

Variables and Structures

Our goal is to use a logical language to describe concrete mathematical situations. To do so, we introduce the notion of a structure, which connects abstract formulas with actual mathematical objects. Before deciding whether, in a given structure, a formula is true or false, it is essential to distinguish free variables from bound variables.

1. Free variables, bound variables

1.1. Idea

Consider a language $\mathcal{L}$ with variables $x, y, z, x_1, x_2, \dots$. In a given formula, a variable may be free or bound. Let us understand the difference through examples before giving the precise definition.

Consider a formula “ $\forall x \ P(x)$ ” or “ $\exists x \ P(x)$ ”. Here the variable “ x ” is *quantified*, that is, *bound* by a quantifier. Likewise, the two variables x and y in “ $\forall x \ \exists y \ Q(x, y)$ ” are quantified.

By contrast, in the formula “ $\forall x \ Q(x, y)$ ”, the variable y is not quantified; we say that this variable is *free* (but the formula is still legitimate).

Here is a formula (under an interpretation):

$$\forall x \in \mathbb{R} \quad \exists y \in \mathbb{R} \quad y > x$$

Since there are no free variables, we can assign this formula a truth value T/F.

By contrast:

$$\forall x \in \mathbb{R} \quad y > x$$

is a formula in which y is a free variable. It makes no sense to ask whether this formula is true or false until a value has been assigned to y .

1.2. Definition

Scope

Definition.

In a formula containing “ $\forall x \ P$ ” or “ $\exists x \ P$ ”, the *scope* of $\forall$ (or $\exists$) is P .

It is worth noting that P may be an arbitrarily complex formula.

Example:

$$\forall x \quad \exists y \quad \underbrace{P(x) \rightarrow Q(x, y)}_{\text{scope of } \exists}$$

$\underbrace{\hspace{10em}}_{\text{scope of } \forall}$

Free or bound variable

Definition.

In a formula, a variable x is a **bound variable** if every occurrence of x lies within the scope of a quantifier $\forall x$ or $\exists x$. Otherwise, x is called a **free variable**.

Examples:

- $\forall x \quad P(x) \vee Q(x, y)$: x is bound, y is free.
- $P(x) \rightarrow Q(x, y)$: x and y are free.
- $(\forall x \quad P(x)) \leftrightarrow (\exists y \quad P(y))$: x and y are bound.
- $(\forall x \quad x = y) \vee (\exists y \quad x = y)$: x and y are free!

There is an important subtlety here, illustrated by this last example. For a variable x to be free, it is enough for at least one occurrence of x not to lie within the scope of a quantifier. In our example, y is a free variable in the left-hand part and x is free in the right-hand part, so x and y are indeed free variables in the formula as a whole.

Closed formula**Definition.**

A formula is **closed** if it contains no free variables.

It is also called a **closed formula**, or a **sentence**.

The idea is that we will be able to assign a truth value T/F to a closed formula, whereas a formula that is not closed still has free variables to which values must be assigned.

Examples:

- $\forall x \quad (P(x) \wedge (\exists y \quad P(y)))$ is a closed formula.
- $\exists x \quad ((\exists y \quad P(x) \wedge P(y)) \rightarrow (\forall y \quad Q(x, y)))$ is also a closed formula.
- $\exists x \quad (Q(x, y) \rightarrow (\exists z \quad P(z)))$ is not closed because y is a free variable.
- $(\forall x \quad P(x)) \vee (\exists x \quad Q(x)) \vee (\neg R(x))$ is not a closed formula because x is a free variable (in the right-hand subformula).

1.3. Substitution

In a language $\mathcal{L}$, let $\mathcal{P}$ be a formula, x a variable, and t a term.

Definition.

$\mathcal{P}[t/x]$ denotes the formula in which every free occurrence of the variable x has been replaced by the term t .

Remark: recall that terms include constants and variables, and, if the language contains a function symbol f , then $f(t_1, \dots, t_n)$ is also a term whenever $t_1, \dots, t_n$ are terms.

Example.

Let $\mathcal{L}$ be a language containing constants a, b , a binary function denoted by $+$, a unary predicate P , and a binary predicate Q .

