CHAPTER 3

First-order logic

Propositional logic allows only for the description of extremely simple language constructions: boolean operations with propositions. It is not powerful enough for representing many constructions used in computer science, linguistics, mathematics or for formalizing significant fragments of reasoning in action, as for example:

- *certain students attend all courses*;
- *no student attends an uninteresting course*;
- can we conclude that *all courses are interesting*?

What is missing in propositional logic is to express elementary facts such as the property of an object, or a relation between several objects. Elementary facts are for example:

- *some course is interesting*;
- *some students attend the computer science course.*

Another type of construction of **first-order logic** is the *quantification* of objects, as in the statement: *all courses are interesting.* **First-order languages** are used in logic for several reasons:

- these languages share certain essential characteristics with natural languages, specification languages and query languages for databases;
- there is a wide variety of first-order languages, each one determined by its own vocabulary;
- these languages allow not only relations, but also functions.

The expression *first-order* distinguishes these languages, which are used most frequently, from those of *higher-order*, such as *second-order*, which allow one to quantify relations as well as functions. The study of first-order languages follows a similar approach to that of propositional language. The first section presents the definition of *languages* and the construction of *formulas*. The second section introduces *semantics* in terms of *structures* and the interpretation of formulas in structures. The last section deals with *normal forms*.

3.1. First-order languages

The *alphabet* of a first-order language requires the following symbols, which are common to all languages:

- propositional operators $\neg, \wedge, \vee, \rightarrow, \leftrightarrow$,
- parentheses $(,)$,
- *universal* $\forall$ and *existential* $\exists$ *quantifiers*,

- an infinite set V, of *variable* symbols, usually chosen from the letters x, y, z, u, v, sometimes indexed.

DEFINITION 3.1. *A first-order logical* **language** $\mathcal{L}$ *is characterized by the following sets of symbols:*

- **relation** *(or* predicate*) symbols usually chosen from the letters* P, Q, R, S *and sometimes indexed; a strictly positive integer, called* arity, *is associated with each relation symbol,*
- **function** *symbols, chosen from* f, g, h, k *and sometimes indexed; a strictly positive integer, called the number of arguments or arity, is associated with each function symbol,*
- **constant** *symbols, chosen from* c, d, e *and sometimes indexed.*

All these symbols are interpreted according to their order, in the structures. The construction of the set of formulas for a language will be defined in a purely syntactic manner and illustrated by two examples.

Example. The first language $\mathcal{L}_1$ contains a symbol for a relation R, and a symbol for a constant c. The second one, denoted by $\mathcal{L}_2$ contains a symbol for a binary relation R, a symbol for a unary function f, two symbols for binary functions g, h, and two symbols for constants c, d.

A language contains, in general, a supplementary binary relation symbol, denoted by $=$. This symbol will always be interpreted as equality and the corresponding language will be referred as *language with equality* . As in the propositional language, the construction of *formulas* uses the *definition by induction*. However, as the power of expression in first-order logic is much higher, it is necessary to construct *terms* which represent objects, and *atomic formulas* which correspond to the relations between objects.

3.1.1. Construction of terms. The set of *terms* in some language $\mathcal{L}$ is the smallest set, containing constants and variables symbols, which is closed under the application of functions symbols.

DEFINITION 3.2. *The set of terms denoted by* $\mathcal{T}$ *is the smallest set satisfying the conditions:*

- *all symbols for constants or variables are terms,*
- *if* f *is a symbol of an* m *arguments function and if* $t_1, t_2, \ldots, t_m$ *are terms, then* $ft_1t_2\ldots, t_m$ *is a term (also denoted by* $f(t_1, t_2, \ldots, t_m)$ *according to the usual notation for functions).*

Example. The only terms of language $\mathcal{L}_1$ are the variables and the constant c.

The following expressions are terms of language $\mathcal{L}_2$:
$hcx, hyz, gyhcx, gdhyz, fgdhyz$.

The corresponding functions are:
$h(c, x), h(y, z), g(y, h(c, x)), g(d, h(y, z)), f(g(d, h(y, z)))$.

In the case of binary functions symbols, as g and h, it is common to write (t_1gt_2) instead of $g(t_1, t_2)$. The notation $t(x_1, x_2, \ldots, x_k)$ signifies that the variables appearing in the term t are among $x_1, x_2, \ldots, x_k$.

DEFINITION 3.3. *A* closed term *is a term with no variables.*

Example. The term $gcfd$ is a closed term but the term fx is not closed.

3.1.2. Construction of formulas. Assume $\mathcal{L}$ is a language with equality.

DEFINITION 3.4. *An* **atomic formula** *is a formula of the form:*

- $t_1 = t_2$ *where* t_1, t_2 *are terms,*
- *or* $Rt_1t_2\ldots, t_n$ *where* R *is a relation symbol of arity* n *and* $t_1, t_2, \ldots, t_n$ *are terms; this formula can also be written as* $R(t_1, t_2, \ldots, t_n)$.

The set of formulas is the smallest set containing atomic formulas and which is closed under the application of operators and quantifiers.

