Electronic Notes in Theoretical Computer Science 71 (2003)

URL: <http://www.elsevier.nl/locate/entcs/volume71.html>

17 pages

Reflection in Membership Equational Logic, Many-Sorted Equational Logic, Horn Logic with Equality, and Rewriting Logic

Manuel Clavel a*,*1, Jos´e Meseguer b*,*2, and Miguel Palomino c*,*3

a *Facultad de Informa´tica, Universidad Complutense de Madrid, Spain.*

b *CS Department, University of Illinois at Urbana-Champaign, USA.*

c *Facultad de Matema´ticas, Universidad Complutense de Madrid, Spain.*

**Abstract**

We show that the generalized variant of rewriting logic where the underlying equa- tional specifications are membership equational theories, and where the rules are conditional and can have equations, memberships and rewrites in the conditions is reflective. We also show that membership equational logic, many-sorted equational logic, and Horn logic with equality are likewise reflective. These results provide logical foundations for reflective languages and tools based on these logics, and in particular for the Maude language itself.

*Key words:* Reflection, reflective logics, universal theory, reflective rewriting logic, reflective membership equational logic, reflective Horn logic with equality

# Introduction

Reflection is a very powerful and useful feature of rewriting logic, motivating work on metalogical reflection theorems. Clavel and Meseguer have formerly given detailed proofs for increasingly general fragments of rewriting logic, namely: (1) unsorted and unconditional [3], (2) unsorted conditional [7]; and

(3) many-sorted conditional [7]. This paper generalizes these previous results to the case of conditional rewrite theories whose underlying equational speci- fications are theories in *membership equational logic* [11]. Conditional rules in this latter case are very general, since they can involve not only other rewrites,

1 Partially supported by a CICYT Project TIC2002-01167.

2 Supported in part by ONR Grant N00014-02-1-0715.

3 Supported by a postgraduate fellowship by the Spanish Ministry for Science and Tech- nology.

*◯*c *2003 Published by Elsevier Science B. V. Open access under* [*CC BY-NC-ND license.*](http://creativecommons.org/licenses/by-nc-nd/3.0/)

but also equations and memberships as conjuncts. The work presented here is also related to Palomino’s own research on rewriting logic reflection [12].

But what about other logics? What about membership equational logic it- self? What about many-sorted equational logic? What about Horn logic with equality? We have for long conjectured that these logics are also reflective, and that the same methods developed for rewriting logic can be used to obtain reflection theorems for these new logics. The present paper confirms the truth of these conjectures. Furthermore, our constructions shed light on the ques- tion of how the universal theories of related logics are themselves related. For example, membership equational logic is itself a sublogic of rewriting logic, and this sublogic relation is expressed at the reflective level by the fact that

the universal theory of memb[ership](#_bookmark22) equational logic is itself a *subtheory* of

the universal theory for the more general version of rewriting logic where the underlying equational specifications are membership equational theories.

Therefore, our results make clear that reflection is available as a very pow- erful feature not only for this more general variant of rewriting logic, namely the one supported by Maude [4,5], but also for other computational logics of great importance in formal specification and declarative programming, such as memb[ership](#_bookmark21) equational logic, many-sorted equational logic, and Horn logic with equality. This can then serve as a basis for the theoretically-grounded design of *declarative reflective programming languages* in those logics.

Having an explicit specification of the corresponding universal theories is of great practical importance for metalogical reasoning. In joint work with David Basin [1,2] we have investigated the good properties of membership equational logic and of rewriting logic as *reflective metalogical frameworks* that combine induction, parameterization, and reflection to support metalogical reasoning about logics represented in them. In such metareasoning, as well as whenever *proof objects* are required to justify reflective proofs, it is essential to make an

explicit use of the corresponding universal theories. In particular, the results in this paper have several important consequences for the Maude language, in that they both serve as a foundation for its META-LEVEL module, and they pro- vide a general method for combining efficient reflective computation using the built-in functionality of the META-LEVEL module with the ability to generate proof objects by means of the universal theories when this is required.

The paper is organized as follows. [First,](#_bookmark19) in Section 2, we summarize the axioms characterizing the notion of a reflective logic. Then, in Sections 3, 4, 5, and 6, we prove, respectively, that membership equational logic, many-sorted equational logic, many-sorted Horn logic with equality, and rewriting logic are reflective in our axiomatic sense. Finally, in Section 7 we compare these results with previous work, and in Section 8 we draw conclusions.

