A General Method for Safely Overwriting Theories in Mechanized Mathematics Systems[*]

William M. Farmer

The MITRE Corporation
202 Burlington Road, A118
Bedford, MA 01730-1420
USA

farmer@mitre.org

21 November 1994

Abstract

We propose a general method for overwriting theories with model conservative extensions in mechanized mathematics systems. Model conservative extensions, which include the definition of new constants and the introduction of new abstract datatypes, are "safe" because they preserve models as well as consistency. The method employs the notions of theory interpretation and theory instantiation. It is illustrated using many-sorted first-order logic, but it works for a variety of underlying logics.

[*]Supported by the MITRE-Sponsored Research program.

1 Introduction

Mathematical reasoning is always performed in some mathematical context consisting of vocabulary and assumptions. The formal counterpart of a context is a *theory* consisting of a formal language plus a set of sentences of the language called *axioms*. (We will denote a theory T by the pair $(\mathcal{L}, \Gamma)$ where $\mathcal{L}$ is the formal language of T and Γ is the set of axioms of T.) An *extension* of a theory T is any theory T' obtained by adding new vocabulary and axioms to T. That is, $T' = (\mathcal{L}', \Gamma')$ is an extension of $T = (\mathcal{L}, \Gamma)$, written $T \leq T'$, if $\mathcal{L}$ is a sublanguage of $\mathcal{L}'$ and $\Gamma \subseteq \Gamma'$. (When we use the word "extension" by itself, without reference to the theory being extended, we will mean a pair (T, T') of theories such that $T \leq T'$.) Theory extension is the process in formal reasoning by which new concepts are created and new structure is introduced.

A *mechanized mathematics system (MMS)* is a computer environment that is intended to support and improve rigorous mathematical activity. MMSs include mechanical theorem provers and systems for specifying and verifying computer software and hardware. Nearly all MMSs provide tools for building and extending theories. An application of an MMS—such as the development of a formal software specification—usually involves the creation of many theory extensions. If each theory extension is represented as a distinct theory, the user of the MMS can easily become overwhelmed by the number of theories that he or she must contend with. The proliferation of theories is often controlled in MMSs by "overwriting" theories with their extensions. (After T is overwritten with T', all references to T automatically become references to T', so T effectively becomes T' in the MMS.) It is not safe, however, to overwrite a theory every time it is extended because an extension may compromise the machinery of the original theory. For instance, an extension may add an axiom that renders the original theory inconsistent.

The danger of overwriting a theory with an "unsafe" extension is avoided in MMSs by restricting overwriting to extensions that are known to preserve the semantics of the original theory. Extensions of this kind are called "conservative" extensions. Actually, there are two distinct notions of a conservative extension. Let $T \leq T'$. T' is a *model conservative extension* of T if every model of T expands to a model of T'.[1] T' is a *consequence conser-*

[1] A precise definition of the "expansion" of a model (in many-sorted first-order logic) is given in Section 3.

vative extension of T if $T' \models A$ implies $T \models A$ whenever A is a sentence of the language of T.[2] In predicate logic, model conservativity is strictly stronger than consequence conservativity, i.e., every model conservative extension is consequence conservative, but there are consequence conservative extensions which are not model conservative (see [3, 22, 20]). Both model and consequence conservativity preserve semantic consistency, i.e., if T' is a model or consequence conservative extension of T and there is a model for T, then there is a model for T'.

Let $T \trianglelefteq T'$ mean that T' is a model conservative extension of T. We shall assume that it is safe to overwrite a theory T with another theory T' iff $T \trianglelefteq T'$. This is a reasonable assumption for two reasons. First, model conservativity is sufficiently strong: both models and consequences are preserved. Second, model conservativity is sufficiently weak: in practice, nearly all MMSs restrict overwriting to model conservative extensions (if they restrict overwriting at all).

For the rest of this section, let $T = (\mathcal{L}, \Gamma)$ and $T' = (\mathcal{L}', \Gamma')$ be theories in some (first-order or higher-order) predicate logic $\mathbf{L}$.

The most important kind of model conservative extension in predicate logic formalizes the practice of giving names to objects that are known to exist. Assume $T \leq T'$ such that:

- $\mathcal{L}'$ is $\mathcal{L}$ plus some new constants $c_1, \ldots, c_m$.
- $\Gamma' = \Gamma \cup \{A(c_1, \ldots, c_m)\}$, where $A(c_1, \ldots, c_m)$ contains $c_1, \ldots, c_m$.

T' is a *nominal extension* of T if

$$T' \models \exists x_1 \ldots \exists x_m \,.\, A(x_1, \ldots, x_m).^3$$

It is easy to see that any nominal extension is model conservative.

T' is a *definitional extension* of T if

$$T' \models \exists! x_1 \ldots \exists! x_m \,.\, A(x_1, \ldots, x_m).^4$$

Definitional extensions are extremely safe: when T' is a definitional extension of T, the following two properties hold:

[2] $T \models A$ means A is valid in every model for T.

[3] Or an equivalent sentence; for example, if $m = 1$ and c_1 is an n-ary function symbol in first-order logic, then the sentence would have the form $\forall x_1 \ldots \forall x_n \exists y \,.\, A'$ since $\exists x_1 \,.\, A(x_1)$ would not be a first-order sentence.

[4] $\exists! x \,.\, A(x)$ means $\exists x \,.\, (A(x) \wedge \forall y \,.\, (A(y) \supset x = y))$.

- T' is a *strong model conservative extension* of T, that is, every model for T expands to a *unique* model for T' (up to isomorphism).

- The constants introduced by T' can be eliminated: for every formula B' of $\mathcal{L}'$, there is some formula B of $\mathcal{L}$ such that $T' \models B' \equiv B$.

Another important kind of model conservative extension, which is employed in many-sorted predicate logic, introduces a new structured collection of objects—what is called an "algebra" in mathematics and an "abstract datatype" in computer science. Let $T \leq T'$. T' is a *datatype extension* of T if $\mathcal{L}'$ is $\mathcal{L}$ plus a new sort α and some new constants $c_1, \ldots, c_m$. α is intended to denote a nonempty set $\mathcal{O}$ of objects and $c_1, \ldots, c_m$ are intended to denote members of $\mathcal{O}$ and operations that involve members of $\mathcal{O}$. Datatype extensions are used extensively in formal reasoning and, in practice, are usually model conservative.

Define a *model conservative extension type (mce-type)* to be a pair (Δ, θ) where Δ is a set of extensions and θ is a function from Δ to sentences such that, for all $(T, T') \in \Delta$, if $T \models \theta(T, T')$, then $T \trianglelefteq T'$. Let $\tau = (\Delta, \theta)$ be an mce-type. The *domain* of τ, written $\mathrm{dom}(\tau)$, is the set $\{(T, T') \in \Delta : T \models \theta(T, T')\}$. An mce-type τ is *definitional* if each extension in $\mathrm{dom}(\tau)$ is definitional. An MMS **S** *supports* τ if it includes a procedure that works more or less as follows. Suppose that a user of **S** would like to overwrite a theory T with a theory T'. The user calls the procedure with arguments T and T'. The procedure first checks whether $(T, T') \in \Delta$. If so, the procedure tries to verify that $T \models \theta(T, T')$. If it is successful, it overwrites T with T'. If $(T, T') \notin \Delta$ or if it cannot verify that $T \models \theta(T, T')$, T is not overwritten with T'.

A typical MMS supports just a small set of mce-types. Most often this set contains only definitional mce-types and is fixed and cannot be extended by the user. A few MMSs support mce-types that are not definitional, including m-EVES [4], EVES [5], and HOL [14].[5] For almost all MMSs, including those that support nondefinitional mce-types, there are many model conservative extensions which are not in the domain of any of the supported mce-types.

