6^{th} International Conference on Applied Informatics
Eger, Hungary, January 27–31, 2004.

Languages of Logic and Applications *

Katalin Pásztor Varga[1], Magda Várterész[2]

[1] Department of Programming Languages and Compilers
Eötvös Loránd University
e-mail: pkata@ludens.elte.hu

[2] Institute of Informatics
University of Debrecen
e-mail: vateres@inf.unideb.hu

Abstract

Concerning the logical description languages, in the past 40-50 years many authors have introduced a number of structurally very different first-order languages. Meanwhile some of these first-order languages follow the structure of a given future model, other ones have been prepared for the description of an arbitrary model. Another ones do not follow the whole structure of any model: they have been prepared only for the relations definable over the universe in order to be able to prove the generalizations of a number of difficult logical results.

We have investigated the different approaches and concluded that they do not mean essential differences. In fact, they have been only motivated by seeking for an easier way to achieve the target. The aim of the semantic of the first order languages is the interpretation of the language. The interpretation of the symbols of a language is based on some model. Differences in the semantics here come from the considerations wether we focus on unique models or concentrate to all models over a given universe. At the same place the naming problem of the universe element of the model arises. The efforts for solving this problem lead to different approaches as well.

Here we present the most important language definitions and some characteristic semantics. Moreover, we try to point out the suitability connections of languages and semantics definitions.

Categories and Subject Descriptors: F.4.1 [Mathematical Logic]: Model theory; F.4.2 [Grammars and Other Rewriting Systems]: Decision problems; I.2.4 [Knowledge Representation Formalisms and Methods]: Predicate logic

Key Words and Phrases: first-order languages, syntax, semantics

1 Syntax

Roughly speaking, development and publications of different versions of the first-order logic languages fell on some periods. Logical parts of logic languages are common (con-

*Supported by the Hungarian National Science Research Grant (OTKA), Grant Nr. T03 77 42.

nectives, quantifiers and a countable set of (individuum) variables). The system of non-logical symbols of first-order languages took shape from the signs of (mathematical and logical) functions in the first general works after 1930s (L. Kalmár, S. C. Kleene, E. Mendelson). The languages had got countable sets of predicate and function symbols for every arity, because in this case the interpretations of languages remained without restrictions. The alphabet of a first-order language holds:

- connectives and quantifiers: $\neg, \wedge, \vee, \supset, \forall, \exists$,
- variables: $x_1, x_2, x_3, \ldots$
- non-logical symbols:
  - predicate symbols: $P_1^1, P_1^2, P_1^3, \ldots, P_k^1, P_k^2, P_k^3, \ldots$ (P_k^n is an n-place predicate symbol),
  - function symbols: $f_1^1, f_1^2, f_1^3, \ldots, f_k^1, f_k^2, f_k^3, \ldots$ (f_k^n is an n-place function symbol),
  - constant symbols: $c_1, c_2, \ldots$,
- punctuation: ')', '(', and ','.

With this system of symbols the language was suitable for description of problems not only in mathematics.

A formal system is a tuple $\langle \mathcal{U}, \mathcal{R}, \mathcal{M}, \mathcal{C} \rangle$ where

- $\mathcal{U}$ is a nonempty set,
- $\mathcal{R}$ is a finite set of relations on $\mathcal{U}$,
- $\mathcal{M}$ is a finite set of operations on $\mathcal{U}$,
- $\mathcal{C}$ is a finite (possibly empty) set of selected elements of $\mathcal{U}$.

Then the formal systems were characterized with signatures. A signature is a mapping that associates some natural number called arity to every relation and operation and its cardinality to set $\mathcal{C}$. Thus the formal system is a quintuple $\langle \mathcal{U}, \mathcal{R}, \mathcal{M}, \mathcal{C}, \mu \rangle$.

