Chapter I of "Logic and Complexity" by Lassaigne & de Rougemont. Springer.

CHAPTER 1

Propositional logic

Propositional logic, or propositional calculus, is an elementary mathematical system that constitutes a minimal kernel common to all logical systems. It plays the role of a simplified construction which will be generalized to more expressive systems. We study the construction of this language and its **semantic** interpretation, i.e. the objects denoted by expressions of the language.

In the first section, we define the set of propositional **formulas**. The second section is dedicated to the interpretation of formulas in terms of truth values (true, false). This interpretation gives rise to notions of *equivalent formulas* and *logical consequence*. The equivalence relation between formulas captures the main properties of propositional logic. The *normal forms* are presented in the third section. Ordered binary decision diagrams (**OBDD**) are defined in the fourth section. They provide a data structure to represent boolean functions and are used in some verification tools, such as model checkers.

1.1. Propositional language

The purpose of this section is to define the set of propositional formulas and to show how this type of definition – present in all branches of mathematical logic – can be used to study properties of formulas in a chosen language.

1.1.1. Construction of formulas. The propositional language is characterized by a collection of symbols, called *alphabet* $\mathcal{A}$, which includes:

- a set $\mathcal{P} = \{p, q, r, \ldots\}$, finite or countable, of symbols called *propositional variables*,
- the set of *connectives* (or logical symbols), which are $\neg$ (*not*), $\wedge$ (*and*), $\vee$ (*or*), $\rightarrow$ (*imply*), $\leftrightarrow$ (*equivalent to*),
- the parentheses (and).

The connective $\neg$ is *unary* and the others are *binary*. A *word*, or an *expression*, is a finite sequence of elements of $\mathcal{A}$. The *length* of a word is equal to the number of symbols composing it. The set of words constructed with the help of alphabet $\mathcal{A}$ is denoted by $\mathcal{A}^*$.

Concatenation is a composition rule defined on $\mathcal{A}^*$ which associates with the two words u, v the word obtained by juxtaposing the sequence u with v: the new word is denoted by uv. A word u is an initial segment of a word v if a word w exists such that $v = uw$. The relation defined on $\mathcal{A}^*$ by "u is an initial segment of v" is an order relation.

> **Example.** The sequences of symbols $\neg p$, $(\neg p \wedge (q \vee r))$ and $(p \wedge \vee qr)$ are words belonging to $\mathcal{A}^*$.
> The word $(\neg p$ is an initial segment of $(\neg p \wedge (q \vee r))$.

From the logical point of view only some of the words of $\mathcal{A}^*$, are interesting: these are what we call **formulas**. In the previous example, only the first two expressions are formulas, in contrast to the last one.

DEFINITION 1.1. *The set of propositional formulas, built with $\mathcal{P}$, is the smallest set $\mathcal{F}$ such that:*

- *all propositional variables are in $\mathcal{F}$,*
- *if $F \in \mathcal{F}$, then $\neg F \in \mathcal{F}$,*
- *if $F, G \in \mathcal{F}$, then $(F \wedge G) \in \mathcal{F}$, $(F \vee G) \in \mathcal{F}$, $(F \rightarrow G) \in \mathcal{F}$ and $(F \leftrightarrow G) \in \mathcal{F}$.*

The set $\mathcal{F}$ is well defined. There are sets satisfying these three conditions: for example, the set $\mathcal{A}^*$ consisting of all words. Among all these sets, there is one set smaller than all the others: their intersection. This intersection is not empty because it contains the set $\mathcal{P}$ of propositional variables. The set of formulas can also be characterized in another way, by using the **induction principle**.

Let P be a property depending on non-negative integers. If P satisfies:

- P is true for 0 (respectively for the non-negative integer n_0),
- if P is true for n, then it is true for $n+1$,

then P is true for all n (respectively, for all $n \geq n_0$).

DEFINITION 1.2. *The sets $\mathcal{F}_n$ are defined by induction on n:*

- $\mathcal{F}_0 = \mathcal{P}$,
- $\mathcal{F}_{n+1} = \mathcal{F}_n \cup \{\neg F : F \in \mathcal{F}_n\} \cup \{(F \,\alpha\, G) : F, G \in \mathcal{F}_n\}$
  where α is $\wedge, \vee, \rightarrow$ or $\leftrightarrow$.

It is easy to see that the sequence $(\mathcal{F}_n)_{n \in \mathbb{N}}$ is increasing (exercise).

PROPOSITION 1.1. *The set $\mathcal{F}$ of propositional formulas is equal to $\bigcup_{n \in N} \mathcal{F}_n$.*

Proof : The set $\bigcup_{n \in N} F_n$ satisfies the conditions of the definition of $\mathcal{F}$:

- all propositional variables are in $\mathcal{F}_0$;
- if $F \in \mathcal{F}_n$, then $\neg F \in \mathcal{F}_{n+1}$;
- if $F, G \in \bigcup_{n \in N} \mathcal{F}_n$, then there exist n, m such that $F \in \mathcal{F}_n$ and $G \in \mathcal{F}_m$; if $p = sup(n, m)$, $F, G \in \mathcal{F}_p$ and the formulas $(F \wedge G)$, $(F \vee G)$, $(F \rightarrow G)$ and $(F \leftrightarrow G)$ are in $\mathcal{F}_{p+1}$.

Therefore, the set $\bigcup_{n \in N} \mathcal{F}_n$ contains $\mathcal{F}$, which is the smallest set satisfying these conditions. In order to obtain the inclusion in the opposite direction, it is sufficient to show that for every non-negative n, $\mathcal{F}_n \subset \mathcal{F}$. This property can be proved by induction on n:

- $\mathcal{F}_0 = P \subset \mathcal{F}$;
- assume that $\mathcal{F}_n \subset \mathcal{F}$ by induction hypothesis. From the definition of $\mathcal{F}_{n+1}$ and the fact that the set $\mathcal{F}$ is closed under all connectives, it follows that $\mathcal{F}_{n+1} \subset \mathcal{F}$.

