$ true;

**Example 31.** *The assignment $\tau_1$ defined above is a model for $\neg(p \wedge q)$. The assignment $\tau_1$ is not a model of the formula $(\neg p \wedge q)$.*

We write $\tau \models \varphi$ (and we read: $\tau$ is a model of the formula $\varphi$; or: $\tau$ satisfies $\varphi$, etc.) iff $\hat{\tau}\Big(\varphi\Big) = 1$. We write $\tau \not\models \varphi$ (and we read: $\tau$ is not a model of the formula $\varphi$; or: $\tau$ does not satisfy $\varphi$) iff $\hat{\tau}\Big(\varphi\Big) = 0$.

**Definition 32** (The satisfaction relation (written $\models$))**.** *The relation $\models$, between assignments and formulae, is called the* satisfaction relation*. It is defined as follows: $\tau \models \varphi$ iff $\hat{\tau}\Big(\varphi\Big) = 1$.*

The satisfaction relation $\models$ is defined as: $\tau \models \varphi$ iff $\hat{\tau}\Big(\varphi\Big) = 1$.

## 5.4 Satisfiability

**Definition 33.** *A formula $\varphi$ is* satisfiable *if, by definition, there exists at least an assignment $\tau$ such that $\tau \models \varphi$ (i.e., if there exists at least a model of $\varphi$).*

**Example 34.** *The formula $(\mathrm{p} \vee \mathrm{q})$ is satisfiable, since it has a model (for example $\tau_1$ above).*

**Example 35.** *The formula $\neg\mathrm{p}$ is also satisfiable: for example, the assignment $\tau_3$ above makes the formula true.*

**Example 36.** *The formula $(\mathrm{p} \wedge \neg\mathrm{p})$ is not satisfiable, since it is false in any assignment.*

**Proof:** *Let us consider an arbitrary assignment $\tau : A \to B$.*

*We have that $\hat{\tau}\Big((\mathrm{p} \wedge \neg\mathrm{p})\Big) = \hat{\tau}\Big(\mathrm{p}\Big) \cdot \hat{\tau}\Big(\neg\mathrm{p}\Big) = \tau(\mathrm{p}) \cdot \overline{\hat{\tau}\Big(\mathrm{p}\Big)} = \tau(\mathrm{p}) \cdot \overline{\tau(\mathrm{p})}$. But $\tau(\mathrm{p})$ can be either 0 or 1:*

1. *in the first case ($\tau(\mathrm{p}) = 0$), we have that $\hat{\tau}\Big((\mathrm{p} \wedge \neg\mathrm{p})\Big) = \ldots = \tau(\mathrm{p}) \cdot \overline{\tau(\mathrm{p})} = 0 \cdot \overline{0} = 0 \cdot 1 = 0$;*

2. *in the second case ($\tau(\mathrm{p}) = 1$) , we have that $\hat{\tau}\Big((\mathrm{p} \wedge \neg\mathrm{p})\Big) = \ldots = \tau(\mathrm{p}) \cdot \overline{\tau(\mathrm{p})} = 1 \cdot \overline{1} = 1 \cdot 0 = 0$.*

*So, in both situations (i.e., $\tau(\mathrm{p})$ is either 0 or 1) we have $\hat{\tau}\Big((\mathrm{p} \wedge \neg\mathrm{p})\Big) = 0$. But $\tau$ was chosen arbitrarily, and therefore the result must hold for any* *assignment $\tau$: $(\mathtt{p} \wedge \neg \mathtt{p})$ is false in any assignment, which means that it is not satisfiable.*

*q.e.d.*

**Definition 37** (Contradiction). *A formula that is not satisfiable is called a* contradiction.

**Example 38.** *As we have seen above, $(\mathtt{p} \wedge \neg \mathtt{p})$ is a contradiction.*

## 5.5 Valid Formulae

**Definition 39.** *A formula $\varphi$ is* valid *if, by definition, any assignment $\tau$ has the property that $\hat{\tau}\Big(\varphi\Big) = 1$ (any assignment is a model of the formula).*

**Definition 40** (Tautology). *A valid formula is also called a* tautology.

**Example 41.** *The formula $(\mathtt{p} \vee \neg \mathtt{p})$ is valid, because it is true in any assignment: let $\tau$ be an arbitrary assignment; we have that $\hat{\tau}\Big((\mathtt{p} \vee \neg \mathtt{p})\Big) = \tau(\mathtt{p}) + \overline{\tau(\neg \mathtt{p})}$, which is either $0+1$ or $1+0$, which is $1$ in any case.*

**Notation.** *We sometimes write $\models \varphi$ instead of $\varphi$ is valid.*

**Example 42.** *The formula $\mathtt{p}$ is not valid (because there is an assignment (for example $\tau_3$) that makes it false).*

## 5.6 Contingent Formulae

**Definition 43.** *A formula that is neither a contradiction nor a tautology is called* contingent.

**Example 44.** *Examples of formulas of each type are shown here:*

1. *$(\mathtt{p} \wedge \neg \mathtt{p})$ is a contradiction.*

2. *$\mathtt{p}$ is contingent.*

3. *$(\mathtt{p} \vee \neg \mathtt{p})$ is a tautology.*

## 5.7 Equivalence

**Definition 45.** *We say that two formulae $\varphi_1, \varphi_2 \in \mathbb{PL}$ are* equivalent *and we write $\varphi_1 \equiv \varphi_2$ if, for any assignment $\tau : A \to B$, $\hat{\tau}\Big(\varphi_1\Big) = \hat{\tau}\Big(\varphi_2\Big)$.*

Intuitively, formulae that are equivalent have the same meaning (express the same thing).

**Example 46.** *At the start of the course, someone asked whether* $\mathsf{p}$ *and* $\neg\neg\mathsf{p}$ *are equal. Of course not, I said, since one has* 1 *symbol and the other* 3 *symbols from the alphabet.*

*However, we are now ready to understand the relation between them:* $\mathsf{p} \equiv \neg\neg\mathsf{p}$*. In other words, even if they are not equal, they are equivalent: they express the same thing.*

*To prove* $\mathsf{p} \equiv \neg\neg\mathsf{p}$*, we have to show they have the same truth value in any assignment. Let* $\tau$ *be an arbitrary assignment. We have that* $\hat{\tau}\Big(\neg\neg\mathsf{p}\Big) = \overline{\overline{\tau(\mathsf{p})}} = \tau(\mathsf{p}) = \hat{\tau}\Big(\mathsf{p}\Big)$*. In summary,* $\hat{\tau}\Big(\neg\neg\mathsf{p}\Big) = \hat{\tau}\Big(\mathsf{p}\Big)$*. As* $\tau$ *was chosen arbitrarily, it follows that* $\hat{\tau}\Big(\neg\neg\mathsf{p}\Big) = \hat{\tau}\Big(\mathsf{p}\Big)$ *for any assignment* $\tau$ *and therefore* $\mathsf{p}$ *is equivalent to* $\neg\neg\mathsf{p}$*.*

**Example 47.** *The following equivalence holds:* $(\mathsf{p} \vee \mathsf{q}) \equiv \neg(\neg\mathsf{p} \wedge \neg\mathsf{q})$ *(check it).*

Here are two more equivalences, known as De Morgan's laws:

**Theorem 48.** *For any formulae* $\varphi_1, \varphi_2 \in \mathbb{PL}$*, we have that:*

1. $\neg(\varphi_1 \vee \varphi_2) \equiv (\neg\varphi_1 \wedge \neg\varphi_2)$;
2. $\neg(\varphi_1 \wedge \varphi_2) \equiv (\neg\varphi_1 \vee \neg\varphi_2)$.

## 5.8 Semantical Consequence

**Definition 49.** *Let* $\Gamma = \{\varphi_1, \ldots, \varphi_n, \ldots\}$ *be a set of formulae. We say that* $\varphi$ *is a* semantical consequence *of* $\Gamma$ *and we write* $\Gamma \models \varphi$*, if is any model of all formulae in* $\Gamma$ *is a model of* $\varphi$ *as well.*

We also say that $\varphi$ is a logical consequence of $\Gamma$ or that $\varphi$ is a tautological consequence of $\Gamma$ instead of $\varphi$ is a semantical consequence of $\Gamma$.

**Example 50.** *Let* $\Gamma = \{\mathsf{p}, (\neg\mathsf{p} \vee \mathsf{q})\}$*. We have that* $\Gamma \models \mathsf{q}$*.*

*Indeed, let* $\tau$ *be a model of* $\mathsf{p}$ *and of* $(\neg\mathsf{p} \vee \mathsf{q})$*. As* $\tau$ *is a model of* $\mathsf{p}$*, by definition, we have that* $\tau(\mathsf{p}) = 1$*.*

*As* $\tau$ *is a model of* $(\neg\mathsf{p} \vee \mathsf{q})$*, it follows that* $\hat{\tau}\Big((\neg\mathsf{p} \vee \mathsf{q})\Big) = 1$*. But* $\hat{\tau}\Big((\neg\mathsf{p} \vee \mathsf{q})\Big)\overline{\tau(\mathsf{p})} + \tau(\mathsf{q})$*. But* $\tau(\mathsf{p}) = 1$*, and therefore* $\hat{\tau}\Big((\neg\mathsf{p} \vee \mathsf{q})\Big) = 0 + \tau(\mathsf{q}) = \tau(\mathsf{q})$*. This means that* $\tau(\mathsf{q}) = 1$*.*

*This means that $\tau$ is a model of $\mathtt{q}$. We assumed that $\tau$ is a model of $\mathtt{p}$ and of $(\neg\mathtt{p} \vee \mathtt{q})$ and we show that necessarily $\tau$ is a model of $\mathtt{q}$. But this is exactly the definition of $\{\mathtt{p}, (\neg\mathtt{p} \vee \mathtt{q})\} \models \mathtt{q}$ which is exactly what we want to prove.*

**Example 51.** *We have that $\mathtt{p}, (\mathtt{p} \vee \mathtt{q}) \not\models \neg\mathtt{q}$, that is $\neg\mathtt{q}$ is not a logical consequence of $\mathtt{p}, (\mathtt{p} \vee \mathtt{q})$. To show this "unconsequence", it is sufficient to find a model of $\mathtt{p}$ and $(\mathtt{p} \vee \mathtt{q})$ that is not a model of $\mathtt{q}$. Any assignment $\tau$ with $\tau(\mathtt{p}) = 1$ and $\tau(\mathtt{q}) = 0$ will do.*

**Notation.** *We sometimes write*

$$\varphi_1, \ldots, \varphi_n \models \varphi$$

*instead of*

$$\{\varphi_1, \ldots, \varphi_n\} \models \varphi.$$

**Remark.** *When $n = 0$, the notation above allows us to we write $\models \varphi$ instead of $\{\} \models \varphi$. This is consistent will the notation for validity, since a formula which is a logical consequence of the empty set is valid (and vice-versa).*

## 5.9 Consistent set of formulae

Another semantical notion is that occurs relatively often in practice and is strongly related to the $\wedge$ is the notion of *(in)consistent set of formulae.*

**Definition 52** (Consistent set of formulae)**.** *A set $\{\varphi_1, \varphi_2, \ldots, \varphi_n\}$ of formulae is* consistent *if there is an assignment $\tau$ that satisfies all $\varphi_i$ $(1 \leq i \leq n)$.*

**Remark.** *Important: the assignment $\tau$ must be the* same *for each formula in the set (not a different assignment $\tau$ for each formula $\varphi_i$, where $1 \leq i \leq n$)!*

A set of formulae is inconsistent if it is not consistent.

**Lemma 53** (The link between consistent sets and the logical connector $\wedge$)**.** *A set of formulae $\{\varphi_1, \varphi_2, \ldots, \varphi_n\}$ is consistent if and only if the formula*

$$(((\varphi_1 \wedge \varphi_2) \wedge \ldots) \wedge \varphi_n)$$

*is satisfiable.*

**Lemma 54** (The link between inconsistent sets and the logical connector $\wedge$)**.** *A set of formulae $\{\varphi_1, \varphi_2, \ldots, \varphi_n\}$ is inconsistent if and only if the formula*

$$(((\varphi_1 \wedge \varphi_2) \wedge \ldots) \wedge \varphi_n)$$

*is not satisfiable.*

**Exercise 55.** *Show that, for any formula $\varphi$, if $\{\varphi_1, \varphi_2, \ldots, \varphi_n\}$ is inconsistent then*

$$\{\varphi_1, \varphi_2, \ldots, \varphi_n\} \models \varphi.$$

**Exercise 56.** *Show that if*

$$\{\varphi_1, \varphi_2, \ldots, \varphi_n\} \models (\mathtt{p} \land \neg \mathtt{p}),$$

*then $\{\varphi_1, \varphi_2, \ldots, \varphi_n\}$ is inconsistent.*

## 5.10 Application 1

John writes the following code:

```
if (((y % 4 == 0) && (y % 100!= 0)) || (y%400 == 0))
   printf("%d is a leap y", y);
else
   printf("%d is not a leap y", y);
```

Jill simplifies the code:

```
if (((y % 4 != 0) || (y % 100 == 0)) && (y%400 != 0))
   printf("%d is not a leap y", y);
else
   printf("%d is a leap y", y);
```

Is Jill right? It is difficult to tell just by looking at the code, but we can use our logic-fu to model the problem above and to determine whether the two programs behave in the same manner.

First of all, we will “translate” the conditions in the if-else statements into propositional logic. Supposed the value of `y` is fixed. We identify the “atomic propositions” and we replace them with propositional variables as follows:

1. the propositional variable $\mathtt{p}$ will stand for `(y % 4 == 0)`;

2. the propositional variable $\mathtt{q}$ will stand for `(y % 100 = 0)`;

3. the propositional variable $\mathtt{r}$ will stand for `(y % 400 == 0)`.

Taking into account the translation key above, we can see that John's condition is, in propositional logic speak, $((\mathtt{p} \land \neg \mathtt{q}) \lor \mathtt{r})$.

Jill's formula is, in propositional logic speak,$((\neg \mathtt{p} \lor \mathtt{q}) \land \neg \mathtt{r})$.

