The Logic of FOLsystems: The Structure of set-theoretic FOLsystems

Richard W. Weyhrauch
Ibuki Inc.
rww@ibuki.com

Carolyn Talcott
Stanford University
clt@sail.stanford.edu

Copyright © 1994 by R. Weyhrauch and C. Talcott

We start by assuming we have a set of symbols, *FOLsym*. These are our basic building blocks.

1. Similarity Type

A *similarity type* (sometimes called a *signature*), is a tuple

$$\tau = \langle \star, j, p, q, n \rangle$$

where $\star$ is an *FOLsym*, used as a designated place holder, j is a positive integer, $p = \langle p_1, \ldots, p_k \rangle$, and $q = \langle q_1, \ldots, q_l \rangle$ are sequences of positive integers whose lengths are k and l respectively, and n is a natural number. We call $\star$ the *star symbol* of τ.

Given two similarity types

$$\tau = \langle \star, j, p, q, n \rangle$$

and

$$\tau' = \langle \star', j', p', q', n' \rangle$$

we say τ *is extended by* τ' or τ' *extends* τ and write $\tau \preceq \tau'$ if

0) $j \leq j'$,

1) $k \leq k'$, and for $1 \leq i \leq k$, $p_i = p'_i$

2) $l \leq l'$, and for $1 \leq i \leq l$, $q_i = q'_i$

3) $n \leq n'$.

2. First order languages

We say $\mathcal{L}$ is a *first order language of type* τ if

$$\begin{aligned}\mathcal{L} = \langle\ &\tau,\\ &\langle\langle s_1, v_{1,1}, \ldots, v_{1,m_1}\rangle, \ldots, \langle s_j, v_{j,1}, \ldots, v_{j,m_j}\rangle\rangle,\\ &\langle r_1, \ldots, r_k\rangle,\\ &\langle f_1, \ldots, f_l\rangle,\\ &\langle c_1, \ldots, c_n\rangle\ \rangle\end{aligned}$$

where the m_i are possibly zero, and all the s_i, r_i, f_i, c_i, and $v_{i,i'}$ are distinct (non star) elements of *FOLsym*. By virtue of its appearance in $\mathcal{L}$:

(i) s_i is called a sort symbol and $v_{i,i'}$ is called an individual variable symbol of sort s_i for $1 \leq i \leq j$, $1 \leq i' \leq m_i$;

(ii) r_i is called a relation symbol of p_i arguments for $1 \leq i \leq k$;

(iii) f_i is called a function symbol of q_i arguments for $1 \leq i \leq l$;

(iv) c_i is called an individual constant symbol for $1 \leq i \leq n$.

We call the sort symbol s_1 the *mostgeneral* sort symbol. This sort plays a distinguished role as will be explained in later sections.

By the *symbols* of $\mathcal{L}$ we mean the set

$$\{s_1, \ldots, s_j, v_{1,1}, \ldots v_{j,m_j}, r_1, \ldots, r_k, f_1, \ldots, f_l, c_1, \ldots, c_n\}$$

Terms and *formulae* of $\mathcal{L}$ are built up in the usual way[1] from the individual variable and constant symbols; the sentential constants, `True` and `False`; the sentential connectives, $\wedge$ (and), $\vee$ (or), $\supset$ (implies), $\equiv$ (equivalant), $\neg$ (not); and the quantifiers, $\forall$ (universal), and $\exists$ (existential) (see Prawitz [4]). We say that e is an *expression* of (the language) $\mathcal{L}$, when e is either a term or a formula of $\mathcal{L}$.

[1] Notice that at this point, we are making an assumption about the construction of terms and formulas. In this paper we will not examine this assumption further. However, we wish to point out that it is possible to make other assumptions or to introduce mechanisms which can extend the language.