Let us study the formula:

$$\mathcal{P} : \quad \forall y \quad P(x) \rightarrow Q(x, y)$$

- $t = a, \mathcal{P}[t/x] : \forall y \quad P(a) \rightarrow Q(a, y)$

- $t = x + b, \mathcal{P}[t/x] : \forall y \quad P(x + b) \rightarrow Q(x + b, y)$
- $t = b, \mathcal{P}[t/y] : \forall y \quad P(x) \rightarrow Q(x, y)$: the formula does not change because every occurrence of y lies within the scope of a quantifier.

Remark: the formal definition of substitution is given by induction, beginning with atomic formulas and then introducing the logical connectives. For formulas with quantifiers “ $\forall x \mathcal{P}$ ” or “ $\exists x \mathcal{P}$ ”, no substitution for the variable x is made because the variable is quantified.

We can also perform a **simultaneous substitution** $\mathcal{P}[t_1/x_1, t_2/x_2, \dots, t_n/x_n]$. It is denoted by $\mathcal{P}[s]$, where s denotes $(t_1/x_1, t_2/x_2, \dots, t_n/x_n)$. It is important to understand that the substitutions are simultaneous: every free occurrence of x_1 is replaced by t_1 and, *at the same time*, every free occurrence of x_2 is replaced by t_2 , and so on.

Example.

Let $\mathcal{P}$ be the formula “ $P(x, y)$ ”. Then:

- $\mathcal{P}[a/x, b/y] : P(a, b)$
- $\mathcal{P}[x + y/x, x/y] : P(x + y, x)$
- $\mathcal{P}[y/x, x/y] : P(y, x)$

Remark: the formula $\mathcal{P}$: “ $P(x) \rightarrow (\forall x \quad Q(x))$ ” is a legitimate formula of the language (it is as though we had two different x ’s: one red and one blue). Substituting a for x gives $\mathcal{P}[a/x]$: “ $P(a) \rightarrow (\forall x \quad Q(x))$ ”, because only the free occurrences of x are replaced. We would prefer to write the initial formula as $\mathcal{P}'$: “ $P(x) \rightarrow (\forall y \quad Q(y))$ ”, which is easier to understand.

Variable capture.

When performing a substitution $\mathcal{P}[t/x]$, we always assume that no variable occurring in t falls within the scope of a quantifier in $\mathcal{P}$. If the term t contains variables that might become bound by quantifiers in $\mathcal{P}$, the substitution could change the logical meaning of the formula: this is called variable capture. For example, let:

$$\mathcal{P} : \quad \forall y \quad P(x) \rightarrow Q(x, y)$$

Substituting $t = y$ for x is not legitimate, because it would give the formula:

$$\mathcal{P}[y/x] : \quad \forall y \quad P(y) \rightarrow Q(y, y)$$

which does not have the same meaning at all.

To avoid this phenomenon, we first rename the bound variables of $\mathcal{P}$ before performing the substitution. In our example, we first rewrite $\mathcal{P}$ in the following form:

$$\mathcal{P} : \quad \forall z \quad P(x) \rightarrow Q(x, z)$$

and then substitute y for x :

$$\mathcal{P}[y/x] : \quad \forall z \quad P(y) \rightarrow Q(y, z)$$

2. Structures

A language remains an abstract shell; it provides a framework for a wide variety of mathematical objects and their properties: integers, groups, vector spaces, set theory, and so on.

2.1. Definition

Definition.

Consider a language $\mathcal{L}$. A **structure** $\mathcal{S}$ for this language consists of:

- a nonempty set E , called the **domain**,
- an element $c^{\mathcal{S}} \in E$ associated with each constant symbol c of $\mathcal{L}$,
- a subset $P^{\mathcal{S}} \subset E^n$ associated with each n -place predicate symbol P , in other words, a function $\phi_P : E^n \rightarrow \{T, F\}$ such that $\phi_P(a_1, \dots, a_n)$ is true if and only if $(a_1, \dots, a_n) \in P^{\mathcal{S}}$,
- a function $f^{\mathcal{S}} : E^n \rightarrow E$ associated with each n -place function symbol f .

$\mathcal{S}$ is also called an **interpretation** of the language $\mathcal{L}$. Several different structures can be associated with the same language.

A structure is therefore a concrete mathematical realization of a language. To define a structure, we must define all the nonlogical symbols: constants, predicates, and functions. The logical symbols are left unchanged: a variable x remains a variable x , except that it now ranges over the domain E .