Also notice that the branches in the two programs are reversed (the if branch of John's program is the else branch of Jill's program and vice-versa). In order for the two programs to have the same behaviour, it is sufficient for the negation of John's formula to be equivalent to Jill's formula. Is this the case? I.e., is it the case that

$$\neg((\mathtt{p} \wedge \neg\mathtt{q}) \vee \mathtt{r}) \equiv ((\neg\mathtt{p} \vee \mathtt{q}) \wedge \neg\mathtt{r})?$$

By applying De Morgan's laws, we can see that the equivalence does hold and therefore the two programs have the same behaviour. So Jill's changes do not break the program – they are correct.

## 5.11 Exercise Sheet

**Exercise 57.** *Let $\tau : A \to B$ be the truth assignment defined as follows: $\tau(\mathtt{p}) = 1$, $\tau(\mathtt{q}) = 0$, $\tau(\mathtt{r}) = 0$, $\tau(a) = 0$ for any other propositional variable $a \in A \setminus \{\mathtt{p}, \mathtt{q}, \mathtt{r}\}$.*

*Find the truth value of the following formulae in the assignment $\tau$:*

1. $(\mathtt{p} \wedge \mathtt{q})$;
2. $(\mathtt{q} \wedge \mathtt{p})$;
3. $\neg\mathtt{q}$;
4. $(\neg\mathtt{q} \wedge \mathtt{r})$;
5. $((\neg\mathtt{q} \wedge \mathtt{r}) \vee \neg\mathtt{p})$.

**Exercise 58.** *Find an assignment $\tau$ that is a model for the following formulae (one assignment for each formula):*

1. $(\mathtt{p} \wedge \mathtt{q})$;
2. $(\mathtt{p} \wedge \neg\mathtt{q})$;
3. $((\mathtt{p} \wedge \neg\mathtt{q}) \vee \mathtt{q})$.

*Is there a single assignment that makes all these formulas true?*

**Exercise 59.** *Find an assignment $\tau$ in which the following formulae are false (one assignment per formula):*

1. $(\mathtt{p} \vee \mathtt{q})$;
2. $(\mathtt{q} \wedge (\mathtt{p} \vee \neg\mathtt{q}))$;
3. $((\mathtt{p} \wedge \neg\mathtt{q}) \vee \mathtt{q})$.

**Exercise 60.** *Which of the following formulae are satisfiable?*

1. $(\mathtt{p} \land \neg\mathtt{p})$;
2. $(\mathtt{p} \lor \neg\mathtt{p})$;
3. $((\mathtt{p} \lor \neg\mathtt{p}) \land \neg\mathtt{q})$;
4. $((\mathtt{p} \lor \neg\mathtt{p}) \land (\neg\mathtt{p} \land \mathtt{q}))$;
5. $((\mathtt{p} \lor \neg\mathtt{q}) \land (\neg\mathtt{p} \lor \mathtt{r}))$.

**Exercise 61.** *Which of the following formulae are valid?*

1. $(\mathtt{p} \land \neg\mathtt{p})$;
2. $(\mathtt{p} \lor \neg\mathtt{p})$;
3. $\mathtt{p}$;
4. $((\mathtt{p} \lor \neg\mathtt{p}) \land \neg\mathtt{q})$;
5. $(\mathtt{p} \rightarrow \neg\mathtt{p})$;
6. $((\mathtt{p} \land \mathtt{q}) \lor (\neg\mathtt{p} \land \mathtt{r}))$.

**Exercise 62.** *Give* 5 *examples of contradictions.*

**Exercise 63.** *Give* 5 *examples of tautologies.*

**Exercise 64.** *Prove that, for any formulae $\varphi_1, \varphi_2, \varphi_3 \in \mathbb{PL}$, the following equivalences hold:*

1. $(\varphi_1 \land (\varphi_2 \land \varphi_3)) \equiv ((\varphi_1 \land \varphi_2) \land \varphi_3)$;
2. $(\varphi_1 \land \varphi_2) \equiv (\varphi_2 \land \varphi_1)$;
3. $(\varphi_1 \lor (\varphi_2 \lor \varphi_3)) \equiv ((\varphi_1 \lor \varphi_2) \lor \varphi_3)$;
4. $(\varphi_1 \lor \varphi_2) \equiv (\varphi_2 \lor \varphi_1)$;
5. $(\neg(\neg\varphi_1)) \equiv \varphi_1$;
6. $(\neg(\varphi_1 \land \varphi_2)) \equiv ((\neg\varphi_1) \lor (\neg\varphi_2))$;
7. $(\neg(\varphi_1 \lor \varphi_2)) \equiv ((\neg\varphi_1) \land (\neg\varphi_2))$.

**Exercise 65.** *Can you prove that, for any formulae $\varphi_1, \varphi_2 \in \mathbb{PL}$, $(\varphi_1 \lor \varphi_2) \equiv \varphi_1$ if and only if $\varphi_1 \in \mathbb{PL}$ is a tautology?*

**Exercise 66.** *Can you prove that, for any formulae $\varphi_1, \varphi_2 \in \mathbb{PL}$, $(\varphi_1 \land \varphi_2) \equiv \varphi_2$ if and only if $\varphi_1 \in \mathbb{PL}$ is a tautology?*

**Exercise 67.** *Can you prove that, for any formulae* $\varphi_1, \varphi_2 \in \mathbb{PL}$, $(\varphi_1 \land \varphi_2) \equiv \varphi_1$ *if and only if* $\varphi_1 \in \mathbb{PL}$ *is a contradiction?*

**Exercise 68.** *Can you prove that, for any formulae* $\varphi_1, \varphi_2 \in \mathbb{PL}$, $(\varphi_1 \lor \varphi_2) \equiv \varphi_2$ *if and only if* $\varphi_1 \in \mathbb{PL}$ *is a contradiction?*

**Exercise 69.** *Show that* $\neg \mathtt{p}$ *is a semantical consequence of* $(\neg \mathtt{p} \lor \neg \mathtt{p})$.

**Exercise 70.** *Show that* $\mathtt{p}$ *is not a semantical consequence of* $(\neg \mathtt{q} \lor \mathtt{p})$.

**Exercise 71.** *Show that* $\mathtt{p}$ *is a semantical consequence of* $(\neg \mathtt{q} \lor \mathtt{p})$ *and* $\mathtt{q}$.

**Exercise 72.** *Show that* $\mathtt{p}_3$ *is a logical consequence of the formulae* $(\neg \mathtt{p}_1 \lor (\mathtt{p}_2 \lor \mathtt{p}_3))$, $((\neg\neg \mathtt{p}_2 \lor \neg \mathtt{p}_4) \land (\neg\neg \mathtt{p}_4 \lor \neg \mathtt{p}_2))$ *and* $(\mathtt{p}_1 \land \neg \mathtt{p}_4)$.

# Chapter 6

# Logical Connectives

There are two more important connectives in propositional logic: the conditional (implication) and the equivalence (double implication).

We use the syntax $(\varphi_1 \rightarrow \varphi_2)$ for an implication and $(\varphi_1 \leftrightarrow \varphi_2)$ for a double implication.

The semantics of an implication $(\varphi_1 \rightarrow \varphi_2)$ is given by

$$\hat{\tau}\Big((\varphi_1 \rightarrow \varphi_2)\Big) = \overline{\hat{\tau}\Big(\varphi_1\Big)} + \hat{\tau}\Big(\varphi_2\Big),$$

for any truth assignment $\tau$.

It is easy to see that $(\varphi_1 \rightarrow \varphi_2) \equiv (\neg\varphi_1 \vee \varphi_2)$.

Similarly, the semantics of a double implication $(\varphi_1 \leftrightarrow \varphi_2)$ is defined such that

$$(\varphi_1 \leftrightarrow \varphi_2) \equiv ((\varphi_1 \rightarrow \varphi_2) \wedge (\varphi_2 \rightarrow \varphi_1)).$$

**Example 73.** *We have that* $(\mathtt{p} \rightarrow \mathtt{p})$ *is a valid formula. Why? The formula* $(\mathtt{p} \rightarrow \mathtt{p})$ *is equivalent to* $(\neg\mathtt{p} \vee \mathtt{p})$*, and this formulae is obviously valid.*

## 6.1 Several Propositional Logics

Up to this point, we have studied the propositional logic of the connectives $\neg, \wedge, \vee$, and we have denoted its set of formulae by $\mathbb{PL}$. In fact, depending on the set of logical connectives that we need, there are several propositional logics. The logic that we have studied up to this point is $\mathbb{PL}_{\neg,\wedge,\vee} = \mathbb{PL}$.

Depending on the logical connectives that are allowed, we can obtain other interesting propositional logics:

1. $\mathbb{PL}_{\neg,\vee}$ is the propositional logic that allows as connectives $\neg$ and $\vee$.

2. $\mathbb{PL}_{\bot,\rightarrow}$ is the propositional logic in which that only connectives are $\bot$ (an arity 0 connective) and $\rightarrow$. The formula $\bot$ (read as: *bottom*) is false in any truth assignment.

3. $\mathbb{PL}_{\vee,\wedge}$ is a logic in which the only connectives are $\vee$ and $\wedge$.

**Exercise 74.** *Write the definition of the formal syntax for each of the logics above.*

What do $\mathbb{PL}, \mathbb{PL}_{\neg,\vee}, \mathbb{PL}_{\bot,\rightarrow}$ have in common? They are *equiexpressive* (equally expressive). This means that for any formula $\varphi \in \mathbb{PL}$ there is a formula $\varphi' \in \mathbb{PL}_{\neg,\vee}$ so that $\varphi \equiv \varphi'$ (and vice-versa, for any formula $\varphi' \in \mathbb{PL}_{\neg,\vee}$ there is an equivalent formula in $\mathbb{PL}$).

How can we show that $\mathbb{PL}$ and $\mathbb{PL}_{\neg,\vee}$ are equally expressive? For one of the directions, it is sufficient to translate all conjunctions in $\mathbb{PL}$ as follows:

$$(\varphi \wedge \varphi') \equiv \neg(\neg\varphi \vee \neg\varphi').$$

At the end we will obtain an equivalent formula that does not contain conjunctions (and is therefore in $\mathbb{PL}_{\neg,\vee}$). Vice-versa, any formula in $\mathbb{PL}_{\neg,\vee}$ is already a formula in $\mathbb{PL}$.

How can we show that $\mathbb{PL}_{\neg,\vee}$ and $\mathbb{PL}_{\bot,\rightarrow}$ are equally expressive?

We translate all disjunctions and negations using the following equivalences:

1. $(\varphi \vee \varphi') \equiv (\neg\varphi \rightarrow \varphi')$;

2. $\neg\varphi \equiv (\varphi \rightarrow \bot)$.

The translation operation stops after a finite number of steps and the result is a formula equivalent to the starter formula, but which uses only the connectives $\bot$ and $\rightarrow$.

The logic $\mathbb{PL}_{\vee,\wedge}$ is strictly less expressive. For example, in this logic there are no unsatisfiable formulae.

**Exercise 75.** *Explain why in* $\mathbb{PL}_{\vee,\wedge}$ *all formulae are satisfiable.*

**Remark.** *By* propositional logic *we understand any logic that is as expressive as* $\mathbb{PL}$*. For example,* $\mathbb{PL}_{\neg,\wedge}$*,* $\mathbb{PL}_{\neg,\vee}$*,* $\mathbb{PL}_{\neg,\rightarrow}$ *are all propositional logics, but* $\mathbb{PL}_{\neg}$ *and* $\mathbb{PL}_{\wedge,\vee}$ *are not propositional logics (they are less expressive).*

## 6.2 The relation between implications and semantical consequence

There is a strong link between implications and the concept of semantical consequence, link which is formalized in the following theorem.

**Theorem 76** (The relation between implications and logical consequences)**.** *For any two formulae $\varphi_1, \varphi_2 \in \mathbb{PL}$, we have that $\varphi_1 \models \varphi_2$ if an only if the formula $(\varphi_1 \rightarrow \varphi_2)$ is valid.*

The following more general theorem also holds:

**Theorem 77** (The generalized relation between implications and logical consequence)**.** *For any formulae $\varphi_1, \varphi_2, \ldots, \varphi_n, \varphi \in \mathbb{PL}$, we have that $\varphi_1, \varphi_2, \ldots, \varphi_n \models \varphi$ if and only if the formula $(((\varphi_1 \land \varphi_2) \land \ldots) \land \varphi_n) \rightarrow \varphi$ is valid.*

A similar relation exists between the double implication logical connective and the concept of semantical equivalence:

**Theorem 78** (The relation between double implication and semantic equivalence)**.** *For any two formulae $\varphi_1, \varphi_2 \in \mathbb{PL}$, we have that $\varphi_1 \equiv \varphi_2$ if and only if the formula $(\varphi_1 \leftrightarrow \varphi_2)$ is valid.*

## 6.3 *Translating* propositions from English into $\mathbb{PL}$

By *translation* we understand modeling an English proposition as a formula of propositional logic. The purpose of this modeling could be: clarifying the meaning of a proposition by eliminating possible syntactical ambiguities, checking whether the proposition is valid, etc.

To *translate* propositions from English into propositional logic, we should perform the following steps:

1. Identifying the atomic propositions and associating propositional variables to them;

2. Identifying the logical connectives and their relative order.

For example, let us *translate* the proposition *I want to learn Logic and pass the examination if the subject is interesting* into propositional logic.

The first step is to identify the atomic propositions. In our case, we find three atomic propositions:

1. *I want to learn Logic*;

2. *I want to pass the examination*; (note that the words *I want* do not appear explicitly in the proposition, but they are *implied*)

3. *the subject is interesting.*

Pay attention! The connectives themselves are not part of the atomic propositions. For example, the third atomic proposition is not *if the subject is interesting.*

We associate a propositional variable to each atomic proposition:

| propositional variable | atomic proposition |
|---|---|
| p | *I want to learn Logic* |
| q | *I want to pass the examination* |
| r | *the subject is interesting.* |

Pay attention! For an accurate *translation*, if a proposition occurs several times (even if it does not use exactly the same words), we should associate to all of its occurences the same propositional variable.

The next step is to identify the logical connectives. In the example above, there are two logical connectives:

1. the connective *and* in the context *[...] Logic and pass [...]*, which indicates a conjunction;

2. the connective *if-then* (even if the word *then* does not appear explicitly) in the context *[...] the examination if the subject [...]*, which indicates an implication.

