Basic logic and model theory

Uri Abraham *

March 26, 2014

Abstract

We describe here propositional languages and their structures, and first-order languages and their Tarskian structures.

1 Propositional languages and structures

In order to describe the state of some complex system, we may make a list of elementary statements which may be true or false, and then the state can be described by saying which of these statements are true and which are false. For example, an elementary statement may be "valve 12A is open", "gate 3 is closed", "the temperature at zone V is greater than 125 degrees" etc. These elementary statements are also called "atomic propositions", "propositions" "atomic sentences". A proposition is atomic in the sense that it is not expressed by means of simpler propositions. If A is the set of atomic propositions then a *state of the system* is a function $\sigma : A \to \{T, F\}$ which assigns to each proposition $X \in A$ its truth value $\sigma(X)$ which can either be "true" ($\sigma(X) = T$) or "false" ($\sigma(X) = F$). We can also say that σ is a "model" of our system at some specific moment: It describes the state in which those statements for which $\sigma(X) = T$ hold and the other statements (for which $\sigma(X) = F$) do not hold.

The logical connectives permit us to form compound statements ("sentences") out of the atomic propositions. We have the following connectives $\wedge$, $\vee$, $\to$, $\neg$. If α and β are any sentences then:

1. $(\alpha \wedge \beta)$ is read "α and β"; this sentence is true if and only if both α and β are true. So $\wedge$ is the logical conjunction connective.

*Models for Concurrency 2014, Ben-Gurion University.

2. $(\alpha \vee \beta)$ is read "α or β", and it holds iff α is true or β is true. (As usual in mathematics, we understand α or β to be true in case α is true, β is true, or both α and β are true. That is, $\vee$ is the "inclusive or" connective. We shall always use the 'or' connective in its inclusive sense.)

3. $(\alpha \to \beta)$ is read "α implies β". This sentence is true if α is false or β is true.

4. $\neg\alpha$ is the negation of α. $\neg\alpha$ is true iff α is false.

We shall not give here a formal definition of the set of sentences that are built over of the set A of atomic sentences; we assume that the reader has reviewed this material. An "assignment" is a function $\sigma : A \to \{\mathbf{true}, \mathbf{false}\}$. As we said above, an assignment is a "model", a description of a specific situation. The Val function gives to every sentence φ (built over the set A of atomic sentences) and assignment σ the truth value $Val_\sigma(\varphi)$ of φ in the model σ. The inductive definition of Val is as follows. (We define by induction on n the value of $Val_\sigma(\varphi) \in \{T, F\}$ when φ has n connective occurrences (and σ is arbitrary).

1. $Val_\sigma((\alpha \wedge \beta)) = T$ iff $Val_\sigma(\alpha) = T$ and $Val_\sigma(\beta) = T$.

2. $Val_\sigma(\alpha \vee \beta) = T$ iff $Val_\sigma(\alpha) = T$ or $Val_\sigma(\beta) = T$.

3. $Val_\sigma(\alpha \to \beta) = F$ iff $Val_\sigma(\alpha) = T$ and $Val_\sigma(\beta) = F$.

4. $Val_\sigma(\neg\alpha) = T$ iff $Val_\sigma(\alpha) = F$.

2 Quantification Languages and Structures

In this lecture we define the notions of quantification languages and their models (structures)—two interrelated concepts that play a central role in logic and in this course. To define a language its signature is needed first. A signature for a language is the list of symbols of the language and some information about their syntax. That is, the signature contains the list of predicates, function symbols and constants, and for every predicate and function symbol an natural number is assigned–its arity–which tells us the number of parameters that should be supplied.

We first define the notion of single-sort signatures and languages, and then we define multi-sorted signatures and languages which are the main type of languages that we shall use.