Example 1.1 (Type of direct definitional extensions) T' is a *direct definitional extension* of T if $\mathcal{L}'$ is $\mathcal{L}$ plus a new constant c and $\Gamma' = \Gamma \cup \{c = E\}$, where E is a closed expression of $\mathcal{L}$. Since $\exists ! x \,.\, x = E$ is universally

[5]NQTHM [1], the so-called Boyer-Moore theorem prover, allows nondefinitional, record-like abstract datatypes called "shells" to be added to a theory. However, the resulting extensions are consequence, but not model, conservative.

valid in standard predicate logic for every expression E, every direct definitional extension is definitional and hence strongly model conservative. Define $\tau = (\Delta, \theta)$ to be the mce-type where Δ is the set of direct definitional extensions of $\mathbf{L}$ and $\theta(T, T') = \mathsf{T}_{\mathcal{L}}$, some universally valid sentence of $\mathcal{L}$. Notice that, as a result of Δ being such a *restricted* set of extensions, it is trivial to verify that a member of Δ is a member of $\mathrm{dom}(\tau)$ (since $\mathrm{dom}(\tau) = \Delta$). This mce-type, the *type of direct definitional extensions*, is supported in most MMSs in which theories may be overwritten. $\Box$

Example 1.2 (Type of nominal extensions) Define (Δ, θ) to be the mce-type where Δ is the set of nominal extensions of $\mathbf{L}$ and

$$\theta(T, T') = \exists x_1 \ldots \exists x_m \,.\, A(x_1, \ldots, x_m),$$

where $A(c_1, \ldots, c_m)$ is the new axiom of T'. Notice that, as a result of Δ being such an *unrestricted* set of extensions, it can be very difficult to verify that a member of Δ is a member of $\mathrm{dom}(\tau)$. This mce-type, the *type of nominal extensions*, is supported by the HOL theorem proving environment [14]. $\Box$

This paper describes a general method for overwriting theories with model conservative extensions in MMSs. The method allows the user to add new mce-types to the set of mce-types supported by an MMS. It employs the notions of theory interpretation and theory instantiation. An *interpretation* of T in T' is a translation from the expressions of T to the expressions of T' which preserves the validity of sentences (see [6, 8, 18]). Given $T \leq T'$, an *instance* of T' via an interpretation Φ of T in a theory U is the result of using Φ to "instantiate" T in T' with U (see [2, 13]).

The following are the basic ingredients of the method. A user-defined mce-type τ is represented as a model conservative extension (T, T') (where T and T' are generally as abstract as possible). $T \trianglelefteq T'$ must be shown before τ can be added to the set of supported mce-types. By the Model Conservative Verification Theorem (see Section 4), the user can show $T \trianglelefteq T'$ by exhibiting an interpretation of T' in T that fixes T. $(U, U') \in \mathrm{dom}(\tau)$ if U' is an instance of T' via an interpretation of T in U. By the Model Conservative Instantiation Theorem (see Section 5), every such instance of T' is a model conservative extension of U since $T \trianglelefteq T'$. To overwrite U with U', the user simply gives the MMS the appropriate interpretation Φ of T in U, then the MMS automatically constructs U' from T' and Φ, and finally the MMS overwrites U with U'.

For the sake of clarity, the method is illustrated using many-sorted first-order logic, but it works for a variety of underlying logics, especially highly expressive logics such as simple type theory. (The method is partially implemented in the IMPS Interactive Mathematical Proof System [9, 11], which is based on the logic LUTINS [7, 8, 15]—a version of simple type theory with partial functions and subtypes.)

The bulk of the paper lays the logical foundation for the method. Section 2 presents a version of many-sorted first-order logic called **MS**. Section 3 discusses theory extension in **MS**. And Section 4 and Section 5 defines theory interpretation and theory instantiation for **MS**, respectively. Then the method itself is described in Section 6. The paper ends with a brief conclusion.

2 Many-sorted first-order logic

This section presents a version of many-sorted first-order logic (with equality) called **MS**. We begin by defining $\Omega(S)$, which is intended to denote the set of sorts built from a set S of base sorts.

Let S be a set of symbols with $* \in S$. $S^- = S \setminus \{*\}$ and $\Omega(S)$ is the set defined inductively by:

(1) $S \subseteq \Omega(S)$.

(2) If $\alpha_1, \ldots, \alpha_n \in S^-$ and $\beta \in S$ $(n \geq 1)$, then $[\alpha_1, \ldots, \alpha_n, \beta] \in \Omega(S)$.

An *S-tagged symbol* is a symbol tagged with a member of $\Omega(S)$. A tagged symbol whose symbol is a and whose tag is α is written a_α. A tagged symbol a_α, where $a = b_i$, will be written as b^i_α (instead of as $(b_i)_\alpha$). Two tagged symbols a_α and b_β are distinct if $a \neq b$.

A *language* of **MS** is a tuple $(\mathcal{B}, \mathcal{V}, \mathcal{C})$ such that:

(1) $\mathcal{B}$ is finite set of symbols called *base sorts* such that $* \in \mathcal{B}$ and $\mathcal{B}^-$ is nonempty. ($*$ is the sort of truth values.)

(2) $\mathcal{V}$ and $\mathcal{C}$ are disjoint countable sets of pairwise distinct $\mathcal{B}$-tagged symbols whose members are called *variables* and *constants*, respectively.

(3) For each $x_\alpha \in \mathcal{V}$, $\alpha \in \mathcal{B}^-$.

(4) For each $\alpha \in \mathcal{B}^-$, there is an infinite subset $\mathcal{V}_\alpha$ of $\mathcal{V}$ such that $x_\beta \in \mathcal{V}_\alpha$ iff $\beta = \alpha$.

c_α is an *individual constant* if $\alpha \in \mathcal{B}$, a *function constant* if $\alpha = [\alpha_1, \ldots, \alpha_n, \beta]$ with $\beta \neq *$, and a *predicate constant* if $\alpha = [\alpha_1, \ldots, \alpha_n, *]$.

For the remainder of this section, let $\mathcal{L} = (\mathcal{B}, \mathcal{V}, \mathcal{C})$ be a language of **MS**. When there is no possibility of confusion, the components of a language $\mathcal{L}_i$ will be denoted by $\mathcal{B}_i, \mathcal{V}_i, \mathcal{C}_i$. A language $\mathcal{L}_1$ is a *sublanguage* of a language $\mathcal{L}_2$, written $\mathcal{L}_1 \leq \mathcal{L}_2$, if $\mathcal{B}_1 \subseteq \mathcal{B}_2$, $\mathcal{V}_1 \subseteq \mathcal{V}_2$, and $\mathcal{C}_1 \subseteq \mathcal{C}_2$.

A *sort* of $\mathcal{L}$ is a member of $\Omega(\mathcal{B})$. An *expression of $\mathcal{L}$ of sort $\alpha \in \mathcal{B}$* is defined inductively by:

(1) Each $x_\alpha \in \mathcal{V}$ and individual constant $c_\alpha \in \mathcal{C}$ is an expression of sort α.

(2) If $c_{[\alpha_1,\ldots,\alpha_n,\beta]} \in \mathcal{C}$ and $E_1, \ldots, E_n$ are expressions of sort $\alpha_1, \ldots, \alpha_n$, then $c_{[\alpha_1,\ldots,\alpha_n,\beta]}(E_1, \ldots, E_n)$ is an expression of sort β.

(3) If E_1 and E_2 are expressions of the same sort, $(E_1 = E_2)$ is an expression of sort $*$.

(4) If E_1 and E_2 are expressions of sort $*$, then $\neg E_1$ and $(E_1 \Box E_2)$ are expressions of sort $*$ for $\Box \in \{\wedge, \vee, \supset, \equiv\}$.