DEFINITION 3.5. *The set of formulas, denoted by* $\mathcal{F}$, *is the smallest set such that:*

- *every atomic formula is a formula,*
- *if* F *is a formula, then* $\neg F$ *is a formula,*
- *if* F, G *are formulas, then* $(F \wedge G)$, $(F \vee G)$, $(F \rightarrow G)$ *and* $(F \leftrightarrow G)$ *are formulas,*
- *if* F *is a formula and* v *a variable, then* $\forall v\ F$ *and* $\exists v\ F$ *are formulas.*

Example. In the following expressions, the first is a formula of the languages $\mathcal{L}_1$ and $\mathcal{L}_2$, the others are formulas of $\mathcal{L}_2$:

$\forall x \forall y \forall z\ ((Rxy \wedge Ryz) \rightarrow Rxz)$
$\forall x \exists y\ (gxy = c \wedge gyx = c)$
$\forall x \forall y (Rxy \rightarrow Rfxfy)$
$\forall x \neg\ (fx = c)$
$\forall x \forall y\ (gxfy = fgxy)$

In the case of binary relations, the notation xRy may also be used instead of Rxy or $R(x, y)$. As in the propositional logic, the existence and uniqueness of the *decomposition* of a formula are assured, due to the definition by induction on the set of formulas.

PROPOSITION 3.1. *Every formula of any first-order language is uniquely decomposed in one, and only one, of the following forms:*

- *an atomic formula,*
- $\neg F$ *where* F *is a formula,*
- $(F \wedge G)$, $(F \vee G)$,$(F \to G)$ *or* $(F \leftrightarrow G)$, *where* F, G *are formulas,*
- $\forall v\ F$ *or* $\exists v\ F$, *where* F *is a formula and* v *a variable.*

The proof is similar to that in propositional logic and is left to the reader. The complete decomposition of a formula can be obtained recursively.

$$
\begin{array}{c}
(Rxz \to \forall z\,(Ryz \vee y = z)) \\
\swarrow \qquad \searrow \\
Rxz \qquad \forall z\,(Ryz \vee y = z) \\
\qquad\qquad | \\
\qquad\qquad (Ryz \vee y = z) \\
\qquad\qquad \swarrow \quad \searrow \\
\qquad\qquad Ryz \qquad y = z
\end{array}
$$

FIGURE 3.1. A formula decomposition.

Example. The decomposition of the formula $(Rxz \to \forall z\ (Ryz \vee y = z))$ is given in Figure 3.1. All formulas appearing in the decomposition of F are subformulas of F.

DEFINITION 3.6. *A formula* G *is a* subformula *of the formula* F *iff it appears in the decomposition of* F.

3.1.3. Free and bound variables. The same variable may appear more than once in a formula. It is necessary to distinguish between these different cases.

DEFINITION 3.7. *An* **occurrence** *of a variable in a formula is given by the position of the variable in a formula where it does not follow a quantifier.*

Example. In a formula $F: (Rxz \to \forall z\,(Ryz \vee y = z))$, the variable x has one occurrence, the variable y two occurrences and the variable z three occurrences (the apparition of z in $\forall z$ does not constitute an occurrence of z).

The definition of the interpretation of a formula in a structure depends on the variables which occur in that formula.

DEFINITION 3.8.

- *An occurrence of a variable* x *in a formula* F *is a* free occurrence *if it does not belong to any subformula of* F, *which begins with a quantifier* $\forall x$ *or* $\exists x$. *Otherwise, the occurrence is bound and the variable* x *is a* bound variable *in* F.
- *A* variable *is* free *in a formula if it has at least one free occurrence in that formula.*
- *A* closed formula *is a formula with no free variables.*

Example. In the formula $F: (Rxz \to \forall z\,(Ryz \vee y = z))$, the only occurrence of x is free, the two occurrences of y are free, the first occurrence of z is free and the others are bounded. The variables x, y, z are thus free in F. In contrast, the formula $\forall x\, \forall z\ (Rxz \to \exists y\ (Ryz \vee y = z))$ is closed.

The notation $F(x_1, x_2, \ldots, x_k)$ signifies that all free variables in formula F are among $x_1, x_2, \ldots, x_k$.

3.2. Semantics

The semantics of first-order logic is defined by interpreting the formulas of a given language in a structure we now define. Such a structure may be of mathematical nature, may represent computer science data, or even the universe of discourse in linguistics.

3.2.1. Structures and languages. The meaning of an expression is only defined for a given *stucture.*

DEFINITION 3.9. *A* **structure** $\mathcal{M}$ *for a language* $\mathcal{L}$ *consists in a non-empty set* M, *called* **domain** *and:*

- *a subset of* M^n, *denoted by* $R^{\mathcal{M}}$, *for each predicate* R *of arity* n;
- *a function from* M^m *to* M, *denoted by* $f^{\mathcal{M}}$, *for each function* f *of* m *arguments;*
- *an element from* M, *denoted by* $c^{\mathcal{M}}$, *for each constant symbol* c.

As examples of structures, consider:

(1) The set D, with the binary relation E and a distinct element a is a structure for the language $\mathcal{L}_1 = \{R, c\}$, denoted by (D, E, a).
(2) The set of real numbers $\mathbf{R}$ allows the construction of a structure for the language $\mathcal{L}_2$, interpreting its symbols as follows:
   - the predicate symbol R as the order relation $\leq$ on the real numbers;
   - the unary function symbol f which associates $r + 1$ to the real r;
   - the binary symbols g, h as addition and multiplication;
   - the symbols c, d as 0 and 1.

