Università di Pisa

Dipartimento di Informatica

Technical Report : TR-97-10

Logical Formalizations of Syntactical Properties (Extended Abstract)

V. Manca

June 23, 1997

ADDR: Corso Italia 40,56125 Pisa,Italy. TEL: +39-50-887111. FAX: +39-50-887226

Logical Formalizations of Syntactical Properties (Extended Abstract) *

Vincenzo Manca
Università di Pisa
Dipartimento di Informatica
Corso Italia, 40 - 56125 Pisa - Italy
e-mail: mancav@di.unipi.it

1 Introduction

Assume the 'marvelous' 7 logical symbols:

$$\rightarrow, \neg, \wedge, \vee, \leftrightarrow, \forall, \exists$$

with the standard syntactical and semantical first order logical notions (equality predicate $=$ is assumed in its usual usage) that can be found in introductory treatises or basic chapters of textbooks in mathematical logic (cf., for example, [1], [6], [15], [16], [3], [2] [13]):

1. variables and constants
2. predicates and functors
3. first order formulae and terms
4. free variables, bound variables, and substitutions
5. connective truth tables

*VI Tarragona Seminar on Formal Syntax and Semantics, University Rovira i Virgili, Tarragona, Spain, October 14-18 1996 (Monday Talk).

6. first order structures, interpretations, models and diagrams

7. first order logical validity $\models$

Let us mention the fundamental results of first order logic. Although they are not completely assumed, a basic familiarity with the main concepts is useful for a deeper understanding of the following discussions:

1. prenex normal forms, Skolem forms, and clause forms.

2. derivability relations and proof systems

3. recursive enumerability and (first order) axiomatizability

4. completeness and compactness (Herbrand, Gödel, Henkin Theorems)

5. non categoricity of first order theories (Löwenheim-Skolem Theorems)

6. non axiomatizability and undecidability of arithmetics (Gödel, Church's Theorems)

7. complete theories, elementary equivalence and elementary extensions.

2 Logical Representability: Hilbert's Thesis

Consider the following (informal) definition.

Definition 1 *Let* $\mathbf{\Sigma}$ *be a signature,* $\mathfrak{M}$ *a* $\mathbf{\Sigma}$*-model and* $\mathcal{S}$ *a mathematical statement about some model (possibly with a signature different from* $\mathbf{\Sigma}$*). The statement* $\mathcal{S}$ *is represented by the formula* φ *of* $\mathcal{L}_{\Sigma}$*, in the* $\mathbf{\Sigma}$*-model* $\mathfrak{M}$*, when:* $\mathcal{S} \Leftrightarrow \mathfrak{M} \models \varphi$*.*

Indicate by Ar the *standard arithmetical model*:

$$Ar = (\mathbf{N}, +, \cdot, 0, 1)$$

where $\mathbf{N}$ is the set of natural numbers, $+$ and $\cdot$ denote respectively the plus and times operations on natural numbers, and 0 and 1 denote respectively numbers zero and one. As usual, we may use, ambiguously, the same symbols to indicate either functors, predicates, and constants of an $\mathbf{\Sigma}$-model, either the corresponding functions, relations and individuals symbols of the signature $\mathbf{\Sigma}$.

Example 1 *The statement: '3 is a prime number' is represented in* Ar, *by the following formula of the first order language of signature* $\{+, \cdot, 0, 1\}$*:*

$$\forall xy(x \cdot y = (1+1)+1 \to x = (1+1)+1 \vee y = (1+1)+1)$$

Notation 1 $\mathcal{S} \equiv_{\mathfrak{M}} \varphi$ *indicates that* φ *represents* $\mathcal{S}$ *within* $\mathfrak{M}$*. When the model* $\mathfrak{M}$ *is clearly understood, the subscript* $\mathfrak{M}$ *may be omitted.*

The following example is a list of very simple formalizations within the standard arithmetical model Ar, where $P(n)$ means that n is a prime number and $|P| = \aleph_0$ means that the set of prime numbers is denumerable, and $<,>$ is Cantor's pairing function:

Example 2

$$x \leq y \equiv \exists z(x + z = y)$$

$$x = 0 \equiv \forall y(x + y = y)$$

$$z = Rem(x, y) \equiv \exists q(x = q \cdot y + z)$$

$$P(n) \equiv \forall xy(x \cdot y = n \to x = n \vee y = n)$$

$$|P| = \aleph_0 \equiv \ \forall x \exists y(P(y) \wedge y \geq x).$$

$$< x, y >= z \equiv \ (z + z = (x + y + 1) \cdot (x + y) + x + x)$$

Indicate by $\varphi(x)$ a formula where the variable x may occur and by $\varphi(c)$ the same formula where the constant c replaces all the occurrences of x. We say that in a model $\mathfrak{M}$ a constant can be eliminated if $\mathfrak{M} \models x = c$ can be formalized in $\mathfrak{M}$ by a formula ψ where does not occur the constant c. It is easy to prove that constants 0 and 1 can be eliminated in Ar.