Given two first order languages

$$
\begin{aligned}
\mathcal{L} = \langle\ & \tau, \\
& \langle\langle s_1, v_{1,1}, \ldots, v_{1,m_1}\rangle, \ldots, \langle s_j, v_{j,1}, \ldots, v_{j,m_j}\rangle\rangle, \\
& \langle r_1, \ldots, r_k\rangle, \\
& \langle f_1, \ldots, f_l\rangle, \\
& \langle c_1, \ldots, c_n\rangle\ \rangle \\
\mathcal{L}' = \langle\ & \tau', \\
& \langle\langle s'_1, v'_{1,1}, \ldots, v'_{1,m'_1}\rangle, \ldots, \langle s'_j, v'_{j',1}, \ldots, v'_{j',m'_{j'}}\rangle\rangle, \\
& \langle r'_1, \ldots, r'_{k'}\rangle, \\
& \langle f'_1, \ldots, f'_{l'}\rangle, \\
& \langle c'_1, \ldots, c'_{n'}\rangle\ \rangle
\end{aligned}
$$

we say $\mathcal{L}$ *is extended by* $\mathcal{L}'$ or $\mathcal{L}'$ *extends* $\mathcal{L}$ and write $\mathcal{L} \preceq \mathcal{L}'$ if $\tau \preceq \tau'$, $m_i \leq m'_i$ for $1 \leq i \leq j$, and corresponding symbols are the same, i.e.:

(i) $s_i = s'_i$ and $v_{i,i'} = v'_{i,i'}$ for $1 \leq i < j$, $1 \leq i' \leq m_i$.

(ii) $r_i = r'_i$ for $1 \leq i \leq k$;

(iii) $f_i = f'_i$ for $1 \leq i \leq l$;

(iv) $c_i = c'_i$ for $1 \leq i < n$;

We have chosen to define a language in terms of two corresponding lists—arities and symbols—rather than as a set of symbols and an associated arity map. This is because similarity type presents the skeleton of the language, with choice of symbols abstracted away. Thus it serves to define both a class of languages and a corresponding class of (partial) models. For the purpose of recognizing the set of well-formed terms and formulas, it is easy to map one form of language presentation to the other.

3. Partial structures

By a *partial structure of type* τ we mean a structure

$$\mathcal{P}s = \langle\ \tau, \langle S_1, \ldots, S_j\rangle, \langle R_1, \ldots, R_k\rangle, \langle F_1, \ldots, F_l\rangle, \langle a_1, \ldots, a_n\rangle\ \rangle$$

where

(i) $S_i = \langle \underline{S}_i, \overline{S}_i\rangle$, $\underline{S}_i \cap \overline{S}_i = \emptyset$, for $1 \leq i \leq j$,

(i.1) $\star \notin \overline{S}_1$,

(i.2) there is some element not in $\underline{S}_i$ for $1 \leq i \leq j$, and

(i.3) $\underline{S}_i \cup \overline{S}_i \subseteq \overline{S}_1$ for $2 \leq i \leq j$,

(ii) $R_i = \langle \underline{R}_i, \overline{R}_i \rangle$, $\underline{R}_i \cap \overline{R}_i = \emptyset$, and $\underline{R}_i \cup \overline{R}_i \subseteq (\overline{S}_1)^{p_i}$, for $1 \leq i \leq k$,

(iii) $F_i : (\overline{S}_1)^{q_i} \mapsto \overline{S}_1 \cup \{\star\}$ for $1 \leq i \leq l$

(iv) $a_i \in \overline{S}_1 \cup \{\star\}$ for $1 \leq i \leq n$.

When $\mathcal{P}s$ is used as an interpretation of a first order language of the same similarity type the elements of $\overline{S}_i$ are definitely elements of the interpretation of the sort symbol s_i, and the elements of $\underline{S}_i$ are definitely not in the interpretation of s_i.[2] The first sort, S_1, (called the mostgeneral sort) is distinguished in that is the domain of the partial structure. We call elements of $\overline{S}_1$ *values* and elements of $\overline{S}_1 \cup \{\star\}$ *answers*.[3] Each relation symbol, r_i, is interpreted similarly, in terms of the p_i-tuples of $\overline{R}_i$ and $\underline{R}_i$. A partial structure does not necessarily determine whether an element belongs to a given sort, or whether a tuple belongs to a given relation. Each F_i determines an answer for each q_i-tuple of values. Unlike the ordinary definition of model we allow $\star$ as the result of a function and in the list of meanings for individual constant symbols. This allows us to construct contexts that contain names for things even though the context itself cannot determine what things they name. The exact technical meaning of $\star$ as it is used in partial structures will be further elaborated below.