   This structure is denoted by $\mathcal{R} = (\mathbf{R}, \leq, s, +, \times, 0, 1)$.

A structure for the language $\mathcal{L}$ may also be called a **realization** of the language $\mathcal{L}$. Because a language is specified by the sequence of its symbols (predicates, functions, constants), for example $\mathcal{L}_2 = \{R, f, g, h, c, d\}$, a realization $\mathcal{M}$ of this language is denoted by:

$$\mathcal{M} = (M, R^{\mathcal{M}}, f^{\mathcal{M}}, g^{\mathcal{M}}, h^{\mathcal{M}}, c^{\mathcal{M}}, d^{\mathcal{M}})$$

3.2.2. Structures and satisfaction of formulas. We define the *satisfaction* and the *value of a formula* $F(x_1, x_2, \ldots, x_k)$ in some structure $\mathcal{M}$ for a sequence of elements $a_1, a_2, \ldots, a_k$ in M, interpreting the free variables of F. First, it is necessary to give the interpretation, or the **value** of a term in a structure.

DEFINITION 3.10. *Given a term $t(x_1, x_2, \ldots, x_k)$ and a sequence of elements in M, $s = (a_1, a_2, \ldots, a_k)$, the value of t in $\mathcal{M}$ for this sequence, denoted by*

$$t^{\mathcal{M}}[a_1, a_2, \ldots, a_k]$$

is defined by induction on the term t:

- *if t is the constant c, then $t^{\mathcal{M}}$ is $c^{\mathcal{M}}$,*
- *if t is the variable x_i ($i = 1, 2, \ldots, k$), then $t^{\mathcal{M}}[a_1, a_2, \ldots, a_k]$ is a_i,*
- *if t is of the form $f(t_1, t_2, \ldots, t_m)$ and if $t_i^{\mathcal{M}}[a_1, a_2, \ldots, a_k]$ is b_i ($i = 1, 2, \ldots, k$), then $t^{\mathcal{M}}[a_1, a_2, \ldots, a_k]$ is $f^{\mathcal{M}}(b_1, b_2, \ldots, b_k)$.*

Example. Let $\mathcal{N}$ be the structure $(\mathbf{N}, \leq, s, +, \times, 0, 1)$, where $\mathbf{N}$ is the set of natural numbers. The value of the term $t_1(x, y) = g(y, h(c, x))\ f(g(d, h(y, z)))$ in $\mathcal{N}$ for the sequence $1, 2$ is 2. That of the term $t_2(y, z) = f(g(d, h(y, z)))$ for the sequence $2, 3$ is 8.

The value of the term $t'(x, y, z) = g(h(x, y), z)$ for the sequence p, q, r, in the structure $(\mathbf{R}, \leq, s, +, \times, 0, 1)$ is the real number $p \times q + r$.

Let $F(x_1, x_2, \ldots, x_k)$ be a formula, $\mathcal{M}$ a structure and $s = (a_1, a_2, \ldots, a_k)$ a sequence of elements in M of length k. The notion of *satisfaction* of the formula F by s in $\mathcal{M}$ is defined by induction on the formula F and is denoted by: $\mathcal{M} \models F[a_1, a_2, \ldots, a_k]$.

DEFINITION 3.11. *The expression* the sequence s satisfies formula F in $\mathcal{M}$ *is defined by induction on formula F:*

- *if F is an atomic formula of the form $Rt_1t_2 \ldots, t_n$ and if $b_1, b_2, \ldots, b_n$ are the corresponding values of the terms $t_1, t_2, \ldots, t_n$ for the sequence s, s satisfies F in $\mathcal{M}$ iff $(b_1, b_2, \ldots, b_n) \in R^{\mathcal{M}}$;*
- *if F is one of the form $\neg G$, $(G \wedge H)$, $(G \vee H)$, $(G \rightarrow H)$ or $(G \leftrightarrow H)$ the satisfaction of F is defined from the satisfaction of G and H as in propositional logic;*
- *if F is of the form $\exists x\ G(x_1, \ldots, x_{i-1}, x, x_i, \ldots, x_k)$, s satisfies F in $\mathcal{M}$ iff there is an element $a \in M$ such that the sequence $s' = (a_1, \ldots, a_{i-1}, a, a_i, \ldots, a_k)$ satisfies G in $\mathcal{M}$;*

- *if F is of the form $\forall x\ G(x_1, \ldots, x_{i-1}, x, x_i, \ldots, x_k)$, s satisfies F in $\mathcal{M}$ iff, for each element $a \in M$, the sequence $s' = (a_1, \ldots, a_{i-1}, a, a_i, \ldots, a_k)$ satisfies G in $\mathcal{M}$.*

We say that formula F is *true* in $\mathcal{M}$ for s when the sequence s satisfies the formula F in $\mathcal{M}$. Otherwise, F is said to be *false* in $\mathcal{M}$ for s. If F is a closed formula, the satisfaction of F in the structure $\mathcal{M}$ does not depend on elements in M, therefore it is either true or false in $\mathcal{M}$. If the closed formula F is true in $\mathcal{M}$, the structure $\mathcal{M}$ is called a **model** of F, and we write: $\mathcal{M} \models F$.