Definition 2.1 (Signature) *We say that L is a* signature *if L is a four-tuple $\langle \overline{P}, \overline{F}, \overline{c}, arity \rangle$ where:*

1. *$\overline{P}$ is a set $\{P_0, \ldots, P_{k-1}\}$ of symbols called predicates. For each predicate P_i, $arity(P_i)$ is a non-zero natural number called "the arity of P_i". The equality predicate symbol $\approx$ is assumed to be in $\overline{P}$ (and its arity is 2).*

2. *$\overline{F} = \{F_0, \ldots, F_{\ell-1}\}$ is a set of "function symbols", and, again, a non-zero natural number $arity(F_i)$ is associated with each function symbol.*

3. *$\overline{c} = \{c_0, \ldots\}$ is set of symbols called "constants".*

We assume that these sets of symbols are disjoint. Usually, in this course, each of these sets of symbols is finite, but signatures with an infinite number of constants can be quite useful. For example, to describe the natural numbers, a signature L may be formed by taking a single predicate $<^*$ with arity 2, two function symbols, $+^*$ and $\times^*$, with arity 2, and a single constant, the symbol 0^* (but it would also be natural to add all constants n^* to represent the natural numbers $n \in \mathbb{N}$). L is thus a set of symbols with associated arities, and no meaning is associated with these symbols. The asterisk, for example in $+^*$, is to emphasize that this is just a symbol rather than the familiar addition operation $+$. The standard interpretation for this signature is obtained by taking as universe of discourse the set of natural numbers $\mathbb{N}$, the familiar ordering relation as an interpretation of $<^*$, the addition and multiplication operations as interpretations of $+^*$ and $\times^*$, and the number zero to interpret 0^*.

Another natural interpretation for L is obtained by taking the real numbers with the standard ordering relation, and with the addition and multiplication operations. But arbitrary interpretations are also possible: indeed any non-empty set with a binary relation, two functions, and a constant can interpret L. Of course the choice of a symbol may indicate the intentions of the users, but these intentions cannot replace a definition.

We are now going to define interpretations of an arbitrary signature.

Definition 2.2 (Interpretation) *Let $L = \langle \overline{P}, \overline{F}, \overline{c}, arity \rangle$ be a signature. $\mathcal{M}$ is called an interpretation (or a structure) for L if $\mathcal{M} = \langle A, \overline{P}^{\mathcal{M}}, \overline{F}^{\mathcal{M}}, \overline{c}^{\mathcal{M}} \rangle$ consists of the following.*

1. *A is a non-empty set denoted $|\mathcal{M}|$ and called the* universe *of $\mathcal{M}$. Members of A are called "individuals" of $\mathcal{M}$.*

2. $\overline{P}^{\mathcal{M}} = \{P^{\mathcal{M}} \mid P \in \overline{P}\}$ *associates with each predicate* P *in* L *of arity* $m = arity(P)$ *an* m*-ary relation* $P^{\mathcal{M}}$ *on* $|\mathcal{M}|$*. That is* $P^{\mathcal{M}} \subseteq A^m$*. (For any set* A*,* A^m *denotes the collection of all* m*-tuples from* A*.) In particular, every unary predicate symbol is associated with a subset of the universe.*

3. $\overline{F} = \{F^{\mathcal{M}} \mid F \in \overline{F}\}$ *is an interpretation of all function symbols. For each function symbol* F*, of arity* m*,* $F^{\mathcal{M}}$ *is an* m*-place function*

$$F^{\mathcal{M}} \colon |\mathcal{M}|^m \to |\mathcal{M}|.$$

(The notation $f : X \to Y$ *means that* f *is a function from* X *to* Y*. Thus* $F^{\mathcal{M}}$ *is defined on* $|\mathcal{M}|^m$*, the set of* m*-tuples of individuals, and takes values in* $|\mathcal{M}|$*.) We identify* $|\mathcal{M}|^1$ *with* $|\mathcal{M}|$*. So when* F *is a function symbol of arity 1, then* $F^{\mathcal{M}} : |\mathcal{M}| \to |\mathcal{M}|$*.*

4. *Finally,* $\overline{c}^{\mathcal{M}} = \{c^{\mathcal{M}} \mid c \in \overline{c}\}$ *interprets the constants of* L*: For every constant* c *in* L*,* $c^{\mathcal{M}}$ *is an individual of* $\mathcal{M}$*. That is,* $c^{\mathcal{M}} \in |\mathcal{M}|$*.*