(5) If $x_\alpha \in \mathcal{V}$ and E is an expression of sort $*$, then $\forall x_\alpha \,.\, E$ and $\exists x_\alpha \,.\, E$ are expressions of sort $*$.

A *term of $\mathcal{L}$ of sort α* is an expression of $\mathcal{L}$ of sort $\alpha \in \mathcal{B}^-$, and a *formula* of $\mathcal{L}$ is an expression of $\mathcal{L}$ of sort $*$. The set of expressions of $\mathcal{L}$ [respectively, $\mathcal{L}_i$] is denoted by $\mathcal{E}$ [respectively, $\mathcal{E}_i$]. "Free variable", "closed expression", and similar notions are defined in the obvious way. A *sentence* is a closed formula. Parentheses in expressions may be suppressed when meaning is not lost.

A *λ-expression of $\mathcal{L}$ of sort* $[\alpha_1, \ldots, \alpha_n, \beta]$ has the form

$$\lambda\{x^1_{\alpha_1}, \ldots, x^n_{\alpha_n} \,.\, E\}$$

where $x^1_{\alpha_1}, \ldots, x^n_{\alpha_n}$ are distinct members of $\mathcal{V}$ and E is an expression of $\mathcal{L}$ of sort β. Given a λ-expression $\lambda\{x^1_{\alpha_1}, \ldots, x^n_{\alpha_n} \,.\, E\}$ and terms $t_1, \ldots, t_n$ of sort $\alpha_1, \ldots, \alpha_n$, $\lambda\{x^1_{\alpha_1}, \ldots, x^n_{\alpha_n} \,.\, E\}(t_1, \ldots, t_n)$ denotes the result of simultaneously substituting t_i for all free occurrences of $x^i_{\alpha_i}$ in E, for all i with $1 \leq i \leq n$ (which is an expression of the sort of E). The set of λ-expressions of $\mathcal{L}$ [$\mathcal{L}_i$] is denoted by Λ [Λ_i].

A *frame* for $\mathcal{B}$ is a set $\{\mathcal{D}_\alpha : \alpha \in \mathcal{B}\}$ of nonempty domains (sets) where $D_* = \{\mathrm{T}, \mathrm{F}\}$ is the domain of truth values. A *model* for $\mathcal{L}$ is a pair $\mathcal{M} =$

$(\{\mathcal{D}_\alpha : \alpha \in \mathcal{B}\}, I)$ where $\{\mathcal{D}_\alpha : \alpha \in \mathcal{B}\}$ is a frame for $\mathcal{B}$ and I is a function defined on $\mathcal{C}$ such that:

(1) If $c_\alpha \in \mathcal{C}$ is an individual constant, $I(c_\alpha) \in \mathcal{D}_\alpha$.

(2) If $c_{[\alpha_1,\ldots,\alpha_n,\beta]} \in \mathcal{C}$ is a function or predicate constant, then $I(c_{[\alpha_1,\ldots,\alpha_n,\beta]}) \in \mathcal{D}_{\alpha_1} \times \cdots \times \mathcal{D}_{\alpha_n} \to \mathcal{D}_\beta$.

A $\mathcal{V}$-*assignment* into $\mathcal{M}$ is a function which maps each $x_\alpha \in \mathcal{V}$ to an element of $\mathcal{D}_\alpha$. There is a binary valuation function $V = V^{\mathcal{M}}$—defined in the obvious way—such that, for all $\mathcal{V}$-assignments φ into $\mathcal{M}$ and all $E \in \mathcal{E} \cup \Lambda$, the following statements hold:

(1) If E is an expression of $\mathcal{L}$ of sort α, then $V_\varphi(E) \in \mathcal{D}_\alpha$.

(2) If E is a λ-expression of $\mathcal{L}$ of sort $[\alpha_1, \ldots, \alpha_n, \beta]$, then $V_\varphi(E) \in \mathcal{D}_{\alpha_1} \times \cdots \times \mathcal{D}_{\alpha_n} \to \mathcal{D}_\beta$.

$V_\varphi(E)$ is the *value* of E in $\mathcal{M}$ with respect to φ. A formula A of $\mathcal{L}$ is *valid in* $\mathcal{M}$ if $V_\varphi(A) = \mathrm{T}$ for every $\mathcal{V}$-assignment φ into $\mathcal{M}$. For a closed $E \in \mathcal{E} \cup \Lambda$, $V_\varphi(E)$ does not depend on φ and thus $V(E)$ is meaningful.

A *theory* of **MS** is a pair $T = (\mathcal{L}, \Gamma)$ where $\mathcal{L}$ is a language of **MS** and Γ is a set of sentences of $\mathcal{L}$. A *model* for T is a model $\mathcal{M}$ for $\mathcal{L}$ such that every member of Γ is valid in $\mathcal{M}$. T is *satisfiable (or semantically consistent)* if there is some model for T. A formula A is a *(semantic) theorem* of T, written $T \models A$, if A is valid in every model for T. T, T', etc. denote theories of **MS**.

3 Theory extension

Let $T_i = (\mathcal{L}_i, \Gamma_i)$ for $i = 1, 2$. T_2 is an *extension* of T_1 (and T_1 is a *subtheory* of T_2), written $T_1 \le T_2$, if $\mathcal{L}_1 \le \mathcal{L}_2$ and $\Gamma_1 \subseteq \Gamma_2$.

Example 3.1 (Idempotent function constant) Let $S^1 = (\mathcal{L}, \emptyset)$, where

$$\mathcal{L} = (\mathcal{B}, \mathcal{V}, \emptyset) = (\{\alpha, *\}, \{x^i_\alpha : i \in \omega\}, \emptyset),$$

be a theory of one abstract sort, and let

$$I = ((\mathcal{B}, \mathcal{V}, \{f_{[\alpha,\alpha]}\}), \{\forall x_\alpha \,.\, f_{[\alpha,\alpha]}(f_{[\alpha,\alpha]}(x_\alpha)) = f_{[\alpha,\alpha]}(x_\alpha)\})$$

be a theory of an abstract idempotent function constant. Obviously, $S^1 \le I$. $\square$

Example 3.2 (Cartesian product) Let $S^2 = (\mathcal{L}, \emptyset)$, where

$$\mathcal{L} = (\mathcal{B}, \mathcal{V}, \emptyset) = (\{\alpha_1, \alpha_2, *\}, \{x^i_{\alpha_1}, y^i_{\alpha_2} : i \in \omega\}, \emptyset),$$

be a theory of two abstract sorts, and let $P = (\mathcal{L}', \Gamma)$, where

$$\mathcal{L}' = (\mathcal{B} \cup \{\gamma\}, \mathcal{V} \cup \{z^i_\gamma : i \in \omega\}, \{p_{[\alpha_1,\alpha_2,\gamma]}, \pi^1_{[\gamma,\alpha_1]}, \pi^2_{[\gamma,\alpha_2]}\}),$$

be a theory of an abstract cartesian product. ($p_{[\alpha_1,\alpha_2,\gamma]}$ is the constructor, and $\pi^1_{[\gamma,\alpha_1]}, \pi^2_{[\gamma,\alpha_2]}$ are the selectors.) Obviously, $S^2 \leq P$. $\Box$