Example. The realization (D, E, a) of the language $\mathcal{L}_1$ is a model of the formula $\forall x \forall y\ (Rxy \rightarrow Ryx)$ iff the relation E is symmetric.

The following closed formula states that in a realization of $\mathcal{L}_1$, R is an equivalence relation.

$(\forall x\ Rxx \wedge \forall x \forall y\ (Rxy \rightarrow Ryx) \wedge \forall x \forall y \forall z\ ((Rxy \wedge Ryz) \rightarrow Rxz))$

The formula $F(x)$: $\forall y\ Rxy$ is true in the structure $(\mathbf{N}, \leq, s, +, \times, 0, 1)$ for 0, but false for all the other natural numbers. The formula $F(x)$ is true for the smallest element, in a structure provided with a linear order.

The formula $G(x)$: $\exists y\ x = fy$ is true in the previous structure $(\mathbf{N}, \leq, s, +, \times, 0, 1)$ for all natural numbers distinct from 0 and false for 0.

The closed formula H: $\forall x \forall z \exists y (x = c \vee g(h(x, y), z) = c)$ of the language $\mathcal{L}_2 = \{R, f, g, h, c, d\}$ is true in the structure $(\mathbf{R}, \leq, s, +, \times, 0, 1)$ and false in the structure $(\mathbf{N}, \leq, s, +, \times, 0, 1)$.

Other examples of models and mathematical theories are given in the last section of the next chapter.

DEFINITION 3.12. *The* **value of a formula** $F(x_1, x_2, \ldots, x_k)$ *in a structure $\mathcal{M}$ is the set of all sequences of elements $(a_1, a_2, \ldots, a_k)$ in M which satisfy F.*

Example. The value of the formula $\exists y\ x = h(y, y)$ in the structure $\mathcal{R}$, where $\mathcal{R} = (\mathbf{R}, \leq, s, +, \times, 0, 1)$ is the set of all positive real numbers.

The definitions of satisfaction and of the value of a formula in a structure, which comes from the intuition concerning the use of quantifiers are not well adapted to prove general results. Meanwhile, they are sufficient for practical use, when it is asked to find the values of particular formulas. It is possible to give a definition of satisfaction by considering a global interpretation of the variables.

DEFINITION 3.13. *An* **interpretation** *s is a function defined from the set of variables V to the domain M of a structure.*

PROPOSITION 3.2. *Every interpretation s has a unique extension to the set T of terms.*

The proof is done by induction on the terms and is left as an exercise.
It is now possible to define the satisfaction of a formula F by an interpretation s and to show that it depends only on the values taken by s on the set of free variables of F. This property justifies the fact that the notion of satisfaction induced by an interpretation is a presentation more general than that of satisfaction of a formula induced by a sequence of elements in a structure.

DEFINITION 3.14. *The* satisfaction *of a formula F by an interpretation s is defined by induction on F:*

- *if F is an atomic formula of the form $Rt_1t_2\ldots,t_n$ and if $t_1^s, t_2^s, \ldots, t_n^s$ are the corresponding interpretations of terms $t_1, t_2, \ldots, t_n$ in the structure $\mathcal{M}$, formula F is satisfied by s if $(t_1^s, t_2^s, \ldots, t_n^s) \in R^{\mathcal{M}}$;*
- *if F is one of the form $\neg G$, $(G \wedge H)$, $(G \vee H)$, $(G \to H)$ or $(G \leftrightarrow H)$, the satisfaction of F is defined from the satisfaction of G and H, as in propositional logic;*
- *if F is of the form $\exists v\, G$, F is satisfied by s if there is an interpretation s', equal to s on $\mathcal{V} - \{v\}$ and satisfies G;*
- *if F is of the form $\forall v\, G$, F is satisfied by s if all interpretations s', equal to s on $\mathcal{V} - \{v\}$, satisfy G.*

PROPOSITION 3.3. *The satisfaction of a formula F by an interpretation s depends only on the values of s in the set of free variables of F.*

The proof is done by induction on the formula F and is left as an exercise.

3.2.3. Valid and equivalent formulas. The equivalence of formulas allows to classify them and to group formulas with the same meaning. One of these classes of formulas is particularly interesting: the formulas that are true in any structure.

DEFINITION 3.15. *Let $\mathcal{L}$ be a language of first-order.*

- *A closed formula is* **valid** *if it is true in any structure.*
- *The* universal closure *of a formula $F(x_1, x_2, \ldots, x_k)$ with free variables is the formula $\forall x_1 \forall x_2 \ldots \forall x_k\; F(x_1, x_2, \ldots, x_k)$.*
- *A formula $F(x_1, x_2, \ldots, x_k)$ is valid if its universal closure is valid.*
- *Two formulas F, G are* equivalent *if for every structure $\mathcal{M}$ and for every sequence of elements in M interpreting the free variables of F and G, they have the same values.*

According to the definition of the value of a formula, two formulas F, G are equivalent if and only if the formula $(F \leftrightarrow G)$ is valid. Another consequence of these definitions is very useful. If in a given formula ψ, we replace a subformula by an equivalent formula, we obtain a new formula equivalent to ψ.