First-order languages

Let $L = \langle \overline{P}, \overline{F}, \overline{c}, arity \rangle$ be a signature. The language of L is the set of all formulas that can be obtained with the symbols of L, and in the following we give a detailed definition. We assume a fixed set *Var* of variables (an infinite set). For example, we may take $Var = \{v_1, v_2, \ldots\}$ to be our variables. We also assume a fixed set of logical connectives: $\wedge$ for conjunction, $\vee$ for disjunction, $\neg$ for negation, and $\rightarrow$ for implication. (It is possible, for convenience to extend this list and to add for example $\leftrightarrow$.) We also have two quantifiers: the "forall" quantifier $\forall$, and the "for some" quantifier $\exists$. We also need left and right parenthesis symbols, (and). Given a signature L, we shall define the following notions: the set of *individual terms* (usually called "terms", but since this name is overloaded in computer-science we need a more specific terminology), the set of *atomic formulas*, and the set of *formulas*. These are all syntactic objects, namely finite sequences of symbols.

The set of individual terms is defined recursively.

1. Every variable $x \in Var$ and every constant c in $\overline{c}$ is an individual term.

2. If F is a function symbol in $\overline{F}$ of arity k, and $\tau_1, \ldots, \tau_k$ are individual terms, then then sequence of symbols $F(\tau_1, \ldots, \tau_k)$ is an individual term. (For the very pedantic: the commas and dots are not part of the term; they are used to express our intension that the sequence of terms includes all the terms in increasing order.)

Now the set of atomic formulas is defined as follows. If P in $\overline{P}$ is a predicate of arity k and $\tau_1, \ldots, \tau_k$ are individual terms, then $P(\tau_1, \ldots, \tau_k)$ is an atomic formula.

The set of formulas is defined recursively as follows.

1. Any atomic formula is a formula.
2. If α and β are formulas then also $(\alpha \wedge \beta)$, $(\alpha \vee \beta)$, $(\alpha \rightarrow \beta)$, and $\neg\alpha$ are formulas.
3. If α is a formula and x is any variable symbol then $\forall\ x\ \alpha$ and $\exists\ x\ \alpha$ are formulas.

As an example take a signature with a unary predicate P, a binary function symbol F, and a constant e. If x, y, z are variables, then for example $F(F(x, F(y, e)), x)$ is an individual term. An example for an atomic formula is $P(F(x, y))$, and an example of a formula is $\forall x \exists y P(F(y, x))$.

Free and bounded occurrences of variables

Take, for example, a formula written in a familiar notation: $x + 4 < 12$. We cannot tell if it is true or false because the value of x in this formula is not determined. It is sometimes said that x is a "place holder", and only when the value of x is determined, as for example in $x + 4 < 12[x = 8]$, that we know that the formula is false because it is not the case that $8 + 4 < 12$. (We shall later speak about "assignments" and then $[x = 8]$ is read "when x is assigned the value 8.) An occurrence of a variable in a formula as a place holder is said to be a "free" occurrence, and an occurrence that falls within the domain of a quantifier is said to be bounded. For example, in $\exists x\ (y < x \wedge x < 3)$ the occurrence of x is bounded and the occurrence of y is free. Another example: $\forall x(P(x, y) \vee \exists R(x))$. Here the x in $P(x, y)$ is bounded by the universal quantifier, and the x in $R(x)$ is bounded by the existential quantifier. The y variable occurs free. The x in $\forall x$ is neither free nor bounded, it is best to view it as forming a compound symbol together with the $\forall$ symbol. We now give a more formal definition of free and bounded occurrences.

A formula is a sequence of symbols, and any symbol may have one or more occurrences in a formula. We shall define now which occurrences of a variable in a formula are free and which are bound, and for any bounded occurrence we will indicate which quantifier occurrence bounds it.

1. Any occurrence of a variable in an atomic formula is a free occurrence.

2. Let α and β be formulas. An occurrence of a variable in α that is free in α remains free in $(\alpha \wedge \beta)$, and an occurrence that is bound in α remains bound in the compound formula and moreover it is the same quantifier occurrence that bounds it. Likewise a free (bound) occurrence of a variable in β retains its status in the compound formula. Just like $\wedge$, the other logical connectives will not change the statuses of variable occurrences.