The languages of the formal systems appear with signatures in the form $\langle \mathcal{R}^*, \mathcal{M}^*, \mathcal{C}^*, \mu \rangle$ where the elements of $\mathcal{R}^*, \mathcal{M}^*, \mathcal{C}^*$ are names of the elements of $\mathcal{R}, \mathcal{M}, \mathcal{C}$ and μ is a signature.

The arithmetic as a formal system is the quintuple $\langle \mathbf{N_0}, \mathcal{R}, \mathcal{M}, \mathcal{C}, \mu \rangle$. The description language is the tuple $\langle \{\leq\}, \{\mathbf{s}, +, \times\}, \{\mathbf{0}\}, \mu \rangle$ where

- $\leq$ is the name of an only relation in $\mathcal{R}$,
- $\mathbf{s}, +, \times$ are the signs of operations in $\mathcal{M}$,
- 0 identifies the smallest element of universe,
- and the signature is the following:

R	$\mu(R)$	m	$\mu(m)$
$\leq$	2	$\mathbf{s}$	1
		$+$	2
		$\times$	2

Later the effect of mentioned above appears in definition of first-order languages. The non-logical part of these languages is given by a tuple $\langle \mathcal{P}, \mathcal{F}, \mathcal{C}, \mu \rangle$:

- $\mathcal{P}$ is a set of predicate symbols,
- $\mathcal{F}$ is a set of function symbols,
- $\mathcal{C}$ is a set of constant symbols and
- μ is a signature.

The sets of predicate and function symbols are finite in M. Davis', A. Dragalin's, K. Pásztor Varga's works: the languages contains

- a finite nonempty set $\{P_1, P_2, \ldots, P_k\}$ of predicate symbols,
- a finite set $f_1, f_2, \ldots, f_l$ of function symbols,
- a finite or countable set $c_1, c_2, \ldots$ of constant symbols and
- a signature that consists of mappings $\mu_{\mathcal{P}}, \mu_{\mathcal{F}}$ and $\mu_{\mathcal{C}}$ where
  - $\mu_{\mathcal{P}}(n)$ gives the arity of the n^{th} predicate symbol,
  - $\mu_{\mathcal{F}}(n)$ gives the arity of the n^{th} function symbol, sometimes designated as

$$\begin{pmatrix} P_1 & P_2 & \ldots & P_k & ; & f_1 & f_2 & \ldots & f_l \\ n_1 & n_2 & \ldots & n_k & ; & m_1 & m_2 & \ldots & m_k \end{pmatrix}$$

    and
  - $\mu_{\mathcal{C}}$ is the cardinality of the set of constant symbols.

There are some authors C. Bell, M. Machover, A. Nerode, R. A. Shore, R. Socher-Ambrosius, P. Johann, U. Schöning, J.-Y. Girard) who work with infinite sets of predicate and function symbols. Yu. L. Ershov allows finite and infinite sets as well. In R. M. Smullyan work the definition of first-order language is the next: the alphabet is a pair $\langle \mathcal{P}, Par \rangle$ where

- $\mathcal{P}$ is a countable list of n-ary predicate symbols for every natural number,
- Par is a countable list of symbols called parameters.

Here the parameter symbol is a new notion.

The grammar can be given for all logic languages in a common way:

(a) Any variable, any constant symbols and any parameter is a term.

(b) If f is an n-ary function symbol and $t_1, t_2, \ldots, t_n$ are terms, then $f(t_1, t_2, \ldots, t_n)$ is a term too.

(c) A string is a term only if it can be made by the conditions (a-b).

(d) If P is an n-ary predicate symbol and $t_1, t_2, \ldots, t_n$ are terms, then $P(t_1, t_2, \ldots, t_n)$ is an (atomic) formula.

(e)
- If A is a formula so is $\neg A$.
- If A and B are formulas, so are $(A \wedge B), (A \vee B)$, $(A \supset B)$.
- If A is a formula and x is a variable, then $\forall x A$ and $\exists x A$ are formulas.