Example. Let H be a tautology of propositional logic whose variables are among $\{p_1, p_2, \ldots, p_n\}$ and $F_1, F_2, \ldots, F_n$ be formulas of the language $\mathcal{L}$.

Then the formula obtained by substituting F_i by p_i $(i = 1, 2, \ldots, n)$ in H, denoted by $H(F_1/p_1, \ldots, F_n/p_n)$, is a valid formula. The formula

$$(\forall x \exists y Rxy) \vee \neg(\forall x \exists y Rxy)$$

is an example of a valid formula.

If F and $(F \to G)$ are valid formulas, then G is valid.

If F is a formula of $\mathcal{L}$, then the following formula is valid:

$$(\exists x \forall y F) \to (\forall y \exists x F)$$

In fact, if a structure $\mathcal{M}$ satisfies $\exists x \forall y F$, then there is an $a \in M$ such that, for all $b \in M$, $\mathcal{M} \models F[a, b]$. To every $b \in M$, we can associate the same $a \in M$, such that, $\mathcal{M} \models F[a, b]$.

The formula $(\forall y \exists x F) \to (\exists x \forall y F)$ is not valid, the structure $(\mathrm{N}, \leq, s, +, ., 0, 1)$ is a model of $(\forall y \exists x Ryx)$, but not of $(\exists x \forall y Ryx)$.

The following pairs of formulas are examples of equivalent formulas.

- F and $\forall x F$ (if x is not free in F)
- F and $\exists x F$ (if x is not free in F)
- $\forall x(F \wedge G)$ and $(\forall x F \wedge \forall x G)$
- $\exists x(F \vee G)$ and $(\exists x F \vee \exists x G)$
- $\exists x(F \to G)$ and $(\exists x \neg F \vee \exists x G)$
- $\exists x F$ and $\exists y F(y/x)$ (if x is free in F and y does not appear in F) (1)
- $\forall x F$ and $\forall y F(y/x)$ (if x is free in F and y does not appear in F) (2)

When x is not free in G, we get the following equivalences:

- $\forall x(F \wedge G)$ and $(\forall x F \wedge G)$ (3)
- $\exists x(F \vee G)$ and $(\exists x F \vee G)$ (4)

It remains to mention equivalences that can be derived using associativity and commutativity of the connectives $\wedge, \vee$, distributivity of $\wedge$ and $\vee$, and properties of negation. These equivalences, which are direct transpositions of those in propositional logic, are not rewritten here. The following equivalences express classical properties of quantifiers and will be used later on to obtain normal forms.

PROPOSITION 3.4. *Let F be a formula, x a variable and G a formula in which x is not free. The following is a list of equivalent formulas:*

- $\neg \forall x F$ *and* $\exists x \neg F$ (5)
- $\neg \exists x F$ *and* $\forall x \neg F$ (6)
- $(\forall x F \vee G)$ *and* $\forall x(F \vee G)$ (7)
- $(\exists x F \wedge G)$ *and* $\exists x(F \wedge G)$ (8)
- $(G \to \forall x F)$ *and* $\forall x(G \to F)$
- $(G \to \exists x F)$ *and* $\exists x(G \to F)$
- $(\forall x F \to G)$ *and* $\exists x(F \to G)$
- $(\exists x F \to G)$ *and* $\forall x(F \to G)$

Proof : A short justification is provided only for the first and the third cases. The other equivalences can be proved in the same way. It is understood that the value of a formula in a structure is considered relatively to an interpretation of the free variables in that structure. Formula $\neg\forall xF$ is true iff formula $\forall xF$ is false, which means there is an element satisfying the negation of F. For the formula $(\forall xF \vee G)$, we use the fact that x is not free in G and thus the value of G does not depend on the interpretation of x. □

3.2.4. Substitution. This operation allows one to replace a variable by a term.

DEFINITION 3.16. *Given a term t and a variable x appearing in this term, we can replace all occurrences of x by another term t'. The new term is said to be obtained by* **substitution** *of t' for x in t and is denoted by $t(t'/x)$.*

> **Example.** The result of the substitution of $f(h(u,y))$ for x in the term $g(y, h(c,x))$ is $g(y, h(c, f(h(u,y))))$. The result of the substitution of $g(x,z)$ for y in this new term is:
>
> $$g(g(x,z), h(c, f(h(u, g(x,z)))))$$

Before substituting any free variable, it is necessary to take some precautions. Otherwise, the meaning of the formula may be completely changed by a phenomenon called *"capture" of a variable.*

> **Example.** Let $F(x)$ be the formula $\exists y\ (gyy = x)$. In the structure $\mathcal{N}$ given by $(\mathbf{N}, \leq, s, +, \cdot, 0, 1)$, where g is interpreted as addition, the significance of $F(x)$ is obvious: $\mathcal{N} \models F[a]$ iff a is even. If we replace the variable x by z, the obtained formula has the same significance as the formula $F(x)$. But if we replace x by y, the resulting formula, $\exists y\ (gyy = y)$ is a closed formula, which is true in the structure $\mathcal{N}$. The meaning of the formula was changed, the variable x was replaced by a variable which is quantified in formula F.