# Reflection in General Logics

We present below in summarized form the axiom characterizing the notion of a *reflective logic*. We introduce first the notions of *syntax* and of *entailment system*, used in our axiomatization. These notions are defined using the lan- guage of category theory, but do not require any acquaintance with categories beyond the basic notions of category and functor.

## Syntax.

Syntax can typically be given by a *signature* Σ providing a grammar on which to build *sentences*. For first-order logic, a typical signature consists of a set of function symbols and a set of predicate symbols, each with a prescribed number of arguments, which are used to build up the usual sentences. We assume that for each logic there is a category **Sign** of possible signatures for it, and a functor *sen* assigning to each signature Σ the set *sen*(Σ) of all its sentences. We call the pair (**Sign***, sen*) a *syntax*.

## Entailment systems.

For a given signature Σ in **Sign**, *entailment* (also called *provability* ) of a sentence *ϕ ∈ sen*[(Σ)](#_bookmark26) from a set of axioms Γ *⊆ sen*(Σ) is a relation Γ *▶*Σ *ϕ*

which holds if and only if we can prove *ϕ* from the axioms Γ using the rules of the logic. We make this relation relative to a signature.

In what follows, *|C|* denotes the collection of objects of a category *C*.

**Definition 2.1** [9] An *entailment system* is a triple *E* = (**Sign***, sen, ▶*) such that

* (**Sign***, sen*) is a syntax,
* *▶* is a function associating to each Σ *∈ |***Sign***|* a binary relation *▶*Σ *⊆ P*(*sen*(Σ)) *× sen*(Σ), called Σ-*entailment*, that satisfies the following prop- erties:

1. *reflexivity*: for any *ϕ ∈ sen*(Σ), *{ϕ} ▶*Σ *ϕ*,
2. *monotonicity*: if Γ *▶*Σ *ϕ* and Γ*' ⊇* Γ then Γ*' ▶*Σ *ϕ*,
3. *transitivity*: if Γ *▶*Σ *ϕ*, for all *ϕ ∈* ∆, and Γ *∪* ∆ *▶*Σ *ψ*, then Γ *▶*Σ *ψ*,
4. *▶*-*translation*: if Γ *▶*Σ *ϕ*, then for any *H* : Σ *→* Σ*'* in **Sign** we have

*sen*(*H*)(Γ) *▶*Σ*' sen*(*H*)(*ϕ*).

**Definition 2.2** [9] Given an entailment system *E* , its category **Th** of *theories* has as objects pairs *T* = (Σ*,* Γ) with Σ a signature and Γ *⊆ sen*(Σ). A *theory morphism H* : (Σ*,* Γ) *→* (Σ*',* Γ*'*) is a signature morphism *H* : Σ *→* Σ*'* such that if *ϕ ∈* Γ, then Γ *▶*Σ*' sen*(*H*)(*ϕ*).

*'*

Note that we can extend the functor *sen* to a functor on theories by taking

*sen*(Σ*,* Γ) = *sen*(Σ).

* 1. *Reflective Logics*

A reflective logic is a logic in which important aspects of its metatheory can be represented at the object level in a consistent way, so that the object- level representation correctly simulates the relevant metatheoretic aspects. Two obvious metatheoretic notions that can be so reflected are theories and the entailment relation *▶*. This leads us to the notion of a universal theory. However, universality may not be absolute, but only relative to a class *C* of *representable* theories.

**Definition 2.3** Given an entailment system *E* and a set of theories *C ⊆ |***Th***|*, a theory *U* is *C*-*universal* if there is a function, called a *representation function*, ( *▶* ) : [ *{T}× sen*(*T* ) *−→ sen*(*U* )

T *∈C*

such that for each *T ∈ C*, *ϕ ∈ sen*(*T* ),

*T ▶ ϕ ⇐⇒ U ▶ T ▶ ϕ.*

If, in addition, *U ∈ C*, then the entailment system *E* is called *C*-*reflective*.

To take into account computability considerations, we should further re- quire that the representation function ( *▶* ) is *recursive*. Finally, to rule out unfaithful representations, we should require that the function ( *▶* ) is *injective*.