(f) A string is a formula only if it can be generated by the conditions (d-e).

2 Semantics

Semantics of a logic language is based on an interpretation of its symbols. A model for the first-order language $\langle \mathcal{P}, \mathcal{F}, \mathcal{C}, \mu \rangle$ is a pair $\langle \mathcal{U}, \mathcal{I} \rangle$ where

- $\mathcal{U}$ is a nonempty set, called the universe,
- $\mathcal{I}$ is a mapping, called interpretation that associates:
  - some n-ary relation $\mathcal{I}(P){:}\mathcal{U}^n \to \{true, false\}$ to every predicate symbol $P \in \mathcal{P}$ whenever $\mu(P) = n$,
  - some n-ary function $\mathcal{I}(f){:}\mathcal{U}^n \to \mathcal{U}$ to every function symbol $f \in \mathcal{F}$ whenever $\mu(f) = n$ and
  - some member $\mathcal{I}(c) \in \mathcal{U}$ to every constant symbol $c \in \mathcal{C}$.

By the grammar, a particular language expression can contain only finite number of symbols. Thus, to specify the meaning of an expressions the symbols appearing in the expression (instead of the whole language) are interpreted only. We interpret the expressions by a corresponding formal system.

This fact approves the reason for the existence of the languages with finite predicate, function and constant symbol sets where the signature of the language determines the structure of interpretations.

We get the same result in the case of languages containing infinitely many predicate, function and constant symbols if we say that the semantics does not mean the the interpretation of the language itself, but the interpretation of symbols of the given formula.

If an expression contains n free variables then the result of the interpretation will be an n-variable mapping in the interpreting structure.

The second step of the semantics is the evaluation of the free variables of the language. In an evaluation, the variables mean elements of the universe, therefore in this evaluation the semantics assigns a truth value or an element of the universe to the interpreted expression.

The first-order language with the interpretation becomes (in classical sense) the description language of the interpreting structure. However, there is no tool for naming the elements of the universe in the languages. If

> the value of an expression should be determined in a given interpretation or
>
> a mapping with n variables of the structure should be defined with the syntax specified algorithm

then the naming of the universe elements and of the value of the expression are necessary. There are two ways to refer to the universe elements:

- with a mapping $\kappa : V \to \mathcal{U}$ or
- with extending the language.

Let $\langle \mathcal{U}, \mathcal{I} \rangle$ be a model for the language $\langle \mathcal{P}, \mathcal{F}, \mathcal{C}, \mu \rangle$, and let κ be an assignment in this model. To each term t of $\langle \mathcal{P}, \mathcal{F}, \mathcal{C}, \mu \rangle$, we associate a value $|c|^{\mathcal{I},\kappa}$ in $\mathcal{U}$ as follows:

- for a constant symbol $c \in C$, $|c|^{\mathcal{I},\kappa}$ is the element $\mathcal{I}(c)$ of $\mathcal{U}$,
- for a variable x, $|x|^{\mathcal{I},\kappa}$ is the element $\kappa(x)$ of $\mathcal{U}$,
- $|f(t_1, t_2, \ldots, t_n)|^{\mathcal{I},\kappa} = \mathcal{I}(f)(|t_1|^{\mathcal{I},\kappa}, |t_2|^{\mathcal{I},\kappa}, \ldots, |t_n|^{\mathcal{I},\kappa})$.

Let x be a variable. The assignment κ^* in the model $\langle \mathcal{U}, \mathcal{I} \rangle$ is an x-variant of the assignment κ, if $\kappa^*(y) = \kappa(y)$ for every variable y except x.

