During the late 1920s, Jacques Herbrand, a young mathematician, developed a method to check the validity of a class of first-order logic formulas. In his thesis, published in 1931, Herbrand discussed what can be considered the first unification procedure. Unification is at the heart of modern implementations of logic programming languages.

In this chapter, we will discuss the model of computation that serves as a basis for the logic programming paradigm. Prolog, one of the most popular logic programming languages, will be discussed in the final part of the chapter.

5.1 The Herbrand universe

In logic programs, the domain of computation is the Herbrand universe, the set of terms defined over a universal alphabet of

- variables, such as $X, Y$, etc., and
- function symbols with fixed arities (the arity of a symbol is as usual the number of arguments associated with it).

Function symbols are usually denoted by $f, g, h, \ldots$, or $a, b, c, \ldots$ if the arity is 0 (i.e., $a, b, c, \ldots$ denote constants). In our examples, we will often use more meaningful names for function symbols.
Definition 5.1 (Terms)

A *term* is either a variable or has the form $f(t_1, \ldots, t_n)$, where $f$ is a function symbol of arity $n$ and $t_1, \ldots, t_n$ are terms. Notice that $n$ may be 0, and in this case we will just write $f$, omitting the brackets.

Example 5.2

If $a$ is a constant, $f$ a binary function, and $g$ a unary function, then $f(f(X, g(a)), Y)$ is a term, where $X$ and $Y$ are variables.

Function symbols in this framework correspond to data constructors; they are used to give structure to the domain of computation. For example, if our algorithm deals with arrays of three elements, a suitable data structure can be defined using a function symbol `array` of arity 3. The array containing the elements 0, 1, 2 is then represented by the term `array(0, 1, 2)`.

There is no definition associated with a function symbol (although in Prolog implementations there are some built-in functions, such as arithmetic operations, that have a specific meaning).

We will not fix the alphabet used to define the Herbrand universe. The names of variables and function symbols needed to represent the problem domain can be freely chosen. In this chapter, names of variables start with capital letters and names of functions start with lower case letters (we follow the conventions used in Prolog’s syntax).

5.2 Logic programs

Once the domain of computation is established, a problem can be described by means of logic formulas involving *predicates*.

Predicates represent properties of terms and are used to build basic formulas that are then composed using operators such as *and*, *not*, and *or*, denoted by $\land$, $\neg$, and $\lor$, respectively.

Definition 5.3

Let $\mathcal{P}$ be a set of *predicate symbols*, each with a fixed arity. If $p$ is a predicate of arity $n$ and $t_1, \ldots, t_n$ are terms, then $p(t_1, \ldots, t_n)$ is an *atomic formula*, or simply an *atom*. Again, $n$ may be 0, and in this case we omit the brackets. A *literal* is an atomic formula or a negated atomic formula.