# Reflection in Membership Equational Logic

* 1. *Membership Equational Logic*

A *signature* in membership equational logic—in short, MEqtl—is a triple Ω = (*K,* Σ*, S*), with *K* a set of *kinds*, Σ a *K*-kinded signature Σ= *{*Σw,k*}*(w,k)*∈*K*∗×*K, and *S* = *{S*k*}*k*∈*K a pairwise disjoint *K*-kinded family of sets. We call *S*k the set of *sorts* of kind *k*. The pair (*K,* Σ) is what is usually called a many- sorted signature of function symbols; however we call the elements of *K* kinds because each kind *k* now has a set *S*k of associated *sorts*, which in the models will be interpreted as subsets of the carrier for the kind. As usual, we denote by *T*Σ the *K*-kinded algebra of ground Σ-terms, and by *T*Σ(*X*) the *K*-kinded algebra of Σ-terms on the *K*-kinded set of variables *X*.

The atomic formulae of MEqtl are either *equations t* = *t'*, where *t* and *t'* are Σ-terms of the same kind, or *membership assertions* of the form *t* : *s*, where the term *t* has kind *k* and *s ∈ S*k. Sentences are Horn clauses on these atomic formulae, i.e., sentences of the form

*∀*(*x*1*,..., x*m)*. A*1 *∧ ... ∧ A*n =*⇒ A*0 *,*

where each *A*i is either an equation or a membership assertion, and each *x*j

is a *K*-kinded variable. A theory in membership equational logic is a pair

(Ω*, E*), where *E* is a set of sentences—(conditional) equations or (conditional) membership axioms—in MEqtl over the signature Ω. To simplify the definition of the universal theory for MEqtl in Section 3.2, we will work with theories with *nonempty kinds*, that is, for each kind, the elements of that kind in the initial algebra form a nonempty set. This is a relatively minor restriction that avoids the well-known complications with quantification in many-sorted

equational deduction [8]. 4 Thus, from now on, we will omit the quantifiers in all sentences.

Finally, we introduce our notions and notations for contexts and substitu- tions. Given a signature Ω = (*K,* Σ*, S*), a *K*-kinded set of variables *X*, and a *K*-kinded set of new constants *{ı*k*}*k*∈*K, a *context* is a term *C*k which contains exactly one subterm *t'* = *ı*k, called its [“hole,”](#_bookmark4) for some *k ∈ K*. We define Cı (*X*) to be the set of contexts. Given a context *C*k and a term *t ∈ T*Σ(*X*) of kind *k*, *C*k[*t*] *∈ T*Σ(*X*) is the term that results from replacing the “hole” *ı*k in *C*k by *t*. When not needed, we omit mentioning the kind of the “hole” in the context. Given a signature Ω = (*K,* Σ*, S*), and a *K*-kinded set of variables *X*, we define *S*(Σ,X) to be the set of *substitutions* 5

Σ

*S*(Σ,X) = *{*(*x*1 *'→ w*1*,..., x*n *'→ w*n) *| ∀i, j ≤ n, i /*= *j ⇒ x*i */*= *x*j*,*

and *∀i ≤ n, x*i and *w*i have the same kind*}.*

Given a term *t* and a substitution *σ* = (*x*1 *'→ w*1*, ..., x*n *'→ w*n), we de- note by *tσ* the term *t*(*w*1*/x*1*,..., w*n*/x*n) obtained from *t* by simultaneously substituting *w*i for *x*i, *i* = 1*,..., n*.

* + 1. *The Rules of* MEqtl

We now introduce the rules of deduction of MEqtl. Our formulation is slightly different from, but equivalent to and simpler for our purposes than, that in [11], in that the congruence rule is removed and is taken into account as part of the “modus ponens” rule.

Given a membership equational theory *T* = (Ω*, E*), we say that *T entails* a sentence *φ* and write *T ▶ φ* if and only if *φ* can be obtained by finite application of the following *rules of deduction* :

* + - 1. **Reflexivity**. For each *t ∈ T*Σ(*X*), *T ▶ t* = *t.*
      2. **Modus ponens**. For each equation (*C*mb *∧ C*eq =*⇒ t* = *t'*) in *E*, with *t*, *t'* of kind *k*, context *C*k *∈* Cı (*X*), and substitution *σ*, where *C*mb , (*u*1 :

Σ

4 The specification and proof of correctness of a universal theory in the more general case in which some kinds can be empty should follow very similar lines. The main difference is that the universal quantifiers need to be metarepresented in sentences, and the inference rules must keep track of such quantifiers.

5 Conceptually, a substitution is a function from variables to terms. For technical conve-

nience, we choose to define substitutions as a special case of lists of pairs formed by variables and terms.