2.2. A first example

Consider a language $\mathcal{L} = (c, P, f)$, where c is a constant, $P(x, y)$ a binary predicate, and $f(x)$ a unary function. As usual, we do not explicitly state that the variables are x, y , that the logical connectives are present, and that there is also the equality predicate “ $x = y$ ”.

We will define a structure that corresponds to the natural numbers and models the successor. Consider the structure $\mathcal{S}$ with domain $E = \mathbb{N}$ and interpretation $(0, <, x \mapsto x + 1)$ of the language, that is:

- for the constant c of $\mathcal{L}$, define $c^{\mathcal{S}} = 0$; this is the interpretation of the symbol c in $\mathcal{S}$,
- for the predicate P of $\mathcal{L}$, define $P^{\mathcal{S}}$ by $P^{\mathcal{S}}(x, y)$: $x < y$,
- for the function f of $\mathcal{L}$, define $f^{\mathcal{S}}$ by $f^{\mathcal{S}}(x)$: the successor of the natural number x , denoted by $x + 1$.

Once the structure has been defined, every formula of $\mathcal{L}$ can be interpreted as a mathematical statement about the structure. Here are some examples:

Formula of $\mathcal{L}$	Formula interpreted in $\mathcal{S}$	True or false in $\mathcal{S}$?
$\forall x \exists y P(x, y)$	$\forall x \in \mathbb{N} \exists y \in \mathbb{N} x < y$	It is true
$\forall x P(c, f(x))$	$\forall x \in \mathbb{N} 0 < x + 1$	It is also true
$\forall x f(x) = y$	$\forall x \in \mathbb{N} x + 1 = y$	The question makes no sense because y is a free variable

When defining this structure $\mathcal{S}$, we will often write, somewhat abusively, $c = 0$, $P(x, y)$: $x < y$, and $f(x) = x + 1$ (instead of $c^{\mathcal{S}}$, $P^{\mathcal{S}}$, and $f^{\mathcal{S}}$).

There are of course many other possible structures for this same language. We can change the domain, the constant, the predicate, or the function:

- Structure $\mathcal{S}_1$ with domain $\mathbb{R}$, again with $(0, <, x \mapsto x + 1)$, except that this time the interpreted formula $\forall x \in \mathbb{R} \quad 0 < x + 1$ is false.
- Structure $\mathcal{S}_2$ with domain $\mathbb{R}$, with $(0, \leq, x \mapsto x^2)$: the formula $\forall x \quad P(c, f(x))$ of the language becomes $\forall x \in \mathbb{R} \quad 0 \leq x^2$, a formula that is true under this interpretation.

2.3. Comments

Here are some important remarks.

Nonempty domain

The domain of a structure must be a nonempty set. This may seem like a detail, but it is quite important. It seems natural to everyone that:

If “ $\forall x \in E \quad P(x)$ ” is true, then “ $\exists x \in E \quad P(x)$ ” is also true.

But suppose for a moment that E were the empty set. Then “ $\forall x \in E \quad P(x)$ ” would always be true (since there is nothing to check, because there is no x to consider). By contrast, “ $\exists x \in E \quad P(x)$ ” would always be false (because E is empty). This situation would be problematic.

We can also argue that if E were empty, then “ $\forall x \in E \quad x \neq x$ ” would be true and “ $\exists x \in E \quad x = x$ ” would be false, which would be very troublesome.

Conclusion: we always assume that the domain E is nonempty.

Functions

Let us return briefly to functions. In the language, f is merely a function symbol that must be followed by n terms (for an n -place function). In the structure, we must define the interpretation of this function using a good old-fashioned function: $f^S : E^n \rightarrow E$. It is a function of n variables in the usual sense. As already mentioned, in practice we simply denote it by $f : E^n \rightarrow E$.

For example, let $\mathcal{L}$ be a language with a unary function f and a binary function g .

- For a domain $\mathbb{N}$, we could define f by $f(x) = x + 1$ or $f(x) = x^2$, and g by $g(x, y) = x + y$ or $g(x, y) = xy$. By contrast, we cannot define f by $f(x) = x - 1$, because $f(0)$ is not in the domain $\mathbb{N}$, nor can we define g by $g(x, y) = x/y$, which does not always yield integers.
- On the domain $] - \infty, 0[$, however, we could define $f(x) = x - 1$ and $g(x, y) = -x/y$.

Predicates

The interpretation of a predicate may be a little more difficult to understand. In the language, an n -ary predicate is merely a symbol that must be followed by n terms. For the structure, we must define each predicate, that is, specify for which objects its value is True.