3. The status of variable x in α does not change in $\forall y\alpha$ when variable y is different from x. Any free occurrence of variable x in α becomes bounded in $\forall x\alpha$ and the leftmost quantifier $\forall x$ is the bounding quantifier of that occurrence. A bounded occurrence of x in α remains bounded by the same quantifier that bounds it in α.

2.0.1 The satisfaction relation

Let M be an interpreting structure of a given signature. An assignment is a function from the set of variables into the universe of M. That is, σ is an assignment if $\sigma : Var \to |M|$. We assume two boolean values $T = \mathbf{true}$ and $F = \mathbf{false}$ and we shall define the function $Val_M(\varphi, \sigma) \in \{T, F\}$ which tells us the truth value of a formula φ in M under the assignment σ. Before we do this, however, we define for any individual term τ its value under the assignment σ, $Val_M(\tau, \sigma) \in |M|$, as follows by induction on (the structure of) τ.

1. For every variable x, $Val_M(x, \sigma) = \sigma(x)$. For every constant c $Val_M(c, \sigma) = c^M$.

2. If τ has the form $F(\tau_1, \ldots, \tau_k)$ (where F is a function symbol of arity k, and $\tau_1, \ldots, \tau_k$ are individual terms) and if $a_i = Val_M(\tau_i, \sigma)$ for every i, then $Val_M(\tau, \sigma) = F^M(a_1, \ldots, a_k)$.

Now we define a boolean valued function, denoted again Val; $Val_M(\varphi, \sigma)$ is the truth value of formula φ in the assignment σ. First, for any atomic formula $P(\tau_1, \ldots, \tau_k)$ we define $Val_M(P(\tau_1, \ldots, \tau_k), \sigma) = \mathbf{true}$ if and only if $\langle a_1, \ldots, a_k \rangle \in P^M$ where $a_i = Val(\tau_i, \sigma)$ for every i.

Then we continue and if φ is any formula (in the given signature) we define $Val_M(\varphi, \sigma)$ (for every assignment σ) by induction on the structure of φ (or by induction on the number of quantifier and logical connectives that occur in φ) as follows.

1. If φ is an atomic formula then $Val_M(\varphi, \sigma)$ was defined above.

2. Suppose α and β are formulas and both of $Val_M(\alpha, \sigma)$ and $Val_M(\beta, \sigma)$ were defined. Then $Val_M((\alpha \to \beta), \sigma) = \mathbf{false}$ if and only if $Val_M(\alpha, \sigma) = \mathbf{true}$ and $Val_M(\beta, \sigma) = \mathbf{false}$. The truth value definitions for the other logical connectives are well known.

3. Suppose a formula φ for which we have defined $Val_M(\varphi, \sigma')$ for every assignment σ'. Then we define $Val_M(\exists x\varphi, \sigma) = \mathbf{true}$ iff there exists an assignment σ' so that

   (a) For every variable y that is different from x $\sigma'(y) = \sigma(y)$, and

   (b) $Val_M(\varphi, \sigma') = \mathbf{true}$.

No one is happy with item 3 in this definition. The reader should play with some examples until she feels that this definition indeed reflects her understanding of what it means for $\exists x\ \varphi$ to be true under assignment σ.

There is alternative definition of the satisfaction relation which is somewhat unusual but which may lead to a better understanding of its meaning. If L is a signature then a larger signature can be obtained by adding new constants to L. Suppose now that whenever an interpretation M of L is considered we automatically extend L be adding constants $\overline{a}$ for every $a \in |M|$. Then we are in the situation where every individual member of the universe of M has a name in the language. In this case, the definition of the satisfaction relation $Val_M(\varphi)$ is easily defined for sentences φ (a sentence is a formula with no free variables) by induction on the length of φ, and there is no need for assignments. Item 3 takes the following particularly simple form: $Val_M(\exists x\varphi) = \mathbf{true}$ iff there exists some $a \in |M|$ such that $Val_M(\varphi') = \mathbf{true}$ where φ' is obtained from φ by subsituting $\overline{a}$ for every free occurence of x in φ.