Now we present a thesis that was never expressed by David Hilbert, but seems to be as an implicit assumption underlying hilbertian formalistic program in metamathematics.

Hilbert's Thesis
Every mathematical statement can be represented by some first order $\mathcal{L}_\Sigma$-formula (for a suitable signature Σ) in some Σ-model $\mathfrak{M}$.

This thesis can be compared with Church's thesis in the theory of computability: 'Any algorithm can be computed by a Turing machine (w.r.t. some encoding of data)'. What exactly is a mathematical statement cannot be specified in a formal way (analogously, the informal notion of algorithm is assumed in Church's thesis). Thus, these assertions can be viewed as claims that some informal notions can be adequately expressed in suitable formal settings. In a synthetical and evocative manner we can paraphrase Hilbert's thesis by saying that:

First order logic is the formal language of mathematics.

In order to appreciate the expressive power of formalization within first order models let us consider some examples of formalization in the *standard syntactic model* Syn:

$$Syn = (\mathbf{N}^*, \_\,\_, |\ |)$$

where $\mathbf{N}^*$ is the set of finite strings of natural numbers, $\_\,\_(\alpha, \beta)$ abbreviated by $\alpha\beta$ is the concatenation of strings α and β, and $|\alpha|$ is the length of string α. In this model numbers can be conceived as symbols of an infinite alphabet, therefore any language can be embedded in its domain (we may identify $0, 1, 2, \ldots$ with letters $a, b, c, \ldots$).

We write: $\alpha(i) = n$ to say that symbol (number) n occurs in the string α at position i, $\alpha \subseteq \beta$ to say that α is a substring of β, i.e., the string constituted by the symbols of α which occur between two (not necessarily distinct) positions; and $a \in \alpha$ iff $a \subseteq \alpha$ and $|a| = 1$.

Example 3

$$y \in \mathbf{N} \ \equiv\ (|y| = ||y||)$$

$$(x(i) = y) \ \equiv\ \exists yz(x = zyw \ \wedge\ y \in \mathbf{N} \ \wedge |zy| = i)$$

$$(x \subseteq y) \equiv \exists zw(y = zxw)$$

$$(x \in y) \equiv (x \subseteq y \wedge |x| = 1)$$

$$(x + y = z) \ \equiv\ \exists uv(x = |u| \ \wedge\ y = |v| \ \wedge z = |uv|).$$

Notice that with the representation of $+$ in Syn we inherit in Ar all the relations whose definendum use such function symbol. The following example is more complicate. How can we represent the product between natural numbers in terms of relationships among strings? We can get a solution to this question by represent the natural process of calculating a product by means of iterated sums. In fact if $m \cdot k = n$, there exists a sequence of natural numbers $m, m \cdot m, m \cdot m \cdot m, \ldots,$ of length k that ends with n.

Example 4

$$
\begin{aligned}
& m \cdot k = n \ \equiv \\
& \exists w(|w| \ = k \ \wedge \ w(1) = m \ \wedge \\
& \forall x(x < k \rightarrow w(x+1) = w(x) + m) \ \wedge \ w(k) = n)
\end{aligned}
$$

The idea of previous example has a wide application. Notice besides that the simple structure of the formula representing multiplication is due mainly to the possibility of Syn-representing computations in a direct way (then avoiding all sort of syntax' encoding). In fact, we have:

Example 5 *The following relations are representable in the model Syn, where $\pi(k)$ is the k-th prime number and $G(n, m, k)$ indicates that m is a prime number and m^k is the maximum power of m that is divisor of n:*

$$n = m^k$$

$$n = m!$$

$$n = \pi(k)$$

$$G(n, m, k).$$

3 Arithmetical Encoding of Syntax

In order to introduce a general notion of representability between models, consider the following theorem saying that first order standard arithmetics can be *expressed* in the syntactic model Syn.

When we refer to a formula φ as a formula of a $\mathbf{\Sigma}$-model $\mathfrak{M}$, we intend that φ is a first order formula over the language $\mathcal{L}_{\mathfrak{M}}$ obtained by extending its original signature with all elements of $|\mathfrak{M}|$, the domain of $\mathfrak{M}$, taken as autoreferential constants (being each of them interpreted into itself).