Consider the example of a unary predicate P in a language $\mathcal{L}$. To define a structure S , we define its domain E and then the interpretation of the predicate, that is, all the values in E that make $P(x)$ true. We can do this by defining a subset $P^S \subset E$, which is exactly the set of all $d \in E$ such that $P(d)$ is true. We can also define it as a function $\phi_P : E \rightarrow \{T, F\}$ that returns True or False. The connection between the two is that $\phi_P(d) = T$ if and only if $d \in P^S$.

Thus, if we want to define an interpretation of “being a real number greater than or equal to 7”, we set $E = \mathbb{R}$ and, for the predicate, either define $P^S = [7, +\infty[$, the set of values that satisfy the property, or define a function:

$$\begin{aligned} \phi_P : E &\longrightarrow \{T, F\} \\ x &\longmapsto \begin{cases} T & \text{if } x \in [7, +\infty[\\ F & \text{otherwise.} \end{cases} \end{aligned}$$

Remark: we emphasized that a domain must be nonempty. By contrast, we do not require a predicate to be realized. In other words, P may always evaluate to false in the structure. Thus, over $\mathbb{R}$, we are allowed to define the predicates “ $x^2 < 0$ ” or “ $x \neq x$ ”, even though no $x \in \mathbb{R}$ will realize them. To define a binary predicate $P(x, y)$ in a structure with domain E , we must define a set $P^S \subset E^2$, which will be the set of pairs (d_1, d_2) such that $P(d_1, d_2)$ is true.

Domain

When we define a structure, we fix a domain E . The variables are then interpreted as objects in this domain. For example, the formula of the language “ $\forall x \exists y \quad Q(x, y)$ ” can be interpreted in a structure where $E = \mathbb{N}$ as “ $\forall x \in \mathbb{N} \exists y \in \mathbb{N} \quad x < y$ ”. But then how can we write a formula in which x and y do not range over exactly the same set, such as “ $\forall \varepsilon > 0 \exists n \in \mathbb{N} \quad n > \frac{1}{\varepsilon}$ ”? We need to use a small trick. Let A and B be two subsets of the same set E . We would like to write the mathematical statement:

$$\forall x \in A \quad \exists y \in B \quad Q(x, y)$$

It cannot come directly from a formula of the language, because x and y must range over the same set. Define a predicate P_A such that $P_A(x)$ is true exactly when $x \in A$. Likewise, define P_B so that $P_B(x)$ is true when $x \in B$. Consider the formula of the language:

$$\forall x \quad (P_A(x) \rightarrow \exists y (P_B(y) \wedge Q(x, y)))$$

It is interpreted in our structure with domain E (with P_A and P_B defined as above) as:

$$\forall x \in E \quad (x \in A \rightarrow \exists y \in E (y \in B) \wedge Q(x, y))$$

If $x \notin A$, then this statement is true (because the implication is true by definition). If $x \in A$, then for this statement to be true, there must exist a $y \in B$ such that $Q(x, y)$ is true. This is indeed what we wanted mathematically.

2.4. Other examples

Inverse

We want to express the existence of an inverse (or an additive inverse). This is a very general notion that appears in many situations and is therefore worth conceptualizing. Let $\mathcal{L}$ be the language with a constant e and a binary function $f(x, y)$, which we denote by $x \circ y$. We can write a formula expressing the existence of a (right) inverse:

$$\forall x \quad \exists y \quad x \circ y = e$$

Such a y is called a (*right*) *inverse* of x .

Concretely, here is what this covers:

- **Additive inverse.** Define the structure $\mathcal{S}$ with domain $\mathbb{Z}$, with $e^S = 0$ and $f^S(x, y) = x + y$. The formula of the language becomes:

$$\forall x \in \mathbb{Z} \quad \exists y \in \mathbb{Z} \quad x + y = 0$$

In other words, $y = -x$ is the *additive inverse* of x here.

- **Multiplicative inverse.** This time define the structure $\mathcal{S}$ with domain $\mathbb{R}^*$, with $e^S = 1$ and $f^S(x, y) = x \times y$. The formula of the language becomes:

$$\forall x \in \mathbb{R}^* \quad \exists y \in \mathbb{R}^* \quad x \times y = 1$$

In other words, $y = \frac{1}{x}$ is the *inverse* of x here (in the algebraic sense).