We call $\mathcal{P}s$ a *total structure* if

(i) $\overline{S}_i \neq \emptyset$ and $\underline{S}_i \cup \overline{S}_i = \overline{S}_1$ for $1 \leq i \leq j$;

(ii) $\underline{R}_i \cup \overline{R}_i = (\overline{S}_1)^{p_i}$ for $1 \leq i \leq k$;

(iii) $F_i : (\overline{S}_1)^{q_i} \mapsto \overline{S}_1$, for $1 \leq i \leq l$;

(iv) $a_i \in \overline{S}_1$ for $1 \leq i \leq n$.

Given two partial structures

$$\mathcal{P}s = \langle\ \tau, \langle S_1, \ldots, S_j \rangle, \langle R_1, \ldots, R_k \rangle, \langle F_1, \ldots, F_l \rangle, \langle a_1, \ldots, a_n \rangle\ \rangle,$$
$$\mathcal{P}s' = \langle\ \tau', \langle S'_1, \ldots, S'_{j'} \rangle, \langle R'_1, \ldots, R'_{k'} \rangle, \langle F'_1, \ldots, F'_{l'} \rangle, \langle a'_1, \ldots, a'_n \rangle\ \rangle$$

we say $\mathcal{P}s$ *is extended by* $\mathcal{P}s'$ or $\mathcal{P}s'$ *extends* $\mathcal{P}s$ and write $\mathcal{P}s \preceq \mathcal{P}s'$ if

---

[2] This notion of partial relation was introduced by Feferman [1].

[3] Note that any $\overline{S}_i$ could be empty. This should be interpreted as meaning that we simply can't recognize any elements of S_i, not that the intended domain is empty, since $\underline{S}_1$ can not include everything.

(i) $\underline{S}_i \subseteq \underline{S}'_i$, $\overline{S}_i \subseteq \overline{S}'_i$, for $1 \leq i \leq j$,

(ii) $\underline{R}_i \subseteq \underline{R}'_i$ and $\overline{R}_i \subseteq \overline{R}'_i$ for $1 \leq i \leq k$,

(iii) if $\langle v_1, \ldots, v_{q_i} \rangle \in \overline{S}_1$, and $F_i(v_1, \ldots, v_{q_i}) = v \in \overline{S}_1$, then $F'_i(v_1, \ldots, v_{q_i}) = v$ for $1 \leq i \leq l$,

(iv) if $a'_i = \star$ or $a_i \neq \star$, then $a_i = a'_i$ for $1 \leq i \leq n$.

The empty partial structure of similarity type τ, is the structure

$$\mathcal{P}s^{\emptyset}_{\tau} = \langle\ \tau, \langle S_1, \ldots, S_j \rangle, \langle R_1, \ldots, R_k \rangle, \langle F_1, \ldots, F_l \rangle, \langle a_1, \ldots, a_n \rangle\ \rangle$$

where

(i) $S_i = \langle \underline{S}_i, \overline{S}_i \rangle$ with $\underline{S}_i = \overline{S}_i = \emptyset$, for $0 \leq i \leq j$,

(ii) $R_i = \langle \underline{R}_i, \overline{R}_i \rangle$ with $\underline{R}_i = \overline{R}_i = \emptyset$, for $0 \leq i \leq k$,

(ii) F_i the function with empty domain for $1 \leq i \leq l$, and

(iii) $a_i = \star$ for $1 \leq i \leq n$.

The relation $\preceq$ defines a partial order on partial structures. For a fixed similarity type, τ, $\mathcal{P}s^{\emptyset}_{\tau}$ is the least element, and for a fixed S_1 the maximal structures of this type are the total structures, i.e. the usual first-order models of $\mathcal{L}$ with domain $\overline{S}_1$.

4. Facts

A *fact* contains a formula along with its associated dependencies. For the set-theoretic formulation of this note, dependencies can be thought of as lists of formulas. That is, a fact is analogous to a single conclusion sequent $F_1, \ldots, F_k \vdash F$ (cf. [2] p. 82, [4]).