Theorem 1 *For every φ of $\mathcal{L}_{Ar}$, the statement $Ar \models \varphi$ can be represented in the model Syn.*

Proof. It is a direct consequence of the examples given in the previous section.

Let $\mathfrak{M}_1$ and $\mathfrak{M}_2$ two first order models.

Definition 2 *A formula* φ *of* $\mathcal{L}_{\mathfrak{M}_1}$ *is represented by the formula* ψ *of* $\mathcal{L}_{\mathfrak{M}_2}$ *iff* $(\mathfrak{M}_1 \models \varphi) \equiv_{M_2} \psi$.

Definition 3 *A set* Φ *of formulae of* $\mathcal{L}_{\mathfrak{M}_1}$ *is representable in a model* $\mathfrak{M}_2$ *iff any formula of* Φ *is represented by some formula of of* $\mathcal{L}_{\mathfrak{M}_2}$.

Definition 4 *A subset* L *of* $|\mathfrak{M}_1|$ *is representable in a model* $\mathfrak{M}_2$ *iff there exists a formula* $\varphi(x)$ *of* $\mathcal{L}_{\mathfrak{M}_2}$ *such that* $(a \in L) \Leftrightarrow \mathfrak{M}_2 \models \varphi(a)$.

The previous definition applies also to any k-ary relation over $|\mathfrak{M}_1|$, viewed as a subset of $|\mathfrak{M}_1|^k$ (in this case we need a formula with k free variables).

Given a model $\mathfrak{M}$, indicate by $Diag(\mathfrak{M})$ the set of *literals* of $\mathfrak{M}$ holding in $\mathfrak{M}$; i.e. the set of atomic formulae, and negated of atomic formulae of $\mathcal{L}_{\mathfrak{M}}$ that hold in $\mathfrak{M}$.

Definition 5 *A model* $\mathfrak{M}_1$ *is representable into a model* $\mathfrak{M}_2$, *say* $\mathfrak{M}_1$ *is* $\mathfrak{M}_2$-*representable, iff the theory* $Diag(\mathfrak{M}_1)$ *is representable in* $\mathfrak{M}_2$.

From the last definition and the previous theorem it follows that Ar is Syn-representable. Now we show that the converse representation also holds: all syntactic relations can be represented into the arithmetic model Ar. This result is based on the following lemma.

Lemma 2 *(Gödel) There exists a function* $\boldsymbol{\beta}$ *such that for any finite sequence of numbers* $a_1, \ldots, a_k$ *there are two numbers* c, d *such that* $\boldsymbol{\beta}(c, d, 0) = k$ *and for all* i *such tat:* $0 < i \leq k$:

$$\boldsymbol{\beta}(c, d, i) = a_i$$

Furthermore, the above equation is representable in the model Ar.

Proof. The proof is a direct consequence of *Chinese Remainder Theorem* (see [13]).

This lemma allows us to obtain the representation of Syn into Ar.

Theorem 3 *Syn is Ar-representable.*

Proof. Let α, δ, γ be strings of $\mathbf{N}^*$ and let $c_1, d_1, c_2, d_2, c_3, d_3$ the pairs associated to them according to the Gödel lemma, then we have:

$$(Syn \models\ |\alpha| = n) \Leftrightarrow \boldsymbol{\beta}(c_1, d_1, 0) = n$$

moreover:

$$Syn \models \alpha\delta = \gamma$$

$$\Leftrightarrow$$

$$\boldsymbol{\beta}(c_3, d_3, 0) = \boldsymbol{\beta}(c_1, d_1, 0) + \boldsymbol{\beta}(c_2, d_2, 0) \wedge$$

$$\forall x(1 \leq x \leq \boldsymbol{\beta}(c_1, d_1, 0) \to \boldsymbol{\beta}(c_3, d_3, x) = \boldsymbol{\beta}(c_1, d_1, x)) \wedge$$

$$\forall x(1 \leq x \leq \boldsymbol{\beta}(c_2, d_2, 0) \to \boldsymbol{\beta}(c_3, d_3, \boldsymbol{\beta}(c_1, d_1, 0) + x) = \boldsymbol{\beta}(c_2, d_2, x))$$

therefore, $Diag(Syn)$ is formalizable within Ar, that is, Syn is Ar-representable. *Q.E.D.*

The representability shown in the previous theorem, can be expressed in the following more precise manner. Indicate by $|M|$ the domain of a model M, in the proof of theorem two functions are given, a *diagram encoding*

$$F : Diag(Syn) \to L(Ar)$$

and a surjective *individual encoding* $\boldsymbol{\beta}$

$$\boldsymbol{\beta} : |Ar| \times |Ar| \ \to |Syn|$$

In this sense we can speak more properly of a *controvariant representation*, for the individual encoding has opposite verse w.r.t. the verse of the diagram encoding. But, Syn can be represented arithmetically also in *covariant* manner, say also Syn is *Ar-corepresentable*, in as much we can also define a pair of codes running in the same verse. In order to show this result we need the following lemma.