This concludes the proof.□

DEFINITION 1.3. *The* **rank** *of a formula F is the smallest non-negative integer n such that $F \in \mathcal{F}_n$.*

> **Example.** The formula $F = (\neg p \wedge ((q \vee r) \rightarrow s))$ has rank 3.
> p, q, r, s have rank 0,
> $\neg p$, $(q \vee r)$ have rank 1,
> $((q \vee r) \rightarrow s)$ has rank 2.

1.1.2. Proof by induction. In this section numerous results will have the form: let P be a property of formulas; then the set of all propositional calculus formulas having the property P is equal to $\mathcal{F}$. In order to prove these results, we will not use a reasoning by induction on non-negative integers but instead, a proof by induction on the formulas. This type of proof is justified by the following proposition:

PROPOSITION 1.2. *Let P be a property of formulas, satisfying the following conditions:*

- *all propositional variables have the property P,*
- *if G is a formula with the property P, then the formula $\neg G$ has the property P,*
- *if G, H are formulas with the property P, then all formulas $(G \wedge H)$, $(G \vee H)$, $(G \rightarrow H)$, $(G \leftrightarrow H)$ have this property.*

Then all propositional formulas have the property P.

Proof : Let $\mathcal{E}$ be a set of formulas in $\mathcal{F}$ with property P. In order to deduce the equality $\mathcal{F} = \mathcal{E}$ it is sufficient to show that $\mathcal{F} \subset \mathcal{E}$. According to the hypothesis, the set $\mathcal{E}$ contains all propositional variables and is closed under the application of operators. Therefore it contains $\mathcal{F}$, which is the smallest set verifying these conditions. □

The following proposition is a simple example of proof by induction on the set of propositional formulas and is left as an exercise.

PROPOSITION 1.3. *Every formula has exactly the same number of opening and closing parentheses.*

1.1.3. Decomposition of a formula. The following proposition provides an answer to the question: given a particular formula, are there different ways to decompose it in "simpler formulas" ?

PROPOSITION 1.4. *Let F be a formula. Then F has one and only one of the following forms:*

(1) *a propositional variable,*
(2) $\neg G$, *where* G *is a formula,*
(3) $(G \wedge H)$, *where* G, H *are formulas,*
(4) $(G \vee H)$, *where* G, H *are formulas,*
(5) $(G \rightarrow H)$, *where* G, H *are formulas,*
(6) $(G \leftrightarrow H)$, *where* G, H *are formulas.*

Furthermore, in cases 2, 3, 4, 5 and 6, the formulas G, H *are uniquely determined.*

The existence of a decomposition is easily obtained from the definition of the set of formulas. The uniqueness is more difficult to establish and requires two intermediary results, which are stated in the exercises.

DEFINITION 1.4. *A* **subformula** *of* F *is a formula which appears in the decomposition of* F.

We define the notion of tree which will be used for the representation of the decomposition of a formula.

DEFINITION 1.5. *A* **tree** *is a set* T *provided with an application* $h : T \longrightarrow \mathbb{N}$ *and a binary relation* $P \subseteq T^2$ *satisfying the following conditions :*

- *there is a unique element* r *of* T*, called the root, such that* $h(x) = 0$,
- *for any element* y *of* T*, except the root, there is a unique element* x *such as* $(x, y) \in P$*, what we also note* $P(x, y)$,
- *for any* $x \in T$*, if* $(x, y) \in P$*, then* $h(y) = h(x) + 1$.

The elements of T are called *nodes*. For any $x \in T$, $h(x)$ is called the *level* of x. If $(x, y) \in P$, x is said the *predecessor* or the *father* of y, and y a *successor* or a *son* of x. A node without successor is a leaf. The decomposition of formulas justifies the following method which allows one:

- to decide whether a given expression is a formula,
- in the case of a positive answer, to construct a **decomposition tree** for this formula, i.e. a tree whose nodes are labelled by subformulas which occur in the expression. If F is a propositional variable (case 1 of the decomposition), the corresponding node a is a leaf. In case 2 of the decomposition, the node a has one successor labelled by G. In cases 3,4,5 and 6 of the decomposition, the node a labelled by a formula F has two descendants nodes labelled by the formulas G and H.

Example. Is the following expression a formula?

$$F = (((\neg p \leftrightarrow q) \vee \neg(r \wedge s)) \rightarrow p)$$

F has the form $(F_1 \rightarrow F_2)$ where $F_2 = p$ is a propositional variable,
F_1 has the form $(F_3 \vee F_4)$,
F_3 has the form $(F_5 \leftrightarrow F_6)$ where $F_5 = \neg F_7$, $F_6 = q$ and $F_7 = p$ are propositional variables,
F_4 has the form $\neg F_8$ and F_8 and is of the form $(F_9 \wedge F_{10})$, where $F_9 = r$ and

$F_{10} = s$ are propositional variables.

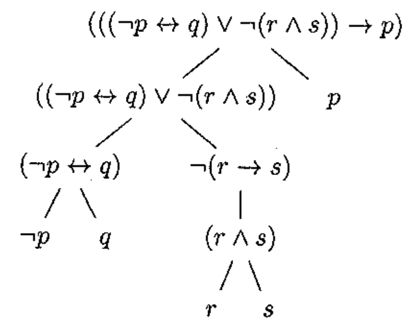

FIGURE 1.1. A tree decomposition.