- **Matrix inverse.** Consider the set of invertible $n \times n$ matrices, equipped with matrix multiplication, and let $e^S = I$ be the identity matrix. In this structure, the formula expresses $A \times B = I$, that is, $B = A^{-1}$ is the inverse of the matrix A .

- **And so on.** We could continue with a group $(G, \diamond)$ with identity element e , to express $g \diamond h = e$; or with the notion of the inverse bijection of a function, to express $g \circ h = \text{id}$; and so on. All of this uses the same language $\mathcal{L}$.

There are of course other formulas of the language that are of interest for all these structures. Associativity:

$$\forall x \quad \forall y \quad \forall z \quad x \circ (y \circ z) = (x \circ y) \circ z$$

The action of the identity element, which changes nothing:

$$\forall x \quad ((e \circ x) = x) \wedge ((x \circ e) = x)$$

Graph

Let us illustrate how the same formula can become true or false depending on the chosen structure. Consider the language $\mathcal{L}$ with the predicate $R(x, y)$, intended to model undirected graphs.

Consider the formula of $\mathcal{L}$:

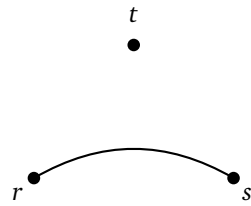
$$\forall x \quad \exists y \quad (x \neq y) \wedge R(x, y)$$

which should be understood as “every vertex is connected to at least one other vertex”.

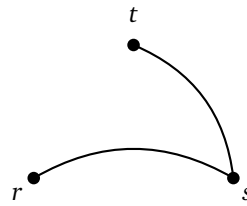
- Define S_1 with domain $E = \{r, s, t\}$, where R^{S_1} is defined by the graph G_1 on the left below; that is, R is true only for the pairs (r, s) and (s, r) (which are connected to one another).

In this structure, the interpreted formula is false because t is not connected to any vertex.

- If, on the other hand, we define S_2 with domain $E = \{r, s, t\}$, where the predicate is now defined by the graph G_2 on the right, then the interpreted formula is true.



Graph G_1



Graph G_2

2.5. Advanced examples

Rings

Consider a language $\mathcal{L} = (0, 1, +, -, \times)$, where $0, 1$ are constants and $+, -, \times$ are binary functions (strictly speaking, we should write these symbols as a, b, f, g, h , because here 0 is only a symbol of the language, named in anticipation of its interpretation). This language is suited to modeling the notion of a ring (a set with addition, subtraction, and multiplication, but not necessarily inverses).

Here are some examples of structures:

- **Integers.** Consider the structure $\mathcal{S}$ with domain $\mathbb{Z}$. The constants are naturally interpreted as the integers 0 and 1 , and the functions are interpreted as the usual operations on integers.
- **Polynomials.** For this structure $\mathcal{S}$, the domain is the set of polynomials $\mathbb{R}[X]$; this time 0 denotes the zero polynomial, 1 the constant polynomial equal to 1 , and the operations are polynomial addition, subtraction, and multiplication.
- **Matrices.** For this new structure, the domain is the set $M_n(\mathbb{R})$ of $n \times n$ matrices. Naturally, 0 denotes the zero matrix, 1 denotes the identity matrix I , and the operations are matrix operations.

The point of the language is therefore to formalize properties that apply in many situations. For a ring, we additionally require the following formulas to be true once interpreted in the structure:

- $\forall x \quad x - x = 0$: subtraction corresponds to taking the additive inverse,
- $\forall x \quad 1 \times x = x$: 1 is an identity element for multiplication,
- $\forall x \quad \forall y \quad \forall z \quad x \times (y + z) = x \times y + x \times z$: distributivity,
- and so on.

Set theory

Consider a language $\mathcal{L} = (\in)$, where the membership symbol “ $\in$ ” denotes a binary predicate $P(x, y)$, written here as $x \in y$.

Warning: in the following structures, the objects will be sets, and the predicate $x \in y$ will ask whether the set x is indeed an *element* of the set y . For example, $\{1\} \in \{\emptyset, \{1\}, \{2\}, \{1, 2\}\}$: a set is indeed an element of another set. The symbol $\in$ will not denote the usual membership $1 \in \{1, 2, 3\}$ (since 1 and $\{1, 2, 3\}$ are different kinds of objects, one being an integer and the other a set of integers), nor the inclusion $\{1, 2\} \subset \{1, 2, 3\}$, which denotes something else.