Lemma 4 *All the arithmetical relations defined in the example 5 can be represented in the model Ar.*

Proof. It is sufficient to use Gödel's $\boldsymbol{\beta}$ function.

Theorem 5 *Syn has a covariant representation into Ar.*

4 Logical Syntax of Formal Languages

Let us recall some classical notions of formal language theory necessary to understand the following discussion. For these notions and further details we refer to [12].

Given a finite alphabet A of symbols, $A = \{a_1, \ldots, a_n\}$, we can embed it into a finite subset N_A of natural numbers by means of a one-to-one mapping f between A and N_A. In this manner a corresponding one-to-one mapping f' between A^* and ${N_A}^*$ is defined in the natural way:

$$f'(\lambda) = f(\lambda) = \lambda$$

$$f'(a_i\alpha) = f(a_i)f'(\alpha) \;\; \textit{for each } a_i \in A, \; \alpha \in A^*$$

Therefore, we can consider languages over alphabet A as particular subsets of the free monoid $\mathbf{N}^*$. This embedding allows us to represent syntactic properties in the model Syn.

Let us consider a Chomsky grammar $G = (T, N, P, S)$, where T is the finite alphabet of terminals, N is the finite set of nonterminals ($V = T \cup N$ is the set of non terminal and terminal symbols), P is the finite set of productions $\alpha \to \beta$, with $\alpha \in V^+$, $\beta \in V^*$ (V^* being the free monoid generated by V, and $V^+ = V^* - \{\lambda\}$) and, finally, $S \in N$ is the start symbol. From S and replacing iteratively in a string the left side of a production with the right one, a grammar G generates a type-0 language $L(G)$ constituted by all $\alpha \in T^*$ generated from S with a finite number of replacements. Type-0 languages coincide with the recursively enumerable subset of T^*, i.e. the subsets generated by some process that is effective w.r.t. some universal computation formalism. If productions of G are all of the form $\alpha \to \beta$, where α and β are any two strings different from λ with $|\alpha| \leq |\beta|$, then G is a *monotone grammar*. These grammars generate the class of *context sensitive* or type-1 languages (a proper subclass type-0 languages). If productions of G are all of the form $A \to \beta$, where A is any symbol in N and β is any string other than λ, then we say that G is a *context-free grammar*, generating a type-2 language [1]. A grammar is called *regular* when it contains only rules

[1] Here and in the following we just consider empty-free grammars, that is grammars without any production $\alpha \to \lambda$, $\alpha \in V^+$. In fact, given a grammar G, there exists a corresponding empty-free grammar G' such that:

- $L(G) = L(G')$ if $\lambda \notin L(G)$ or of form $A \to aB$ or $A \to a$, with $a \in T$ and $A, B \in N$. Regular grammars generate type-3 languages, properly included into type-2 languages, that are properly included into type-1 languages.

A *regular expression* is defined in the following recursive way:

- $\{\lambda\}$, $\{a\}$ are regular expressions, denoting respectively the language with only the empty sentence and languages with only symbol a;
- if e_1 and e_2 are two regular expressions denoting languages $L(e_1)$ and $L(e_2)$ then e_1+e_2 is a regular expression denoting the union of languages $L(e_1)$ and $L(e_2)$;
- if e_1 and e_2 are two regular expression denoting languages $L(e_1)$ and $L(e_2)$ then e_1e_2 is a regular expression denoting the concatenation of languages $L(e_1)$ and $L(e_2)$;
- if e is a regular expression then e^* is also a regular expression that denotes the language $L(e^*) = \cup_{i=0}^{\infty} L(e^i)$, where $L(e^0) = \lambda$, $L(e^{i+1}) = L(e)L(e^i) = \{\alpha\beta | \alpha \in L(e), \beta \in L(e^i)\}$ for $i > 0$.

Regular grammars and regular expressions are equivalent formalisms (see [12]).