To each formula A of $\langle \mathcal{P}, \mathcal{F}, \mathcal{C}, \mu \rangle$, we associate a truth value $|A|^{\mathcal{I},\kappa}$ as follows:

- $|P(t_1, t_2, \ldots, t_n)|^{\mathcal{I},\kappa} = \mathcal{I}(P)(|t_1|^{\mathcal{I},\kappa}, |t_2|^{\mathcal{I},\kappa}, \ldots, |t_n|^{\mathcal{I},\kappa})$.
- $|\neg A|^{\mathcal{I},\kappa} = true$ if and only if $|A|^{\mathcal{I},\kappa} = false$,
- $|A \wedge B|^{\mathcal{I},\kappa} = true$ if and only if $|A|^{\mathcal{I},\kappa} = true$ and $|B|^{\mathcal{I},\kappa} = true$,
- $|A \vee B|^{\mathcal{I},\kappa} = true$ if and only if $|A|^{\mathcal{I},\kappa} = true$ or $|B|^{\mathcal{I},\kappa} = true$,
- $|A \supset B|^{\mathcal{I},\kappa} = true$ if and only if $|A|^{\mathcal{I},\kappa} = false$ or $|B|^{\mathcal{I},\kappa} = true$,
- $|\forall x A|^{\mathcal{I},\kappa} = true$ if and only if $|A|^{\mathcal{I},\kappa^*} = true$ for every assignment κ^* that is an x-variants of κ,
- $|\exists x A|^{\mathcal{I},\kappa} = true$ if and only if $|A|^{\mathcal{I},\kappa} = true$ for some assignment κ^* that is an x-variants of κ.

As is well-known (Dragalin, Girard, Nerode, Shore), the languages can be extended to universes. Usage of the extended languages is comfortable in applications. If $\mathcal{U}$ is an universe of a model of a first-order language, we introduce a new $\underline{u}$ for naming of all elements $u \in \mathcal{U}$. Then we extend the interpretation for the new symbols: $\mathcal{I}(\underline{u}) = u$. Following the *Girard*'s idea, a model $\mathcal{M}$ for $\mathcal{L}$ consists of the next data:

- a nonempty set $\mathcal{U}$, the domain of the model $\mathcal{M}$,
- for all n-ary predicate symbol P_k^n, a relation $\mathcal{M}(P_k^n) : \mathcal{U}^\backslash \to \{\sqcup\nabla\sqcap\rceil, \{\dashv\updownarrow\int\rceil\}$,
- for all n-ary function symbol f_k^n, a function $\mathcal{M}(f) : \mathcal{U}^\backslash \to \mathcal{U}$.

We extend $\mathcal{L}$ to $\mathcal{L}[\mathcal{M}]$ by introducing new constant symbols $\underline{c}$ for all $c \in \mathcal{U}$ where is $\mathcal{M}(\underline{c}) = c \in \mathcal{U}$ for all $\underline{c}$.

We associate a value to each closed expression of $\mathcal{L}[\mathcal{M}]$ as follows:

- $\mathcal{M}(\underline{c}) = c$,
- $\mathcal{M}(f_k^n(t_1, t_2, \ldots, t_n)) = \mathcal{M}(f)(\mathcal{M}(t_1), \mathcal{M}(t_2), \ldots, \mathcal{M}(t_n))$,
- $\mathcal{M}(P(t_1, t_2, \ldots, t_n)) = \mathcal{M}(P)(\mathcal{M}(t_1), \mathcal{M}(t_2), \ldots, \mathcal{M}(t_n))$.

- $\mathcal{M}(\neg A) = true$ if and only if $\mathcal{M}(A) = false$,
- $\mathcal{M}(A \wedge B) = true$ if and only if $\mathcal{M}(A) = true$ and $\mathcal{M}(B) = true$,
- $\mathcal{M}(A \vee B) = true$ if and only if $\mathcal{M}(A) = true$ or $\mathcal{M}(B) = true$,
- $\mathcal{M}(A \supset B) = true$ if and only if $\mathcal{M}(A) = false$ or $\mathcal{M}(B) = true$,
- $\mathcal{M}(\forall x A) = true$ if and only if $\mathcal{M}(A^x_{\underline{c}}) = true$ for all $c \in \mathcal{U}$,
- $\mathcal{M}(\exists x A) = true$ if and only if there is a $c \in \mathcal{U}$ that $\mathcal{M}(A^x_{\underline{c}}) = true$.