Example 3.3 (Vector space) Let $F = (\mathcal{L}, \Gamma)$, where

$$\mathcal{L} = (\mathcal{B}, \mathcal{V}, \mathcal{C}) = (\{\mathbf{f}, *\}, \{x^i_{\mathbf{f}} : i \in \omega\}, \{0_{\mathbf{f}}, 1_{\mathbf{f}}, +_{[\mathbf{f},\mathbf{f},\mathbf{f}]}, *_{[\mathbf{f},\mathbf{f},\mathbf{f}]}\}),$$

be a theory of an abstract field, and let $V = (\mathcal{L}', \Gamma \cup \Gamma')$, where

$$\mathcal{L}' = (\mathcal{B} \cup \{\mathbf{v}\}, \mathcal{V} \cup \{y^i_{\mathbf{v}} : i \in \omega\}, \mathcal{C} \cup \{0_{\mathbf{v}}, +_{[\mathbf{v},\mathbf{v},\mathbf{v}]}, *_{[\mathbf{f},\mathbf{v},\mathbf{v}]}\}),$$

be a theory of an abstract vector space. Obviously, $F \leq V$. $\Box$

If $\mathcal{L}_1 \leq \mathcal{L}_2$ and $\mathcal{M}_i = (\{\mathcal{D}^i_\alpha : \alpha \in \mathcal{B}_i\}, I_i)$ is a model for $\mathcal{L}_i$ for $i = 1, 2$, then $\mathcal{M}_2$ is an *expansion* of $\mathcal{M}_1$ to $\mathcal{L}_2$ (and $\mathcal{M}_1$ is the *reduct* of $\mathcal{M}_2$ to $\mathcal{L}_1$) provided $\mathcal{D}^1_\alpha = \mathcal{D}^2_\alpha$ for all $\alpha \in \mathcal{B}_1$ and I_1 is a subfunction of I_2. T_2 is a *model conservative extension* of T_1, written $T_1 \trianglelefteq T_2$, if $T_1 \leq T_2$ and, for every model $\mathcal{M}_1$ for T_1, there is a model for T_2 which is an expansion of $\mathcal{M}_1$. We will see in the next section that the extensions in the previous three examples are model conservative.

The following two basic lemmas about model conservative extensions are very easy to prove:

Lemma 3.4 (Transitivity) *If $T_1 \trianglelefteq T_2 \trianglelefteq T_3$, then $T_1 \trianglelefteq T_3$.*

Lemma 3.5 (Reductivity) *If $T_1 \leq T_2 \leq T_3$ and $T_1 \trianglelefteq T_3$, then $T_1 \trianglelefteq T_2$.*

The next example shows that the other possible conclusion of the previous lemma does not hold.

Example 3.6 Let

$$\mathcal{L} = (\mathcal{B}, \mathcal{V}, \emptyset) = (\{\alpha, *\}, \{x^i_\alpha : i \in \omega\}, \emptyset)$$

and

$$\mathcal{L}' = (\mathcal{B}, \mathcal{V}, \{a_\alpha, b_\alpha\}).$$

Define $T_1 = (\mathcal{L}, \emptyset)$, $T_2 = (\mathcal{L}', \emptyset)$, and $T_3 = (\mathcal{L}', \{a_\alpha = b_\alpha\})$. Obviously, $T_1 \leq T_2 \leq T_3$, $T_1 \trianglelefteq T_2$, and $T_1 \trianglelefteq T_3$ hold, but $T_2 \trianglelefteq T_3$ does not hold. $\Box$

4 Theory interpretation

This section presents the concept of an interpretation of one **MS** theory in another. Our approach adapts a combination of the approaches in [6, 17, 18] for many-sorted first-order logic. (See also [8, 16].)

An interpretation of a theory is determined by how the primitive symbols of the theory (i.e., the base sorts, variables, and constants of the theory) are interpreted. In the definition given here, base sorts are interpreted by sorts or closed λ-expressions of sort $[\alpha, *]$ with $\alpha \in \mathcal{B}^-$ (i.e., unary predicates); variables are interpreted by variables; and constants are interpreted by constants, expressions, or λ-expressions. The set of closed λ-expressions of $\mathcal{L}$ $[\mathcal{L}_i]$ of sort $[\alpha, *]$ with $\alpha \in \mathcal{B}^-$ is denoted by $\mathcal{U}$ $[\mathcal{U}_i]$.

Let $T_i = (\mathcal{L}_i, \Gamma_i)$ for $i = 1, 2$. Given a function $\mu : \mathcal{B}_1 \rightarrow \mathcal{B}_2 \cup \mathcal{U}_2$ and $\alpha \in \mathcal{B}_1$, $\mu[\alpha]$ denotes the member of $\mathcal{B}_2$ defined by

$$\mu[\alpha] = \begin{cases} \mu(\alpha) & \text{if } \mu(\alpha) \in \mathcal{B}_2 \\ \beta & \text{if } \mu(\alpha) = \lambda\{x_\beta \,.\, E\} \in \mathcal{U}_2 \end{cases}$$

A *translation* Φ *from* T_1 *to* T_2, written $\Phi : T_1 \rightarrow T_2$, is a pair (μ, ν) where $\mu : \mathcal{B}_1 \rightarrow \mathcal{B}_2 \cup \mathcal{U}_2$ and $\nu : \mathcal{V}_1 \cup \mathcal{C}_1 \rightarrow \mathcal{C}_2 \cup \mathcal{E}_2 \cup \Lambda_2$ such that:

(1) $\mu(\alpha) = *$ iff $\alpha = *$.

(2) For all $x_\alpha \in \mathcal{V}_1$, $\nu(x_\alpha)$ is a variable of sort $\mu[\alpha]$.

(3) ν is injective on $\mathcal{V}_1$.

(4) If $c_\alpha \in \mathcal{C}_1$ is an individual constant, then $\nu(c_\alpha)$ is a closed expression of sort $\mu[\alpha]$.

(5) If $c_{[\alpha_1,\ldots,\alpha_n,\beta]} \in \mathcal{C}_1$ is a function or predicate constant, then $\nu(c_{[\alpha_1,\ldots,\alpha_n,\beta]})$ is a constant or closed λ-expression of sort $[\mu[\alpha_1], \ldots, \mu[\alpha_n], \mu[\beta]]$.

T_1 and T_2 are called the *source* and *target* theories of Φ, respectively.

Given $E \in \mathcal{E}_1$, $\Phi(E)$ denotes the member of $\mathcal{E}_2$ defined inductively by:

(1) $\Phi(a_\alpha) = \nu(a_\alpha)$, where a_α is a variable or individual constant.

(2) $\Phi(c_{[\alpha_1,\ldots,\alpha_n,\beta]}(t_1, \ldots, t_n)) = \nu(c_{[\alpha_1,\ldots,\alpha_n,\beta]})(\Phi(t_1), \ldots, \Phi(t_n))$.

(3) $\Phi(\neg A) = \neg\Phi(A)$.

(4) $\Phi(A \mathbin{\Box} B) = (\Phi(A) \mathbin{\Box} \Phi(B))$, where $\Box \in \{=, \wedge, \vee, \supset, \equiv\}$.

(5) $\Phi(\forall x_\alpha \,.\, A) = \begin{cases} \forall \nu(x_\alpha) \,.\, \Phi(A) & \text{if } \mu(\alpha) \in \mathcal{B}_2 \\ \forall \nu(x_\alpha) \,.\, (\mu(\alpha)(x_\alpha) \supset \Phi(A)) & \text{if } \mu(\alpha) \in \mathcal{U}_2 \end{cases}$

(6) $\Phi(\exists x_\alpha \,.\, A) = \begin{cases} \exists \nu(x_\alpha) \,.\, \Phi(A) & \text{if } \mu(\alpha) \in \mathcal{B}_2 \\ \exists \nu(x_\alpha) \,.\, (\mu(\alpha)(x_\alpha) \wedge \Phi(A)) & \text{if } \mu(\alpha) \in \mathcal{U}_2 \end{cases}$

If $E \in \mathcal{E}_1$ is of sort α, then $\Phi(E)$ is of sort $\mu[\alpha]$.

An *obligation* of Φ is any sentence $\Phi(A)$ where A is one of the following theorems of T_1:

(1) An axiom of T_1.

(2) $\exists x_\alpha \,.\, (x_\alpha = x_\alpha)$ where $\alpha \in \mathcal{B}_1^-$.