A *substitution* f is a mapping from finite alphabet A to the powerset of A:

$$f : A \to 2^A$$

Thus f associates some language with each symbol of A. Such a mapping can naturally be extended to string in A^* in the following manner (here juxtaposition indicates concatenation between languages):

$$f(\lambda) = \lambda$$

$$f(ax) = f(a)f(x), \ a \in A, \ x \in A^*$$

Furter, we can also extend f to languages $L \subseteq A^*$ by defining

$$f(L) = \cup_{x \in L} f(x)$$

- $L(G) = L(G') \cup \{\lambda\}$ if $\lambda \in L(G)$.

So, we can focus our attention just on empty-free grammars (for further details, see [12], page 62, section 4.6).

If $f(a)$ is a singleton for all $a \in A$, then we say that f is a *homomorphism* in A and we identify $f(a)$ with its unique element.

Finally, a *Dyck* language is a context-free language generated by a grammar

$$G_k = \{\{a_1, a_2, \ldots, a_k, b_1, b_2, \ldots, b_k\}, \{S\}, P, S\}$$

where P consists of productions $\{S \to SS,\ S \to \lambda\ ,\ S \to a_i S b_i | 1 \leq i \leq k\}$. $L(G_k)$ can be thought as the language of correct strings over parentheses of k types (being a_i and b_i each pair of left and right parentheses of type i).

Consider the following syntactical relation:

- $sub(\alpha, i, j, \beta)$ means that the string β is the substring of α starting at position $i+1$ and ending at position j:

  $$\alpha[i, j] = \beta,\ i < j$$

  For example:

  $$\alpha = \overbrace{b\ \ a}^{\gamma}\overbrace{b\ \ c\ \ a\ \ d}^{\beta}\overbrace{\ b\ }^{\delta}\ ,\ \text{with } i = 2 \text{ and } j = 6$$

- $occur(\alpha, a, n)$ means that in the string α the symbol a occurs exactly n times:

  $$|\alpha|_a = n$$

- $hom_h(\alpha, \beta)$ means:

  $$h(\alpha) = \beta$$

  where h is a string homomorphism.

- $perm(\alpha, \beta)$ means that α is a permutation of β.

The following lemma is useful in the further discussion.

Lemma 6 *Syntactical relations: $sub, occur, hom_h,$ and $perm$ are Syn-representable.*

Theorem 7 *Any Dyck language is Syn-representable.*

The following theorem gives us a first result of logical representability for an important class of formal languages.

Theorem 8 *Any regular language is Syn-representable.*

From theorems given so far we get easily the following theorem.

Theorem 9 *Any context-free language is Syn-representable.*

Proof. A language L is context-free if and only if there exist a homomorphism h, a *Dyck* language D and a regular language R such that $L = h(D \cap R)$. Therefore, it is sufficient to combine the representability into Syn of Dyck languages, of regular languages and of homomorphisms. *Q.E.D.*

From this and previous results, we can deduce the representability of any type-0 language L in the model Syn; in fact we know [12] that $L = h(L_1 \cap L_2)$ for a suitable homomorphism h and two context-free languages L_1, L_2. However, by using the techniques already introduced we have also a direct proof of this general theorem.

Theorem 10 *Any type-0 language is Syn-representable.*

Proof. (Sketch) Let L be a language generated by a type-0 grammar G. A string α belongs to L if there exist two strings β and ν, the first one constituted by the concatenation of all strings of a derivation of α, the second one (whose length is the length of the derivation + 1) being a vector that allows us to extract from β, for every pair of values of consecutive indices, the derivation steps of α. In order to characterize all strings of L and only them, it is sufficient to express that, for any derivation of them, any step is obtained by the previous one by applying some production of G; but this is easily representable by a simple logical condition expressed by concatenation and existential quantification. *Q.E.D.*

Is any Syn-representable language a type-0 language? If not, which is the class of formulae that represents type-0 languages? In order to answer this questions, we will introduce the notion of first order axiomatic representability.

5 First Order Axiomatic Representability

Let us recall some basic notions about first order theories. A set Φ of $\mathcal{L}_\Sigma$-formulae is *recursively enumerable* (r.e.) or *semidecidable* if we have an algorithm for effectively generating all the formulae of Φ. It is *decidable* or *recursive* iff we have an algorithm for deciding when a given $\mathcal{L}_\Sigma$-formula belongs to Φ or not, or equivalently, iff we can effectively generate all the formulae of Φ and all the $\mathcal{L}_\Sigma$-formulae that do not belong to Φ. An $\mathcal{L}_\Sigma$-*theory* Φ is a set of formulae closed w.r.t. the first order logical consequence, that is, for every $\mathcal{L}_\Sigma$-formula: $\Phi \models \varphi \Rightarrow \varphi \in \Phi$.