What is the order of the two connectives? In other words, it the entire proposition a conjunction or an implication? What is the main connective? Our English-language intuition tells us that it is more likely an implication (for example, because the verb *want* is implied (not explicit) in the proposition *I want to pass the examination*). The *translation* is therefore

$$(\mathtt{r} \rightarrow (\mathtt{p} \wedge \mathtt{q})),$$

because the proposition associated to the propositional variable $\mathtt{r}$ is the antecedent of the implication, even if syntactically it appears at the end of the sentence.

Pay attention! In spite of the name, *propositional variables* are not variables in a mathematical sense. A frequent mistake is to write

$$\texttt{p} = \textit{I want to learn Logic},$$

which is not correct, because $\texttt{p}$ is only equal to $\texttt{p}$ and nothing else. Any variable (in a mathematical sense) in the lecture is written with a regular black font. For example, we often use the mathematical variable $\varphi$ for propositional formulae.

**Exercise 79.** *Translate the following proposition into propositional logic:* Either I pass Logic, or I don't.

*Pay attention to the words that do not explictly appear in the atomic propositions. Pay attention to the meaning of the connective* either-or *(hint: it is not among the connectives that we discussed so far, but it can be* emulated/simulated*).*

## 6.4 Application 2

The following puzzle is from the book *Peter Smith. An Introduction to Formal Logic.* Either the butler or the cook committed the murder. The victim died from poison if the cook did the murder. The butler did the murder only if the victim was stabbed. The victim didn't die from poison.

Does it follow that the victim was stabbed?

We associate to every atomic proposition a propositional variable as follows:

1. For the proposition "the butler commited the murder" the propositional variable $\texttt{p}$;

2. For the proposition "the cook commited the murder" the propositional variable $\texttt{q}$;

3. For the proposition "the victim died of posion" the propositional variable $\texttt{r}_1$;

4. For the proposition "the victim was stabbed" the propositional variable $\texttt{r}_2$.

The hypotheses of the puzzle are modeled as propositional formulae in propositional logic as follows:

1. $((\mathtt{p} \lor \mathtt{q}) \land \neg(\mathtt{p} \land \mathtt{q}))$ (exclusive disjunction between $\mathtt{p}$ and $\mathtt{q}$);
2. $(\mathtt{q} \rightarrow \mathtt{r}_1)$;
3. $(\mathtt{p} \rightarrow \mathtt{r}_2)$ (mind the direction of the implication);
4. $\neg\mathtt{r}_1$.

The question of the puzzle is simply the formula $\mathtt{r}_2$.
To answer the puzzle with yes or no, it is sufficient to check whether

$$\Big\{((\mathtt{p} \lor \mathtt{q}) \land \neg(\mathtt{p} \land \mathtt{q})), (\mathtt{q} \rightarrow \mathtt{r}_1), (\mathtt{p} \rightarrow \mathtt{r}_2), \neg\mathtt{r}_1\Big\} \models \mathtt{r}_2.$$

The logical consequences does hold (left as an exercise to the reader).

## 6.5 Exercise sheet

**Exercise 80.** *Associate to each of the following statements a formula in $\mathbb{PL}$ that models its meaning in English.*

1. *If it rains outside, I stay inside or go to the mall. I don't stay inside unless I'm bored. It rains and I'm not bored.*
2. *I study logic only if it is not possible to go outside. It is possible to go outside if it is not raining and if it is hot. As I am not studying logic and outside is hot, it means it is raining.*
3. *Thing go well in the country if the leaders of the country are not thiefs and if the economy is healthy. People go abroad if and only if things do not go well in the country. The economy is healthy, but people leave abroad.*

# Chapter 7

# Natural Deduction

In the previous lecture we have discussed some important notions about the *semantics* of propositional logic.

1. the truth value of a formula in an assignment;
2. satisfiability;
3. validity;
4. equivalence;
5. semantical consequence.

We have seen that in order to establish that two formulae are equivalent, a semantical reasoning process is necessary (meaning an argument that uses semantical notions such as truth values, assignments etc).

One of the main purposes of logic in computer science is to design mechanical models (meaning models that are suitable to computer implementation) for reasoning instead of semantical-based reasoning.

In this lecture, we will study a method for mechanizing the notion of semantical consequence. By mechnization, we understand a method for proving consequeneces with the following properties:

1. we should be able to convince someone that the consequence does take place, without the person having to following a semantical argument;
2. in particular, each proof step should be easy to check (mechanically, no RI needed);
3. the method must be so precise as to allow the reasoning to be performed by a computer.

## 7.1 Sequents

**Definition 81** (Sequent)**.** *A* sequent *is a pair consisting of a set of formulae* $\{\varphi_1, \ldots, \varphi_n\}$ *and a formula* $\varphi$*, pair denoted by*

$$\{\varphi_1, \ldots, \varphi_n\} \vdash \varphi.$$

The word *sequent* does not really exist in English. It is borrowed from the German word *Sequentz*.

Sometimes we read $\{\varphi_1, \ldots, \varphi_n\} \vdash \varphi$ as $\varphi$ *is a syntactical consequence of* $\{\varphi_1, \ldots, \varphi_n\}$. We will often use $\Gamma$ to denote the set of premisses $\{\varphi_1, \ldots, \varphi_n\}$ of the sequent and in this case we will write $\Gamma \vdash \varphi$. Also, the usual notion in the literature permits the writing $\varphi_1, \ldots, \varphi_n \vdash \varphi$ (without curly braces) instead of $\{\varphi_1, \ldots, \varphi_n\} \vdash \varphi$, but we have to keep in mind that to the left hand side of the $\vdash$ symbol we always have a set of formulae.

**Example 82.** *Here are some examples of sequents:*

1. $\{\mathtt{p}, \mathtt{q}\} \vdash (\mathtt{p} \land \mathtt{q})$;
2. $\{\mathtt{p}, (\mathtt{q} \land \neg\mathtt{r})\} \vdash (\mathtt{p} \lor \mathtt{q})$;
3. $\{(\mathtt{p} \land \mathtt{q}), (\mathtt{p} \lor \mathtt{q})\} \vdash (\mathtt{p} \land \mathtt{r})$.

*Later on, we shall see that the first two sequents are* valid*, and the last sequent is not.*

## 7.2 Inference Rules

**Definition 83.** *An* inference rule *is a tuple consisting of:*

1. *a set of sequents* $S_1, \ldots, S_n$*, called the* hypotheses *of the rule;*
2. *a sequent* $S$ *that is called the* conclusion *of the rule;*
3. *a possible* condition *for applying the rule;*
4. *a name.*

An inference rule is denoted as follows:

$$\text{NAME}\ \frac{S_1 \qquad \ldots \qquad S_n}{S}\ \textit{condition.}$$

**Remark.** *It is possible for a sequent to have* $n = 0$ *hypotheses. Such inference rules, having* $0$ *hypotheses, are called* axioms.

**Remark.** *Furthermore, the* condition *in an inference rule is optional.*

**Example 84.** *Here are some examples of inference rules:*

$$\wedge i \frac{\Gamma \vdash \varphi \qquad \Gamma \vdash \varphi'}{\Gamma \vdash (\varphi \wedge \varphi'),} \qquad \wedge e_1 \frac{\Gamma \vdash (\varphi \wedge \varphi')}{\Gamma \vdash \varphi,}$$

$$\wedge e_2 \frac{\Gamma \vdash (\varphi \wedge \varphi')}{\Gamma \vdash \varphi'.}$$

*All three rules above are* sound, *in a sense that we will define later. None of the three rules above has a condition. Here is an example of a rule with $n = 0$ hypotheses, but having a condition:*

$$\text{PREMISS} \frac{}{\Gamma \vdash \varphi} \varphi \in \Gamma.$$

*Here is an example of an unsound inference rule (we will see later in what sense it is unsound):*

$$\text{UNSOUND RULE} \frac{\Gamma \vdash \varphi'}{\Gamma \vdash (\varphi \wedge \varphi').}$$

**Remark.** *To be precise, the hypotheses of the inference rule, as well as the conclusion, are really* sequent schemes *and not sequents. This means that an inference rules might have several* instances, *obtained by replacing the mathematical variables $\varphi, \varphi', \Gamma$ with actual formulae. For example, here are two instances of the rule $\wedge i$ above:*

$$\wedge i \frac{\{\mathtt{p}, \mathtt{q}\} \vdash \mathtt{p} \qquad \{\mathtt{p}, \mathtt{q}\} \vdash \mathtt{q}}{\{\mathtt{p}, \mathtt{q}\} \vdash (\mathtt{p} \wedge \mathtt{q});} \qquad \wedge i \frac{\{\mathtt{p}, \mathtt{q}, \mathtt{r}\} \vdash (\mathtt{p} \wedge \mathtt{q}) \qquad \{\mathtt{p}, \mathtt{q}, \mathtt{r}\} \vdash \mathtt{p}}{\{\mathtt{p}, \mathtt{q}, \mathtt{r}\} \vdash ((\mathtt{p} \wedge \mathtt{q}) \wedge \mathtt{p}).}$$

*In the first instance, we have replaced the mathematical variable $\Gamma$ by the set of formulae $\{\mathtt{p}, \mathtt{q}\}$, the mathematical variable $\varphi$ by the formula $\mathtt{p}$ and the mathematical variable $\varphi'$ by the formula $\mathtt{q}$.*

**Exercise 85.** *Determine what each variable was replaced with in the second instance.*

**Exercise 86.** *Explain why the following rule is not an instance of the $\wedge i$ rule:*

$$\wedge i \frac{\{\mathtt{p}, \mathtt{q}\} \vdash \mathtt{p} \qquad \{\mathtt{p}, \mathtt{q}\} \vdash \mathtt{q}}{\{\mathtt{p}, \mathtt{q}\} \vdash (\mathtt{p} \wedge \mathtt{p}).}$$

## 7.3 Proof system

**Definition 87.** *A* proof system *is a set of inference rules.*

**Example 88.** *Consider the proof system $D_1$, containing the following inference rules:*

$$\textsc{Premiss}\ \frac{}{\Gamma \vdash \varphi,}\ \varphi \in \Gamma \qquad \wedge i\ \frac{\Gamma \vdash \varphi \qquad \Gamma \vdash \varphi'}{\Gamma \vdash (\varphi \wedge \varphi'),} \qquad \wedge e_1\ \frac{\Gamma \vdash (\varphi \wedge \varphi')}{\Gamma \vdash \varphi,}$$

$$\wedge e_2\ \frac{\Gamma \vdash (\varphi \wedge \varphi')}{\Gamma \vdash \varphi'.}$$

## 7.4 Formal Proofs

**Definition 89** (Formal Proof). *A* formal proof *in a proof system is a list of sequents*

*1.* $S_1$*;*

*2.* $S_2$*;*

...

*n.* $S_n$

*such that for each* $1 \leq i \leq n$*,*

> *the sequent $S_i$ is justified by an inference rule in the proof system from the previous sequents ($S_1, \ldots, S_{i-1}$), in the sense that $S_i$ is the conclusion of an instance of an inference rule in the proof system that uses hypotheses only among the sequents $S_1, \ldots, S_{i-1}$ and such that the condition of the inference rule is true (if the inference rule has a condition).*

**Example 90.** *Here is an example of a formal proof in the proof system $D_1$ defined above:*

*1.* $\{\mathtt{p}, \mathtt{q}\} \vdash \mathtt{p}$; *(*Premiss*)*

*2.* $\{\mathtt{p}, \mathtt{q}\} \vdash \mathtt{q}$; *(*Premiss*)*

*3.* $\{\mathtt{p}, \mathtt{q}\} \vdash (\mathtt{p} \wedge \mathtt{q})$; *($\wedge i$, 1, 2)*

*4.* $\{\mathtt{p}, \mathtt{q}\} \vdash (\mathtt{q} \wedge (\mathtt{p} \wedge \mathtt{q}))$. *($\wedge i$, 2, 3)*

*Note that each line is annotated by the name of the inference rule that was applied, plus the lines where the hypotheses of the rule are found.*

**Definition 91** (Valid Sequent). *A sequent* $\Gamma \vdash \varphi$ *is* valid in a proof system $D$ *if there exists a formal proof* $S_1, \ldots, S_n$ *in* $D$ *such that* $S_n = \Gamma \vdash \varphi$.

**Example 92.** *The sequent* $\{p, q\} \vdash (p \wedge q)$ *is valid in the proof system* $D_1$ *above, because it is the last sequent in the following formal proof:*

*1.* $\{p, q\} \vdash p$; *(*Premiss*)*

*2.* $\{p, q\} \vdash q$; *(*Premiss*)*

*3.* $\{p, q\} \vdash (p \wedge q)$. *(*$\wedge i$, 1, 2*)*

**Remark.** *Do not mistake the notion of* sequent valid in a proof system *for the notion of* valid formula.

## 7.5 Natural Deduction

*Natural deduction* is a proof system for $\mathbb{PL}_{\neg,\wedge,\vee,\rightarrow,\perp}$. In this proof system, each logical connective has one or more introduction rules and one or more elimination rules.

### 7.5.1 The Rules for Conjunction

We have already seen the introduction and elimination rules for the connective *and*:

$$\wedge i\ \frac{\Gamma \vdash \varphi \qquad \Gamma \vdash \varphi'}{\Gamma \vdash (\varphi \wedge \varphi'),} \qquad \wedge e_1\ \frac{\Gamma \vdash (\varphi \wedge \varphi')}{\Gamma \vdash \varphi,} \qquad \wedge e_2\ \frac{\Gamma \vdash (\varphi \wedge \varphi')}{\Gamma \vdash \varphi'.}$$

The proof system is called *natural* deduction because it mimics the human reasoning process:

• The rule for introducing *and* indicates that we can prove a conjunction $(\varphi \wedge \varphi')$ from the antecedents $\Gamma$ if we know that each conjunct, $\varphi$ and $\varphi'$ respectively, is a consequence of $\Gamma$.

In other words, to show that a conjunction follows from $\Gamma$, it is enough to show individually that each conjunct follows from $\Gamma$.

• There are two elimination rules for *and.* The first elimination rule indicates that if we have already established that a conjunction $(\varphi \wedge \varphi')$ follows from $\Gamma$, then the left-hand side conjunct, $\varphi$, also follows from $\Gamma$.