Hence the expression F is a formula. Furthermore, its tree decomposition shows how it is constructed. Propositional variables are the labels of the leaves and the subformulas are the labels of the nodes. The formula F is the label of the root of the tree. The tree decomposition of F is presented in the Figure 1.1.

Example. The formulas $(\neg p \leftrightarrow q)$, $(r \wedge s)$ and p are subformulas of the formula F in the previous example.

1.2. Semantics

Semantics is defined as the interpretation of formulas in terms of their truth values (true, false). A **valuation**, that is a distribution of truth values on the set of propositional variables, allows the determination of the truth value of the formula. In what follows, **0** represents the value *false* and **1** the value *true*. We use *iff* as an abbreviation for *if and only if*.

DEFINITION 1.6. *A valuation* $\mathcal{V}$ *is a function from the set of propositional variables* $\mathcal{P}$ *into* $\{0, 1\}$.

PROPOSITION 1.5. *Let* $\mathcal{V}$ *be a valuation. There is a unique extension* $\overline{\mathcal{V}}$ *of* $\mathcal{V}$ *on* $\mathcal{F}$ *satisfying the following conditions:*

(1) *for all* $p \in \mathcal{P}$, $\overline{\mathcal{V}}(p) = \mathcal{V}(p)$,
(2) *if* F *is* $\neg G$*, then* $\overline{\mathcal{V}}(F) = 1$ *iff* $\overline{\mathcal{V}}(G) = 0$,
(3) *if* F *is* $(G \wedge H)$*, then* $\overline{\mathcal{V}}(F) = 1$ *iff* $\overline{\mathcal{V}}(G) = \overline{\mathcal{V}}(H) = 1$,
(4) *if* F *is* $(G \vee H)$*, then* $\overline{\mathcal{V}}(F) = 0$ *iff* $\overline{\mathcal{V}}(G) = \overline{\mathcal{V}}(H) = 0$,

(5) *if* F *is* $(G \to H)$, *then* $\overline{\mathcal{V}}(F) = 0$ *iff* $\overline{\mathcal{V}}(G) = 1$ *and* $\overline{\mathcal{V}}(H) = 0$,
(6) *if* F *is* $(G \leftrightarrow H)$, *then* $\overline{\mathcal{V}}(F) = 1$ *iff* $\overline{\mathcal{V}}(G) = \overline{\mathcal{V}}(H)$.

Proof : The distribution $\overline{\mathcal{V}}$ is defined by induction on formulas.

- Case 1 gives the definition for propositional variables.
- If F is $\neg G$ and $\overline{\mathcal{V}}(G)$ is already defined (induction hypothesis), we put $\overline{\mathcal{V}}(F) = 1$ if $\overline{\mathcal{V}}(G) = 0$ and $\overline{\mathcal{V}}(F) = 0$ otherwise.
- If F is $(G \wedge H)$ and $\overline{\mathcal{V}}(G), \overline{\mathcal{V}}(H)$ are already defined (induction hypothesis), we put $\overline{\mathcal{V}}(F) = 1$ if $\overline{\mathcal{V}}(G) = \overline{\mathcal{V}}(H) = 1$ and $\overline{\mathcal{V}}(F) = 0$ otherwise.

The proofs for the other cases are similar: the values $\overline{\mathcal{V}}(G)$ and $\overline{\mathcal{V}}(H)$ allow one to define $\overline{\mathcal{V}}(F)$ satisfying conditions (4), (5) or (6), respectively. The function $\overline{\mathcal{V}}$ is well defined in a unique way according to the decomposition. The uniqueness of the extension of $\mathcal{V}$ is left as an exercise: if we suppose that there are two extensions, it is easy to show by induction on formulas, that they are equal. □

Example. The value of the formula $((p \to q) \wedge (q \vee r))$ for the valuation of $\mathcal{V}$ defined by $\mathcal{V}(p) = \mathcal{V}(q) = 0$ and $\mathcal{V}(r) = 1$ is 1.

One way to represent the conditions stated in the previous proposition is to construct a table giving the values of $\mathcal{V}(F)$, as a function of the different possible values of $\mathcal{V}$, from the immediate subformulas of F. It is easy to construct **truth tables** for the binary operators $\wedge, \vee, \to$:

G	H	$G \wedge H$	$G \vee H$	$G \to H$
0	0	0	0	1
0	1	0	1	1
1	0	0	1	0
1	1	1	1	1

Henceforth, each valuation $\mathcal{V}$ given on $\mathcal{P}$ is extended to the set of all formulas, $\mathcal{F}$, and we also denote its extension by $\mathcal{V}$.

1.2.1. Tautologies. Equivalent formulas. The interpretation of formulas allows their classification: two formulas with the same interpretation will be grouped in the same class. A particularly interesting class is the class of formulas which are always true.

DEFINITION 1.7.

- *A formula F is* **satisfied** *by a valuation $\mathcal{V}$ if $\mathcal{V}(F) = 1$.*
- *A* **tautology** *is a formula satisfied by all valuations.*
- *Two formulas F, G are said to be* **equivalent** *if for every valuation $\mathcal{V}$, $\mathcal{V}(F) = \mathcal{V}(G)$; we write $F \equiv G$.*

Example. The following formulas are examples of tautologies:

$$(p \to p)$$

$$(p \to (q \to p))$$
$$((p \to (q \to r)) \to ((p \to q) \to (p \to r)))$$
$$((\neg p \to q) \to ((\neg p \to \neg q) \to p))$$

The following pairs of formulas are examples of equivalences:

$$\neg\neg p \text{ and } p$$
$$(p \to q) \text{ and } (\neg p \vee q)$$
$$(p \leftrightarrow q) \text{ and } ((p \to q) \wedge (q \to p))$$
$$(p \wedge (q \vee r)) \text{ and } ((p \wedge q) \vee (p \wedge r))$$

We note that two formulas F, G are equivalent iff the formula $(F \leftrightarrow G)$ is a tautology. The binary relation $\equiv$ defined on the set of formulas by: $F \equiv G$ iff F, G are equivalent, is an equivalence relation (exercise).