A theory Φ is *axiomatizable* iff $\Phi = \{\varphi | Ax \models \varphi\}$ for some r.e. set of formulae Ax called *axioms* of Φ. If Ax is a decidable (recursive) set, then Φ is an *axiomatic* theory (if Ax is finite, Φ is finitely axiomatizable). Given the effective nature of first order logical consequence (see deductive completeness of the proof systems), the set of theorems of an axiomatizable theory is recursively enumerable.

An $\mathcal{L}_\Sigma$-theory Φ is *complete* iff, for any L-formula φ, $\Phi \models \varphi$ or $\Phi \models \neg\varphi$ holds (Φ is *sound* iff it is not the case that both conditions hold). A very simple, but fundamental result establishes that any axiomatizable and complete theory is also decidable.

The following is our starting definition.

Definition 6 *A mathematical statement $\mathcal{S}$ is axiomatically represented by the $\mathcal{L}_\Sigma$-formula φ in a $\mathcal{L}_\Sigma$-theory Φ, that is, $\mathcal{S}$ is Φ-representable, iff $\mathcal{S} \Leftrightarrow (\Phi \models \varphi)$.*

The notion of axiomatic representability can be considered as a generalization of first order model representability. In fact, a statement is represented in a model $\mathfrak{M}$ iff it is represented in the theory $Th(\mathfrak{M})$. Therefore all the notions definited for first order model representability can be naturally extended to first order axiomatic representability, by replacing a model $\mathfrak{M}$ with the theory $Th(\mathfrak{M})$.

Consider the following theory RR^* on the signature $\{\omega, --, ||, \leq, \subseteq, \lambda\}$ where ω indicates a denumerable set of constants for natural numbers (denoted by the usual symbols). This theory, strongly related to the arithmetical theory RR due to Raphael Robinson (cf. [13]), consists of the following axiom schemata, for all $n, m, n_1, \ldots, n_k, m_1, \ldots, m_h \in \omega$ where $n \neq m$:

- $(n_1 \ldots n_k)\,(m_1 \ldots m_h) = (n_1 \ldots n_k m_1 \ldots m_h)$
- $|(n_1 \ldots n_k)| = k$

- $\neg n = m$
- $\forall x(x \leq n \leftrightarrow (x = 0 \vee x = 1 \vee \ldots \vee x = n))$
- $\forall x(x \subseteq (n_1 \ldots n_k) \leftrightarrow \bigvee_{i \leq j \leq k} x = (n_i \ldots n_j))$

The following are interesting applications of first order axiomatic representability, where a, b, c stand for any three natural numbers.

Example 6 *$\{a^n b^n | n \in \omega\}$ is representable in RR^*, extended by these axioms:*

- $L(\lambda)$
- $L(x) \rightarrow L(axb)$

In fact: $\alpha \in \{a^n b^n | n \in \omega\} \Leftrightarrow RR^* \cup Ax \models L(\alpha)$.

Example 7 *$\{a^n b^n c^n | n \in \omega\}$ is representable in RR^*, extended by these axioms:*

- $L(\lambda)$
- $L(xby) \rightarrow L(axbbyc)$

In fact: $\alpha \in \{a^n b^n c^n | n \in \omega\} \Leftrightarrow RR^* \cup Ax \models L(\alpha)$.

In general we can prove that:

Theorem 11 *Any type-0 language L is representable in some extension of RR^* with a suitable finite set of axioms Ax_L.*

Robinson's RR first order arithmetical theory of signature $\{\omega, +, \cdot, \leq\}$ consists of the following axiom schemata:

- $n + k = m$ iff $(n + k = m) \in Diag(Ar)$
- $n \cdot k = m$ iff $(n \cdot k = m) \in Diag(Ar)$
- $\neg n = m$ for all $n \neq m \in \omega$

- $\forall x(x \leq n \leftrightarrow (x = 0 \vee x = 1 \vee \ldots \vee x = n))$ for all $n \in \omega$

Let us extend the signatures of $\mathcal{L}_{Ar}$ and $\mathcal{L}_{Syn}$ with the predicate symbols $\leq$ and $\subseteq$ (meaning in Ar and Syn respectively the usual order on natural numbers and the substring inclusion), and consider the following abbreviations:

$$\forall x \leq t\,.\varphi \;\equiv\; \forall x(x \leq t \rightarrow \varphi)$$

$$\forall x \subseteq t\,.\varphi \;\equiv\; \forall x(x \subseteq t \rightarrow \varphi)$$

$$\exists x \leq t\,.\varphi \;\equiv\; \exists x(x \leq t \wedge \varphi)$$

$$\exists x \subseteq t\,.\varphi \;\equiv\; \exists x(x \subseteq t \wedge \varphi)$$

where the term t is said to bound the quantifiers (a normal quantification considered as an *unbounded* quantification). A (extended) formula φ of $\mathcal{L}_{Ar}$ is a Σ_1-formula iff it is constructed starting from atomic formulae by means of connectives, bounded quantifications and existential unbounded quantifications. Analogously, a formula φ of $\mathcal{L}_{Syn}$ is a ${\Sigma^*}_1$-formula iff any universal quantification that occurs in it is a bounded quantification.