A set-theoretic fact F in a language $\mathcal{L}$ is true in a model $\mathcal{P}s$ of $\mathcal{L}$ if whenever the conjunction of the dependencies of F holds, then the formula of F holds in $\mathcal{P}s$. To say what it means for an `FOL` formula to be true in a model we define a translation that eliminates variables not of the most-general sort, and then eliminates occurrences of the most-general sort symbol. An `FOL` formula is true in a model just if its translation is true in the ordinary semantics of first-order-logic, treating the variables as unsorted. The translation is that of Kleene [3](Chapter XIV. Example 13.) For example if N is a sort symbol of $\mathcal{L}$ other than the most general sort, n is a variable symbol of sort N, x is variable symbol of most general sort, and $\Phi(x)$ expands to $\Phi'(x)$, then $(\forall n)(\Phi(n))$ expands to $(\forall x)(N(x) \supset \Phi'(x))$ and

$(\exists n)(\Phi(n))$ expands to $(\exists x)(N(x) \wedge \Phi'(x))$. Note that the mostgeneral sort symbol can be eliminated from formulas since, by definition, (supposing M is the mostgeneral sort symbol) $\forall x.M(x)$ holds in any model of $\mathcal{L}$.

5. Set-theoretic `FOL` Contexts

A set-theoretic `FOL`context of type τ is a structure with three parts: a language, $\mathcal{L}$, of type τ, some objects the language is talking about (a partial structure of type τ), $\mathcal{P}s$, and a list, $\mathcal{F}$, of facts.

$$\mathcal{C} = \langle \mathcal{L}, \mathcal{P}s, \mathcal{F} \rangle.$$

We say F is a fact of $\mathcal{C}$ if F is an element of $\mathcal{F}$.

6. Values and Satisfaction

Let $\mathcal{C} = \langle \mathcal{L}, \mathcal{P}s, \mathcal{F} \rangle$ be an `FOL`context of type τ. Suppose t is a term of $\mathcal{C}$. If t is the individual constant c_i, then

$$val[\mathcal{C}](c_i) = a_i.$$

Recall that a_i may be $\star$. If t is of the form $f_i(t_1, \ldots, t_{q_i})$, and if $val[\mathcal{C}](t_j) = a_j$ for $1 \leq j \leq q_i$, then

$$val[\mathcal{C}](t) = \begin{cases} F_i(a_1, \ldots, a_{q_i}) & \text{if } \langle a_1, \ldots, a_{q_i} \rangle \in \overline{S}_1 \\ \star & \text{otherwise.} \end{cases}$$

Thus if any of $val[\mathcal{C}](t_1), \ldots, val[\mathcal{C}](t_{q_i})$ are $\star$ then the value of $val[\mathcal{C}](t)$ is also $\star$.

If A is an atomic well-formed formula of the form $r_i(t_1, \ldots, t_{p_i})$, then

$$sat[\mathcal{C}](A) = \begin{cases} \mathit{True} & \text{if } \langle val[\mathcal{C}](t_1), \ldots, val[\mathcal{C}](t_{p_i}) \rangle \in \overline{R}_i \\ \mathit{False} & \text{if } \langle val[\mathcal{C}](t_1), \ldots, val[\mathcal{C}](t_{p_i}) \rangle \in \underline{R}_i \\ \star & \text{otherwise.} \end{cases}$$

From the point of view satisfaction, sort symbols are considered to be unary relation symbols. Since both $\underline{R}_i$ and $\overline{R}_i$ are subsets of $\overline{S}_0^{p_i}$ if any of the

$$val[\mathcal{C}](t_1), \ldots, val[\mathcal{C}](t_{p_i})$$

are $\star$ then the value of $sat[\mathcal{C}](A)$ is also $\star$.