1.2.2. Logical consequence. From the point of view of semantics, one of the fundamental questions is to determine whether one formula is a consequence of a given set of formulas.

DEFINITION 1.8. *Let Σ be a set of formulas and F a formula.*

- *A formula F is said to be a* **consequence** *of Σ if every valuation satisfying all formulas of Σ, also satisfies the formula F.*
- *A set of formulas Σ is said to be* **satisfiable** *if there exists a valuation which satisfies all formulas of Σ.*

Example. The formula q is a consequence of the set $\{p, (p \to q)\}$.
The set of formulas $\{p, (p \to q), \neg q\}$ is not satisfiable.

PROPOSITION 1.6. *Any formula F is a consequence of the set of formulas Σ iff the set $\Sigma \cup \{\neg F\}$ is not satisfiable.*

Proof : If every valuation satisfying Σ also satisfies F, then there is no valuation satisfying both Σ and $\neg F$. The converse is easily shown by contraposition: if there is a valuation satisfying Σ and not satisfying F, then this valuation satisfies both Σ and $\neg F$. □

1.2.3. Value of a formula and substitution. The value of a formula, for example $((p \to q) \wedge (q \vee r))$ can be determined for any given valuation $\mathcal{V}$. But how can we calculate the truth value of a complex formula using truth values of simpler formulas? In this paragraph we answer this question: it is sufficient to compose truth values as in the case of propositional variables. In the first reading, it is possible to omit the general case treated within the theorem and its corollary. In fact, the study of properties stated in the following examples are sufficient for the construction of normal forms.

The notation $F(p_1, p_2, \ldots, p_n)$ specifies that the formula F contains propositional variables among $p_1, p_2, \ldots, p_n$. The following proposition expresses a rather intuitive property: in order to compute the truth value of a formula, it is sufficient to check the values taken by the propositional variables involved in this formula.

PROPOSITION 1.7. *Let $F(p_1, \ldots, p_n)$ be some formula and $\mathcal{V}$ a valuation. Then the value $\mathcal{V}(F)$ depends only on the values $\mathcal{V}$ on $p_1, \ldots, p_n$.*

Proof : By induction on formulas.

- If F is a propositional variable p_1, the statement is true.
- Suppose that the values $\mathcal{V}(G), \mathcal{V}(H)$ only depend on those of $\mathcal{V}$ on $p_1, \ldots, p_n$ (induction hypothesis). If F is $\neg G$ then the property is equally true for F since $\mathcal{V}(F)$ only depends on those of $\mathcal{V}(G)$. If F is $(G \wedge H)$, the property is still true for F since $\mathcal{V}(F)$ only depends on $\mathcal{V}(G), \mathcal{V}(H)$. The proof is similar for all other cases.

□

DEFINITION 1.9. *An* **occurrence** *of the variable p in some formula F is a position where it appears in F.*

Let G be a formula. The formula obtained by the **substitution** of G for p in F, denoted $F(G/p)$, is the formula obtained by replacing all occurrences of p in F by the formula G.

DEFINITION 1.10. *The formula $F(G/p)$ is defined by induction on formula F:*

- *if F is a propositional variable p, $F(G/p)$ is the formula G;*
- *if F is a propositional variable q distinct from p, $F(G/p)$ is the formula F;*
- *if F has the form $\neg H$, $F(G/p)$ is the formula $\neg H(G/p)$;*
- *if F has form $F_1 \alpha F_2$, where α is a binary connective, $F(G/p)$ is the formula $F_1(G/p)\alpha F_2(G/p)$.*

Example. The substitution of the formula $(q \to r)$ for the variable q in the formula $((p \to q) \wedge (q \vee r))$ is:

$$((p \to (q \to r)) \wedge ((q \to r) \vee r))$$

Let F, G, H be some formulas. Suppose we know the values $\mathcal{V}(F), \mathcal{V}(G), \mathcal{V}(H)$. How could we, for example, determine the value of $\mathcal{V}(((F \to G) \wedge (G \vee H)))$? The following proposition answers this question.

PROPOSITION 1.8. *Let F, G be two formulas and $\mathcal{V}$ a valuation whose value for G is known. The value of the formula $F(G/p)$ for the valuation $\mathcal{V}$ is equal to the value of the formula F for a valuation $\mathcal{V}'$ satisfying: $\mathcal{V}'(p) = \mathcal{V}(G)$ and $\mathcal{V}'(q) = \mathcal{V}(q)$ for each q distinct from p.*

Proof : The proof is by induction on formula F.

- If F is a propositional variable p, then the statement is true. If F is some variable q distinct from p, $F(G/p)$ is q and $\mathcal{V}'(F) = \mathcal{V}(q) = \mathcal{V}(F)$.
- Assume that the property is true for the formula H, i.e.

$$\mathcal{V}(H(G/p)) = \mathcal{V}'(H)$$

If F has the form $\neg H$, then:

$$\mathcal{V}(F(G/p)) = 1 \text{ iff } \mathcal{V}(H(G/p)) = 0$$

$$\mathcal{V}'(F) = 1 \text{ iff } \mathcal{V}'(H) = 0$$

The induction hypothesis allows one to obtain the desired condition for F.