Finally we show the semantics of Smullyan's language $\langle \mathcal{P}, Par \rangle$. First we give a nonempty set $\mathcal{U}$ called universe, then we introduce the notion of formulas with constans in $\mathcal{U}$ or more briefly $\mathcal{U}$-formulas:

- If P is an n-ary predicate symbol and $t_1, t_2, \ldots, t_n$ are either variables or elements of $\mathcal{U}$, then $P(t_1, t_2, \ldots, t_n)$ is an atomic $\mathcal{U}$-formula.
-
  - If A is an $\mathcal{U}$-formula so is $\neg A$.
  - If A and B are $\mathcal{U}$-formulas, so are $(A \wedge B), (A \vee B), (A \supset B)$.
  - If A is an $\mathcal{U}$-formula and x is a variable, then $\forall x A$ and $\exists x A$ are $\mathcal{U}$-formulas.

Note that a $\mathcal{U}$-formula does not contain any parameter, moreover it is not in the original language.

Over a fixed universe $\mathcal{U}$ the meaning of predicate symbols of language is given by an interpretation $\mathcal{I}$ which assigns to each n-ary predicate symbol $P \in \mathcal{P}$ a relation $\mathcal{U}^n \to \{true, false\}$. In an interpretation $\mathcal{I}$ we can get truth values to all $\mathcal{U}$-sentence:

- $|P(u_1, u_2, \ldots, u_n)|^{\mathcal{I}} = \mathcal{I}(P)(u_1, u_2, \ldots, u_n)$.
- $|\neg A|^{\mathcal{I}} = true$ if and only if $|A|^{\mathcal{I}} = false$,
- $|A \wedge B|^{\mathcal{I}} = true$ if and only if $|A|^{\mathcal{I}} = true$ and $|B|^{\mathcal{I}} = true$,
- $|A \vee B|^{\mathcal{I}} = true$ if and only if $|A|^{\mathcal{I}} = true$ or $|B|^{\mathcal{I}} = true$,
- $|A \supset B|^{\mathcal{I}} = true$ if and only if $|A|^{\mathcal{I}} = false$ or $|B|^{\mathcal{I}} = true$,
- $|\forall x A|^{\mathcal{I}} = true$ if and only if $|A^x_u|^{\mathcal{I}} = true$ for all $u \in \mathcal{U}$,
- $|\exists x A|^{\mathcal{I}} = true$ if and only if there is a $u \in \mathcal{U}$ that $|A^x_u|^{\mathcal{I}} = true$.

Let now $A(a_1, a_2, \ldots, a_n)$ be a sentence containing exactly the parameters $a_1, a_2, \ldots, a_n$. For any universe $\mathcal{U}$ and any elements $u_1, u_2, \ldots, u_n$ of $\mathcal{U}$ we obtain $A(u_1, u_2, \ldots, u_n)$ by substituting u_1 for a_1 ...... u_n for a_n in the sentence $A(a_1, a_2, \ldots, a_n)$. Such formulas are called pre-interpreted formulas.

- $A(a_1, a_2, \ldots, a_n)$ is satisfiable under interpretation $\mathcal{I}$ if there exists at least one n-tuple $(u_1, u_2, \ldots, u_n)$ of $\mathcal{U}$ such that $|A(u_1, u_2, \ldots, u_n)|^{\mathcal{I}} = true$,
- $A(a_1, a_2, \ldots, a_n)$ is valid under interpretation $\mathcal{I}$ if for every n-tuple $(u_1, u_2, \ldots, u_n)$ of $\mathcal{U}$ $|A(u_1, u_2, \ldots, u_n)|^{\mathcal{I}} = true$.