The second elimination rule is symmetrical.

Here is an example of a formal proof that uses the inference rules for *and*:

1. $\{(\mathtt{p} \wedge \mathtt{q}), \mathtt{r}\} \vdash (\mathtt{p} \wedge \mathtt{q})$; (Premiss)
2. $\{(\mathtt{p} \wedge \mathtt{q}), \mathtt{r}\} \vdash \mathtt{r}$; (Premiss)
3. $\{(\mathtt{p} \wedge \mathtt{q}), \mathtt{r}\} \vdash \mathtt{p}$; ($\wedge e_1$, 1)
4. $\{(\mathtt{p} \wedge \mathtt{q}), \mathtt{r}\} \vdash (\mathtt{p} \wedge \mathtt{r})$. ($\wedge i$, 3, 2)

**Exercise 93.** *Show that the following sequents are valid:*

*1.* $\{((\mathtt{p} \wedge \mathtt{q}) \wedge \mathtt{r})\} \vdash (\mathtt{q} \wedge \mathtt{r})$;

*2.* $\{((\mathtt{p} \wedge \mathtt{q}) \wedge \mathtt{r}), \mathtt{r}'\} \vdash (\mathtt{r}' \wedge \mathtt{q})$;

*3.* $\{((\mathtt{p} \wedge \mathtt{q}) \wedge \mathtt{r})\} \vdash ((\mathtt{r} \wedge \mathtt{q}) \wedge \mathtt{p})$.

### 7.5.2 The Rules for Implications

The rule for eliminating implications, also known as *modus ponens* in latin, is one of the most important inference rules.

$$\rightarrow e \; \frac{\Gamma \vdash (\varphi \rightarrow \varphi') \qquad \Gamma \vdash \varphi}{\Gamma \vdash \varphi'.}$$

The rule tells us that, assuming that we already know that $\varphi \rightarrow \varphi'$ follows from $\Gamma$ and that $\varphi$ follows from $\Gamma$, then we also have that $\varphi'$ follows from $\Gamma$.

Here is an example of formal proof that uses the modus ponens rule:

1. $\{(\mathtt{p} \rightarrow \mathtt{r}), (\mathtt{p} \wedge \mathtt{q})\} \vdash (\mathtt{p} \wedge \mathtt{q})$; (Premiss)
2. $\{(\mathtt{p} \rightarrow \mathtt{r}), (\mathtt{p} \wedge \mathtt{q})\} \vdash \mathtt{p}$; ($\wedge e_1$, 1)
3. $\{(\mathtt{p} \rightarrow \mathtt{r}), (\mathtt{p} \wedge \mathtt{q})\} \vdash (\mathtt{p} \rightarrow \mathtt{r})$; (Premiss)
4. $\{(\mathtt{p} \rightarrow \mathtt{r}), (\mathtt{p} \wedge \mathtt{q})\} \vdash \mathtt{r}$. ($\rightarrow e$, 3, 2)

This formal proof shows that the sequent $\{(\mathtt{p} \rightarrow \mathtt{r}), (\mathtt{p} \wedge \mathtt{q})\} \vdash \mathtt{r}$ is valid, meaning that the formula $\mathtt{r}$ is a consequence of the formulae $(\mathtt{p} \rightarrow \mathtt{r})$ and $(\mathtt{p} \wedge \mathtt{q})$. Note the order of the line numbers 3 and 2 in the annotation for line 4 (they follow the order in the inference rule).

**Exercise 94.** *Show that the following sequents are valid:*

*1.* $\{((\mathtt{p} \wedge \mathtt{q}) \rightarrow \mathtt{r}), \mathtt{p}, \mathtt{q}\} \vdash \mathtt{r}$;

*2.* $\{(\mathtt{p} \rightarrow \mathtt{r}), \mathtt{p}, \mathtt{q}\} \vdash (\mathtt{q} \wedge \mathtt{r})$.

The rule for introducing implication is more subtle. To show that an implication $(\varphi \rightarrow \varphi')$ follows from $\Gamma$, we *assume* $\varphi$ (in addition to $\Gamma$) and we show $\varphi'$. In other words, in the hypothesis of the rule, we add the formula $\varphi$ to the formulae in $\Gamma$. The rule can be written in two equivalent ways, that differ only in their use of the convention regarding the curly braces:

$$\rightarrow i \, \frac{\Gamma, \varphi \vdash \varphi'}{\Gamma \vdash (\varphi \rightarrow \varphi'),} \qquad\qquad \rightarrow i \, \frac{\Gamma \cup \{\varphi\} \vdash \varphi'}{\Gamma \vdash (\varphi \rightarrow \varphi').}$$

What is important to note and understand for the rule for introducing implication is that the premisses of the sequent in the hypotheses are different from the premisses of the sequent in the conclusion. Whereas in the conclusion we have that the formula $(\varphi \rightarrow \varphi')$ follows from $\Gamma$, in the hypothesis we have to show that $\varphi'$ follows from $\Gamma \cup \{\varphi\}$. In other words, intuitively speaking, to prove an implication $(\varphi \rightarrow \varphi')$, we assume the antecedent $\varphi$ and we show the consequent $\varphi'$.

**Example 95.** *Let us show that the sequent* $\{\} \vdash (\mathtt{p} \rightarrow \mathtt{p})$ *is valid:*

*1.* $\{\mathtt{p}\} \vdash \mathtt{p}$; *(*PREMISS*)*

*2.* $\{\} \vdash (\mathtt{p} \rightarrow \mathtt{p})$. *(*$\rightarrow i$, 1*)*

**Example 96.** *Let us show that the sequent* $\{(\mathtt{p} \rightarrow \mathtt{q})\} \vdash (\mathtt{p} \rightarrow \mathtt{q})$ *is valid. A simple proof is:*

*1.* $\{(\mathtt{p} \rightarrow \mathtt{q})\} \vdash (\mathtt{p} \rightarrow \mathtt{q})$. *(*PREMISS*)*

*A longer proof is:*

*1.* $\{(\mathtt{p} \rightarrow \mathtt{q}), \mathtt{p}\} \vdash (\mathtt{p} \rightarrow \mathtt{q})$*;* *(*PREMISS*)*

*2.* $\{(\mathtt{p} \rightarrow \mathtt{q}), \mathtt{p}\} \vdash \mathtt{p}$*;* *(*PREMISS*)*

*3.* $\{(\mathtt{p} \rightarrow \mathtt{q}), \mathtt{p}\} \vdash \mathtt{q}$*;* *(*$\rightarrow e$, 1, 2*)*

*4.* $\{(\mathtt{p} \rightarrow \mathtt{q})\} \vdash (\mathtt{p} \rightarrow \mathtt{q})$. *(*$\rightarrow i$, 3*)*

**Example 97.** *Let us show that the sequent* $\{(\mathtt{p} \rightarrow \mathtt{q}), (\mathtt{q} \rightarrow \mathtt{r})\} \vdash (\mathtt{p} \rightarrow \mathtt{r})$ *is valid:*

*1.* $\{(\mathtt{p} \rightarrow \mathtt{q}), (\mathtt{q} \rightarrow \mathtt{r}), \mathtt{p}\} \vdash (\mathtt{p} \rightarrow \mathtt{q})$; *(*PREMISS*)*

*2.* $\{(\mathtt{p} \rightarrow \mathtt{q}), (\mathtt{q} \rightarrow \mathtt{r}), \mathtt{p}\} \vdash \mathtt{p}$; *(*PREMISS*)*

*3.* $\{(p \rightarrow q), (q \rightarrow r), p\} \vdash q$; *(*$\rightarrow e$, 1, 2*)*

*4.* $\{(p \rightarrow q), (q \rightarrow r), p\} \vdash (q \rightarrow r)$; *(*PREMISS*)*

*5.* $\{(p \rightarrow q), (q \rightarrow r), p\} \vdash r$; *(*$\rightarrow e$, 4, 3*)*

*6.* $\{(p \rightarrow q), (q \rightarrow r)\} \vdash (p \rightarrow r)$. *(*$\rightarrow i$, 5*)*

**Exercise 98.** *Show that the following sequents are valid:*

*1.* $\{((p \wedge q) \rightarrow r), p, q\} \vdash r$;

*2.* $\{((p \wedge q) \rightarrow r)\} \vdash (p \rightarrow (q \rightarrow r))$;

*3.* $\{(p \rightarrow (q \rightarrow r))\} \vdash ((p \wedge q) \rightarrow r)$.

### 7.5.3 Rules for Disjunctions

The connective *or* has two introduction rules:

$$\vee i_1 \frac{\Gamma \vdash \varphi_1}{\Gamma \vdash (\varphi_1 \vee \varphi_2),} \qquad \vee i_2 \frac{\Gamma \vdash \varphi_2}{\Gamma \vdash (\varphi_1 \vee \varphi_2).}$$

The first rule indicates that if we already know $\varphi_1$ follows from $\Gamma$, then $(\varphi_1 \vee \varphi_2)$ must also follow from $\Gamma$, what $\varphi_2$ is. In other words, intuitively speaking, if we know $\varphi_1$, then we also know $(\varphi_1 \vee$ *whatever else*$)$. The second introduction rule for disjunction is symmetrical.

**Example 99.** *Let us show that the sequent* $\{(p \wedge q)\} \vdash (p \vee q)$ *is valid:*

*1.* $\{(p \wedge q)\} \vdash (p \wedge q)$; *(*PREMISS*)*

*2.* $\{(p \wedge q)\} \vdash p$; *(*$\wedge e_1$, 1*)*

*3.* $\{(p \wedge q)\} \vdash (p \vee q)$. *(*$\vee i_1$, 2*)*

*Another formal proof for the same sequent is:*

*1.* $\{(p \wedge q)\} \vdash (p \wedge q)$; *(*PREMISS*)*

*2.* $\{(p \wedge q)\} \vdash q$; *(*$\wedge e_2$, 1*)*

*3.* $\{(p \wedge q)\} \vdash (p \vee q)$. *(*$\vee 2_1$, 2*)*

**Exercise 100.** *Show that the sequent* $\{(p \wedge q)\} \vdash (r \vee p)$ *is valid.*

The rule for eliminating disjunctions is slightly more complicated, being another rule where the set of premisses of the sequents changes from hypotheses to conclusion:

$$\vee e \, \frac{\Gamma \vdash (\varphi_1 \vee \varphi_2) \qquad \Gamma, \varphi_1 \vdash \varphi' \qquad \Gamma, \varphi_2 \vdash \varphi'}{\Gamma \vdash \varphi'}$$

The first hypothesis of the rule, $\Gamma \vdash (\varphi_1 \vee \varphi_2)$, is easy to understand: to *eliminate* a disjunction, we must have such a disjunction among the hypotheses (disjunction that we will *eliminate* in the conclusion). The following two hypotheses for disjunction elimination must be understood intuitively as follows. From the first hypothesis, we have that $(\varphi_1 \vee \varphi_2)$ follows from $\Gamma$; in other words, at least one of the formulae $\varphi_1$ and $\varphi_2$ follows from $\Gamma$. The hypotheses 2 and 3 indicate that, no matter which of $\varphi_1$ and $\varphi_2$ would hold, in any case $\varphi'$ holds. Meaning that if we assume $\varphi_1$ (in addition to $\Gamma$), $\varphi'$ holds, and if we assume $\varphi_2$ (in addition to $\Gamma$), $\varphi'$ still holds. And then the conclusion indicates that $\varphi'$ holds independently of which of $\varphi_1$ and $\varphi_2$ holds.

**Example 101.** *Let us show that the sequent* $\{(\mathtt{p} \vee \mathtt{q})\} \vdash (\mathtt{q} \vee \mathtt{p})$ *is valid:*

1. $\{(\mathtt{p} \vee \mathtt{q}), \mathtt{p}\} \vdash \mathtt{p}$; (Premiss)
2. $\{(\mathtt{p} \vee \mathtt{q}), \mathtt{p}\} \vdash (\mathtt{q} \vee \mathtt{p})$; $(\vee i_2, 1)$
3. $\{(\mathtt{p} \vee \mathtt{q}), \mathtt{q}\} \vdash \mathtt{q}$; (Premiss)
4. $\{(\mathtt{p} \vee \mathtt{q}), \mathtt{q}\} \vdash (\mathtt{q} \vee \mathtt{p})$; $(\vee i_1, 3)$
5. $\{(\mathtt{p} \vee \mathtt{q})\} \vdash (\mathtt{p} \vee \mathtt{q})$; (Premiss)
6. $\{(\mathtt{p} \vee \mathtt{q})\} \vdash (\mathtt{q} \vee \mathtt{p})$. $(\vee e, 5, 2, 4)$

*Note that way in which the set of premisses varies from one sequent to another along the formal proof, obbeying the inference rules.*

**Exercise 102.** *Show that the sequent* $\{(\mathtt{p} \vee \mathtt{q}), (\mathtt{p} \rightarrow \mathtt{r}), (\mathtt{q} \rightarrow \mathtt{r})\} \vdash \mathtt{r}$ *is valid.*

**Exercise 103.** *Show that the sequent* $\{(\mathtt{p} \rightarrow \mathtt{r}), (\mathtt{q} \rightarrow \mathtt{r})\} \vdash ((\mathtt{p} \vee \mathtt{q}) \rightarrow \mathtt{r})$ *is valid.*

### 7.5.4 The Rules for Negations

The rules for introducing and respectively eliminating negations must be presented at the same time as the rules for $\bot$:

$$\neg i\ \frac{\Gamma, \varphi \vdash \bot}{\Gamma \vdash \neg\varphi,} \qquad \neg e\ \frac{\Gamma \vdash \varphi \qquad \Gamma \vdash \neg\varphi}{\Gamma \vdash \bot,} \qquad \bot e\ \frac{\Gamma \vdash \bot}{\Gamma \vdash \varphi.}$$

Recall that $\bot$ is a nullary logical connective (it has arity 0), meaning that $\bot$ connects 0 formulae among themselves. In other words, the connective $\bot$ is by itself a formula. The semantics of $\bot$ is that it is false in any assignment. In other words, $\bot$ is a contradiction.