DEFINITION 3.17. *The* substitution of a term t *for a* free variable x *in a* formula *F is achieved by replacing all free occurrences of this variable by the term t, under the condition that for each variable y appearing in t, any subformula of F starting with a quantifier $\forall y$ or $\exists y$ does not have free occurrences of x. When such substitution is possible, the result is a formula denoted by $F(t/x)$.*

> **Example.** The result of the substitution of the term fz for the variable x in the formula $F(x)$: $((Rcx \wedge \neg x = c) \wedge (\exists y\ gyy = x))$ is the formula $((Rcfz \wedge \neg fz = c) \wedge (\exists y\ gyy = fz))$.

PROPOSITION 3.5. *If F is a formula, x a free variable in F and t a term such that the substitution of t for x in F is defined, then the formulas $(\forall xF \rightarrow F(t/x))$ and $(F(t/x) \rightarrow \exists xF)$ are valid.*

Proof : We show by induction on formula F that the satisfaction of the formula $F(t/x)$ by the interpretation s is equivalent to that of the formula $F(x)$ by the interpretation s_1, where s_1 is obtained from s by assigning the value t^s to the variable x. The only case that requires justification is that when the formula F has the form $\forall y\ G$ or $\exists y\ G$. According to the hypothesis of substitution, the quantification can act only on a variable y which is distinct from x and distinct from every variable of t. It then suffices to examine the satisfaction of the formula $G(t/x)$ by an interpretation s', equal to s except for y. According to the induction hypothesis on G, the formula $G(t/x)$ is satisfied by the interpretation s' iff G is satisfied by the interpretation ${s'}_1$, where ${s'}_1$ is obtained from s' by assigning to the variable x the value $t^{s'}$, which is equal to t^s. In fact, s and s' are equal for every variable occurring in the term t. □

3.3. Prenex formulas and Skolem forms

The objective of this section is to determine a "standard" form for first-order formulas. The treatment applied to the quantifiers of a given formula aims to arrange them all as a prefix, followed by a formula without quantifiers. Afterwards it is possible to apply the transformation methods of propositional logic, in order to obtain a normal form.

3.3.1. Prenex formulas and normal forms. Some formulas have a specific structure and each formula can be equivalent to such specific formulas.

DEFINITION 3.18. *A* **prenex** *formula is a formula of the form*

$$Q_1x_1Q_2x_2\ldots, Q_kx_kG$$

where Q_i ($i = 1, 2, \ldots, k$) is a quantifier and G is a formula without any quantifier.

> **Example.** The formula $\forall x\exists y\forall z\ (Rxy \wedge (Rxz \rightarrow (Ryz \vee y = z)))$ is a prenex formula.
> The formula $\forall x\forall y\ ((Rxy \wedge \neg x = y) \leftrightarrow \exists z\ (y = g(x, h(z,z))))$ is not a prenex formula.

THEOREM 3.1. *Every formula is equivalent to a prenex formula.*

Proof : The proof is by induction on formulas.

- The property is obvious for atomic formulas.
- Let F be of the form $\neg G$, where G is assumed equivalent to a prenex formula $Q_1x_1Q_2x_2\ldots, Q_kx_kG'$. Applying property (5) or (6) k times, we get a prenex formula equivalent to F: $\overline{Q}_1x_1\overline{Q}_2x_2\ldots\overline{Q}_kx_k\neg G'$, where $\overline{Q}_i$ is $\exists$ if Q_i is $\forall$, and $\overline{Q}_i$ is $\forall$ if Q_i is $\exists$.
- Let F be of the form $\forall x\ G$, where G is assumed equivalent to a prenex formula G'. F is then equivalent to $\forall x\ G'$, which is a prenex formula.
- Let F be of the form $\exists x\ G$, where G is assumed to be equivalent to a prenex formula G'. F is then equivalent to $\exists x\ G'$, which is a prenex formula.

- Let F be of the form $(G \wedge H)$, where G, H are assumed to be equivalent to the prenex formulas G', H' respectively. F is then equivalent to $(G' \wedge H')$. If G' is of the form $\forall x\ G''$ or $\exists x\ G''$, and if x is not free in H', we apply property (3) or (8), in order to remove the quantifiers. If x is free in H', we then apply property (2) or (1). If y is a variable which does not appear in neither G' nor H', the formula $\forall x\ G''$, respectively $\exists x\ G''$, is equivalent to $\forall y\ G''(y/x)$, respectively to $\exists y\ G''(y/x)$. Then we can apply property (3) or (8), since y is not free in H'. Finally, formula F is equivalent to one of the formulas $\forall x\ (G'' \wedge H')$, $\exists x\ (G'' \wedge H')$, $\forall y\ (G''(y/x) \wedge H')$ or $\exists y\ (G''(y/x) \wedge H')$. Formula $(G'' \wedge H')$ contains one quantifier less than the formula $(G' \wedge H')$. Repeating this transformation as long as the formula $(G' \wedge H')$ contains quantifiers, we obtain a prenex formula equivalent to F.
- If F is of the form $(G \vee H)$, the sequence of transformations applied is analogous and uses properties (1), (2), (4) and (7).
- Let F be of the form $(G \rightarrow H)$, where G, H are assumed to be equivalent to the prenex formulas G', H' respectively. Then F is equivalent to $(G' \rightarrow H')$, and thus to $(\neg G' \vee H')$. The applied method uses transformations applicable to the operators $\neg$ and $\vee$.
- If F is of the form $(G \leftrightarrow H)$, where G, H are assumed to be equivalent to the prenex formulas G', H' respectively, F is equivalent to $(G' \leftrightarrow H')$. By equivalence, we can eliminate the operator $\leftrightarrow$ using operators $\wedge, \rightarrow$ and then use the transformations concerning them.