- Assume the property is true for formulas F_1, F_2. If F has the form $(F_1 \wedge F_2)$, the values of $\mathcal{V}, \mathcal{V}'$ in F will satisfy:

$$\mathcal{V}(F(G/p)) = 1 \text{ iff } \mathcal{V}(F_1(G/p)) = \mathcal{V}(F_2(G/p)) = 1$$

$$\mathcal{V}'(F) = 1 \text{ iff } \mathcal{V}'(F_1) = \mathcal{V}'(F) = 1$$

Once again, the induction hypothesis leads us to the desired conclusion. The proof is similar for the other binary connectives.

□

COROLLARY 1.1. *Let F, F', G, G' be formulas and p a propositional variable.*

- *If the formula F is a tautology, then the formula $F(G/p)$ is also a tautology.*
- *If the formulas F and F' are equivalent, then the formulas $F(G/p)$ and $F'(G/p)$ are also equivalent.*
- *If the formulas G and G' are equivalent, then the formulas $F(G/p)$ and $F(G'/p)$ are also equivalent.*

The proof is left as an exercise. The following examples are direct consequences of the corollary.

For every formula F, G, H, the following formulas are tautologies:

(1)

$$(F \to F)$$

$$(F \to (G \to F))$$

$$((F \to (G \to H)) \to ((F \to G) \to (F \to H)))$$

(2) For every formula F, G, H, the following formulas are equivalent:

$$\neg\neg F \equiv F$$

$$(F \to G) \equiv (\neg F \vee G)$$

$$(F \leftrightarrow G) \equiv ((F \to G) \wedge (G \to F))$$

(3) If $F \equiv F'$ and $G \equiv G'$, then the following formulas are equivalent:

$$\neg F \equiv \neg F'$$

$$(F \wedge G) \equiv (F' \wedge G')$$

$$(F \vee G) \equiv (F' \vee G')$$

$$(F \to G) \equiv (F' \to G')$$

$$(F \leftrightarrow G) \equiv (F' \leftrightarrow G')$$

The following equivalences of formulas express the main properties of connectives:

(1) commutativity:

$$(F \wedge G) \equiv (G \wedge F)$$

$$(F \vee G) \equiv (G \vee F)$$

(2) associativity:

$$(F \wedge (G \wedge H)) \equiv ((F \wedge G) \wedge H)$$

$$(F \vee (G \vee H)) \equiv ((F \vee G) \vee H)$$

(3) idempotence:

$$(F \wedge F) \equiv F$$

$$(F \vee F) \equiv F$$

(4) De Morgan's rules:

$$\neg(F \wedge G) \equiv (\neg F \vee \neg G)$$

$$\neg(F \vee G) \equiv (\neg F \wedge \neg G)$$

(5) distributivity:

$$(F \wedge (G \vee H)) \equiv ((F \wedge G) \vee (F \wedge H))$$

$$(F \vee (G \wedge H)) \equiv ((F \vee G) \wedge (F \vee H))$$

(6) absorption:

$$(F \wedge (F \vee G)) \equiv F$$

$$(F \vee (F \wedge G) \equiv F$$

1.2.4. Complete systems of connectives. The following proposition introduces a set of connectives which allows to express all propositional formulas.

PROPOSITION 1.9. *Any propositional formula is equivalent to a formula constructed only with the connectives $\neg$ and $\wedge$.*

Proof : The proof is obtained by induction on the formulas. It is true for propositional variables. Assume the property is true for G, H, i.e. the formula G (respectively H) is equivalent to the formula G' (respectively H') built only with the connectives $\neg, \wedge$.

- Let $F = \neg G$, so F is equivalent to $\neg G'$, which is a formula built only with the connectives $\neg, \wedge$, according to the induction hypothesis.
- Let $F = (G \wedge H)$, so F is equivalent to $(G' \wedge H')$, which is a formula built only with the connectives $\neg, \wedge$, according to the induction hypothesis.
- Let $F = (G \vee H)$, $F \equiv (G' \vee H')$. Using De Morgan's second rule and the fact that $K \equiv \neg\neg K$, we get that $F \equiv \neg(\neg G' \wedge \neg H')$, which is a formula built only with the connectives $\neg, \wedge$.
- Let $F = (G \to H)$, $F \equiv (G' \to H')$, $F \equiv (\neg G' \vee H')$, which is equivalent to a formula built only with the connectives $\neg, \wedge$,, according to the previous case.
- Let $F = (G \leftrightarrow H)$, $F \equiv (G' \leftrightarrow H')$, $F \equiv ((G' \to H') \wedge (H' \to G'))$, which is equivalent to a formula built only with the connectives $\neg, \wedge$, according to the previous case.

This concludes the proof. □

Notice that the expression $F \equiv G$ is a relation between two formulas and is not a propositional formula!

DEFINITION 1.11. *A set of connectives having the property stated in the above proposition for $\{\neg, \wedge\}$ is called a* **complete system.**

From the previous result, it is easy to deduce that the systems of operators $\{\neg, \vee\}$, $\{\neg, \to\}$ are complete; the proof is left as an exercise.

1.3. Normal forms

Normal forms are special formulas such that any formula can be transformed into an equivalent normal form. We consider disjunctive and conjunctive normal forms.

1.3.1. Disjunctive and conjunctive normal forms.

DEFINITION 1.12. *A literal is a propositional variable or the negation of a propositional variable.*

DEFINITION 1.13. *A* **disjunctive normal form** *is a disjunction $(F_1 \vee F_2 \vee ... \vee F_k)$ of k formulas ($k \geq 1$), where each formula F_i ($i = 1, 2, ..., k$) is a conjunction $(G_1 \wedge G_2 \wedge ... \wedge G_l)$ of l literals ($l \geq 1$).*

Example. The following formulas are disjunctive normal forms:

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

DEFINITION 1.14. *A* **conjunctive normal form** *is a conjunction* $(F_1 \wedge F_2 \wedge \cdots \wedge F_k)$ *of k formulas* $(k \geq 1)$*, where each formula* F_i $(i = 1, 2, \ldots, k)$ *is a disjunction* $(G_1 \vee G_2 \vee \cdots \vee G_l)$ *of l literals* $(l \geq 1)$.