The first rule above is that for introducing negations. It is easy to explain intuitively: how can we show that a formula of the form$\neg\varphi$ follows from $\Gamma$? We assume $\varphi$ in addition to $\Gamma$ and we show that from $\Gamma$ and $\varphi$ follows a contradiction ($\Gamma, \varphi \vdash \bot$). In this way, we have that $\neg\varphi$ follows from $\Gamma$.

The second rule, for eliminating negations, indicates that if both a formula, $\varphi$, and its negation, $\neg\varphi$, follow from the same set of premisses $\Gamma$, then from $\Gamma$ follows a contradiction, $\bot$. A set $\Gamma$ from which a contradiction follows is called *inconsistent*.

The third rule indicates that, if $\Gamma$ is an inconsistent set, then any formula $\varphi$ follows from $\Gamma$.

There is no rule for introducing $\bot$ (or, equivallently, the rule for eliminating negation can be considered as the rule for introducing $\bot$).

**Example 104.** *Let us show that the sequent* $\{\mathtt{p}\} \vdash \neg\neg\mathtt{p}$ *is valid:*

1. $\{\mathtt{p}, \neg\mathtt{p}\} \vdash \mathtt{p}$; (PREMISS)
2. $\{\mathtt{p}, \neg\mathtt{p}\} \vdash \neg\mathtt{p}$; (PREMISS)
3. $\{\mathtt{p}, \neg\mathtt{p}\} \vdash \bot$; ($\neg e$, 1, 2)
4. $\{\mathtt{p}\} \vdash \neg\neg\mathtt{p}$. ($\neg i$, 3)

**Example 105.** *Let us show that the sequent* $\{\mathtt{p}, \neg\mathtt{p}\} \vdash \mathtt{r}$ *is valid:*

1. $\{\mathtt{p}, \neg\mathtt{p}\} \vdash \mathtt{p}$; (PREMISS)
2. $\{\mathtt{p}, \neg\mathtt{p}\} \vdash \neg\mathtt{p}$; (PREMISS)
3. $\{\mathtt{p}, \neg\mathtt{p}\} \vdash \bot$; ($\neg e$, 1, 2)
4. $\{\mathtt{p}, \neg\mathtt{p}\} \vdash \mathtt{r}$. ($\bot e$, 3)

**Exercise 106.** *Show that the following sequents are valid:*

1. $\{(p \lor q)\} \vdash \neg(\neg p \land \neg q)$;
2. $\{(p \land q)\} \vdash \neg(\neg p \lor \neg q)$;
3. $\{(\neg p \lor \neg q)\} \vdash \neg(p \land q)$;
4. $\{(\neg p \land \neg q)\} \vdash \neg(p \lor q)$;
5. $\{\neg(p \lor q)\} \vdash (\neg p \land \neg q)$.

### 7.5.5 Alte reguli

Another useful rule (especially when proving derived rules – which we will discuss shortly), that is not related to any given connective is the Extend rule:

$$\text{Extend}\ \frac{\Gamma \vdash \varphi}{\Gamma, \varphi' \vdash \varphi}$$

This rule indicates that, if $\varphi$ is a consequence of a set of formulae $\Gamma$, then $\varphi$ is also a consequence of $\Gamma \cup \{\varphi'\}$ (whatever $\varphi'$ is). In other words, we can extend the set of premisses of a valid sequent and still get a valid sequent.

**Example 107.** *Let us show that the sequent* $\{p, \neg q, r, (q_1 \land q_2)\} \vdash \neg\neg p$ *is valid:*

1. $\{p, \neg p\} \vdash p$; (Premiss)
2. $\{p, \neg p\} \vdash \neg p$; (Premiss)
3. $\{p, \neg p\} \vdash \bot$; $(\neg e, 1, 2)$
4. $\{p\} \vdash \neg\neg p$; $(\neg i, 3)$
5. $\{p, \neg q\} \vdash \neg\neg p$; (Extend, 4)
6. $\{p, \neg q, r\} \vdash \neg\neg p$; (Extend, 5)
7. $\{p, \neg q, r, (q_1 \land q_2)\} \vdash \neg\neg p$. (Extend, 6)

The rules above are natural deduction for a logic that is called *intuitionistic propositional logic*. In this lecture, we study *classical propositional logic*. The two logics have the same syntax, but their semantics are different. Intuitionist logic is important in computer science, because formal proofs in this logic are in a 1-to-1 correspondence with computer programs. Meaning that to each intuitionistic proof we can associate a program, and to each program, an intuitionistic proof. This mean essentially that each intuitionistic proof *is* a program. Intuitionistic logic is a *constructive logic*, meaning a logic where any proof gives an algorithm.

To obtain a proof system for classical propositional logic, which is the object of our study, we need one more rule:

$$\neg\neg e \, \frac{\Gamma \vdash \neg\neg\varphi}{\Gamma \vdash \varphi.}$$

**Example 108.** *Let us show that the sequent* $\{(\neg p \rightarrow q), \neg q\} \vdash p$ *is valid:*

1. $\{(\neg p \rightarrow q), \neg q, \neg p\} \vdash \neg p$; (Premiss)
2. $\{(\neg p \rightarrow q), \neg q, \neg p\} \vdash (\neg p \rightarrow q)$; (Premiss)
3. $\{(\neg p \rightarrow q), \neg q, \neg p\} \vdash q$; $(\rightarrow e, 2, 1)$
4. $\{(\neg p \rightarrow q), \neg q, \neg p\} \vdash \neg q$; (Premiss)
5. $\{(\neg p \rightarrow q), \neg q, \neg p\} \vdash \bot$; $(\neg i, 4, 3)$
6. $\{(\neg p \rightarrow q), \neg q\} \vdash \neg\neg p$; $(\neg i, 5)$
7. $\{(\neg p \rightarrow q), \neg q\} \vdash p$. $(\neg\neg e, 6)$

**Example 109.** *Let us show that the sequent* $\{\} \vdash (p \vee \neg p)$ *is valid:*

1. $\{\neg(p \vee \neg p), p\} \vdash \neg(p \vee \neg p)$; (Premiss)
2. $\{\neg(p \vee \neg p), p\} \vdash p$; (Premiss)
3. $\{\neg(p \vee \neg p), p\} \vdash (p \vee \neg p)$; $(\vee i_1, 2)$
4. $\{\neg(p \vee \neg p), p\} \vdash \bot$; $(\neg e, 1, 3)$
5. $\{\neg(p \vee \neg p)\} \vdash \neg p$; $(\neg i, 4)$
6. $\{\neg(p \vee \neg p)\} \vdash (p \vee \neg p)$; $(\vee i_2, 5)$
7. $\{\neg(p \vee \neg p)\} \vdash \neg(p \vee \neg p)$; (Premiss)
8. $\{\neg(p \vee \neg p)\} \vdash \bot$; $(\neg e, 7, 6)$
9. $\{\} \vdash \neg\neg(p \vee \neg p)$; $(\neg i, 8)$
10. $\{\} \vdash (p \vee \neg p)$. $(\neg\neg e, 9)$

**Exercise 110.** *Show that the following sequents are valid:*

1. $\{\neg(p \wedge q)\} \vdash (\neg p \vee \neg q)$;
2. $\{\neg(\neg p \vee \neg q)\} \vdash (p \wedge q)$;
3. $\{\neg(\neg p \wedge \neg q)\} \vdash (p \vee q)$.

## 7.6 Natural Deduction

Natural deduction is the proof system containing all rules above. Here is a summary of all rules:

$$\wedge i \ \frac{\Gamma \vdash \varphi \qquad \Gamma \vdash \varphi'}{\Gamma \vdash (\varphi \wedge \varphi'),} \qquad \wedge e_1 \ \frac{\Gamma \vdash (\varphi \wedge \varphi')}{\Gamma \vdash \varphi,}$$

$$\wedge e_2 \ \frac{\Gamma \vdash (\varphi \wedge \varphi')}{\Gamma \vdash \varphi',} \qquad \to e \ \frac{\Gamma \vdash (\varphi \to \varphi') \qquad \Gamma \vdash \varphi}{\Gamma \vdash \varphi',} \qquad \to i \ \frac{\Gamma, \varphi \vdash \varphi'}{\Gamma \vdash (\varphi \to \varphi'),}$$

$$\vee i_1 \ \frac{\Gamma \vdash \varphi_1}{\Gamma \vdash (\varphi_1 \vee \varphi_2),} \qquad \vee i_2 \ \frac{\Gamma \vdash \varphi_2}{\Gamma \vdash (\varphi_1 \vee \varphi_2),}$$

$$\vee e \ \frac{\Gamma \vdash (\varphi_1 \vee \varphi_2) \qquad \Gamma, \varphi_1 \vdash \varphi' \qquad \Gamma, \varphi_2 \vdash \varphi'}{\Gamma \vdash \varphi',}$$

$$\neg e \ \frac{\Gamma \vdash \varphi \qquad \Gamma \vdash \neg\varphi}{\Gamma \vdash \bot,} \qquad \neg i \ \frac{\Gamma, \varphi \vdash \bot}{\Gamma \vdash \neg\varphi,} \qquad \bot e \ \frac{\Gamma \vdash \bot}{\Gamma \vdash \varphi,}$$

$$\textsc{Premiss} \ \frac{}{\Gamma \vdash \varphi} \ \varphi \in \Gamma, \qquad \textsc{Extend} \ \frac{\Gamma \vdash \varphi}{\Gamma, \varphi' \vdash \varphi,} \qquad \neg\neg e \ \frac{\Gamma \vdash \neg\neg\varphi}{\Gamma \vdash \varphi.}$$

## 7.7 Derived Rules

A *derived rule* is a proof rule that we can prove using the other proof rules in a given proof system, through a formal proof.

An example of a derived rule is the rule for introducing double negation:

$$\neg\neg i \ \frac{\Gamma \vdash \varphi}{\Gamma \vdash \neg\neg\varphi.}$$

Here is a formal proof for the rule for introducing double negation. We start with the hypotheses of the rule, and the last sequent in the proof must be the conclusion of the rule.

1. $\Gamma \vdash \varphi$; (hypothesis of the derived rule)
2. $\Gamma, \neg\varphi \vdash \varphi$; (Extend, 1)
3. $\Gamma, \neg\varphi \vdash \neg\varphi$; (Premiss)

4. $\Gamma, \neg\varphi \vdash \bot$; $(\neg e, 2, 3)$

5. $\Gamma \vdash \neg\neg\varphi$. $(\neg i, 4)$

After having been shown derived (as above), such a proof rule can be used to shorten other proofs, similarly to how we shorten code by writing subprograms (functions) in certain programming languages. Here is an example of a formal proof that uses the derived rule $\neg\neg i$:

1. $\{(\neg\neg p \rightarrow q), p\} \vdash p$; (Premiss)
2. $\{(\neg\neg p \rightarrow q), p\} \vdash \neg\neg p$; $(\neg\neg i, 1)$
3. $\{(\neg\neg p \rightarrow q), p\} \vdash (\neg\neg p \rightarrow q)$; (Premiss)
4. $\{(\neg\neg p \rightarrow q), p\} \vdash q$. $(\rightarrow e, 3, 2)$

In the absence of the derived rule, our proof would have been longer:

1. $\{(\neg\neg p \rightarrow q), p\} \vdash p$; (Premiss)
2. $\{(\neg\neg p \rightarrow q), p, \neg p\} \vdash p$; (Extend, 1)
3. $\{(\neg\neg p \rightarrow q), p, \neg p\} \vdash \neg p$; (Premiss)
4. $\{(\neg\neg p \rightarrow q), p, \neg p\} \vdash \bot$; $(\neg e, 2, 3)$
5. $\{(\neg\neg p \rightarrow q), p\} \vdash \neg\neg p$. $(\neg i, 4)$
6. $\{(\neg\neg p \rightarrow q), p\} \vdash (\neg\neg p \rightarrow q)$; (Premiss)
7. $\{(\neg\neg p \rightarrow q), p\} \vdash q$. $(\rightarrow e, 6, 5)$

Note that the lines 1–5 in the longer proof are an instance of the formal proof for the derived rule.

## 7.8 Soundness and Completeness for Natural Deduction

**Theorem 111** (Soundness of Natural Deduction)**.** *For any set of formulae $\Gamma$ and any formula $\varphi$, if the sequent $\Gamma \vdash \varphi$ is valid, then $\Gamma \models \varphi$.*

Exercise: prove this theorem.

**Theorem 112** (Completeness of Natural Deduction)**.** *For any set of formulae $\Gamma$ and any formula $\varphi$, if $\Gamma \models \varphi$ then the sequent $\Gamma \vdash \varphi$ is valid.*

The proof of the completeness theorem is complex and we do not address it in this course.

**Remark.** *Note that, due to the soundness and completeness theorems, the relation $\vdash$ coincides with the relation $\models$, even if they have totally different definitions.*

**Exercise 113.** *Using the above theorems, prove that a set $\{\varphi_1, \varphi_2, \ldots, \varphi_n\}$ of formulae is inconsistent if the sequent $\{\varphi_1, \varphi_2, \ldots, \varphi_n\} \vdash \bot$ is valid.*

## 7.9 Exercise Sheet

**Exercise 114.** *Give a formal proof of $(\mathtt{p} \land \mathtt{r})$ from $\{((\mathtt{q} \land \mathtt{r}) \land \mathtt{q}), (\mathtt{p} \land \mathtt{p})\}$.*

**Exercise 115.** *Show the validity of the following sequents:*

1. $(\mathtt{p} \land \mathtt{q}), \mathtt{r} \vdash (\mathtt{p} \land (\mathtt{r} \lor \mathtt{r}')))$;
2. $(\mathtt{p} \to (\mathtt{q} \to \mathtt{r})) \vdash ((\mathtt{p} \land \mathtt{q}) \to \mathtt{r})$;
3. $((\mathtt{p} \land \neg\mathtt{r}) \to \mathtt{q}), \neg\mathtt{q}, \mathtt{p} \vdash \mathtt{r}$;

**Exercise 116.** *Finish the game at* `https://profs.info.uaic.ro/~stefan.ciobaca/lnd.html`*. Do not cheat. It is considered cheating if you change the JavaScript source code, if someone else solves a level for you or if you prove the derived rules using the derived rules themselves.*

**Exercise 117.** *Prove that the following inference rules are derivable:*

1. $\neg\neg i$;
2. *LEM (law of excluded middle):* $\textsc{Lem}\ \dfrac{}{\Gamma \vdash (\varphi \lor \neg\varphi)}$;
3. *PBC (proof by contradiction):* $\textsc{Pbc}\ \dfrac{\Gamma, \neg\varphi \vdash \bot}{\Gamma \vdash \varphi}$;
4. *MT (modus tollens):* $\text{MT}\ \dfrac{\Gamma \vdash (\varphi \to \varphi') \qquad \Gamma \vdash \neg\varphi'}{\Gamma \vdash \neg\varphi}$.