This proof provides an effective method to construct a prenex formula equivalent to a given formula. □

Example. The formula $\forall x \forall y\ ((Rxy \wedge \neg x = y) \rightarrow \exists z\ (y = gxhzz))$ is transformed to its prenex form using the following steps:
$\forall x \forall y\ ((Rxy \wedge \neg x = y) \rightarrow \exists z\ (y = gxhzz))$
$\forall x \forall y\ ((\neg Rxy \vee x = y) \vee \exists z\ (y = gxhzz))$
$\forall x \forall y \exists z\ ((\neg Rxy \vee x = y) \vee (y = gxhzz))$

DEFINITION 3.19.

- *A* literal *is an atomic formula or a negation of an atomic formula.*
- *A* clause *is a disjunction of literals.*
- *A prenex formula $Q_1x_1Q_2x_2 \ldots, Q_kx_kG$ is in* conjunctive normal form *if the quantifier free formula G is a conjunction of clauses.*

For example, the prenex formula obtained in the previous example is already in conjunctive normal form. Similarly the notion of a prenex formula in disjunctive normal form may be defined.

COROLLARY 3.1. *Every formula is equivalent to a prenex formula in conjunctive (respectively disjunctive) normal form.*

Proof : Let F be a formula and $Q_1x_1Q_2x_2 \ldots, Q_kx_kG$ a prenex formula equivalent to F. We denote by $A_1, A_2, \ldots, A_k$ all atomic formulas occurring in G. There is a propositional formula H constructed with the variables $\{p_1, p_2, \ldots, p_k\}$, such that formula G is equal to the formula $H(A_1/p_1, A_2/p_2, \ldots, A_k/p_k)$. Let H' be a conjunctive normal form equivalent to H, as constructed in propositional logic. The following transformations are used for this construction:

- elimination of the connectives $\rightarrow, \leftrightarrow$ using the connectives $\neg, \vee, \wedge$,
- moving negations as far inside as possible, by using De Morgan's rules such that negations are applied only to propositional variables,
- distributing conjunctions over disjunctions.

If we want a disjunctive normal form, the last transformation must use the distribution of $\vee$ over $\wedge$. In the case of a normal conjunctive form, the formula G is equivalent to the formula $H'(A_1/p_1, A_2/p_2, \ldots, A_k/p_k)$. The formula F is thus equivalent to some prenex formula in conjunctive normal form. □

3.3.2. Skolem forms. We will present a method for transforming every prenex formula to a formula whose prefix is composed only of universal quantifiers. The property of equivalence between formulas will be lost in the process; but that of the existence of a structure satisfying the formula will be conserved.

DEFINITION 3.20. *Let $\mathcal{L}$ be a first-order language.*

- *A formula F is said to be* universal *if it is a prenex formula and all quantifiers occurring in F are universal.*
- *A language $\mathcal{L}'$ is a* Skolem extension *of $\mathcal{L}$ if it is obtained from $\mathcal{L}$ by adding an infinite number of function symbols of each arity and an infinite number of constant symbols.*

A closed prenex formula F of $\mathcal{L}'$ is either universal or of the form $\forall x_1 \forall x_2 \ldots \forall x_k \exists x\ G$, where G is a prenex formula. In the latter case, k may be 0 and F is of the form $\exists x\ G$. The transformation applied to F, if it has at least one existential quantifier, consists of associating to F the formula $\forall x_1 \forall x_2 \ldots \forall x_k\ G(f(x_1, x_2, \ldots, x_k)/x)$, where f is a function symbol which does not appear in formula G. In the special case when $F = \exists x\ G$, we associate to it the formula $G(c/x)$, where c is a constant symbol, which does not appear in the formula G. The resulting formula F_1 has one existential quantifier less than the formula F.

Example. We associate the formula

$$\forall x \forall y\ (Rf(x)g(x,y) \rightarrow (Rf(x)k(x,y) \wedge Rk(x,y)h(x,y)))$$

to the formula $\forall x \forall y \exists z\ (Rf(x)g(x,y) \rightarrow (Rf(x)z \wedge Rzh(x,y)))$ in the language $\mathcal{L} = \{R, f, g, h, c, d\}$; here k is a new binary function symbol.

DEFINITION 3.21. *Let F be a closed prenex formula (in the language $\mathcal{L}'$) having n existential quantifiers.*

- *A* Skolem form *of F is a formula obtained by applying the previous transformation n times.*

- *The new functions and constants introduced during these transformations are called* Skolem functions *and* Skolem constants.