The most important fact about Σ_1-formulae is a theorem, that, in our framework, is as an easy consequence of axiomatic representability. Classically it is derived as a consequence of Kleene's Normal Form Theorem of Computability theory (cf. Th. I.11.7 and related topics in [13]): A set L of natural numbers is recursively enumerable iff it is Ar-representable by some Σ_1-formula (cf. Theorem 16).

Σ_1-formulae and ${\Sigma^*}_1$-formulae, intrinsically related to theories RR and RR^*, allow us to analyze fundamental arithmetical and syntactical notions in terms of logical representability.

Definition 7 *A theory Φ is Σ_1-sound iff for every Σ_1-formula φ of $\mathcal{L}_{Ar}$:*
$\Phi \models \varphi \Rightarrow Ar \models \varphi$.

Definition 8 *A theory Φ is ${\Sigma^*}_1$-sound iff for every ${\Sigma^*}_1$-formula φ of $\mathcal{L}_{Syn}$:*
$\Phi \models \varphi \Rightarrow Syn \models \varphi$.

Theorem 12 **(Σ_1-completeness of RR)** *Let φ a Σ_1-formula with k free variables and let $n_1, \ldots, n_k \in \omega$, then:*

$$Ar \models \varphi(n_1, \ldots, n_k) \Leftrightarrow RR \vdash \varphi(n_1, \ldots, n_k)$$

Proof. By induction on the complexity of φ (see Th. III.6.13 in [13]).

Corollary 13 *If Φ is a Σ_1-sound extension of theory RR, then it is also a Σ_1-complete theory.*

Theorem 14 ($\Sigma^*{}_1$-completeness of RR^*) *Let φ a $\Sigma^*{}_1$-formula with k free variables and let $\alpha_1, \ldots, \alpha_k \in \omega^*$, then:*

$$Syn \models \varphi(\alpha_1, \ldots, \alpha_k) \Leftrightarrow RR^* \vdash \varphi(\alpha_1, \ldots, \alpha_k)$$

Proof. By induction on the complexity of φ, analogously to the proof of the Σ_1-completeness of RR.

Corollary 15 *If Φ is a $\Sigma^*{}_1$-sound extension of theory RR^*, then it is also a $\Sigma^*{}_1$-complete theory.*

Finally, we can answer the question raised at end of the previous section.

Theorem 16 *A language L, i.e. a subset of A^* for some finite alphabet $A \subset \omega$, is r.e. iff it is Syn-representable by a $\Sigma^*{}_1$-formula.*

Theorem 17 *A set L of natural numbers is recursively enumerable iff it is Ar-representable by some Σ_1-formula.*

Corollary 18 (Effective Representability of RR) *Let Φ any Σ_1-sound recursively enumerable extension of RR, then a set L of natural numbers is Φ-representable iff it is recursively enumerable.*

Corollary 19 (Effective Representability of RR^*) *Let Φ any $\Sigma^*{}_1$-sound recursively enumerable extension of RR^*, then a language, i.e. a set of strings over a finite set of numbers, is RR^*-representable if it is recursively enumerable.*

Theorem 20 *All syntactical relations considered in the previous section are RR^*-representable.*

Proof. It is sufficient check that all the formulae there considered are or can be equivalently transformed into $\Sigma^*{}_1$-formulae, and therefore, for the previous $\Sigma^*{}_1$-completeness theorem, they are RR^*-representable. *Q.E.D.*

Another interesting result showing the intrinsic interest of theory RR^*, is that, w.r.t. the representability of languages, its expressive power cannot be increased by finite extensions (a similar results holds also for RR):

Theorem 21 *Any language L representable in a theory Φ that is a finite extension of RR^* is also representable in RR^*.*