*s*1 *∧ ... ∧ u*j : *s*j) and *C*eq , (*v*1 = *v' ∧ ... ∧ v*k = *v'* ),

1 k

*T ▶ u*1*σ* : *s*1 *··· T ▶ u*j*σ* : *s*j *T ▶ v*1*σ* = *v' σ ··· T ▶ v*k*σ* = *v' σ*

1 k

*.*

*T ▶ C*[*tσ*] = *C*[*t'σ*]

Similarly, for each membership axiom (*C*mb *∧ C*eq =*⇒ t* : *s*) in *E*, and substitution *σ*, where *C*mb and *C*eq are as before,

*T ▶ u*1*σ* : *s*1 *··· T ▶ u*j*σ* : *s*j *T ▶ v*1*σ* = *v' σ ··· T ▶ v*k*σ* = *v' σ*

## Symmetry.

*T ▶ t* = *t' T ▶ t'* = *t .*

1 k *.*

*T ▶ tσ* : *s*

## Transitivity.

* + - 1. **Membership**.

*T ▶ t* = *t'' T ▶ t''* = *t'*

*T ▶ t* = *t' .*

*T ▶ t* = *u T ▶ u* : *s*

*.*

*T ▶ t* : *s*

* + - 1. **Implication introduction**. For each sentence *A*1 *∧...∧A*n =*⇒ A*0 over

the signature of *T* , where each *A*i is either an equation or a membership assertion,

(Ω(*X*)*, E ∪ {A*1*,..., A*n*}*) *▶ A*0 *,*

(Ω*, E*) *▶ A*1 *∧ ... ∧ A*n =*⇒ A*0

where Ω(*X*) is the signature Ω extended with the elements of *X* as addi- tional new constants.

* 1. *A Universal Theory for* MEqtl

In this section, we introduce the universal theory UMEqtl and a representation function ( *▶* ) that encodes pairs consisting of a theory *T* and a sentence

over its signature, as a sentence in UMEqtl. We also sketch the proof of the

universality of UMEqtl, as formalized in Definition 2.3. In what follows, we will be dealing with finitely presentable theories in MEqtl.

* + 1. *The Signature of* UMEqtl

The signature of the theory UMEqtl contains constructors to represent terms, contexts, substitutions, kinds, sorts, signatures, axioms, and theories. For brevity we only declare the subsignature of UMEqtl explicitly used in the paper. We use Maude notation and write a kind enclosed in square brackets.

To represent the decomposition of a [term](#_bookmark6) *t* = *C*[*t'*] into a context *C* and a potential redex *t'*, the signature of UMEqtl includes the constructor

op \_[\_] : [Context] [Term] -> [Term] .

and to represent the decomposition of a term *t* = *t'σ* into a term *t'* and a substitution *σ*, it includes the constructor 6

6 Notice that we indeed use these two operations as constructors, and not as defined

op : [Term] [Substitution] -> [Term] .

To represent the context consisting only of a “hole,” and to represent the empty substitution, the signature of UMEqtl includes the constructors

op \* : -> [Context] . op - : -> [Substitution] .

Also, to represent (possibly conditional) equations and membership axioms, the signature of UMEqtl includes the constructors

op \_=\_ : [Term] [Term] -> [Atom] [comm] .

op \_:\_ : [Term] [Sort] -> [Atom] . op none : -> [Condition] .

op \_/\\_ : [Atom] [Condition] -> [Condition] . op \_if\_ : [Atom] [Condition] -> [Axiom] .

where the constant none is used to represent the empty set of conditions, and the attribute comm indicates that matching a term built with this constructor is done *modulo* commutativity.

In addition, the signature of UMEqtl contains a Boolean function parse to decide whether a term is well-formed with respect to a many-kinded signature, a function extVar to extract the variables occurring in a sentence, a function addVarAsCnst to extend a signature by adding some variables as new con- stants, and a function addEq to extend a set of axioms by adding the atomic formulae in a condition as new axioms.

op parse : [Term] [Signature] -> [Bool] .

op extVar : [Term] [Term] [Condition] -> [VarSet] . op extVar : [Term] [Condition] -> [VarSet] .

op addVarAsCnst : [Signature] [VarSet] -> [Signature] . op addEq : [AxiomSet] [Condition] -> [AxiomSet] .