Example. The following formulas are conjunctive normal forms:

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

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

$$(\neg p \vee q)$$

1.3.2. Functions associated to formulas. The set $\mathcal{P}$ is now supposed to be finite: $\mathcal{P} = \{p_1, p_2, ..., p_n\}$. Thus there are 2^n distinct valuations. Each formula $F(p_1, p_2, ..., p_n)$ defines a function f_F from the set of valuations $\{0,1\}^{\mathcal{P}}$ into $\{0,1\}$ which associates the value $\mathcal{V}(F)$ to each valuation $\mathcal{V}$. The following property is obvious.

PROPOSITION 1.10. *Two formulas* F, G *are equivalent iff their associated functions are equal.*

COROLLARY 1.2. *There are at most* 2^{2^n} *propositional formulas, pairwise non-equivalent, built with n variables.*

Proof : Let $\mathcal{F}/\equiv$ be the quotient of the set of formulas by the equivalence relation $\equiv$. We associate the function f_F to the equivalence class of some formula F. There are 2^{2^n} such functions because each one associates a valuation to $\{0,1\}$ and there are 2^n such valuations. This application is an injection and therefore there are at most 2^{2^n} non-equivalent formulas. □

The following theorem claims that the function defined in the proof of the previous corollary is in fact a bijection.

THEOREM 1.1. *Every function f from* $\{0,1\}^{\mathcal{P}}$ *to* $\{0,1\}$ *can be represented by a formula* $F(p_1, p_2, ..., p_n)$*, meaning that there is some formula* $F(p_1, p_2, ..., p_n)$ *such that, for all valuations* $\mathcal{V}$, $f(\mathcal{V}) = \mathcal{V}(F)$.

Proof : The proof is by induction on the number of propositional variables n.

- If $n = 1$, there are four functions from $\{0,1\}^{\mathcal{P}}$ to $\{0,1\}$: these functions can be represented by the formulas $p, \neg p, (p \vee \neg p), (p \wedge \neg p)$.
- Assume the property true for $n - 1$ propositional variables.
Let $\mathcal{P} = \{p_1, p_2, ..., p_n\}$ and f be a function from $\{0,1\}^{\mathcal{P}}$ to $\{0,1\}$. Every valuation $\mathcal{V}'$ on $\{p_1, p_2, ..., p_{n-1}\}$ can be considered as a restriction of a valuation $\mathcal{V}$ on $\{p_1, p_2, ..., p_n\}$. Let the function f_0, respectively f_1, be the restriction of f to the valuation $\mathcal{V}$ such that $\mathcal{V}(p_n) = 0$, respectively $\mathcal{V}(p_n) = 1$: f_0, f_1 are functions from the set of valuations defined on $\{p_1, p_2, ..., p_n\}$ to $\{0,1\}$ and are represented by the formulas $G(p_1, ..., p_{n-1})$ and $H(p_1, ..., p_{n-1})$, according to the induction hypothesis. The function f can be represented by the formula:

$$(\neg p_n \wedge G(p_1, ..., p_{n-1})) \vee (p_n \wedge H(p_1, ..., p_{n-1})))$$

We conclude that any function from $\{0,1\}^{\mathcal{P}}$ to $\{0,1\}$ can be represented by a formula, whatever the number of propositional variables. □

We obtain the existence of normal forms for all propositional formulas.

COROLLARY 1.3. *Every formula is equivalent to a disjunctive normal form and to a conjunctive normal form.*

Proof : As in the previous proof, we prove this corollary by induction on the number n of propositional variables.

- In the case $n = 1$, we consider the formulas above, which are both disjunctive and conjunctive normal forms.
- Assume the property true for $n - 1$ variables. Let f_F be the function associated to the formula $F(p_1, p_2, ..., p_n)$. It is possible to construct a formula which represents f_F, as in the proof of the previous theorem. The formula F is equivalent to a formula of the form $(\neg p_n \wedge G) \vee (p_n \wedge H))$, where G, H are equivalent to disjunctive normal forms:

$$G \equiv (G_1 \vee G_2 \vee ... \vee G_k)$$

$$H \equiv (H_1 \vee H_2 \vee ... \vee H_l)$$

$$(\neg p_n \wedge G) \equiv (\neg p_n \wedge G_1) \vee (\neg p_n \wedge G_2) \vee ... \vee (\neg p_n \wedge G_k))$$

which is a disjunctive normal form,

$$(p_n \wedge H) \equiv ((p_n \wedge H_1) \vee (p_n \wedge H_2) \vee ... \vee (p_n \wedge H_l))$$

which is also a disjunctive normal form. The formula F is then equivalent to the disjunction of these two disjunctive normal.