3 Naming elements of the universe

We could see that the two groups of the first-order languages are the classical languages $\langle \mathcal{P}, \mathcal{F}, \mathcal{C} \rangle$ and the language $\langle \mathcal{P}, Par \rangle$.

- The symbol system and uniform syntax make the different languages suitable for formalization of an arbitrary first-order problem, but language $\langle \mathcal{P}, Par \rangle$. Missing of function symbols does not cause any problem, because an n-ary operation can be defined by an $(n+1)$-ary relation.
- The semantics is uniform, so both the semantical properties of a formula or set of formulas and the notion of semantical consequence can be defined for every language in the same way. Similarly the proof of deduction theorem and the drafting of the semantical decision problem does not depend on the language.

The solving of the semantical decision problem raises the issue of perspicuity of all interpretations over a given universe.

- In the case of languages $\langle \mathcal{P}, \mathcal{F}, \mathcal{C} \rangle$, besides the elements of $\mathcal{P}$ we must interpret the elements of $\mathcal{F}$ for a complete interpretation.
- In the case of languages $\langle \mathcal{P}, \emptyset, \mathcal{C} \rangle$ and $\langle \mathcal{P}, Par \rangle$ we can give any interpretation with the evaluation of atomic formulas. Here the interpretations can be regarded as points of a space determined by all ground atoms:

  a sequence of all ground atoms is called a base and

  the interpretation is a base in which every ground atom has a truth value.

  All of the interpretations can be given by a semantical tree based on the base.

Let $\mathcal{U} = \{\dashv, \lfloor\}$ be the universe and $P(a,a), P(b,b), P(a,b), P(b,a)$ be the base. The semantical tree based on this base is the next: "b"

Notice that the description of ground atoms needs the naming elements of the universe. Accordingly, if we fix a universe, then we have to extend the language with new symbols denoting different elements of universe in a pre-interpretation.

The introduction of symbols for naming of elements of universe is routine in the description language of some mathematical structures. For example we can extend the description language of the arithmetics with naming of natural number. The name of a number can be a sequence of digits $0, 1, \ldots, 9$. We do this in spite of the successor function guarantees the naming of natural numbers.

The following well-known result led to the development of resolution calculus:

A set of Skolem formulas is unsatisfiable if and only if it is unsatisfiable over the Herbrand universe, that is over the set of all ground terms generated from function and constant symbols of the language.

This is an important result because it traces back the examination of a formula $\langle \mathcal{P}, \mathcal{F}, \mathcal{C} \rangle$ to the examination of the formula of $\langle \mathcal{P}, \mathcal{C} \rangle$ over the Herbrand universe. Observe that

- the elements of Herbrand universe are ground terms and at the same time names of elements. Here the names of elements depend on the language. (In some works names depend on the particular model.)

- Listing of the elements of the Herbrand universe determines an interpretation for the function symbols: naming with $h_1, h_2, \ldots$ of the elements of the universe we get an interpretation for the function symbols over the set $h_1, h_2, \ldots$.

The Hintikka sets play an important role in the tableau calculus. A Hintikka set can be defined over a given universe. To recognize a Hintikka set on a branch of the tableau, the naming in the language of elements of the universe is needed. The rules of the tableau depend on the description language through their formulas.

For example the first-order tableau rules are the following:

$$\text{language } \langle \mathcal{P}, \mathcal{F}, \mathcal{C} \rangle : \quad \frac{\forall x A(x)}{A(t)} \qquad \frac{\exists x A(x)}{A(y)} \ (y \text{ is a critical variable})$$

$$\text{language } \langle \mathcal{P}, Par \rangle : \quad \frac{\forall x A(x)}{A(a)} \qquad \frac{\exists x A(x)}{A(a)} \ (a \text{ is a critical parameter})$$

The tableau rules of the language

- $\langle \mathcal{P}, Par \rangle$ provide to arise Hintikka sets defined over the set of critical parameters as a universe on the open branches of the tableau. It is well-known that a Hintikka set is satisfiable.
- $\langle \mathcal{P}, \mathcal{F}, \mathcal{C} \rangle$ provide only, that the set of formulas arising on a branch without any complement pair fulfil the rules of the Hintikka sets concerning the critical parameters, but we can substitute arbitrary terms into the matrix of the quantifier formulas. According to *C. Bell*'s and *M. Machover*'s works these sets are satisfiable over the set of equivalence classes of terms introduced the rules.