**Exercise 118.** *Prove the soundness theorem for natural deduction (by induction on the number of sequents in the formal proof).*

**Exercise 119.** *Show that the rule $\neg\neg e$ is derivable using the LEM (i.e., you may use LEM in the derivation, but not $\neg\neg e$).*

**Exercise 120.** *Prove, using the soundness and completeness theorems, that $\varphi_1 \dashv\vdash \varphi_2$ if and only if $\varphi_1 \equiv \varphi_2$.*

# Chapter 8

# Normal Forms

So far, we have discussed the syntax and the semantics of propositional logic, and in the last chapter, we introduced a first method for *mechanizing* formal proofs, namely *natural deduction*.

An essential concern in Logic is *automated proof*, which means *automatic search* for formal proofs by a computer for a specific conjecture/theorem.

Natural deduction is a deductive system with certain features (for instance, formal proofs can be theoretically verified and easily understood). However, the performance of automated provers using natural deduction is lacking in practice.

For this reason, we will study another deductive system called *resolution*, which allows for practical and efficient implementations, even though resolution proofs may not be as elegant as natural deduction proofs.

Unlike natural deduction, which works with arbitrary formulae, resolution is limited to formulae in a specific *form*, known as *clausal normal form*. The goal of this chapter is to study clausal normal forms.

## 8.1 Notations

In practice, when we write formulae in $\mathbb{PL}$, we only use parentheses when strictly necessary.

Just as when we write $5 \times -3 + 2$, we understand it as $(5 \times (-3)) + 2$, similarly, instead of $\mathtt{p} \wedge \neg\mathtt{q} \vee \mathtt{r}$, we will understand $((\mathtt{p} \wedge \neg\mathtt{q}) \vee \mathtt{r})$. In this sense, the negation $\neg$ has the highest priority, followed by $\wedge$ and then $\vee$. We can remember this convention by associating:

1. $\neg$ with unary minus;
2. $\wedge$ with multiplication; and

3. $\vee$ with addition.

Furthermore, the operators $\wedge$ and $\vee$ are both associative and commutative. For this reason, we allow writing $p \wedge q \wedge r$ instead of $((p \wedge q) \wedge r)$ or $(p \wedge (q \wedge r))$.

In the rest of the course, we will allow to write formulae without explicit brackets, respecting the following priority order of the logical connectives:

$$\bot, \neg, \wedge, \vee, \rightarrow, \leftrightarrow.$$

For example, by

$$\neg\neg p \vee \bot \wedge p \rightarrow \neg p \wedge q \leftrightarrow q$$

we will understand the formula

$$(((\neg\neg p \vee (\bot \wedge p)) \rightarrow (\neg p \wedge q)) \leftrightarrow q).$$

We will also write $((\varphi_1 \vee \varphi_2) \vee \varphi_3)$ as $\varphi_1 \vee \varphi_2 \vee \varphi_3$ (meaning that a series of $\vee$ will be implicitly left-associated).

We will also write $((\varphi_1 \wedge \varphi_2) \wedge \varphi_3)$ as $\varphi_1 \wedge \varphi_2 \wedge \varphi_3$ (meaning that a series of $\wedge$s will also be implicitly left-associated).

**Example 121.** *We write*

$$p \wedge q \wedge r \vee \neg p \wedge \neg q \wedge \neg r$$

*for the formula*

$$(((p \wedge q) \wedge r) \vee ((\neg p \wedge \neg q) \wedge \neg r)).$$

**Example 122.** *We write*

$$p_1 \wedge p_2 \wedge p_3 \wedge p_4$$

*for the*

$$(((p_1 \wedge p_2) \wedge p_3) \wedge p_4).$$

**Example 123.** *We write*

$$p_1 \vee p_2 \vee p_3 \vee p_4$$

*for the formula*

$$(((p_1 \vee p_2) \vee p_3) \vee p_4).$$

## 8.2 Replacement Theorem

We present a theorem that we have used implicitly so far:

**Theorem 124** (The Replacement Theorem). *Let $\varphi, \varphi'$ be two formulae such that $\varphi \equiv \varphi'$. Let $\varphi_1$ be a formula that contains $\varphi$ as a subformula. Let $\varphi_2$ be the formula obtained from $\varphi_1$ by replacing one occurence of $\varphi$ by $\varphi'$.*

*Then $\varphi_1 \equiv \varphi_2$.*

In other words, the relation $\equiv$ is a *congruence*.

**Example 125.** *Let $\varphi = \mathtt{p}$ and $\varphi' = \neg\neg\mathtt{p}$.*

*Let $\varphi_1 = (\mathtt{p} \vee \mathtt{q})$ and $\varphi_2 = (\neg\neg\mathtt{p} \vee \mathtt{q})$.*

*By the replacement theorem, we obtain $\varphi_1 \equiv \varphi_2$. In other words, by replacing a subformula with an equivalent one, the new formula is equivalent to the initial formula.*

## 8.3 Literal

**Definition 126** (Literal). *A formula $\varphi$ is called a* literal *if there exists a propositional variable $a \in A$ such that*

$$\varphi = a \qquad or \qquad \varphi = \neg a.$$

**Example 127.** *The formulae $\mathtt{p}, \mathtt{q}, \neg\mathtt{p}, \neg\mathtt{p}', \mathtt{q}_1$ are literals, but the formulae $(\mathtt{p} \vee \mathtt{q}), \neg\neg\mathtt{p}, \neg\neg\neg\mathtt{q}_1, (\neg\mathtt{p} \wedge \mathtt{q})$ are not literals.*

## 8.4 Clauses

**Definition 128.** *A formula $\varphi$ is called a* clause *if there exist $n$ literals $\varphi_1, \ldots, \varphi_n$ such that*

$$\varphi = \varphi_1 \vee \varphi_2 \vee \ldots \vee \varphi_n.$$

In other words, a clause is a *disjunction of literals.*

**Example 129.** *The following formulae are clauses:*

1. $\mathtt{p} \vee \mathtt{q} \vee \mathtt{r}$;
2. $\mathtt{p} \vee \neg\mathtt{q} \vee \neg\mathtt{r}$;
3. $\neg\mathtt{p} \vee \neg\mathtt{q} \vee \neg\mathtt{r}$;
4. $\neg\mathtt{p}_1 \vee \mathtt{p}_1 \vee \mathtt{p}_2 \vee \neg\mathtt{q} \vee \neg\mathtt{r}$;

5. $\neg p_1 \vee p_1$;
6. $\neg p_1$ ($n = 1$).
7. $p$ ($n = 1$).

*The following formulae are not clauses:*

1. $p \wedge r$; *(conjunction instead of disjunction)*
2. $p \vee \neg\neg q \vee \neg r$; *(not a disjunction of literals)*
3. $\neg\neg p \vee p \wedge \neg p_1$. *(we have $\neg\neg$, so no literals; we also have $\wedge$)*

**Remark.** *Definition 128 explicitly states that we require $n$ literals, where $n \geq 1$. However, for the particular case when $n = 0$, we get a special clause called the* empty clause, *which has a dedicated notation: $\square$. We consider that $\square$ represents an unsatisfiable formula (semantically, the empty clause has the same meaning as $\bot$). This clause is used in a later chapter which will discuss about resolution in propositional logic.*

## 8.5 Conjunctive Normal Form

**Definition 130** (CNF)**.** *A formula $\varphi$* is in CNF *if there exist $n \geq 1$ clauses $\varphi_1, \ldots, \varphi_n$ such that*

$$\varphi = \varphi_1 \wedge \varphi_2 \wedge \ldots \wedge \varphi_n.$$

In other words, a formula in CNF is a conjunction of disjunctions of literals. Or, a formula in CNF is a conjunction of clauses. CNF is conveniently also an abbreviation of *clausal normal form*, which means the same thing as conjunctive normal form.

**Example 131.** *The following formulae are in CNF:*

1. $(\neg p \vee q) \wedge (r \vee \neg p \vee r') \wedge (\neg p \vee \neg r)$;
2. $\neg p \wedge (r \vee \neg p \vee r') \wedge \neg r$;
3. $\neg p \wedge r \wedge \neg r$;
4. $\neg p$;
5. $p$.

*The following formulae are not in CNF:*

1. $\neg(\neg p \vee q) \wedge (r \vee \neg p \vee r') \wedge (\neg p \vee \neg r)$; *(the first clause is negated)*

2. $\neg p \wedge \neg(r \vee \neg p \vee r')$; *(second clause is negated)*
3. $\neg p \vee (r \wedge \neg p \wedge r')$; *(the main connective is the disjunction, instead of a conjunction)*
4. $\neg\neg p$; *(we have $\neg\neg$, so no literals)*
5. $p \vee (q \wedge r)$. *(disjunction of conjunctions, not a conjunction of disjunctions (of literals))*

## 8.6 Bringing a Formula into CNF

**Theorem 132** (Theorem for Bringing a Formula into CNF)**.** *For any formula $\varphi \in \mathbb{PL}$, there exists a formula $\varphi' \in \mathbb{PL}$ that is in CNF such that $\varphi \equiv \varphi'$.*

**Proof:** [Proof Sketch]

By repeatedly applying the replacement theorem using the following equivalences:

1. $(\varphi_1 \leftrightarrow \varphi_2) \equiv ((\varphi_1 \rightarrow \varphi_2) \wedge (\varphi_2 \rightarrow \varphi_1))$;
2. $(\varphi_1 \rightarrow \varphi_2) \equiv (\neg\varphi_1 \vee \varphi_2)$;
3. $(\varphi_1 \vee (\varphi_2 \wedge \varphi_3)) \equiv ((\varphi_1 \vee \varphi_2) \wedge (\varphi_1 \vee \varphi_3))$;
4. $((\varphi_2 \wedge \varphi_3) \vee \varphi_1) \equiv ((\varphi_2 \vee \varphi_1) \wedge (\varphi_3 \vee \varphi_1))$;
5. $(\varphi_1 \vee (\varphi_2 \vee \varphi_3)) \equiv ((\varphi_1 \vee \varphi_2) \vee \varphi_3)$;
6. $(\varphi_1 \wedge (\varphi_2 \wedge \varphi_3)) \equiv ((\varphi_1 \wedge \varphi_2) \wedge \varphi_3)$;
7. $\neg(\varphi_1 \vee \varphi_2) \equiv (\neg\varphi_1 \wedge \neg\varphi_2)$;
8. $\neg(\varphi_1 \wedge \varphi_2) \equiv (\neg\varphi_1 \vee \neg\varphi_2)$;
9. $\neg\neg\varphi \equiv \varphi$.

The first two equivalences ensure the fact that all double implications and all implications disappear from the formula.

The third and the fourth equivalence ensure that in the tree of the formula, all disjunctions “go under” the conjunctions.

Equivalences 5 and 6 ensure the associativity of the chains of disjunctions and conjunctions, respectively (it allows us to write $\varphi_1 \vee \varphi_2 \vee \varphi_3 \vee \ldots \vee \varphi_n$ instead of $((((\varphi_1 \vee \varphi_2) \vee \varphi_3) \vee \ldots) \vee \varphi_n)$.

Equivalences 7 and 8 ensure that the negations end up “under” the conjunctions and disjunctions in the tree of the formula.

The last equivalence ensures that there are no double negations “one under the other” in the abstract syntax tree.

Applying the equivalences above eventually stops (they cannot be applied ad infinitum – why?).

The result will be a formula in which all conjunctions are “above” all disjunctions, which are “above” all possible negations, meaning a formula in CNF. q.e.d.

**Example 133.** *Let us bring the formula* $((\neg p \rightarrow \neg q) \leftrightarrow (q \rightarrow p))$ *into CNF*

$$\begin{aligned}
& ((\neg p \rightarrow \neg q) \leftrightarrow (q \rightarrow p)) \\
\overset{1}{\equiv}\ & \Big((\neg p \rightarrow \neg q) \rightarrow (q \rightarrow p)\Big) \wedge \Big((q \rightarrow p) \rightarrow (\neg p \rightarrow \neg q)\Big) \\
\overset{2}{\equiv}\ & \Big((\neg\neg p \vee \neg q) \rightarrow (\neg q \vee p)\Big) \wedge \Big((\neg q \vee p) \rightarrow (\neg\neg p \vee \neg q)\Big) \\
\overset{2}{\equiv}\ & \Big(\neg(\neg\neg p \vee \neg q) \vee (\neg q \vee p)\Big) \wedge \Big(\neg(\neg q \vee p) \vee (\neg\neg p \vee \neg q)\Big) \\
\overset{7}{\equiv}\ & \Big((\neg\neg\neg p \wedge \neg\neg q) \vee (\neg q \vee p)\Big) \wedge \Big((\neg\neg q \wedge \neg p) \vee (\neg\neg p \vee \neg q)\Big) \\
\overset{9}{\equiv}\ & \Big((\neg p \wedge q) \vee (\neg q \vee p)\Big) \wedge \Big((q \wedge \neg p) \vee (p \vee \neg q)\Big) \\
\overset{4}{\equiv}\ & \Big((\neg p \vee (\neg q \vee p)) \wedge (q \vee (\neg q \vee p))\Big) \wedge \Big((q \vee (p \vee \neg q)) \wedge (\neg p \vee (p \vee \neg q))\Big) \\
\overset{5}{\equiv}\ & \Big(((\neg p \vee \neg q) \vee p) \wedge ((q \vee \neg q) \vee p)\Big) \wedge \Big(((q \vee p) \vee \neg q) \wedge ((\neg p \vee p) \vee \neg q)\Big) \\
=\ & ((\neg p \vee \neg q \vee p) \wedge (q \vee \neg q \vee p)) \wedge ((q \vee p \vee \neg q) \wedge (\neg p \vee p \vee \neg q)) \\
\overset{6}{\equiv}\ & \Big(((\neg p \vee \neg q \vee p) \wedge (q \vee \neg q \vee p)) \wedge (q \vee p \vee \neg q)\Big) \wedge (\neg p \vee p \vee \neg q) \\
=\ & (\neg p \vee \neg q \vee p) \wedge (q \vee \neg q \vee p) \wedge (q \vee p \vee \neg q) \wedge (\neg p \vee p \vee \neg q).
\end{aligned}$$

## 8.7 Disjunctive Normal Form

**Definition 134** (DNF). *A formula is in DNF if it is a disjunction of conjunctions of literals.*

**Example 135.** *The following formulae are in DNF:*

1. $(\neg p \wedge q) \vee (r \wedge \neg p \wedge r') \vee (\neg p \wedge \neg r)$;
2. $\neg p \vee (r \wedge \neg p \wedge r') \vee \neg r$;
3. $\neg p \vee r \vee \neg r$;
4. $\neg p$;
5. $p$.