In order to obtain a conjunctive normal form, the induction hypothesis produces two conjunctive normal forms G and H. The equivalence used in this case is:

$$F \equiv ((\neg p_n \vee H) \wedge (p_n \vee G))$$

and we obtain a conjunctive normal form for the formula F. □

1.3.3. Transformation methods. In practice, two main methods are used to obtain a disjunctive or conjunctive normal form. The first method consists in transforming formulas by successive equivalences using the following rules, applied in this order:

(1) eliminate the connectives $\rightarrow$ and $\leftrightarrow$ by using the following equivalences:

$$(F \rightarrow G) \equiv (\neg F \vee G)$$

$$(F \leftrightarrow G) \equiv ((\neg F \vee G) \wedge (F \vee \neg G))$$

(2) push the negation as far inside as possible:

$$\neg(F \wedge G) \equiv (\neg F \vee \neg G)$$

$$\neg(F \vee G) \equiv (\neg F \wedge \neg G)$$

(3) use the distributivity of $\wedge$ and $\vee$:

$$(F \wedge (G \vee H)) \equiv ((F \wedge G) \vee (F \wedge H))$$
$$(F \vee (G \wedge H)) \equiv ((F \vee G) \wedge (F \vee H))$$

Example. Determine disjunctive and conjunctive normal forms of the formula $\neg(p \leftrightarrow (q \to r))$. The formula is transformed by successive equivalences:

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

which is a disjunctive normal form.
Consider one of the formula above: $((p \wedge q \wedge \neg r)) \vee ((\neg q \vee r) \wedge \neg p))$ and apply the distributivity. We obtain:

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

which is a conjunctive normal form.

The second method to obtaining a disjunctive or conjunctive normal form equivalent to a given formula F, consists first in determining the associated function f_F. Then we can build a normal form G which represents the function f_F and which is equivalent to F:

(1) we determine the valuations $\mathcal{V}$ such that $\mathcal{V}(F) = 1$;
(2) to each valuation $\mathcal{V}_j$ such that $\mathcal{V}_j(F) = 1$, we associate a formula G_j, which is of the form $(e_1p_1 \wedge e_2p_2 \wedge \dots \wedge e_np_n)$, where for each $i = 1, 2, \dots, n$, e_ip_i is p_i if $\mathcal{V}_j(p_i) = 1$ and e_ip_i is $\neg p_i$ if $\mathcal{V}_j(p_i) = 0$;
(3) the formula G, obtained by taking the disjunction of the formulas G_j, is a disjunctive normal form.

It is easy to verify that the function associated with the formula G is equal to f_F Therefore G is equivalent to the given formula F.

Example. Let us apply this method to the formula F, from the previous example:

$$\neg(p \leftrightarrow (q \to r))$$

There are four valuations $\mathcal{V}$ such that $\mathcal{V}(F) = 1$: $\mathcal{V}_1 = (1,1,0)$, $\mathcal{V}_2 = (0,1,1)$, $\mathcal{V}_3 = (0,0,1)$, $\mathcal{V}_4 = (0,0,0)$, where we denote the valuation $\mathcal{V}$ by $(\epsilon_1, \epsilon_2, \epsilon_3)$ if $\mathcal{V}(p_i) = \epsilon_i$, for $i = 1, 2, 3$. The obtained formula G is a disjunctive normal form:

$$G = ((p \wedge q \wedge \neg r) \vee (\neg p \wedge q \wedge r) \vee (\neg p \wedge \neg q \wedge r) \vee (\neg p \wedge \neg q \wedge \neg r))$$

The construction of a conjunctive normal form for a given formula follows a similar method: we exchange systematically the roles between valuations giving value 1 and those giving value 0, between propositional variables and negation of propositional variables, between disjunction and conjunction.

PROPOSITION 1.11. *Determining a disjunctive normal (respectively conjunctive) form for F is equivalent to determining a conjunctive normal (respectively disjunctive) form for $\neg F$.*

Proof : Assume that G is a disjunctive normal (respectively conjunctive) form for F. The formula $\neg F$ is equivalent to $\neg G$: if we apply De Morgan's rules to $\neg G$, we obtain a conjunctive (respectively disjunctive) normal form, equivalent to $\neg G$ and to $\neg F$. □

The previous proposition express the duality between a formula and its negation, disjunction and conjunction, disjunctive and conjunctive normal forms.

1.3.4. Clausal form. Clausal form is an alternative presentation of conjunctive normal form, which is used in some automatic deduction methods.

DEFINITION 1.15.

- *A* **clause** *C is a disjunction $(G_1 \vee G_2 \vee \dots \vee G_l)$ of l formulas ($l \geq 1$), where each G_j ($j = 1, 2, \dots, l$) is a literal.*
- *The propositional variables which appear in the clause C without negation are called* **positive variables***; propositional variables preceded by a negation are called* **negative variables***.*

The following proposition is a direct consequence of the corollary on the existence of conjunctive normal form.

PROPOSITION 1.12. *Every propositional formula is equivalent to a conjunction of clauses.*

Clause C is equivalent to a clause of the form:

$$(\neg a_1 \vee \neg a_2 \vee \dots \vee \neg a_n \vee b_1 \vee b_2 \vee \dots \vee b_m)$$

where all a_i ($i = 1, 2, \dots, n$) and all b_j ($j = 1, 2, \dots, m$) are propositional variables. A clause which has at least one negative variable and one positive variable is also equivalent to a formula of the form:

$$((a_1 \wedge a_2 \wedge \dots \wedge a_n) \to (b_1 \vee b_2 \vee \dots \vee b_m))$$

The notation (Γ, Δ) is often used as a representation of a clause $C : \Gamma$ is the set of all negative propositional variables in C and Δ the set of all positive variables in C. If Δ is reduced to a unique variable b_1, we have a *Horn clause*.

Example. The formula $\neg(p \leftrightarrow (q \to r))$ is equivalent to the conjunction of the following clauses:

$$C_1 \; : \; (\neg q \vee p \vee r) \equiv (q \to (p \vee r))$$
$$C_2 \; : \; (\neg p \vee q) \equiv (p \to q)$$

$$C_3 : (\neg p \vee \neg r)$$

Only C_2 is a Horn clause.

Among particular clauses we can distinguish the following three cases:

(1) a clause $C = (\Gamma, \Delta)$ is negative if $\Delta = \emptyset$, such as clause C_3 for example;
(2) a clause $C = (\Gamma, \Delta)$ is positive if $\Gamma = \emptyset$;
(3) an empty clause, defined by $\Gamma = \Delta = \emptyset$.