Finally, the signature of UMEqtl contains two Boolean op[erations](#_bookmark12)

op \_:\_if\_in\_ : [Term] [Sort] [Condition] [Theory] -> [Bool] . op \_=\_if\_in\_ : [Term] [Term] [Condition] [Theory] -> [Bool] .

to represent entailment of sentences in a given membership equational the- ory; the main axioms of UMEqtl, including those in Figure 1, define these two operations.

operations; that is, application of a substitution to a term is not really defined. Instead, in addition to equations dealing with the rules of deduction of MEqtl, the universal theory UMEqtl has equations for composing/decomposing both terms appearing in an equation into different contexts and substitutions (these equations are not included here due to space limitations), so that the “modus ponens” rule can be applied. In particular, a term *t* can for example be represented as \*[*t* -], where *t* is the metarepresentation of *t* itself, and \* and - represent, respectively, the “hole” and the empty substitution as explained below. The reason for this choice is that it makes our proofs simpler.

* + 1. *The Representation Function*

We next define the representation function ( *▶* ). For all membership equa- tional theories *T* , and sentences *φ* over the signature of *T* ,

* if *φ* = (*A*1 *∧ ... ∧ A*n =*⇒ t* : *s*), where each *A*i is an atomic formula, then

*T ▶ φ* , (\*[*t* -]:*s* if *A*1 *∧ ... ∧ A*n in *T* ) = true, and

* if *φ* = (*A*1 *∧ ... ∧ A*n =*⇒ t* = *t'*), where each *A*i is an atomic formula, then

*T ▶ φ* , (\*[*t* -] = \*[*t'* -] if *A ∧ ... ∧ A* in *T* ) = true,

1

n

where ( ) is a representation function defined recursively over theories, signa- tures, sets of axioms, individual equational and membership axioms, conjunc- tions of atomic formulae, terms and sorts, and returns, respectively, ground terms of kind [Theory], [Signature], [AxiomSet], [Axiom], [Condition], [Term], and [Sort]. We also define a representation function ( ) over contexts and substitutions that returns, respectively, ground terms of kind [Context] and [Substitution].

* + 1. *The Axioms of* UMEqtl

Finally, we define the axioms of UMEqtl, which include equations that corre- spond to the inference rules of MEqtl, along with equations to decide whether a term is well-formed with respect to a many-kinded [signature,](#_bookmark8) to compose/de- compose a term (see footnote 6), to extract the variables occurring in a sen- tence, to extend a signature by adding some variables as new constants, and to extend a set of axioms by adding the atomic formulae in a condition as new axioms. Due to space limitations we only show here, in Figure 1, the equations

in UMEqtl that correspond to the inference rules of MEqtl, 7 and state below some essential properties satisfied by its axioms.

**Proposition 3.1** *For T* = (Ω*, E*) *a ﬁnitely presentable membership equa- tional theory with nonempty kinds, with* Ω = (*K,* Σ*, S*)*, for all terms t ∈ T*Σ(*X*)*, it holds that*

UMEqtl *▶* parse(*t*,Ω) = true *.*

**Proposition 3.2** *For T* = (Ω*, E*) *a ﬁnitely presentable membership equa- tional theory with nonempty kinds, with* Ω = (*K,* Σ*, S*)*, for all terms w in* UMEqtl *of kind* [Term]*, it holds that if*

UMEqtl *▶* parse(*w*,Ω) = true *,*

7 To ease the understanding of these equations, we replace the usual variable notation by the corresponding representations of the entities to be placed in such variable positions. For example, Ω is a normal variable, but the notation suggests that the terms that the variable will match will typically be representations of signatures. Also, in the actual theory UMEqtl, the conditions of the “modus ponens” equations are formalized using a Boolean function that checks at the metalevel whether an instantiated conjunction of conditions holds in a given membership equational theory. To ease readability, in Figure 1 we have “expanded out” this Boolean function as a conjunction of checks for each of the atoms in the condition.

*then there is a term t ∈ T*Σ(*X*) *such that w* = *t.*

**Proposition 3.3** *For T* = (Ω*, E*) *a ﬁnitely presentable membership equa- tional theory with nonempty kinds, with* Ω = (*K,* Σ*, S*)*, for all terms t, t', u, u' ∈ T*Σ(*X*)*, contexts C ∈* Cı (*X*)*, and substitutions σ ∈ S*(Σ,X)*, it holds that, if t* = *C*[*uσ*] *and t'* = *C*[*u'σ*]*, then*

Σ

MEqtl

U *▶* (\*[*t* -]= \*[*t'* -] if none in *T* ) = (*C*[*u σ*]= *C*[*u'σ*] if none in *T* ) *.*

*Similarly, for all terms t, u ∈ T*Σ(*X*) *and substitutions σ ∈ S*(Σ,X)*, it holds that, if t* = *uσ then*

UMEqtl *▶* (\*[*t* -]:*s* if none in *T* ) = (\*[*u σ*]:*s* if none in *T* ) *.*

* 1. *The Correctness of the Universal Theory* UMEqtl

We sketch now the proof of the correctness of the universal theory UMEqtl.