Soundness of deduction systems:
The definition of the soundness depends on the calculus itself.

- For the Hilbert system: If a formula A is deducible from a set Γ of formulas, then $\Gamma \models A$. Or if Γ is inconsistent, then Γ is unsatisfiable.
- For the resolution calculus: If the empty clause has resolution deduction from a set S of clauses, then S is unsatisfiable.
- For the tableau calculus: If the tableau of a formula A is closed, then A is unsatisfiable.

The key of the proof is the soundness of the deduction rule.

Completeness of deduction systems:

A calculus is complete, if a set of formulas have the semantical property given by soundness, then the calculus works successfully.

- By the semantics a set of formulas is either satisfiable or not. This property divides into two disjoint parts the set Ω of sets of formulas.

- For a given calculus we can define properties for the sets of formulas dividing into two disjoint parts the set Ω. These are so called consistency and inconsistency properties.

The inconsistency property

- for the Hilbert system: A set Γ of formulas is inconsistent if A and $\neg A$ are deducible from Γ.
- for the resolution calculus: a set S of clauses is inconsistent if the empty clause has resolution deduction from S.
- for the tableau calculus: A formula A is inconsistent if the tableau of A is closed.

The completeness is expressed by the (in)consistency property.

- If a set of formulas is unsatisfiable, then it is inconsistent.
- If a set of formulas is satisfiable, then it is consistent.

So the semantical and the syntactical properties divide into the same two parts the set Ω of sets of formulas.

- For the Hilbert system
  - the classical consistency property: A set Γ of formulas is consistent if its finite subsets are satisfiable.
  - the completeness theorem: If a set Γ of formulas is consistent, then it is satisfiable.The proof uses the compactness theorem and the Tukey lemma.
- For the tableau method
  - the consistency property: A formula A is consistent if its tableau is not closed say it has open branches.
  - the completeness theorem: Using the language $\langle P, Par \rangle$ on an open branch, a Hintikka set appears on the set of the critical parameters. Consequently if A is consistent, then it can be embedded into a Hintikka set, so A is satisfiable

Γ-consistency (for $\alpha-, \beta-, \gamma-, \delta-$ formulas): S is a Γ-consistent set of formulas if

- S does not contain complement formula pair,
- if $\alpha \in S$ then $\{S, \alpha_1\}$ and $\{S, \alpha_2\}$ are Γ-consistent,
- if $\beta \in S$ then $\{S, \beta_1\}$ or $\{S, \beta_2\}$ is Γ-consistent,
- if $\gamma \in S$ then $\{S, \gamma(a)\}$ is Γ-consistent,
- if $\delta \in S$ then $\{S, \delta(a)\}$ is Γ-consistent, if a is not in S.

Clear that it is possible to embed an Γ-consistent set S of formulas into a Hintikka set.

Important result is for the calculi in logic: the consistent sets are Γ-consistent.

The main applications of logic in the artificial intelligence belong to the automatical theorem proving. We will see that the naming of the element of universe is inevitable.

The main steps:

- Preparation. Create the relations and operations necessary.
- Give a description language. The non-logical part consists of predicate and function symbols naming the relations and operations.
- To identify the elements of the universe, constant symbols are introduced.
- In case of using resolution calculus the constant symbols are the constant elements of the Herbrand universe.

The preparation for using PROLOG is the same.

Another field of application is the relational database theory.

The description and query language DATALOG uses a logical language to denominate the relations and describe their connections. In this language there are only predicate symbols, sorts, countable many variables and constant symbols for every sort.