The following properties are useful:

(1) The valuation $\mathcal{V}$ satisfies a clause $C = (\Gamma, \Delta)$ iff there is some variable $p \in \Gamma$ such that $\mathcal{V}(p) = 0$ or there is some variable $q \in \Delta$ such that $\mathcal{V}(q) = 1$.
(2) The valuation $\mathcal{V}$ does not satisfy a clause $C = (\Gamma, \Delta)$ iff for all variables $p \in \Gamma$, $\mathcal{V}(p) = 1$ and for all variables $q \in \Delta$, $\mathcal{V}(q) = 0$.
(3) The empty clause is satisfied by no valuation.

1.3.5. OBDD: Ordered Binary Decision Diagrams. OBDD's are important structures which represent boolean functions. In order to define these structures, we need basic notions on graphs.

1.3.5.1. *Graphs.* A **directed graph** G consists of a finite set V of vertices or nodes and a set E of **edges** between two vertices. The set E is a subset of $V \times V$ and defines a binary relation on V. Two nodes u and v are connected by an edge starting in u iff $(u, v) \in E$. The *in-degree* of a node v is the number of edges leading to v. Analogously, the *out-degree* of v is the number of edges starting in v. A node is called a *sink* if it has out-degree 0. If the out-degree of v is larger than 0, v is called an *internal node*. A node is called a *root* if it has indegree 0. If (u, v) is an edge, then u is called a *predecessor* of v, and v is called a *successor* of u. A *path* of length k is a sequence $u_0, u_1, \ldots, u_k$ of $k+1$ nodes where u_{i+1} is a successor of u_i $(i = 0, 1, \ldots, k-1)$. If $u_0 = u_k$, the path is called *cyclic*. A graph is said *acyclic* if there does not exist a cyclic path.

1.3.5.2. *Decision diagrams.* Consider the *boolean algebra* on the set $\{0, 1\}$, which is defined by the operations $+, \cdot, ^{-}$ as follows:

$$a + b = max\{a, b\} \quad a \cdot b = min\{a, b\} \quad \overline{0} = 1 \quad \overline{1} = 0.$$

In the sequel, the term $a \cdot b$ is abbreviated by ab.

DEFINITION 1.16. *Let $<$ be a linear order on the set of variables $x_1, \ldots, x_n$. An ordered binary decision diagram (OBDD) with respect to the variable order $<$ is a directed acyclic graph with exactly one root, which satisfies the following properties:*

- *There are exactly two sinks, labelled by the constants 1 and 0.*
- *Each non-sink node is labelled by a variable x_i, and has two outgoing edges which are labelled by 1 (1-edge) and 0 (0-edge), respectively.*
- *The order, in which the variables appear on a path in the graph, is consistent with the variable order $<$, i.e. for each edge connecting a node labelled by x_i to a node labelled by x_j, we have $x_i < x_j$.*

Nodes labelled by a variable are called internal nodes.

The variable of a node v is denoted by $var(v)$. The successor node of a node v, which is determined by the 1-edge, is denoted by $l(v)$ and the successor which is determined by the 0-edge, is denoted by $r(v)$.

DEFINITION 1.17.

- *The* computation path *of an input $\overline{a} = (a_1, \ldots, a_n) \in \{0,1\}^n$ is the path from the root to a sink in the OBDD, defined by the input, i.e. the computation path begins at the root, and in each node labelled by x_i the path follows the edge with label a_i.*
- *An OBDD represents a given boolean function $f : \{0,1\}^n \longrightarrow \{0,1\}$ if for any input $\overline{a} \in \{0,1\}^n$, the computation path of $\overline{a}$ reaches the sink with label $f(a)$.*

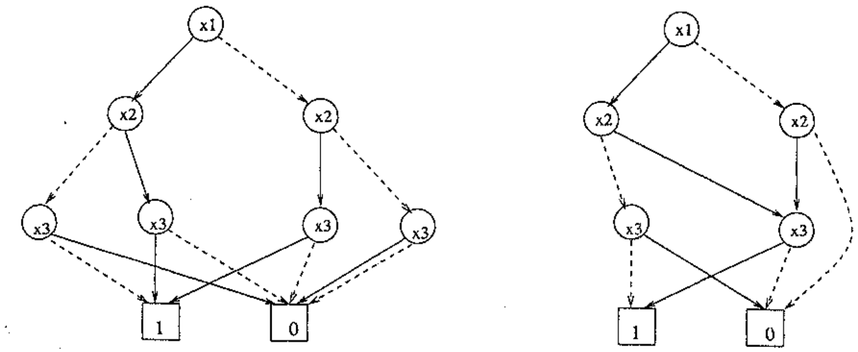

FIGURE 1.2. Two OBDDs for the function f.

Example. Let $<$ be the variable order $x_1 < x_2 < x_3$. Figure 1.2 shows two OBDD representations of the function:

$$f(x_1, x_2, x_3) = x_1 x_2 x_3 + x_1 \overline{x_2}\,\overline{x_3} + \overline{x_1} x_2 x_3$$

with respect to the variable order $<$. The 1-edges use plain edges whereas the 0-edges use dotted edges.

OBDDs in the sense of the previous definition are not uniquely determined. The notion of *reduced OBDD* provides a canonical representation of boolean functions.

1.3.5.3. *Reduced OBDDs.* Two OBDDs are said *isomorphic* if they are isomorphic as labelled graphs, i.e. if there is a bijection between the set of vertices which preserves the 1 and $0 -$ *edges*.

DEFINITION 1.18. *An OBDD is called reduced if*