The definition of $sat[\mathcal{C}]$ for the propositional connectives uses the following tables.

p	$\neg p$
t	f
$\star$	$\star$
f	t

$\wedge$	t	$\star$	f
t	t	$\star$	f
$\star$	$\star$	$\star$	f
f	f	f	f

$\vee$	t	$\star$	f
t	t	t	t
$\star$	t	$\star$	$\star$
f	t	$\star$	f

$\supset$	t	$\star$	f
t	t	$\star$	f
$\star$	t	$\star$	$\star$
f	t	t	t

$\equiv$	t	$\star$	f
t	t	$\star$	f
$\star$	$\star$	$\star$	$\star$
f	f	$\star$	t

These tables look like the Kleene strong 3-valued logic [3], but we will provide a different interpretation below. Our interpretation is closer to that found in Feferman's [1]. For us, $\star$ is read as "I don't know". In all cases $sat[\mathcal{C}](w) = \star$ if w is a quantified formula.

We have written $sat[\mathcal{C}](w)$ and $val[\mathcal{C}](t)$ rather than $sat(\mathcal{C}, w)$ and $val(\mathcal{C}, t)$ to point out that val and sat are uniform in $\mathcal{C}$. This uniformity assures that val and sat are computable relative to $\mathcal{C}$. We restrict the domain of $sat[\mathcal{C}]$ and $val[\mathcal{C}]$ to be expressions of $\mathcal{C}$.

A context $\mathcal{C} = \langle \mathcal{L}, \mathcal{P}s, \mathcal{F} \rangle$ is said to be *consistent* (or *satisfiable*) if there is a total structure $\mathcal{P}s'$ where $\mathcal{P}s \preceq \mathcal{P}s'$ and all the facts of $\mathcal{F}$ are are true in $\mathcal{P}s'$. We call such a structure a model of $\mathcal{C}$. Our notion of model is not quite the standard notion, since we allow the language to expand. However, the notion of consistency is not changed if, in the above definition, we require $\mathcal{P}s'$ to have the same similarity type as $\mathcal{P}s$.

7. The meaning of $\star$

A reasonable reading of $\star$ is "I don't know". We carry this interpretation of $\star$ into our definition of val and sat. Because we are considering only *partial* structures, it is reasonable that we should not be able to determine the "meanings" of some expressions of $\mathcal{L}$.

In this interpretation, $\star$ is **not** to be thought of as a third truth value or as an arbitrary value which is taken by functions when they would have otherwise been "partial functions". The correct way of thinking about $\star$ is as follows:

When $val[\mathcal{C}](t) = \star$ for some term t it means that the function val together with the structure $\mathcal{C}$ is simply not complete enough to find a value for t. If $\mathcal{C}$ is consistent, then t has an interpretation that might be found by some stronger *val* or in some extension of $\mathcal{C}$, but these are not the tools we have available in $\mathcal{C}$.

The same applies to *sat*. We simply return $\star$ when we do not have the tools to determine the truth or falsity of some sentence with respect to a partial structure.

The above definition of *sat* may at first seem too weak. There are at least two complaints you might have. One is that a notion of satisfiability that does not decide **any** quantified formulas can not be very useful. To this we can only say: read on and judge later. As a matter of philosophical interest we might realize that the usual finitist position doesn't allow for the infinite searches implicit in deciding these sentences either.

Secondly, you might say that even if we don't allow such infinite searches we might still do better than we have. For example, we might be able to determine the truth of some existential sentences because we may know the interpretations of some constants. Our answer to this is simple—yes, sometimes we can do better, but what we propose will turn out to be enough. More important, is to recognize that this is an instance of the principle of *tolerance*. That is, nothing in our theory depends on the details the choice we have made as to *when* to stop looking for an value. If you want to work harder, to determine the value of a formula, then then chosing *sat* to this is fine. On the other hand, if you want to be lazy and simply return $\star$ for the value of all formulas, that is also an acceptable definition of *sat*.

> *The key requirement is that it not be important whether or not we get a value (in $\mathcal{P}s$) but that that we can stop looking (that is, return $\star$) whenever we choose, without changing the semantics.*

The phrase "stop looking" can not be explained using the description of set-theoretic `FOL`contexts given in this paper. We explain how to do this in `FOL`Note[Pgm] using the restartable computations described in `FOL`Note[Csys].