- A query describes a relation with special structure. This is the theorem to prove.
- The condition formulas are formulas expressing the connection among the relation named by the predicate symbols.
- The model of the language is a suitable database system where constant symbols can be used for naming the universe elements.
- The language has a tool to name the data elements. Therefore, while DATALOG executes the steps of resolution deduction, it uses the relational calculus instead of unification.

4 Conclusion

- The applications require from the description language the ability to name the elements of the domain of applications. So the problem of an interpretation depending extension is important.
- Sometimes the language $\langle \mathcal{P}, Par \rangle$ is convenient to solve this problem.
- It is clear that every interpretation depending extension of $\mathcal{L}$ involves that the set of models of the extended language is a subset of the set of models of $\mathcal{L}$.
- Some trend in logic offers the facility to fix a domain first and extends the alphabet of $\mathcal{L}$ by the names of the domain. In this case the formulas are pre-interpreted before the interpretation. That means the constant and the parameter symbols are substituted by these names in the formulas.

References

[1] Bell, C., Machover, M., *A Course in Mathematical Logic*, Elsevier / North-Holland, Amsterdam, London, New York, 1977.

[2] Buzási Sz., Dragálin, A., *Bevezetés a matematikai logikába*, KLTE egyetemi jegyzet, Debrecen, 1986.

[3] Fitting, M., *First-order Logic and Automated Theorem Proving*, Springer-Verlag, New York, 1996.

[4] Frege, G., *Begriffsschrift, eine der arithmetischen nachgebildete Formelsprache des reinen Denkens*, Halle 1879.

[5] Gallier, J.H., *Logic for Compute Science: Foundation of Automatic Theorem Proving*, Harper and Row, New York, 1986.

[6] Girard, J.-Y., *Proof Theory and Logical Complexity* Volume 1, Bibliopolis, Napoli, 1987.

[7] Kalmár, L., *A matematika alapjai I-IV.*, JATE egyetemi jegyzet, Szeged, 1971.

[8] Kleene, S.C., *Mathematical Logic*, John Wiley and Sons, New York, 1967.

[9] Manna, Z., Waldinger, R., *The Logical Basis for Computer Programming: Volume 1: Deductive Reasoning*, Addison-Wesley, Reading, Massachusetts, 1992.

[10] Mendelson, E., *Introduction to Mathematical Logic*, D. Van Nostrand Company, Inc., 1964.

[11] Nerode, A., Shore, R.A., *Logic for Applications*, Springer-Verlag, New York, 1997.

[12] Pásztorné Varga K., *A matematikai logia és alkalmazásai*, ELTE egyetemi jegyzet, Budapest, 1989.

[13] Pásztorné Varga K., *Matematikai logia alkalmazásokhoz*, ELTE egyetemi jegyzet, Budapest, 1997.

[14] Peano, G., *Arithmetices principia, nova methodo exposita*, Fratres Bocca, Turin, 1889.

[15] Ruzsa, I., *A matematika néhány filozófiai problémájáról*, Ruzsa, I., Urbán, J., *Matematikai logika*, Tankönyvkiadó, Budapest, 1966.

[16] Ruzsa, I., *Klasszikus, modális és intenzionális logika*, Akadémiai Kiadó, Budapest, 1984.

[17] Ruzsa, I., Máté, A., *Bevezetés a modern logikába*, Osiris Kiadó, Budapest, 1997.

[18] Smullyan, R.M., *First-Order Logic*, Springer-Verlag, Berlin, Heidelberg, 1968.

Postal addresses

Katalin Pásztor Varga
Dept. of Progr. Languages and Compilers
Eötvös Loránd University
1117 Budapest, Pázmány Péter sétány 1/C.
Hungary

Magda Várterész
Institute of Informatics
University of Debrecen
4010 Debrecen, Egyetem tér 1.
Hungary