We often write $M \models \varphi[\sigma]$ for $Val_M(\varphi, \sigma) = \mathbf{true}$, and we say then that M *satisfies* φ under the assignment σ. Likewise, we write $M \not\models \varphi[\sigma]$ for $Val_M(\varphi, \sigma) = \mathbf{false}$

An assignment is by definition defined over all variables. This was done in order to simplify the definition of satisfaction. But in practice the assessment of $Val_M(\varphi, \sigma)$ depends only on the values of σ on the free variables of φ.

Sometimes we write the assignment function σ explicitly: for example if $\sigma(x_i) = a_i$ then we can write

$$\mathcal{S} \models \phi[x_1, \ldots, x_n \ / \ a_1, \ldots, a_n].$$

We may even substitute individuals for variables and write an impure formula such as $3 < 4$ instead of $x < y[x, y \ / \ 3, 4]$, when clarity is preferred to rigor. (Of course, if 3 and 4 are constants in L then there is nothing wrong with $3 < 4$.)

Exercise 2.3 *Let M be a structure and let σ and σ' be two assignments that agree on all variables that occur in the term τ. (That is $\sigma(x) = \sigma'(x)$ for every variable x that occurs in τ.) Then $Val_M(\tau, \sigma) = Val_M(\tau, \sigma')$.*

Exercise 2.4 *Let M be a structure and φ a formula. Let σ and σ' be two assignments that agree on all variables that have a free occurrence in φ. Then $Val_M(\varphi, \sigma) = Val_M(\varphi, \sigma)$.*

2.0.2 Isomorphisms of structures

When understood, the notion of isomorphism is very simple, and it is very important. So it pays well to insist and go through what may appear at first to be a technical definition.

Let $L = \langle \overline{P}, \overline{F}, \overline{c}, \mathit{arity} \rangle$ be a fixed signature, and suppose that $\mathcal{M}_1 = \langle A_1, \overline{P}^{\mathcal{M}_1}, \overline{F}^{\mathcal{M}_1}, \overline{c}^{\mathcal{M}_1} \rangle$ and $\mathcal{M}_2 = \langle A_2, \overline{P}^{\mathcal{M}_2}, \overline{F}^{\mathcal{M}_2}, \overline{c}^{\mathcal{M}_2} \rangle$ are two interpreting structures. Then $\mathcal{M}_1$ and $\mathcal{M}_2$ are said to be isomorphic if there exists a bijection $g : A_1 \to A_2$ (i.e. a one-to-one function that is onto A_2) so that the following hold (we then say that g is an "isomorphism").

1. For every predicate $P \in \overline{P}$ of arity k, for every k-tuple of members of A_1, $(a_1, \ldots, a_k)$,

$$(a_1, \ldots, a_k) \in P^{\mathcal{M}_1} \text{ iff } (g(a_1), \ldots, g(a_k)) \in P^{\mathcal{M}_2}.$$

2. For every function symbol $F \in \overline{F}$ of arity k, and for every k-tuple $(a_1, \ldots, a_k)$ of members of A_1,

$$g(F^{\mathcal{M}_1}(a_1, \ldots, a_k) = F^{\mathcal{M}_2}(g(a_1), \ldots, g(a_k)).$$

   (A picture can clarify this situation.)

Intuitively, two isomorphic structures are the same: they differ only by the color of their universe (so to speak) but not by their organization. But since the way that the members of the universe are organized is the essence of the interpreting structure, any two isomorphic structures are essentially the same. As we have said, this is an important notion for us when we seek to find models that describe activity of concurrently operating processes. Usually, the modeling structures and more specifically the names that we choose for their elements (members of their universes), names of their predicates and functions are reflections of the external 'real' objects that we want to model. These external object may have more refined properties and each such object may be itself a rather complicated structure. When we analyze the system informally, knowledge about the inner structure of these objects may help us to argue and derive conclusions that cannot be obtained in the abstract setting, and this may lead to erroneous and unwanted results. Thinking about isomorphic structures, however, make it very clear that all our reasoning should rely only on the structures themselves, on abstract properties of their predicates and functions, properties that do not change when an isomorphic structure is considered. Thus, if we ensure that any conclusion that we reach must hold in all isomorphic structures, we help to remove those argument that rely on some non-stated properties of the members of the structure.