(3) $\exists x_\alpha \,.\, (c_\alpha = x_\alpha)$ where c_α is an individual constant in $\mathcal{C}_1$.

(4) $\forall x^1_{\alpha_1} \,.\, \forall x^2_{\alpha_2} \ldots \forall x^n_{\alpha_n} \,.\, \exists y_\beta \,.\, (c_{[\alpha_1,\ldots,\alpha_n,\beta]}(x^1_{\alpha_1} \cdots x^n_{\alpha_n}) = y_\beta)$ where $c_{[\alpha_1,\ldots,\alpha_n,\beta]}$ is a function constant in $\mathcal{C}_1$.

The four kinds of obligations are called, in order, *axiom*, *sort nonemptiness*, *individual constant sort*, and *function constant sort* obligations. Note: The last three kinds of obligations are trivial theorems of T_2 unless there are sorts in A which are mapped into $\mathcal{U}_2$ by μ.

Let $\Phi = (\mu, \nu) : T_1 \rightarrow T_2$ and $\Phi' = (\mu', \nu') : T_1' \rightarrow T_2'$ be translations. Φ' is an *extension* of Φ, written $\Phi \leq \Phi'$, if $T_i \leq T_i'$ for $i = 1, 2$, μ is a subfunction of μ', and ν is a subfunction of ν'.

A translation $\Phi : T_1 \rightarrow T_2$ is an *interpretation of* T_1 *in* T_2, written $\Phi : T_1 \hookrightarrow T_2$, if, for each theorem A of T_1, $\Phi(A)$ is a theorem of T_2. We shall prove shortly the Interpretation Theorem for **MS** which gives a sufficient condition for a translation to be an interpretation.

Fix a translation $\Phi = (\mu, \nu) : T_1 \rightarrow T_2$, and let $\mathcal{M}_2 = (\{\mathcal{D}^2_\alpha : \alpha \in \mathcal{B}_2\}, I_2)$ be a model for T_2. $\mathcal{M}_1 = (\{\mathcal{D}^1_\alpha : \alpha \in \mathcal{B}_1\}, I_1)$ is defined as follows. For $\alpha \in \mathcal{B}_1$, if $\mu(\alpha) \in \mathcal{B}_2$, then $\mathcal{D}^1_\alpha = \mathcal{D}^2_{\mu(\alpha)}$; otherwise

$$\mathcal{D}^1_\alpha = \{a \in \mathcal{D}^2_{\mu[\alpha]} : V^{\mathcal{M}_2}(\mu(\alpha))(a) = \mathrm{T}\}.$$

Notice that $\mathcal{D}^1_\alpha \subseteq \mathcal{D}^2_{\mu[\alpha]}$ for all $\alpha \in \mathcal{B}_1$. I_1 is defined by:

(1) For an individual constant $c_\alpha \in \mathcal{C}_1$, $I_1(c_\alpha) = V^{\mathcal{M}_2}(\nu(c_\alpha))$.

(2) For a function or predicate constant $c_{[\alpha_1,\ldots,\alpha_n,\beta]} \in \mathcal{C}_1$, the function $I_1(c_{[\alpha_1,\ldots,\alpha_n,\beta]})$ is the restriction of $V^{\mathcal{M}_2}(\nu(c_{[\alpha_1,\ldots,\alpha_n,\beta]})$ to $\mathcal{D}^1_{\alpha_1} \times \cdots \times \mathcal{D}^1_{\alpha_n}$.

The proofs of the next two lemmas are straightforward:

Lemma 4.1 *Suppose each obligation of* Φ *is valid in* $\mathcal{M}_2$. *Then* $\mathcal{M}_1$ *is a model for* $\mathcal{L}_1$.

Lemma 4.2 *Suppose each obligation of* Φ *is valid in* $\mathcal{M}_2$. *For all* $\mathcal{V}_1$-*assignments* φ *into* $\mathcal{M}_1$ *and all expressions* $E \in \mathcal{E}_1$,

$$V^{\mathcal{M}_1}_{\varphi}(E) = V^{\mathcal{M}_2}_{\Phi(\varphi)}(\Phi(E)),$$

where $\Phi(\varphi)$ *is any* $\mathcal{V}_2$-*assignment into* $\mathcal{M}_2$ *such that, for all* $x_\alpha \in \mathcal{V}_1$, $\Phi(\varphi)(\Phi(x_\alpha)) = \varphi(x_\alpha)$. *Furthermore, for all formulas* $A \in \mathcal{E}_1$, A *is valid in* $\mathcal{M}_1$ *iff* $\Phi(A)$ *is valid in* $\mathcal{M}_2$.

Theorem 4.3 (Relative Satisfiability) *Suppose* $\Phi : T_1 \hookrightarrow T_2$. *Then* T_1 *is satisfiable if* T_2 *is satisfiable.*

Proof Let $\mathcal{M}_2$ be a model for T_2 and $\mathcal{M}_1$ be defined as above. Since Φ is an interpretation, $\Phi(A)$ is valid in $\mathcal{M}_2$ for each theorem A of T_1. This implies that each obligation of Φ is valid in $\mathcal{M}_2$. Hence, by Lemma 4.1, $\mathcal{M}_1$ is a model for $\mathcal{L}_1$. Let $B \in \Gamma_1$. B is a theorem of T_1, so $\Phi(B)$ is valid in $\mathcal{M}_2$; hence, by Lemma 4.2, B is valid in $\mathcal{M}_1$. Therefore, each axiom of T_1 is valid in $\mathcal{M}_1$, and so $\mathcal{M}_1$ is a model for T_1. $\Box$

Theorem 4.4 (Interpretation Theorem) *Suppose* $\Phi : T_1 \to T_2$. *Then* Φ *is an interpretation if each of its obligations is a theorem of* T_2.

Proof Suppose each obligation of Φ is a theorem of T_2. Choose a theorem A of T_1. We may assume that there are models for T_2, since otherwise Φ would be trivially an interpretation. Let $\mathcal{M}_2$ be a model for T_2, and let $\mathcal{M}_1$ be defined as above. Obviously, each obligation of Φ is valid in $\mathcal{M}_2$, and so by the proof of Relative Satisfiability, $\mathcal{M}_1$ is a model for T_1. Then A is valid in $\mathcal{M}_1$, and by Lemma 4.2, $\Phi(A)$ is valid in $\mathcal{M}_2$. Therefore, we can conclude that $\Phi(A)$ is a theorem of T_2 and Φ is an interpretation. $\Box$

$\Phi : T_1 \to T_2$ *fixes* a theory $U = ((\mathcal{B}, \mathcal{V}, \mathcal{C}), \Gamma)$ if $U \leq T_1$, $U \leq T_2$, μ is the identity function on $\mathcal{B}$, and ν is the identity function on $\mathcal{C}$ (but ν need not be the identity function on $\mathcal{V}$).

Theorem 4.5 (Model Conservative Verification Theorem) *Suppose $T_1 \leq T_2$. Then $T_1 \trianglelefteq T_2$ if there is some $\Phi : T_2 \hookrightarrow T_1$ which fixes T_1.*

Proof Let $\Phi : T_2 \hookrightarrow T_1$ fix T_1. We may assume that there are models for T_1 since otherwise T_2 would be trivially a model conservative extension of T_1. Let $\mathcal{M}_1$ be a model for T_1, and let $\mathcal{M}_2$ be the model for T_2 constructed from $\mathcal{M}_1$ as in the proof of Relative Satisfiability. By the definition of $\mathcal{M}_2$, $\mathcal{M}_2$ is an expansion of $\mathcal{M}_1$ since Φ fixes T_1. Therefore, $T_1 \trianglelefteq T_2$. $\Box$

Example 4.6 (Idempotent function constant (continued)) Let $\Phi = (\mu, \nu)$ be the interpretation of I in S^1 defined by:

(1) μ is the identity function on $\mathcal{B}$.