8. Rules of inference

A function on `FOL`contexts is said to be *consistency preserving* if it maps consistent contexts onto consistent contexts. We call such a consistency preserving map a *rule of inference*. A special case are maps that preserve the set of models. As is the case in ordinary first-order semantics, we call such a map a *valid rule of inference*.

Notice that natural deduction rules of inference used in the setting of `FOL`contexts give rise to consistency preserving maps on classical contexts that leave the language constant, extend the list of facts, and are valid rules of inference. The facts of a context are not deductively closed. `FOL`contexts and rules of inference allow us to distinguish between the notion of *a fact of a context*, namely those facts which actually appear in the context, and *a consequence of a context*, which are those facts which could appear in some successor of a context after the application of some number of validity preserving rules of inference.

At first, it might seem that simply preserving the consistency of a context is a notion that is too weak to be interesting. However, taking this view, many ideas can be formulated in a natural way that are hard to understand when you take the usual view that the only interesting rules are the valid ones. Interesting consistency perserving maps on contexts arises when we look at formalizing the *activity* of reasoning. Consider the following:

- language additions are always consistency preserving;
- introducing explicit definitions changes both language and facts;
- adding skolem functions changes both the language and the facts;
- most proofs of conservative extension results can be reinterpreted as proofs that certain maps on contexts are consistency preserving;
- forgetting facts is another useful operation that cannot be formulated in the traditional setting.

Other examples of consistency preserving operations on contexts are: making interpretations of one theory in another; modeling reasoning as an activity that takes place over time; and realizing various metatheoretic results, such as interpreting the fundamental theorem of projective geometry (the duality of points and lines) as a rule for adding duals of existing facts.

There are many natural interpretations of other mappings. Consider contexts $\mathcal{C}$ whose language only contains individual constants and whose set of facts is empty. Logicians might call these contexts interpretations, computer scientists might call them memories or environments and mathematicians might say they are finite functions. Notice that **any** map from one of these structures to another is consistency preserving.

9. FOL Systems

An FOLsystem, S, is a finite sequence of FOLcontexts, written:

$$\mathcal{S} = [\![\mathcal{C}_1, \ldots, \mathcal{C}_n]\!].$$

For any i, $1 \leq i \leq n$ we say that the context $\mathcal{C}_i$ is in $\mathcal{S}$. An FOLsystem is consistent if all the contexts it contains are consistent. We say ρ is a *consistency preserving map* on FOLsystems if ρ maps consistent FOLsystems to consistent FOLsystems. We will also call these maps *rules of inference*. Let

$$\rho : [\![\mathcal{C}_1, \ldots, \mathcal{C}_n]\!] \mapsto [\![\mathcal{C}'_1, \ldots, \mathcal{C}'_m]\!]$$

be a rule of inference. Sometimes we will want to put additional constraints on these maps, that is, there is some additional properties P that ρ should preserve.

$$P([\![\mathcal{C}_1, \ldots, \mathcal{C}_n]\!]) \supset P(\rho([\![\mathcal{C}_1, \ldots, \mathcal{C}_n]\!]))$$

One obvious idea is to fix a set of rules of inference and then ask what properties of systems are preserved by these rules. An even more important idea is to fix a property P and ask what rules preserve P. Such a P represents a class of systems, and the allowed rules tell you the ways in which a system can evolve. Many problems of philosophy, cognitive science, and artificial intelligence can be formulated as an appropriate choice of a property P and a set of P-preserving rules of inference. Some examples are simple theories (mathematical and common sense), meta theoretic reasoning, states of knowledge that change over time, the use of more than one modal operator at a time, non-monotonic principles of reasoning, etc.

10. References

[1] S. Feferman. Non-extensional type-free theories of partial operations and classifications, i. In J. Diller and G. H. Müller, editors, *Proof theory symposium, Kiel 1974*, number 500 in Lecture notes in mathematics, pages 73–118. Springer, Berlin, 1975.

[2] G. Gentzen. *The Collected Papers of Gerhard Gentzen.* North-Holland, 1969.

[3] S. C. Kleene. *Introduction to metamathematics.* North-Holland, Amsterdam, 1952.

[4] D. Prawitz. *Natural Deduction: A Proof-theoretical Study.* Almquist and Wiksell, 1965.