Example of isomorphic structures. Take a very simple signature consisting of just one binary predicate R and one constant symbol 0. A possible interpreting structure for this signature is $\mathcal{M}_1 = (A_1, R^{\mathcal{M}_1}, 0^{\mathcal{M}_1})$ where $A_1 = \mathcal{P}(\omega)$,$R^{\mathcal{M}_1} = \{(X,Y) \mid X \subseteq Y \subseteq \omega\}$, and $0^{\mathcal{M}_1} = \emptyset$. Here we use the following notations: $\mathcal{P}(U) = \{T \mid T \subseteq U\}$ is the power-set of U, namely the set of all subsets of U. $\omega = \{0, 1, \ldots\}$ is the set of all natural numbers; so the universe of $\mathcal{M}_1$, $\mathcal{P}(\omega)$, is the collection of all sets of natural numbers, and $R^{\mathcal{M}_1}$, the interpretation of predicate R, is the inclusion relation. The constant symbol 0 is interpreted as the empty set in our structure. We brought this trivial example here in order to draw the reader's attention to the following point. The members of the universe of $\mathcal{M}_1$ should not be thought of as subsets of ω but rather as abstract points devoid of any inner structure. That is, as a member of the universe of $\mathcal{M}_1$, the set of even numbers for example is just a point and it is meaningless to ask what are its members. The notion of isomorphism can help to understand this point. We can find an isomorphic copy of $\mathcal{M}_1$ whose universe is quite different, but since isomorphic structures are considered to be "the same" it makes no sense to insist on the members of our abstract structure to be subsets of ω. Let $\mathcal{M}_2$ be defined as follow. The universe of $\mathcal{M}_2$ is the set $\{0,1\}^\omega$ of all functions from ω to $\{0,1\}$, the R relation is interpreted as $R^{\mathcal{M}_2} = \{(u,v) \mid u, v \in \{0,1\}^\omega \text{ and } \forall n \in \omega(u(n) = 1 \to v(n) = 1)\}$.

Exercise 2.5 *Prove that $\mathcal{M}_1$ and $\mathcal{M}_2$ are isomorphic.*

2.0.3 Logical implication

When proving properties of a program, we will follow three steps: (1) express these properties as first-order sentences, (2) define a family S of structures that describe the program's executions, (3) prove that the sentences hold in every structure from S. These correctness proofs have the form $\Phi \implies \psi$ where Φ is a set of sentences defining the structures and ψ is a single sentence expressing the correctness conditions. We therefore have to define exactly what $\Phi \implies \psi$ means, and how one proves such implications.

Fix a language (and signature) L. Suppose that Φ and Ψ are two sets of sentences in L. Then $\Phi \implies \Psi$ means that for every structure $\mathcal{M}$ of L, if $\mathcal{M}$ is a model for Φ then it is also a model for Ψ[1]. In case $\Psi = \{\psi\}$ is a singleton we write $\Phi \implies \psi$ rather than $\Phi \implies \{\psi\}$. The relation $\Phi \implies \Psi$ is also meaningful when Φ and Ψ are sets of formulas (which may contain free variables). This relation means then that for every structure $\mathcal{M}$ of L and every assignment σ in $\mathcal{M}$, if $\mathcal{M}$ (under σ) satisfies every formula in Φ then it also satisfies every formula in Ψ. Again, for singletons, we write $\phi \implies \psi$ for simplicity.

2.1 Substructures and reducts

Let L be a first-order language. Given two structures $\mathcal{S}_1$ and $\mathcal{S}_2$ for L, we say that $\mathcal{S}_1$ is a substructure of $\mathcal{S}_2$ iff:

[1] $\mathcal{M}$ is a model for Φ when it satisfies each of the sentences of Φ

1. $|\mathcal{S}_1| \subseteq |\mathcal{S}_2|$ (the universe of $\mathcal{S}_1$ is a subset of the universe of $\mathcal{S}_2$.
2. $\mathcal{S}_1$ is obtained by restricting the interpretation of $\mathcal{S}_2$ to the universe of $\mathcal{S}_1$. This means the following:
   (a) For every predicate P of arity n in L and for every n-tuple of individuals from $|\mathcal{S}_1|$,
   $$\langle a_1, \ldots, a_n \rangle \in P^{\mathcal{S}_1} \text{ iff } \langle a_1, \ldots, a_n \rangle \in P^{\mathcal{S}_2}.$$
   (b) For every function symbol F, $F^{\mathcal{S}_1}$ is the restriction of $F^{\mathcal{S}_2}$ to $|\mathcal{S}_1|$. (The restriction of a function $f : A \to B$ to a subset of its domain $A' \subseteq A$ is the function $f' : A' \to B$ defined by $f'(x) = f(x)$ for $x \in A'$. We write $f' = f \restriction A'$ for the restriction.)
   (c) If c is a constant in L then $c^{\mathcal{S}_1} = c^{\mathcal{S}_2}$.

We shall see that the notion of a *reduct* is very useful. A signature L_1 is said to be a sub-signature of L_2 if L_1 is a signature that consists only of sorts, predicates, function symbols and constants from L_2, but it does not change their arity and sort. (Since L_1 is a signature, if F is a function symbol in L_1, for example, then both the domain and range of F are sorts in L_1).

Definition 2.6 (Reduct) *If L_1 is a sub-signature of L_2 and $\mathcal{S}_2$ is a structure for L_2, then the reduct of $\mathcal{S}_2$ to L_1 is the following structure $\mathcal{S}_1$ obtained by "forgetting" the symbols not in L_1:*

- *The universe of $\mathcal{S}_1$ is obtained by taking the sorts of $\mathcal{S}_2$ that are in L_1. (The universe of every structure is the union of its sorts, and so $|\mathcal{S}_1|$ is the subset of $|\mathcal{S}_2|$ obtained as $\bigcup\{S^{\mathcal{S}_2} : S \text{ is a sort in } L_1\}$).*
- *The interpretation of each predicate, function symbol, and constant of L_1 is the same in $\mathcal{S}_1$ and $\mathcal{S}_2$.*

It is convenient to use the notation

$$S_1 = S_2 \restriction L_1$$

for reducts (little danger of confusion with function restriction). We also say in this case that S_2 is an expansion of S_1 (to L_2).

2.1.1 Examples and problems in modelling with first-order structures

Although the first steps in logic were related to philosophical investigations (Aristotles, fourth century BC, comes immediately to mind) mathematical logic was developed with the aim of elucidating the methods of mathematics. It is therefore not a surprise that examples for first-order structures abound in mathematics. We will

Suppose that we want to model vector spaces, that is to represent vector spaces as structures that interpret a certain language signature. What should be that

signature? We have two sorts of objects: scalars (members of some field such as the field of real numbers $\Re$) and vectors (for example the vectors in the three-dimensional space $\Re^3$). So it is reasonable to have two unary predicates F and V to qualify these objects in the interpreting structures. Then we have the field operations represented by the function symbols $+_F$ and $\times_F$ (for addition and multiplication) and the 0_F, 1_F constants. We also have the vector addition function $+_V$, and the scalar multiplication function symbol $\cdot$. Now suppose that we want to take the usual three dimensional vector space as an interpreting structure $\mathcal{M}$ for this signature. Then we define the universe of $\mathcal{M}$ as the union $\Re \cup \Re^3$, we define $F^{\mathcal{M}} = \Re$, and $V^{\mathcal{M}} = \Re^3$. Then we continue and define the function symbols in the standard way. $+_F^{\mathcal{M}}$ for example is the usual addition function on the set of real numbers, and $\cdot^{\mathcal{M}}$ is the usual scalar function that takes the pair α, (a, b, c) to $\alpha \cdot (a, b, c) = (\alpha a, \alpha b, \alpha c)$ etc. But here we have a problem and it is in order to describe this problem that I brought this example here.

Function symbols must be interpreted in any structure as functions that are defined over all members of the universe of that structure. But here, for example, the universe is the union $\Re \cup \Re^3$, and the $+_F^{\mathcal{M}}$ function is only defined over $\Re$. We cannot add a vector to a scalar. This problem occurs in many situations and we must face it if we want to model systems of concurrently operation systems.

Exercise 2.7 *Devise at least two possible solutions to this problem of partially defined functions, and show how your solutions handle the example of vector spaces.*