For example the following formulae $\varphi(w), \psi(w)$ represent in RR^* the two languages $\{a^n b^n | n \in \omega\}, \{a^n b^n c^n | n \in \omega\}$ considered in the examples 9 and 10:

$$\varphi(w) \equiv \exists xy(w = xy \wedge |x| = |y| \wedge$$
$$\forall uv((u \subseteq x \wedge |u| = 1 \to u = a) \wedge (v \subseteq y \wedge |v| = 1 \to v = b))$$

$$\psi(w) \equiv \exists xyz(w = xyz \wedge |x| = |y| \wedge |x| = |z| \wedge$$
$$\forall uvw((u \subseteq x \wedge |u| = 1 \to u = a) \wedge$$
$$(v \subseteq y \wedge |v| = 1 \to v = b) \wedge$$
$$(w \subseteq z \wedge |w| = 1 \to w = c)))$$

Usually, given a representation of a language L in a finite extension of RR^* it is no so easy to find a representation of L in RR^*. However, it would be possible to define a general, but rarely efficient, algorithm which provides this representation, in a uniform way, by encoding derivations in RR^* by suitable strings.

Theorem 15 is essentially equivalent to an analogous representation theorem formulated in terms of *rudimentary predicates* (cf. [12] Ch. III Th. 12.5, and [14]). The exact equivalence of two theorems relies on the following theorem.

Theorem 22 $|\alpha| = |\beta|$ *is representable in the model* $(\mathbf{N}^*, --, \varpi)$ *by a* ${\Sigma^*}_1$-*formula, where* ϖ *denotes a predicate that holds on atomic strings (i.e. single numbers).*

6 Conclusions

Our results, based on first order representability, provides an interesting understanding of the connection between arithmetics and syntax, and gives insights for extending the approach of ${\Sigma^*}_1$-completeness, to any denumerable model. In fact, consider a first order language including a set of binary predicates B. We can define the notion of ${\Sigma^B}_1$-formula by making B to play the role of $\{\leq, \subseteq\}$ in $\Sigma^*{}_1$-formulae. We say predicates of B to be bounding predicates w.r.t. a model M iff for any $b \in |M|$ and for any $p \in B$ the set $\{a \in |M| \mid p(a,b)\}$ is finite. Under these assumptions, we can develop an intrinsic notion of computability over M, in terms of first order representability and $\Sigma^B{}_1$-formulae, and thus we may search for theories that for M could play the role that RR^* plays for Syn.

Another interesting aspect of logical formalization of formal syntax is that we can represent in the same formal machinery very different syntactical aspects, often specific of different generative, or analytical or algebraic devices present in literature. This fact suggest to use Syn-representations and RR^*-representations to compare and integrate several symbolic formalisms in a unique framework. Moreover, logical representations could indicate significative extensions of the notion of symbolic transformation rule. This aspect is presently under investigation for dealing with formal descriptions of phenomena that are peculiar of *metabolic* processes (cf. [7]).

References

[1] Barwise J. (ed.), Handbook of Mathematical logic, North-Holland, Amsterdam, 1977.

[2] Bell J. L., Machover M., A Course in Mathematical Logic, North-Holland, Amsterdam, 1977.

[3] Crossley J. N. et al., What is Mathematical Logic?, Oxford University Press, London, 1972.

[4] Csuhaj-Varju E., Dassow J., Kelemen J., Paun G., Grammar Systems: A grammatical approach to distribution and cooperation, Gordon and Breach Sciences Publishers, Singapore, 1994.

[5] Dassow J., Paun G., Regulated Rewriting in Formal Language Theory, Springer Verlag, Berlin, 1989.

[6] Gabbay D., Guenthner F.(eds.), Handbook of Philosophical Logic, Vol.I: Elements of Classical Logic, Reidel Publishing company, Dordrecht, 1983.

[7] Manca V., Martino D., From string rewriting to logical metabolic systems, Technical Report 97-01, Dipartimento di Informatica, Università di Pisa, Pisa, 1997.

[8] Paun G. (ed), Artificial Life, Black Sea University Press, Bucharest, 1995.

[9] Rosenberg G., Salomaa A., The Mathematical Theory of L Systems, Academic Press, New York, 1980.

[10] Rosenberg G., Salomaa A.(eds.), Handbook of Language Theory, Springer Verlag, Berlin, 1997.

[11] Salomaa A., Formal Languages, Academic Press, New York, 1973.

[12] Smoryński C., Logical Number Theory, Springer Verlag, Berlin, 1991.

[13] Smullyan R. M., Theory of Formal Systems, Princeton Univ. Press, Princeton, New Jersey, 1961.

[14] Tarski A., Introduction to Logic, Oxford Universiry Press, London, 1965.

[15] Van Dalen D., Logic and Structure, Springer Verlag, Berlin, 1981.