Let us construct a first structure S_1 . Its domain is $E = \{\emptyset, \{\emptyset\}, \{\{\emptyset\}\}, \{\{\{\emptyset\}\}\}, \dots\}$; every element is a singleton except the empty set. The predicate is defined by the following relations:

$$\emptyset \in \{\emptyset\} \quad \{\emptyset\} \in \{\{\emptyset\}\} \quad \dots$$

No other membership relations hold.

Formally, define $E = \bigcup_{n \geq 0} E_n$, where:

$$E_0 = \{\emptyset\} \quad \text{and} \quad E_{n+1} = E_n \cup \{\{x\} \mid x \in E_n\}$$

Thus, E_1 is formed from $\emptyset$ and $\{\emptyset\}$, so $E_1 = \{\emptyset, \{\emptyset\}\}$. Then E_2 is formed by the union of E_1 and $\{\{\emptyset\}, \{\{\emptyset\}\}\}$, so $E_2 = \{\emptyset, \{\emptyset\}, \{\{\emptyset\}\}\}$ (we do not count $\{\emptyset\}$ twice).

To define the predicate in this structure, we give the set of $(x, y) \in E^2$ such that $x \in y$:

$$\in^S : \quad \{(x, \{x\}) \mid x \in E\}$$

In other words, $x \in \{x\}$ and no other membership relation holds. In this structure, if $x \in y$, then $y = \{x\}$. The set $\{x\}$ is called the *successor* of x .

What could such a complicated construction possibly be used for? If we write $\emptyset = 0$, $\{\emptyset\} = 1$, and $\{\{\emptyset\}\} = 2$, then 1 is the successor of 0, 2 is the successor of 1, and so on. We have constructed a model for the set of natural numbers $\mathbb{N}$, whose only property here is that every natural number has a successor.

When constructing a set theory, we add axioms or additional notions. Here are some formulas that we require to be true when interpreted in the structure:

- Equality of two sets. Two sets are equal if and only if they have the same elements:

$$\forall x \quad \forall y \quad (x = y) \leftrightarrow (\forall z \quad (z \in x \leftrightarrow z \in y))$$

- Empty set axiom:

$$\exists x \quad \forall y \quad \neg(y \in x)$$

That is, there exists a set containing no elements.

- Inclusion is defined by:

$$x \subset y \quad \text{means by definition} \quad \forall z \quad (z \in x \rightarrow z \in y)$$

- Union axiom:

$$\forall x \quad \exists u \quad \forall z \quad (z \in u) \leftrightarrow (\exists y \quad (y \in x) \wedge (z \in y))$$

For example, if $x = \{\{1, 2\}, \{3\}, \{4, 5\}\}$, then $u = \bigcup_x = \{1, 2, 3, 4, 5\}$ (y could, for example, be $\{1, 2\}$, in which case z would be 1 or 2).

We can define another structure related to set theory. This time define $E = \bigcup_{n \geq 0} E_n$ by:

$$E_0 = \emptyset \quad \text{and} \quad E_{n+1} = \mathcal{P}(E_n)$$

where $\mathcal{P}(X)$ denotes the power set of a set X . Thus:

- $E_0 = \emptyset$
- $E_1 = \{\emptyset\}$, because the empty set is the only subset of E_0 ,
- $E_2 = \{\emptyset, \{\emptyset\}\}$, the subsets of E_1 ,
- $E_3 = \{\emptyset, \{\emptyset\}, \{\{\emptyset\}\}, \{\emptyset, \{\emptyset\}\}\}$: the empty set, two singletons, and one pair,
- ... E_n contains a number of elements that grows very rapidly.

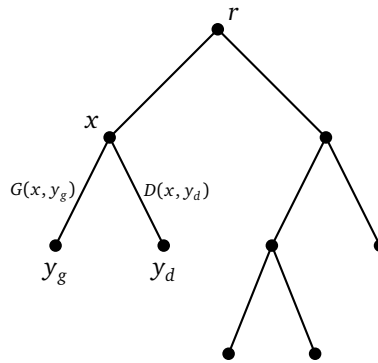
Membership here is ordinary membership; for example, $\{\emptyset\} \in \{\emptyset, \{\emptyset\}\}$, whereas $\emptyset \notin \{\{\emptyset\}\}$.