A Skolem form of F is a universal formula.

Example. Let

$F = \exists x \forall y \forall x' \exists y' \forall z\ (Rxy \to (Rx'y' \wedge (Rx'z \to (Ry'z \vee y' = z))))$

The Skolem form of F is:

$\forall y \forall x' \forall z\ (Rey \to (Rx'k(y,x') \wedge (Rx'z \to (Rk(y,x')z \vee k(y,x') = z))))$

where e is a new constant symbol and k a new binary function symbol.

The set Σ' of Skolem forms, of a set of closed prenex formulas Σ, is well defined if we respect the condition stating that newly introduced Skolem symbols are pairwise distinct. The property justifying the use of Skolem forms is provided by the following theorem.

THEOREM 3.2. *Let Σ' be the set of Skolem forms for a set of closed prenex formulas Σ. Then Σ has a model iff Σ' has one.*

Proof : Assume that $\mathcal{M}$ is a model of Σ'. It is enough to prove that for any F' obtained by transforming F, if F' is true in $\mathcal{M}$, then F is also true. If $F = \forall x_1 \forall x_2 \ldots \forall x_k \exists x\ G$, then $F' = \forall x_1 \forall x_2 \ldots \forall x_k\ G(f(x_1, x_2, \ldots, x_k)/x)$. The result follows from the validity of the formula:

$$\forall x_1 \forall x_2 \ldots \forall x_k\ G(f(x_1, x_2, \ldots, x_k)/x) \to \forall x_1 \forall x_2 \ldots \forall x_k \exists x\ G$$

If $F = \exists x\ G$, then $F' = G(c/x)$. The result follows from the the validity of:

$$G(c/x) \to \exists x\ G$$

To prove the converse, we consider a model $\mathcal{M}$ of Σ and we will enlarge it in order to obtain a model of Σ'. This extension of $\mathcal{M}$ is defined in successive stages, each stage corresponding to a transformation. It suffices to define, for each stage, the interpretation of the corresponding Skolem function or the corresponding Skolem constant. If $F = \forall x_1 \forall x_2 \ldots \forall x_k \exists x\ G$, the interpretation of the Skolem function f is defined by associating to each sequence $a_1, a_2, \ldots, a_k$ in M, an element $f^{\mathcal{M}}(a_1, a_2, \ldots, a_k)$ such that:

$$\mathcal{M} \models G(a, a_1, a_2, \ldots, a_k)$$

which is possible since $\mathcal{M}$ is a model of F. If $F = \exists x\ G$, the interpretation of the Skolem constant c is defined by taking an element $c^{\mathcal{M}}$ among $b \in M$ such that it will satisfy G in $\mathcal{M}$. □

3.4. Exercises

(1) Show that the following expression is a term of the language $\mathcal{L}_2$:
$hgyhcxfgdhyz$

(2) Give the decomposition of the following formula:
$F\ :\ \neg(\exists y\ Rxy \to \forall x(\exists y\ Rxy \wedge \forall z(Rxz \to (Ryz \vee y = z))))$

(3) Let M be the domain of a structure $\mathcal{M}$ for the language $\mathcal{L}$, N a non-empty subset of M containing the interpretation of every constant of $\mathcal{L}$ and stable under the application of every function defined on M. We associate a structure $\mathcal{N}$ to N, called the **substructure** of $\mathcal{M}$, as follows:
   - $R^{\mathcal{M}} \cap N^n$ is the relation associated to the n-ary predicate R,
   - the restriction to N^m of the corresponding function is associated to a function symbol f with m arguments,
   - the same element as in M is the interpretation of the constant symbol c.

   Prove that the set of the non-negative integers defines a substructure of $(\mathbb{R}, \leq, s, +, \times, 0, 1)$, which we denote by $(\mathbb{N}, \leq, s, +, \times, 0, 1)$.

(4) What is the result of the substitution of $f(h(u,y))$ for x in the following term:
$h(g(y, h(c,x)), f(g(d, h(y,z))))$?
What is the result of the substitution of $g(x,z)$ for y in this new term?

(5) Let $F(x,y)\ :\ \neg(\exists y\ Rxy \to \forall x\ (\exists y\ Rxy \wedge \forall z\ (Rxz \to (Ryz \vee y = z))))$. Can we substitute $f(h(x,y))$ for the free variable x and $g(x,z)$ for the free variable y in $F(x,y)$? Can we substitute $f(h(x,z))$ for the free variable x in $F(x,y)$?

(6) Show that if F is a formula, x a free variable in F and t a term such that the substitution of t for x in F is defined, then the formulas $(\forall x\ F \to F(t/x))$ and $(F(t/x) \to \exists x\ F)$ are valid.

(7) Find an equivalent prenex formula for each of the following two formulas:
$(\forall x \exists y\ Rxy \to \forall x \exists y\ (Rxy \wedge \forall z\ (Rxz \to (Ryz \vee y = z))))$
$\forall x \forall y\ ((Rxy \wedge \neg x = y) \leftrightarrow \exists z\ (y = gxhzz))$

(8) Give a Skolem form for each of these two formulas.