*The following formulae are not in DNF:*

1. $\neg(\neg p \wedge q) \vee (r \wedge \neg p \wedge r') \vee (\neg p \wedge \neg r)$;
2. $\neg p \vee \neg(r \wedge \neg p \wedge r')$;
3. $\neg p \wedge (r \vee \neg p \vee r')$;
4. $\neg\neg p$;
5. $p \wedge (q \vee r)$.

**Exercise 136.** *State and sketch the proof of a constructive theorem to bring a formula into DNF.*

## 8.8 The Link between DNF and CNF

**Definition 137.** *The complement of a formula $\varphi \in \mathbb{PL}_{\neg,\vee,\wedge}$ is denoted by $\varphi^c$ and is defined as follows:*

1. $a^c = \neg a$, *for any* $a \in A$;
2. $(\neg\varphi)^c = \varphi$, *for any* $\varphi \in \mathbb{PL}$;
3. $((\varphi_1 \vee \varphi_2))^c = ({\varphi_1}^c \wedge {\varphi_2}^c)$, *for any* $\varphi_1, \varphi_2 \in \mathbb{PL}$;
4. $((\varphi_1 \wedge \varphi_2))^c = ({\varphi_1}^c \vee {\varphi_2}^c)$, *for any* $\varphi_1, \varphi_2 \in \mathbb{PL}$.

**Example 138.**
1. $(\neg p)^c = p$ *(the complement of $\neg p$ is not $\neg\neg p$)*;
2. $(\neg(\neg p \wedge q)\vee(r \wedge \neg p \wedge r')\vee(\neg p \wedge \neg r))^c =$
   $(\neg p \wedge q)\wedge(\neg r \vee p \vee \neg r')\wedge(p \vee r)$;
3. $((\neg p \vee q) \wedge (r \vee \neg p \vee r') \wedge (\neg p \vee \neg r))^c =$
   $(p \wedge \neg q) \vee (\neg r \wedge p \wedge \neg r') \vee (p \wedge r)$.

**Theorem 139** (The Complement is Equivalent to the Negation)**.** *For any formula $\varphi \in \mathbb{PL}_{\neg,\wedge,\vee}$, we have that*

$$\varphi^c \equiv \neg\varphi.$$

**Exercise 140.** *Prove the theorem above by structural induction.*

**Example 141.** $(p \wedge (q \vee \neg r))^c = (\neg p \vee (\neg q \wedge r)) \equiv \neg(p \wedge (q \vee \neg r))$.

**Theorem 142** (The Link between CNFs and DNFs)**.** *Let $\varphi_1$ be a formula in CNF and $\varphi_2$ a formula in DNF. Then*

$${\varphi_1}^c \textit{ is in DNF}$$

*and*

$${\varphi_2}^c \textit{ is in CNF.}$$

## 8.9 Exercise Sheet

**Exercise 143.** *Write the following formulae with as few brackets as possible:*
*1.* $((p \wedge \neg q) \wedge r)$; *2.* $((p \vee \neg q) \wedge r)$; *3.* $\neg((p \vee \neg q) \wedge r)$.

**Exercise 144.** *Compute the abstract syntax tree for the following formulae (pay attention to the priority of the logical connectives):*
*1.* $p \wedge q \vee r$; *2.* $\neg p \wedge \neg q \vee \neg r$; *3.* $p \wedge \neg q \vee r$; *4.* $\neg p \wedge \neg(q \vee r)$; *5.* $p \vee \neg p \wedge \neg(q \vee r)$.

**Exercise 145.** *Bring the following formulae into CNF:*

1. $((p \wedge q) \vee r)$;
2. $((p \vee q) \wedge r)$;
3. $\neg((p \vee q) \wedge r)$;
4. $\neg((p \wedge q) \vee r)$;
5. $((p \wedge q) \vee (\neg p \wedge \neg q))$;
6. $((p \wedge (q \wedge r)) \vee \neg p)$;
7. $\neg(\neg(p \wedge q) \vee (p \vee q))$;
8. $(\neg(p \wedge q) \rightarrow (\neg p \wedge \neg q))$;
9. $(p \leftrightarrow (q \rightarrow (\neg p \wedge \neg q)))$;
10. $((p \rightarrow q) \leftrightarrow (\neg q \rightarrow \neg p))$;
11. $(p_1 \wedge q_1) \vee (p_2 \wedge q_2) \vee \ldots \vee (p_n \wedge q_n)$ *(first solve the cases* $n = 2$ *and* $n = 3$*, then generalize)*;

**Exercise 146.** *Bring into CNF the formula* $p \vee q \rightarrow p \wedge \neg r$ *(mind the priority of the logical connectives).*

**Exercise 147.** *Compute the complement of the formulae in CNF computed above.*

# Chapter 9

# Resolution in Propositional Logic

Natural deduction is a proof system that was invented by Gentzen to be *as close as possible to actual reasoning.*

However, a downside of natural deduction is that, given a sequent

$$\varphi_1, \varphi_2, \ldots, \varphi_n \vdash \varphi,$$

it is difficult for a computer program to find a formal proof of the sequent (i.e., it is not terribly efficient) due to the multitude of potential rules that can be applied.

Resolution is a proof system, just like natural deduction, but tailored to computers and not humans. In particular, it is easy for computers to find resolution proofs (at least, easier than finding natural deduction proofs). However, resolution proofs do not resemble human reasoning very well. There are programming languages (Prolog) that use (a variant of) resolution as their basic execution step.

## 9.1 Clauses as Sets of Literals

We have that disjunction is:

1. associative: for all $\varphi_1, \varphi_2, \varphi_3 \in \mathbb{PL}$,
   $(\varphi_1 \lor (\varphi_2 \lor \varphi_3)) \equiv ((\varphi_1 \lor \varphi_2) \lor \varphi_3)$;

2. commutative: for all $\varphi_1, \varphi_2 \in \mathbb{PL}$,
   $(\varphi_1 \lor \varphi_2) \equiv (\varphi_2 \lor \varphi_1)$;

3. idempotent: for all $\varphi \in \mathbb{PL}$, $(\varphi \lor \varphi) \equiv \varphi$.

**Exercise 148.** *Prove the three equivalences above.*

This means, for example, that the clauses $\mathtt{p} \lor \mathtt{p} \lor \mathtt{p} \lor \neg\mathtt{q} \lor \neg\mathtt{q}$, $\neg\mathtt{q} \lor \mathtt{p} \lor \neg\mathtt{q} \lor \mathtt{p}$, $\mathtt{p} \lor \neg\mathtt{q}$ are all equivalent. The three equivalences above justify the use of the notation:

**Notation.** *For any literals $\varphi_1, \ldots, \varphi_n$ the set*

$$\{\varphi_1, \varphi_2, \ldots, \varphi_n\}$$

*is a representation of the following formula:*

$$\varphi_1 \lor \varphi_2 \lor \ldots \lor \varphi_n.$$

That is, a clause is the set of its literals.

**Example 149.** *The clause $\mathtt{p} \lor \mathtt{q} \lor \neg\mathtt{r}$ is represented by the set $\{\mathtt{p}, \mathtt{q}, \neg\mathtt{r}\}$. This set of literals, $\{ \mathtt{p}, \mathtt{q}, \neg\mathtt{r} \}$, also represents the following clauses:*

1. $\mathtt{q} \lor \mathtt{p} \lor \neg\mathtt{r}$;
2. $\mathtt{q} \lor \mathtt{p} \lor \neg\mathtt{r} \lor \mathtt{p}$;
3. $\mathtt{p} \lor \mathtt{q} \lor \mathtt{p} \lor \neg\mathtt{r} \lor \mathtt{p}$;
4. *etc.*

*(Due to the fact that disjunction is associative, commutative, and idempotent, all these formulas are equivalent.)*

**Remark.** *Attention! A set of literals represents multiple clauses, but all these clauses are equivalent.*

**Exercise 150.** *Make sure you understand the notations above and can easily switch between notations.*

**Remark.** *Furthermore, we will allow a slight abuse of language and say* clause *to refer to a set of literals.*

*Also, we will allow using clauses seen as sets of literals instead of formulas when it is not relevant to which of the formulas represented by the set of literals we are referring.*

*For example, using the language abuse explained above, we have the fact that the clause $\{ \mathtt{p}, \neg\mathtt{p} \}$ is valid. Why? Because any formula represented by the clause $\{ \mathtt{p}, \neg\mathtt{p} \}$ (for example, $(\mathtt{p} \lor \neg\mathtt{p})$ or $((\mathtt{p} \lor \neg\mathtt{p}) \lor \mathtt{p})$) is true in any assignment.*

**Remark.** *Recall that the notation $\square$ is called the empty clause and represents the particular case of a disjunction of $0$ literals, thus representing any contradiction.*

## 9.2 CNFs as Sets of Clauses

Similarly to disjunction, conjunction also enjoys the following three properties:

1. associative: for all $\varphi_1, \varphi_2, \varphi_3 \in \mathbb{PL}$,
   $(\varphi_1 \land (\varphi_2 \land \varphi_3)) \equiv ((\varphi_1 \land \varphi_2) \land \varphi_3)$;
2. commutative: for all $\varphi_1, \varphi_2 \in \mathbb{PL}$,
   $(\varphi_1 \land \varphi_2) \equiv (\varphi_2 \land \varphi_1)$;
3. idempotent: for all $\varphi \in \mathbb{PL}$, $(\varphi \land \varphi) \equiv \varphi$.

**Exercise 151.** *Prove the three equivalences above.*

The three equivalences above justify the following notation:

**Notation.** *Given clauses $\varphi_1, \ldots, \varphi_n$, we write*

$$\{\varphi_1, \ldots, \varphi_n\}$$

*instead of the CNF formula*

$$\varphi_1 \land \ldots \land \varphi_n.$$

For example, we write $\{\mathtt{p} \lor \neg\mathtt{q}, \mathtt{q} \lor \mathtt{r} \lor \mathtt{p}\}$ instead of $(\mathtt{p} \lor \neg\mathtt{q}) \land (\mathtt{q} \lor \mathtt{r} \lor \mathtt{p})$.

Going further, we will combine the two notations above and write, for example, $\{\{\mathtt{p}, \neg\mathtt{q}\}, \{\mathtt{q}, \mathtt{r}, \mathtt{p}\}\}$ for $(\mathtt{p} \lor \neg\mathtt{q}) \land (\mathtt{q} \lor \mathtt{r} \lor \mathtt{p})$. That is, we write a CNF formula as a set of sets of literals.

**Exercise 152.** *Write the CNF formula $(\mathtt{p} \lor \mathtt{q})$ as a set of sets of literals.*

*Write the CNF formula $(\mathtt{p} \land \mathtt{q})$ as a set of sets of literals.*

## 9.3 Resolution

Resolution is a proof system with only one inference rule. The hypotheses and the conclusion of the inference rule are all clauses (unlike natural deduction, where the hypotheses and the conclusion were sequents).

Here is the resolution rule:

$$\textsc{Binary Resolution}\ \frac{C \cup \{a\} \qquad D \cup \{\neg a\}}{C \cup D}$$

In the rule above, $C$ and $D$ denote arbitrary clauses and $a \in A$ denotes an arbitrary propositional variable. As $C$ is a clause and $a \in A$ is a propositional variable, it follows that $C \cup \{a\}$ is also a clause. In fact $C \cup \{a\}$ is the first hypothesis of the inference rule.

The second hypothesis is the clause $D \cup \{\neg a\}$. The conclusion is the clause $C \cup D$.

Here is an example of applying the resolution rule:

1. $\{\mathtt{p}, \mathtt{q}\}$; (premiss)
2. $\{\neg\mathtt{p}, \neg\mathtt{r}\}$; (premiss)
3. $\{\mathtt{q}, \neg\mathtt{r}\}$. (by Resolution, lines 1, 2, $a = \mathtt{p}$)

**Definition 153.** *The clause $C \cup D$ resulting from applying resolution is called the* resolvent *of $C \cup \{a\}$ and $D \cup \{\neg a\}$.*

Note that the resolvent of two clauses is not unique. For example, the clauses $\{\mathtt{p}, \mathtt{q}\}$ and $\{\neg\mathtt{p}, \neg\mathtt{q}, \mathtt{r}\}$ have two resolvents: $\{\mathtt{p}, \neg\mathtt{p}, \mathtt{r}\}$ and respectively $\{\mathtt{q}, \neg\mathtt{q}, \mathtt{r}\}$. We may distinguish among the various resolvents by stating *on which propositional variable $a$* we have performed the resolution:

**Definition 154.** *The clause $C \cup D$ resulting from applying resolution is called the* resolvent *of $C \cup \{a\}$ and $D \cup \{\neg a\}$* on $a$.

For example, $\{\mathtt{p}, \neg\mathtt{p}, \mathtt{r}\}$ is the resolvent on $\mathtt{q}$ of the clauses $\{\mathtt{p}, \mathtt{q}\}$ and $\{\neg\mathtt{p}, \neg\mathtt{q}, \mathtt{r}\}$, while $\{\mathtt{q}, \neg\mathtt{q}, \mathtt{r}\}$ is their resolvent on $\mathtt{p}$.