Binary trees

We want to model the notion of a binary tree (each vertex has 0 or 2 children). Let $\mathcal{L} = (r, G, D)$ be a language where r is a constant and $G(x, y)$ and $D(x, y)$ are binary predicates.

Define a structure $\mathcal{S}$ for this language:

- E is a set of vertices (possibly infinite),
- $r \in E$ is a distinguished vertex, called the *root*,
- $G \subset E^2$ is the *left-child* relation; $G(x, y)$ is true if y is the left child of x ,
- $D \subset E^2$ is the *right-child* relation; $D(x, y)$ is true if y is the right child of x .



Define a list of *axioms*, that is, formulas that must be true for every structure defining a binary tree.

- Every vertex has at most one left child and at most one right child:

$$\forall x \quad \forall y \quad \forall z \quad G(x, y) \wedge G(x, z) \rightarrow y = z$$

$$\forall x \quad \forall y \quad \forall z \quad D(x, y) \wedge D(x, z) \rightarrow y = z$$

- The left and right children are distinct:

$$\forall x \quad \forall y \quad \forall z \quad G(x, y) \wedge D(x, z) \rightarrow y \neq z$$

- A vertex has only one parent:

$$\forall x \quad \forall y \quad \forall z \quad G(x, z) \wedge G(y, z) \rightarrow x = y$$

$$\forall x \quad \forall y \quad \forall z \quad D(x, z) \wedge D(y, z) \rightarrow x = y$$

$$\forall x \quad \forall y \quad \forall z \quad G(x, z) \wedge D(y, z) \rightarrow x = y$$

- The root is nobody's child:

$$\forall x \quad \neg G(x, r)$$

$$\forall x \quad \neg D(x, r)$$

- Every vertex has either zero children or two children:

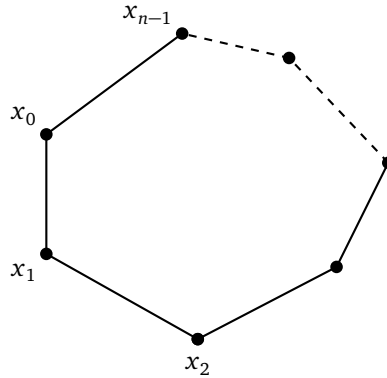
$$\forall x \quad (\exists y \quad G(x, y)) \leftrightarrow (\exists z \quad D(x, z))$$

- Optional axiom requiring the tree to be connected (in one piece): every vertex other than the root must have a parent:

$$\forall y \quad (y \neq r) \rightarrow (\exists x \quad G(x, y) \vee D(x, y))$$

(This ensures connectedness for finite trees.)

One subtle point remains to be expressed: a tree has no cycle (when following a path, one cannot return to its starting point).



Cycle in a graph

The absence of cycles cannot be expressed by a single formula, or even by finitely many formulas.

For every integer $n \geq 1$ and every sequence of relations $R_1, \dots, R_n \in \{G, D\}$, we require that the corresponding path not return to its starting point. This is expressed by the formula:

$$\forall x_0 \quad \forall x_1 \quad \dots \quad \forall x_{n-1} \quad \neg (R_1(x_0, x_1) \wedge R_2(x_1, x_2) \wedge \dots \wedge R_n(x_{n-1}, x_0))$$

Exercises

Free variables, bound variables

Exercise 1. For each of the following formulas, indicate the scope of each quantifier and list the free and bound variables among x, y, z . Is the formula closed?

1. $\forall x \quad P(x) \rightarrow Q(a, x)$
2. $\exists z \quad Q(x, y) \wedge Q(y, z) \wedge Q(z, x)$
3. $\forall x \quad (T(x, y, c) \rightarrow (\forall z \quad T(a, y, z)))$
4. $(\forall x \quad P(x)) \wedge (\exists y \quad P(y)) \wedge (x \neq y)$
5. $((\exists x \quad P(x)) \vee (\exists y \quad Q(x, y))) \rightarrow (\forall z \quad P(z))$
6. $(\exists x \quad \exists y \quad P(x) \wedge P(y) \wedge (x \neq y)) \rightarrow P(z)$
7. $\forall x \quad (Q(x, x) \rightarrow (\exists x \quad P(x)))$

Exercise 2. For each of the following formulas in a given structure, determine the free and bound variables (among x, y). If the formula is closed, state whether the sentence is true or false. If the formula is not closed, state for which values of the free variables the formula becomes true.