(2) ν is the identity function on $\mathcal{V}$.

(3) $\nu(f_{[\alpha,\alpha]}) = \lambda\{x_\alpha \,.\, x_\alpha\}$.

Clearly, Φ fixes S^1, and hence $S^1 \trianglelefteq I$ by the Model Conservative Verification Theorem. $\Box$

Example 4.7 (Cartesian product (continued)) There is no interpretation of P in S^2; so the Model Conservative Verification Theorem is not applicable. Nevertheless, it is easy to see that P is indeed a model conservative extension of S^2. In second-order logic with λ-notation or in simple type theory, one can construct an interpretation of P in S^2 which fixes S^2, by interpreting the sort γ as a suitable function sort built from α_1 and α_2. This illustrates an important benefit of working in an expressive logic: there exist more interpretations. $\Box$

Example 4.8 (Vector space (continued)) Let $\Phi = (\mu, \nu)$ be the interpretation of V in F defined by:

(1) $\mu(\mathbf{f}) = \mu(\mathbf{v}) = \mathbf{f}$.

(2) $\nu(x^i_{\mathbf{f}}) = x^{2i}_{\mathbf{f}}$.

(3) $\nu(y^i_{\mathbf{v}}) = x^{2i+1}_{\mathbf{f}}$.

(4) ν is the identity function on $\mathcal{C}$.

(5) $\nu(0_{\mathbf{v}}) = 0_{\mathbf{f}}$.

(6) $\nu(+_{[\mathbf{v},\mathbf{v},\mathbf{v}]}) = +_{[\mathbf{f},\mathbf{f},\mathbf{f}]}$.

(7) $\nu(*_{[\mathbf{f},\mathbf{v},\mathbf{v}]}) = *_{[\mathbf{f},\mathbf{f},\mathbf{f}]}$.

Φ is a formalization of the well-known fact that a field can be viewed as a one-dimensional vector space. Clearly, Φ fixes F, and hence $F \trianglelefteq V$ by the Model Conservative Verification Theorem. □

5 Theory instantiation

Let $T_i = (\mathcal{L}_i, \Gamma_i)$ for $i = 1, 2$ and $T'_1 = (\mathcal{L}'_1, \Gamma'_1)$. Suppose $T_1 \leq T'_1$ and $\Phi = (\mu, \nu) : T_1 \hookrightarrow T_2$. An *instance* of T'_1 under the interpretation Φ is an extension T'_2 of T_2 constructed as follows.

Intuitively, T'_2 is simply the result of substituting T_2 for T_1 in T'_1. How T_2 is cemented to the part of T'_1 outside of T_1 is determined by Φ. Also, the members of $(\mathcal{B}'_1 \cup \mathcal{V}'_1 \cup \mathcal{C}'_1) \setminus (\mathcal{B}_1 \cup \mathcal{V}_1 \cup \mathcal{C}_1)$ are renamed to avoid name conflicts.

Let $\mathcal{B}$ be a set of symbols such that $|\mathcal{B}| = |\mathcal{B}'_1 \setminus \mathcal{B}_1|$ and $\mathcal{B} \cap \mathcal{B}_2 = \emptyset$. Define $\mathcal{B}'_2 = \mathcal{B}_2 \cup \mathcal{B}$. Then let $\mu' : \mathcal{B}'_1 \to \mathcal{B} \cup \mathcal{B}_2 \cup \mathcal{U}_2$ be any extension of μ which is a bijection from $\mathcal{B}'_1 \setminus \mathcal{B}_1$ onto $\mathcal{B}$.

Let $\mathcal{V}$ be a set of symbols such that $|\mathcal{V}| = |\mathcal{V}'_1 \setminus \mathcal{V}_1|$ and $\mathcal{V} \cap \mathcal{V}_2 \cap \mathcal{C}_2 = \emptyset$. Define $\mathcal{V}'_2 = \mathcal{V}_2 \cup \mathcal{V}$. Let $\mathcal{C}$ be a set of symbols such that $|\mathcal{C}| = |\mathcal{C}'_1 \setminus \mathcal{C}_1|$ and $\mathcal{C} \cap \mathcal{C}_2 \cap \mathcal{V}'_2 = \emptyset$. Define $\mathcal{C}'_2 = \mathcal{C}_2 \cup \mathcal{C}$. Then let $\nu' : \mathcal{V}'_1 \cup \mathcal{C}'_1 \to \mathcal{V} \cup \mathcal{C} \cup \mathcal{C}_2 \cup \mathcal{E}_2 \cup \Lambda_2$ be any extension of ν which is a bijection from $\mathcal{V}'_1 \setminus \mathcal{V}_1$ onto $\mathcal{V}$ and a bijection from $\mathcal{C}'_1 \setminus \mathcal{C}_1$ onto $\mathcal{C}$.

Let $a \in \mathcal{V} \cup \mathcal{C}$ and α be the sort of $(\nu')^{-1}(a)$. If $\alpha \in \mathcal{B}'_1$, tag a with $\mu'[\alpha]$, and if $\alpha = [\alpha_1, \ldots, \alpha_n, \beta]$, tag a with $[\mu'[\alpha_1], \ldots, \mu'[\alpha_n], \mu'[\beta]]$.

The following two lemmas are easily verified:

Lemma 5.1 *$\mathcal{L}'_2 = (\mathcal{B}'_2, \mathcal{V}'_2, \mathcal{C}'_2)$ is a language of* **MS**.

Lemma 5.2 $\Phi' = (\mu', \nu') : T'_1 \to (\mathcal{L}'_2, \emptyset)$.

Let $T'_2 = (\mathcal{L}'_2, \Gamma'_2)$ where Γ'_2 is the union of Γ_2 and the set of obligations of Φ'. Since $(\mathcal{L}'_2, \emptyset) \leq T'_2$, $\Phi' : T'_1 \to T'_2$. In fact, $\Phi' : T'_1 \hookrightarrow T'_2$ because each obligation of Φ' is an axiom, and hence, a theorem of T'_2.

We have thus shown the following theorem:

Theorem 5.3 (Instantiation Theorem) *Suppose $T_1 \leq T'_1$ and $\Phi : T_1 \hookrightarrow T_2$. Then there is an extension T'_2 of T_2 and an extension $\Phi' = (\mu', \nu') : T'_1 \hookrightarrow T'_2$ of Φ such that μ' and ν' are bijections outside of T_1.*

Example 5.4 Suppose T_1 and T_2 are "minimal" theories (i.e., $|\mathcal{B}_1| = |\mathcal{B}_2| = 2$, $\mathcal{C}_1 = \mathcal{C}_2 = \emptyset$, and $\Gamma_1 = \Gamma_2 = \emptyset$) and $\Phi : T_1 \hookrightarrow T_2$. Then an instance of T'_1 under Φ is a copy of T'_1 whose base sorts and constants have been possibly renamed. $\Box$

Example 5.5 Suppose $T'_1 = T_2$ and Φ is the identity interpretation on T_1. Then an instance of T'_1 under Φ would generally be a proper extension of itself which contains copies of some of its machinery. A more reasonable "instance" of T'_1 under Φ would be just T'_1 itself. This suggests that, in practice, one would like to have a more sophisticated notion of the instance of a theory under an interpretation where new machinery is created only if it is not already present in the target theory of the interpretation. $\Box$

Our notion of a theory instantiation is closely related to the notion of theory instantiation proposed by Burstall and Goguen [2, 12, 13]; in both approaches a theory is instantiated via an interpretation. However, in our approach, any theory can be instantiated with respect to any of its subtheories. In the Burstall-Goguen approach, only "parameterized theories" can be instantiated and only with respect to the explicit parameter of the parameterized theory.