**Remark.** *Note that when we apply inference rules, we must follow them to the letter. For example, a common mistake in applying resolution is:*

*1.* $\{\mathtt{p}, \mathtt{q}, \mathtt{r}\}$*;* *(premiss)*
*2.* $\{\neg\mathtt{p}, \neg\mathtt{q}\}$*;* *(premiss)*
*3.* $\{\mathtt{r}\}$*.* *(***wrong** *application of resolution)*

*On line* 3*, we may conclude either the resolvent on* $\mathtt{p}$ *or on* $\mathtt{q}$*, but it makes no sense to mix the two in order to have a single resolvent on both* $\mathtt{p}$ *and* $\mathtt{q}$*. Make sure you understand this point very well.*

*Here is the first correct variant:*

*1.* $\{\mathtt{p}, \mathtt{q}, \mathtt{r}\}$*;* *(premiss)*
*2.* $\{\neg\mathtt{p}, \neg\mathtt{q}\}$*;* *(premiss)*
*3.* $\{\mathtt{q}, \mathtt{r}, \neg\mathtt{q}\}$*;* *(resolvent of 1, 2 on $a = \mathtt{p}$)*

*and the second one:*

*1.* $\{\mathtt{p}, \mathtt{q}, \mathtt{r}\}$*;* *(premiss)*
*2.* $\{\neg\mathtt{p}, \neg\mathtt{q}\}$*;* *(premiss)*
*3.* $\{\mathtt{p}, \mathtt{r}, \neg\mathtt{p}\}$*.* *(resolvent of 1, 2 on $a = \mathtt{q}$)*

## 9.4 Formal Proofs

Just as in the case of natural deduction,

**Definition 155.** *A* formal proof *of a clause $\varphi$ from a set of clauses $\varphi_1, \varphi_2, \ldots, \varphi_n$ is a sequence of clauses $\psi_1, \psi_2, \ldots, \psi_m$ such that, for all $1 \leq i \leq m$:*

- *either $\psi_i \in \{\varphi_1, \ldots, \varphi_n\}$,*
- *or $\psi_i$ is obtained by resolution from some clauses $\psi_j, \psi_k$ that appear earlier in the formal proof: $1 \leq j, k < i$.*

*Furthermore $\psi_m$ must be the same as $\varphi$.*

The clauses $\varphi_1, \ldots, \varphi_n$ are called the premisses of the formal proof and $\varphi$ is the conclusion.

**Example 156.** *Here is an example of a proof of $\{\mathtt{p}, \neg\mathtt{q}\}$ from $\{\neg\mathtt{r}, \mathtt{p}, \neg\mathtt{r}'\}$, $\{\mathtt{r}, \mathtt{p}\}$ and $\{\mathtt{r}', \neg\mathtt{q}\}$:*

1. $\{\neg\mathtt{r}, \mathtt{p}, \neg\mathtt{r}'\}$; *(premiss)*
2. $\{\mathtt{r}, \mathtt{p}\}$; *(premiss)*
3. $\{\mathtt{r}', \neg\mathtt{q}\}$; *(premiss)*
4. $\{\mathtt{p}, \neg\mathtt{r}'\}$; *(B.R., 2, 1, $a = \mathtt{r}$)*
5. $\{\mathtt{p}, \neg\mathtt{q}\}$. *(resolution, 3, 4, $a = \mathtt{r}'$)*

We sometimes also say *derivation (by resolution) of $\varphi$* instead of *formal proof of $\varphi$*.

**Example 157.** *Here is a derivation of the empty clause from $\{\mathtt{p}, \neg\mathtt{q}\}, \mathtt{q}, \neg\mathtt{p}$:*

1. $\{\mathtt{p}, \neg\mathtt{q}\}$; *(premiss)*
2. $\{\mathtt{q}\}$; *(premiss)*
3. $\{\neg\mathtt{p}\}$; *(premiss)*
4. $\{\mathtt{p}\}$; *(B.R., 2, 1, $a = \mathtt{q}$)*
5. $\square$. *(B.R., 4, 3, $a = \mathtt{p}$)*

**Definition 158** (Refutation)**.** *A derivation of $\square$ from $\varphi_1, \ldots, \varphi_n$ is sometimes also called a* refutation of *$\varphi_1, \ldots, \varphi_n$.*

**Exercise 159.** *Starting with the same premisses, find a different proof by resolution of $\square$.*

## 9.5 Soundness

Like natural deduction, resolution is sound. This section shows that this is indeed the case.

**Lemma 160.** *Let $\varphi \in \{\varphi_1, \ldots, \varphi_n\}$. We have that*

$$\varphi_1, \ldots \varphi_n \models \varphi.$$

**Exercise 161.** *Prove Lemma 160.*

**Lemma 162.** *Let $C, D$ be two clauses and let $a \in A$ be a propositional variable. We have that*

$$C \cup \{a\}, D \cup \{\neg a\} \models C \cup D.$$

**Exercise 163.** *Prove Lemma 162.*

**Theorem 164** (Soundness of Resolution)**.** *If there is a proof by resolution of $\varphi$ from $\varphi_1, \ldots, \varphi_n$, then*

$$\varphi_1, \ldots, \varphi_n \models \varphi.$$

**Proof:** Let $\psi_1, \ldots, \psi_m$ be a proof by resolution of $\varphi$ from $\varphi_1, \ldots, \varphi_n$.

We will prove by induction on $i \in \{1, 2, \ldots, m\}$ that

$$\varphi_1, \ldots, \varphi_n \models \psi_i.$$

Let $i \in \{1, 2, \ldots, m\}$ be an integer. We assume by the induction hypothesis that

$$\varphi_1, \ldots, \varphi_n \models \psi_l \text{ for any } l \in \{1, 2, \ldots, i-1\}$$

and we prove that

$$\varphi_1, \ldots, \varphi_n \models \psi_i.$$

By the definition of a formal proof by resolution, we must be in one of the following two cases:

1. $\psi_i \in \{\varphi_1, \ldots, \varphi_n\}$. In this case we have

   $$\varphi_1, \ldots, \varphi_n \models \psi_i$$

   by Lemma 160, which is what we had to show.

2. $\psi_i$ was obtained by resolution from $\psi_j, \psi_k$ with $1 \leq j, k < i$. In this case, $\psi_j$ must be of the form $\psi_j = C \cup \{a\}$, $\psi_k$ must be of the form $\psi_k = D \cup \{\neg a\}$ and $\psi_i = C \cup D$, where $C, D$ are clauses and $a \in A$ is a propositional variable.

By the induction hypotheses that $\varphi_1, \ldots, \varphi_n \models \psi_j$ and that $\varphi_1, \ldots, \varphi_n \models \psi_k$. Replacing $\psi_j$ and $\psi_k$ as detailed above, we have that

$$\varphi_1, \ldots, \varphi_n \models C \cup \{a\}$$

and that

$$\varphi_1, \ldots, \varphi_n \models D \cup \{\neg a\}.$$

We prove that

$$\varphi_1, \ldots, \varphi_n \models C \cup D.$$

Let $\tau$ be a model of $\varphi_1$, …, and $\varphi_n$. We have that $\tau$ is a model of $C \cup \{a\}$ and of $D \cup \{\neg a\}$ by the semantical consequences above. By Lemma 162, it follows that $\tau$ is a model of $C \cup D$. But $\psi_i = C \cup D$ and therefore $\tau$ is a model $\psi_i$. As $\tau$ was any model of all of $\varphi_1$, …, and $\varphi_n$, it follows that

$$\varphi_1, \ldots, \varphi_n \models \psi_i,$$

which is what we had to prove.

In both cases, we have established

$$\varphi_1, \ldots, \varphi_n \models \psi_i$$

for all $1 \leq i \leq m$. As $\psi_1, \ldots, \psi_m$ is a proof of $\varphi$, it follows that $\psi_m = \varphi$ and therefore, for $i = m$, we have

$$\varphi_1, \ldots, \varphi_n \models \varphi,$$

which is what we had to prove.

q.e.d.

## 9.6 Completeness

A proof system must be sound in order to be of any use (otherwise, we could use it to prove something false).

However, it is also nice when a proof system is complete, in the sense of allowing to prove any true statement.

Unfortunately, resolution is not complete, as shown by the following example:

**Theorem 165** (Incompleteness of Resolution)**.** *There exist clauses $\varphi_1, \ldots, \varphi_n, \varphi$ such that*

$$\varphi_1, \ldots, \varphi_n \models \varphi,$$

*but there is no resolution proof of $\varphi$ from $\varphi_1, \ldots, \varphi_n$.*

**Proof:** Let $n = 2$, $\varphi_1 = \mathtt{p}$, $\varphi_2 = \mathtt{q}$ and $\varphi = \{\mathtt{p}, \mathtt{q}\}$. We clearly have that $\mathtt{p}, \mathtt{q} \models (\mathtt{p} \vee \mathtt{q})$, but there is no way to continue the following resolution proof:

1. $\{\mathtt{p}\}$; (premiss)
2. $\{\mathtt{q}\}$; (premiss)
3. $\cdots$

because there is no negative literal anywhere. Therefore, only $\mathtt{p}$ and $\mathtt{q}$ can be derived by resolution from $\mathtt{p}, \mathtt{q}$. q.e.d.

However, resolution still has a weaker form of completeness called refutational completeness:

**Theorem 166** (Refutational Completeness of Resolution)**.** *If the CNF formula* $\varphi_1 \wedge \ldots \wedge \varphi_n$ *is unsatisfiable, then there is a derivation by resolution of* $\square$ *starting from the clauses* $\varphi_1, \varphi_2, \ldots, \varphi_n$.

The proof is beyond the scope of the course, but it is not too complicated in case you want to prove it yourself and I recommend this exercise for the more curious minds among you.

## 9.7 Proving Validity and Logical Consequences by Resolution

We may use the soundness and refutational completeness of resolution to construct an algorithm for validity checking and logical consequence testing.

**Validity Testing** Here is an example of how to test the validity of the formula $\mathtt{p} \vee \mathtt{q} \rightarrow \mathtt{q} \vee \mathtt{p}$.

First of all, recall that:

**Theorem 167.** *A formula is valid iff its negation is a contradiction.*

**Exercise 168.** *Prove the theorem above.*

Therefore, establishing validity of $\mathtt{p} \vee \mathtt{q} \rightarrow \mathtt{q} \vee \mathtt{p}$ is equivalent to establishing unsatisfiability of $\neg(\mathtt{p} \vee \mathtt{q} \rightarrow \mathtt{q} \vee \mathtt{p})$.

A formula is satisfiable iff its CNF is satisfiable.

Let us compute a CNF of $\neg(\mathtt{p} \vee \mathtt{q} \rightarrow \mathtt{q} \vee \mathtt{p})$:

$$\begin{aligned}
\neg(\mathtt{p} \vee \mathtt{q} \rightarrow \mathtt{q} \vee \mathtt{p}) &\equiv \neg(\neg(\mathtt{p} \vee \mathtt{q}) \vee (\mathtt{q} \vee \mathtt{p})) \\
&\equiv \neg\neg(\mathtt{p} \vee \mathtt{q}) \wedge \neg(\mathtt{q} \vee \mathtt{p}) \\
&\equiv (\mathtt{p} \vee \mathtt{q}) \wedge (\neg\mathtt{q}) \wedge (\neg\mathtt{p}).
\end{aligned}$$

We have reached a CNF. Starting with the clauses that make up the CNF, we can derive $\square$:

1. $\{p, q\}$; (premiss)
2. $\{\neg q\}$; (premiss)
3. $\{\neg p\}$; (premiss)
4. $\{p\}$; (resolution, 1, 2, $a = q$)
5. $\square$. (resolution, 4, 3, $a = p$)

We may reason by applying the following corrolary of the soundness theorem:

**Corollary 169.** *If $\square$ is derivable by resolution from $\varphi_1, \ldots, \varphi_n$, then the set of clauses $\{\varphi_1, \ldots, \varphi_n\}$ is inconsistent.*

**Proof:** By the soundness theorem, we have that $\varphi_1, \ldots \varphi_n \models \square$. Assume by contradiction that $\{\varphi_1, \ldots, \varphi_n\}$ is consistent; then there is a $\tau : A \to B$ such that $\hat{\tau}\big(\varphi_1\big) = \ldots = \hat{\tau}\big(\varphi_n\big) = 1$. As $\varphi_1, \ldots \varphi_n \models \square$, we also have that $\hat{\tau}\big(\square\big) = 1$. But, by definition, there is no such assignment (that makes $\square$ true), and therefore our assumption must have been false. Therefore $\{\varphi_1, \ldots, \varphi_n\}$ is not consistent. q.e.d.

Continuing our example, our set of clauses is inconsistent, which means that $\neg(p \vee q \to q \vee p)$ is unsatisfiable, which further means that the formula $(p \vee q \to q \vee p)$ is valid (what we had to show in the first place).

**Testing Logical Consequence** How to prove by resolution that $\varphi_1, \ldots, \varphi_n \models \varphi$?

We use the following theorem, which reduces logical consequence to validity:

**Theorem 170.** *Let $\varphi_1, \ldots, \varphi_n, \varphi$ be any propositional formulae. We have that*

$$\varphi_1, \ldots, \varphi_n \models \varphi$$

*iff*

$$\varphi_1 \wedge \ldots \wedge \varphi_n \to \varphi$$

*is valid.*

**Exercise 171.** *Prove the theorem above.*

To establish a logical consequence by resolution, first apply the theorem above and then apply the method shown above for proving validity.

## 9.8 Exercise Sheet

**Exercise 172.** *Using resolution, show that the following formulae are valid:*

1. $((p \land q) \rightarrow (p \lor q))$;
2. $(p \rightarrow (q \rightarrow p))$;
3. $((p \rightarrow q) \rightarrow (\neg q \rightarrow \neg p))$;

**Exercise 173.** *Using resolution, show the following semantical consequences:*

1. $p \models (p \lor q)$;
2. $(p \land q) \models (p \lor q)$;
3. $((p \land q) \rightarrow r) \models (p \rightarrow (q \rightarrow r))$;
4. $(p \rightarrow p'), (q \rightarrow q') \models ((p \land q) \rightarrow (p' \lor q'))$.