1. $\forall x \in \mathbb{N} \quad x^2 \geq y$
2. $\forall x \in \mathbb{N} \quad \exists y \in \mathbb{N} \quad x \geq y + 1$
3. $\forall y \in \mathbb{N} \quad \exists x \in \mathbb{N} \quad x \geq y + 1$
4. $\exists y \in \mathbb{R} \quad x \neq y$
5. $\forall y \in \mathbb{R} \quad x \neq y$
6. $(\forall x \in \mathbb{R} \quad x^2 \geq 0) \rightarrow (y = x^2 \rightarrow y \geq 0)$

Exercise 3. Rewrite the formulas after each indicated substitution.

1. $\mathcal{P}_1 : \forall x \quad (P(x) \wedge Q(x, y)) ; \mathcal{P}_1[b/y], \text{ then } \mathcal{P}_1[a/x], \text{ then } \mathcal{P}_1[x/y].$
2. $\mathcal{P}_2 : P(x) \rightarrow (\exists y \quad Q(x, y)) ; \mathcal{P}_2[x + a/x], \text{ then } \mathcal{P}_2[y + b/y].$
3. $\mathcal{P}_3 : \exists z \quad (x \neq y \rightarrow T(x, y, z)) ; \mathcal{P}_3[a/x, b/y, c/z].$
4. $\mathcal{P}_4 : (\forall x \quad P(x)) \wedge (\exists y \quad Q(x, y)) \wedge (\forall z \quad \exists x \quad T(x, y, z)) ; \mathcal{P}_4[y/x, x/y, c/z].$

Structures

Exercise 4. Consider the language $\mathcal{L} = (a, f)$ with a constant a and a binary function $f(x, y)$. For each formula and each structure, give the interpretation of the formula and state whether this interpreted formula is true or false.

Formulas:

- $\mathcal{P}_1 : \forall x \quad \exists y \quad f(x, y) = a$
- $\mathcal{P}_2 : \forall x \quad \exists y \quad f(x, x) = f(f(y, y), y)$

- $\mathcal{P}_3 : \forall x \quad \forall y \quad f(x, y) = a \rightarrow x = a$

Structures:

1. $\mathcal{S}_i : E = \mathbb{Z}, a = 0, f(x, y) = x + y$
2. $\mathcal{S}_{ii} : E = \mathbb{R}, a = 0, f(x, y) = x - y$
3. $\mathcal{S}_{iii} : E =]0, +\infty[, a = 1, f(x, y) = xy$

Exercise 5. Consider the language $\mathcal{L} = (a, b, P, Q)$, where a, b are constants and $P(x)$ and $Q(x, y)$ are predicates (the variables are x, y). For each formula and each structure, give the interpretation of the formula and state whether this interpreted formula is true or false.

Formulas:

- $\mathcal{P}_1 : \forall x \quad Q(x, a)$
- $\mathcal{P}_2 : \exists x \quad \forall y \quad P(x) \wedge Q(x, y)$
- $\mathcal{P}_3 : (\forall x \quad P(x)) \rightarrow (\exists y \quad Q(a, y))$

Structures:

1. $\mathcal{S}_i : E = \mathbb{N}, a = 0, b = 1, P(x): "x \neq 1", Q(x, y): "x > y"$
2. $\mathcal{S}_{ii} : E = M_2$, the set of 2×2 matrices with real entries; a is the zero matrix, b the identity matrix, $P(x): "\det(x) \neq 0"$ (the determinant is nonzero), $Q(x, y): "xy = I"$ (the product is the identity)

Exercise 6. Let $\mathcal{L}$ be a language with a predicate $R(x, y)$. Consider $E = \{r, s, t, u\}$. For each of the following formulas, define an interpretation of the predicate $R(x, y)$ in the form of an undirected graph so that the formula is true in this structure. (There may be several possible solutions.)

1. $\forall x \quad \forall y \quad (x \neq y) \rightarrow R(x, y)$
2. $(\forall x \quad R(x, x)) \wedge (\forall x \quad \exists y \quad \neg R(x, y))$
3. $\forall x \quad \forall y \quad \forall z \quad (R(x, y) \wedge R(x, z)) \rightarrow (y = z)$
4. $\neg (\forall x \quad \forall y \quad \forall z \quad (R(x, z) \wedge R(z, y)) \rightarrow R(x, y))$