Theorem 5.6 (Model Conservative Instantiation Theorem) *Suppose $T_1 \trianglelefteq T'_1$ and T'_2 be an instance of T'_1 under $\Phi : T_1 \hookrightarrow T_2$. Then $T_2 \trianglelefteq T'_2$.*

Proof Let $\Phi' = (\mu', \nu') : T'_1 \hookrightarrow T'_2$ be an extension of Φ such that μ' and ν' are bijections outside of T_1. Also, let $\mathcal{M}_2 = (\{\mathcal{D}^2_\alpha : \alpha \in \mathcal{B}_2\}, I_2)$ be a model for T_2, and let $\mathcal{M}_1 = (\{\mathcal{D}^1_\alpha : \alpha \in \mathcal{B}_1, I_1)$ be the model for T_1 constructed from $\mathcal{M}_2$ as in the proof of Relative Satisfiability. Since $T_1 \trianglelefteq T'_1$, there is a model $\mathcal{M}'_1 = (\{\mathcal{D}'_{1,\alpha} : \alpha \in \mathcal{B}'_1\}, I'_1)$ for T'_1 which is an expansion of $\mathcal{M}_1$.

$\mathcal{M}'_2 = (\{\mathcal{D}'_{2,\alpha} : \alpha \in \mathcal{B}'_2, I'_2)$ is defined as follows. For $\alpha \in \mathcal{B}'_2$, if $\alpha \in \mathcal{B}_2$, then $\mathcal{D}'_{2,\alpha} = \mathcal{D}^2_\alpha$; otherwise $\mathcal{D}'_{2,\alpha} = \mathcal{D}'_{1,(\mu')^{-1}(\alpha)}$. Notice that $\mathcal{D}'_{1,\alpha} \subseteq \mathcal{D}'_{2,\mu'[\alpha]}$ for all $\alpha \in \mathcal{B}'_1$. I'_2 is defined by:

(1) For $c_\alpha \in \mathcal{C}_2$, $I'_2(c_\alpha) = I_2(c_\alpha)$.

(2) For an individual constant $c_\alpha \in \mathcal{C}'_2 \setminus \mathcal{C}_2$, $I'_2(c_\alpha) = I'_1((\nu')^{-1}(c_\alpha))$.

(3) For a function or predicate constant $c_{[\alpha_1,\ldots,\alpha_n,\beta]} \in \mathcal{C}'_2 \setminus \mathcal{C}_2$, $I'_2(c_{[\alpha_1,\ldots,\alpha_n,\beta]})$ is any extension $F : \mathcal{D}'_{2,\alpha_1} \times \cdots \times \mathcal{D}'_{2,\alpha_n} \to \mathcal{D}'_{2,\beta}$ of the function $I'_1((\nu')^{-1}(c_{[\alpha_1,\ldots,\alpha_n,\beta]}))$.

By a straightforward proof, $\mathcal{M}_2'$ is a model for T_2' and an expansion of $\mathcal{M}_2$. Therefore, $T_2 \trianglelefteq T_2'$. □

Remark 5.7 The analogue of the Model Conservative Instantiation Theorem for consequence conservativity is sometimes referred to as the "modularization theorem" (e.g., see [19]). For a proof of this theorem for many-sorted first-order logic, see [21].

Example 5.8 (Idempotent function constant (completed)) Let U be a theory with β among its sorts. We can add an abstract idempotent function constant to U by instantiating I under the trivial interpretation that maps α to β. The resulting theory is a model conservative extension of U by the Model Conservative Instantiation Theorem. Similar theories to I can be used for adding other "underspecified" constants to a theory. □

Example 5.9 (Cartesian product (completed)) Let U be a theory with β_1 and β_2 among its sorts. We can add a cartesian product of β_1 and β_2 to U by instantiating P under the trivial interpretation that maps α_i to β_i for $i = 1, 2$. The resulting theory is a model conservative extension of U by the Model Conservative Instantiation Theorem. Similar theories to P can be used for adding other abstract data types to a theory. □

Example 5.10 (Vector space (completed)) Let U be a theory which contains the structure of a field, i.e., there is some interpretation Φ of F in U. Suppose we would like to extend U to a theory in which we can reason about abstract vectors over this field. A suitable theory would be an instance U' of V under Φ. By the Model Conservative Instantiation Theorem, U' would be a model conservative extension of U. □

6 The method

We present in this section our method for safely overwriting theories in MMSs. Let **S** be some MMS whose underlying logic is **MS**. We will describe the method in terms of **S**.

We begin by making a few definitions.

A *theory module* is a structure written as $[T_0, T]$ where $T_0 \trianglelefteq T$. Theory modules will be used to represent "working" theories and model conservative extensions.

An *interpretation module* is a structure written as $[\Phi, [U_0, U], [V_0, V]]$ where $\Phi : T_1 \hookrightarrow T_2$, $[U_0, U]$ is a theory module such that $U_0 \trianglelefteq T_1$ and $T_1 \trianglelefteq U$, and $[V_0, V]$ is a theory module such that $T_2 \leq V$. Interpretation modules will be used to represent "working" interpretations.

A *theory library* is a set $\mathcal{M}$ of theory and interpretation modules such that, for all $[\Phi, [U_0, U], [V_0, V]] \in \mathcal{M}$, $[U_0, U], [V_0, V] \in \mathcal{M}$.

We will assume that **S** contains a theory library **TL** as well as the following procedures for modifying **TL**:

(1) *Install-theory.* Given T as input, this procedure adds a new theory module $[T, T]$ to **TL**.

(2) *Install-theory-extension.* Given T and T' as input, this procedure adds a new theory module $[T, T']$ to **TL**, provided that **S** can verify that $T \trianglelefteq T'$.

(3) *Splice-theory-extensions.* Given $[T_0, T_1], [T_1, T_2] \in$ **TL** as input, this procedure replaces every occurrence of $[T_0, T_1]$ and $[T_1, T_2]$ in **TL** with $[T_0, T_2]$. (Note: By Transitivity, $T_0 \trianglelefteq T_2$.)

(4) *Install-interpretation.* Given $\Phi : T_1 \rightarrow T_2$ and $[U_0, U], [V_0, V] \in$ **TL** as input, this procedure adds a new interpretation module $[\Phi, [U_0, U], [V_0, V]]$ to **TL**, provided **S** can verify that $\Phi : T_1 \hookrightarrow T_2$, $U_0 \trianglelefteq T_1$, $T_1 \trianglelefteq U$, and $T_2 \leq V$.

(5) *Extend-theory.* Given $[\Phi, [U_0, U], [V_0, V]] \in$ **TL** as input, this procedure replaces $[\Phi, [U_0, U], [V_0, V]]$ and every occurrence of $[V_0, V]$ in **TL** with $[\Phi', [U_0, U], [V_0, V']]$ and $[V_0, V']$, respectively, where V' is an instance of U via Φ such that $V \leq V'$ and where Φ' is the corresponding extension of Φ to U. (Note: By the Model Conservative Instantiation Theorem, $V \trianglelefteq V'$, and hence, by Transitivity, $V_0 \trianglelefteq V'$.)

It is easy to prove that, if **TL** is theory library, then the result of applying any one of these five procedures to **TL** is also a theory library.

Remarks:

(1) Install-theory is a special case of Install-theory-extension.

(2) By the Model Conservative Verification Theorem, **S** can verify that $T \trianglelefteq T'$ by verifying that there is an interpretation module $[\Phi, [T', T'], [T, T]] \in$ **TL** such that Φ fixes T.

(3) If theory and interpretation modules are implemented in **S** so that there is at most one module with the same components, then "replacing each occurrence of a module X in **TL** with a module Y" would normally be implemented by overwriting X with Y.

(4) Suppose $[\Phi, [U_0, U], [V_0, V]]$ in **TL** where $\Phi : U \hookrightarrow V$. If $[U_0, U]$ is extended to $[U_0, U']$ as a result of applying Extend-theory to some other interpretation module, then $[\Phi, [U_0, U], [V_0, V]]$ will become an interpretation module of the form $[\Phi, [U_0, U'], [V_0, V']]$. Later, if desired, Φ can be extended to an interpretation $\Phi' : U' \hookrightarrow V'$ by applying Extend-theory to $[\Phi, [U_0, U'], [V_0, V']]$.

We will now describe how this machinery for managing theories and interpretations can be used as a basis for a general method for overwriting theories with model conservative extensions.

Suppose that a user of **S** would like to install an mce-type τ where each member of $\mathrm{dom}(\tau)$ has the same "form". The user would choose a fully abstract member (T, T') of $\mathrm{dom}(\tau)$ and then install the theory module $[T, T']$ in **TL** using Install-theory-extension. (If **S** could not verify that $T \trianglelefteq T'$, the user might first use Install-interpretation to install an interpretation module $[\Phi, [T', T'], [T, T]]$ in **TL** such that Φ fixes T.) $\mathrm{dom}(\tau)$ would then consist of all (U, U') such that U' is an instance of T' under an interpretation of T in U.

Now suppose that someone wants to overwrite a theory module $[U_0, U]$ in **TL** with $[U_0, U']$ where U' is an instance of T' under an interpretation $\Phi : T \hookrightarrow U$. First, $[\Phi, [T, T'], [U_0, U]]$ would be installed in **TL** using Install-interpretation. Then, $[U_0, U]$ would be overwritten in **TL** with $[U_0, U']$ by applying Extend-theory to $[\Phi, [T, T'], [U_0, U]]$. Interpretation modules affected by the overwriting can be "updated" when necessary by using Extend-theory.

Notice that, by this method, the task of overwriting a theory T with a model conservative extension T' is factored into two steps. The first step is to install an mce-type τ as described above such that $(T, T') \in \mathrm{dom}(\tau)$. The second step is to overwrite T with T': The user installs an appropriate interpretation Φ with target theory T, then **S** automatically constructs T', and finally **S** overwrites T with T'. After the first step is performed, the second step can be repeated again and again for different members of τ.

7 Conclusion

We have proposed a method for overwriting theories with model conservative extensions. Module conservative extensions are safe since they preserve models and semantic consistency. The method can be used in a predicate logic for which there is:

(1) A notion of an interpretation of one theory in another.

(2) A notion of theory instantiation via theory interpretation.

(3) A "model conservative verification theorem" that says T' is a model conservative extension of T if there is an interpretation of T' in T which fixes T.

(4) A "model conservative instantiation theorem" that says model conservativity is preserved by theory instantiation.

The main idea of the method is to represent a model conservative extension type by an abstract theory extension. The abstract extension is shown to be model conservative by the model conservative verification theorem, and instances of the extension type are constructed from the abstract extension by means of theory instantiation. Each instance of the extension type is automatically model conservative by the model conservative instantiation theorem.

We have illustrated the method with a version of many-sorted first-order logic called **MS**. It is more effective with highly expressive predicate logics, such as simple type theory, which have many more interpretations than a first-order logic. The method is primarily intended for use in mechanized mathematics systems—especially systems which support the "little theories" version, as opposed to the "big theory" version, of the axiomatic method.[6]

Acknowledgments

Many of the ideas in this paper grew out of conversations I have had with Dr. Joshua Guttman and Dr. Javier Thayer in the course of developing the IMPS Interactive Mathematical Proof System [9, 11].

[6]In the big theory version, all reasoning is performed within a single powerful and highly expressive axiomatic theory, such as Zermelo-Fraenkel set theory; in the little theories version, reasoning is distributed across a network of theories linked by interpretations which serve as conduits to pass results from one theory to another. We argue in [10] that this way of organizing mathematics is very advantageous for managing complex mathematics within mechanized mathematics systems.

References

[1] R. Boyer and J Moore. *A Computational Logic Handbook.* Academic Press, 1988.

[2] R. Burstall and J. Goguen. The semantics of Clear, a specification language. In *Advanced Course on Abstract Software Specifications*, volume 86 of *Lecture Notes in Computer Science*, pages 292–332. Springer-Verlag, 1980.

[3] P. Byers and D. Pitt. Conservative extensions: A cautionary note. *Bulletin of the EATCS*, (41):196–201, June 1990.

[4] D. Craigen, S. Kromodimoeljo, I. Meisels, A. Neilson, B. Pase, and M. Saaltink. m-EVES: A tool for verifying software. In *Proceedings of the 10th International Conference on Software Engineering (ICSE), Singapore.* IEEE Computer Society Press, 1988.

[5] D. Craigen, S. Kromodimoeljo, I. Meisels, B. Pase, and M. Saaltink. EVES: An overview. Technical Report CP-91-5402-43, ORA Corporation, 1991.

[6] H. B. Enderton. *A Mathematical Introduction to Logic.* Academic Press, 1972.

[7] W. M. Farmer. A simple type theory with partial functions and subtypes. *Annals of Pure and Applied Logic*, 64:211–240, 1993.

[8] W. M. Farmer. Theory interpretation in simple type theory. In J. Heering et al., editor, *Higher-Order Algebra, Logic, and Term Rewriting*, volume 816 of *Lecture Notes in Computer Science*, pages 96–123. Springer-Verlag, 1994.

[9] W. M. Farmer, J. D. Guttman, and F. J. Thayer. IMPS: System description. In D. Kapur, editor, *Automated Deduction—CADE-11*, volume 607 of *Lecture Notes in Computer Science*, pages 701–705. Springer-Verlag, 1992.

[10] W. M. Farmer, J. D. Guttman, and F. J. Thayer. Little theories. In D. Kapur, editor, *Automated Deduction—CADE-11*, volume 607 of *Lecture Notes in Computer Science*, pages 567–581. Springer-Verlag, 1992.

[11] W. M. Farmer, J. D. Guttman, and F. J. Thayer. IMPS: An Interactive Mathematical Proof System. *Journal of Automated Reasoning*, 11:213–248, 1993.

[12] J. A. Goguen. Principles of parameterized programming. Technical report, SRI International, 1987.

[13] J. A. Goguen and R. M. Burstall. Introducing institutions. In *Logic of Programs*, volume 164 of *Lecture Notes in Computer Science*, pages 221–256. Springer-Verlag, 1984.

[14] M. J. C. Gordon and T. F. Melham. *Introduction to HOL: A Theorem Proving Environment for Higher Order Logic.* Cambridge University Press, 1993.

[15] J. D. Guttman. A proposed interface logic for verification environments. Technical Report M91-19, The MITRE Corporation, 1991.

[16] J. L. Hook. A note on interpretations of many-sorted theories. *Journal of Symbolic Logic*, 50:372–374, 1985.

[17] J. D. Monk. *Mathematical Logic.* Springer-Verlag, 1976.

[18] J. R. Shoenfield. *Mathematical Logic.* Addison-Wesley, 1967.

[19] W. M. Turski and T. S. E. Maibaum. *The Specification of Computer Programs.* Addison-Wesley, 1987.

[20] P. A. S. Veloso. Yet another cautionary note on conservative extensions: A simple case with a computing flavour. *Bulletin of the EATCS*, (46):188–192, February 1992.

[21] P. A. S. Veloso. A new, simpler proof of the modularization theorem for logical specifications. *Bulletin of the IGPL*, 1:3–12, 1993.

[22] P. A. S. Veloso and S. R. M. Veloso. Some remarks on conservative extensions: A Socratic dialogue. *Bulletin of the EATCS*, (43):189–198, February 1991.