**Theorem 3.4** *For all ﬁnitely presentable membership equational theories with nonempty kinds T* = (Ω*, E*)*, and sentences A*1 *∧.. .∧A*n =*⇒ t* : *s over* Ω*, where each A*i *is an atomic formula,*

*T ▶ A*1 *∧ ... ∧ A*n =*⇒ t* : *s*

*⇐⇒* UMEqtl *▶* (\*[*t* -]:*s* if *A*1 *∧ ... ∧ A*n in *T* ) = true *.*

*Similarly, for all sentences A*1 *∧ ... ∧ A*n =*⇒ t* = *t' over* Ω*, where each A*i *is* *an atomic formula,*

*T ▶ A*1 *∧ ... ∧ A*n =[*⇒*](#_bookmark23) *t* = *t'*

MEqtl 1 n

*⇐⇒* U *▶* (\*[*t* -] = \*[*t'* -] if *A ∧ ... ∧ A* in *T* ) = true *.*

The (*⇒*)-direction of this theorem is proved by structural induction on MEqtl proofs, using Propositions 3.1 and 3.3. Examples of similar proofs are given in detail in [3,7]. The (*⇐*)-direction of the theorem is proved as a corollary of the following

**Lemma 3.5** *For all ﬁnitely presentable membership equational theories with nonempty kinds T* = (Ω*, E*)*, with* Ω = (*K,* Σ*, S*)*, terms t in T*Σ(*X*)*, and sorts s in some S*k*,*

*T ▶ t* : *s ⇐*= UMEqtl *▶* (\*[*t* -]:*s* if none in *T* ) = true *.*

*Similarly, for all terms t, t' in T*Σ(*X*)*,*

*T ▶ t* = *t' ⇐*= U *▶* (\*[*t* -]= \*[*t'* -] if none in *T* ) = true *.*

**Proof.** Notice, first, that a proof in UMEqtl of an equality of the form

MEqtl

(\*[*t* -]:*s* if none in *T* ) = true or (\*[*t* -]= \*[*t'* -] if none in *T* ) = true

(1)

\*\*\* reflexivity

eq (\*[*t* -] = \*[*t* -] if none in (Ω, *E*)) = true.

\*\*\* modus ponens

\*\*\* where *E* = *{t* = *t' if C*mb *∧ C*eq*}∪ E'*, with

\*\*\* *C*mb = (*u*1 : *s*1 *∧ ... ∧ u*j :*s*j) and *C*eq = (*v*1 = *v' ∧ ... ∧ v*k = *v'* ).

1 k

ceq (*C*[*tσ*] = *C*[*t'σ*] if none in (Ω, *E*)) = true if (\*[*u*1*σ*]:*s*1 if none in (Ω, *E*)) = true

.

/\ (\*[*u*j*σ*]:*s*j if none in (Ω, *E*)) = true

/\ (\*[*v*1*σ*] = \*[*v' σ*] if none in (Ω, *E*)) = true

1

.

/\ (\*[*v*k*σ*] = \*[*v' σ*] if none in (Ω, *E*)) = true .

k

\*\*\* modus ponens

\*\*\* where *E* = *{t* : *s if C*mb *∧ C*eq*}∪ E'*, with

\*\*\* *C*mb = (*u*1 : *s*1 *∧ ... ∧ u*j :*s*j) and *C*eq = (*v*1 = *v' ∧ ... ∧ v*k = *v'* ).

1 k

ceq (\*[*tσ*]:*s* if none in (Ω, *E*)) = true

if (\*[*u*1*σ*]:*s*1 if none in (Ω, *E*)) = true

.

/\ (\*[*u*j*σ*]:*s*j if none in (Ω, *E*)) = true

/\ (\*[*v*1*σ*] = \*[*v' σ*] if none in (Ω, *E*)) = true

1

.

/\ (\*[*v*k*σ*] = \*[*v' σ*] if none in (Ω, *E*)) = true .

k

\*\*\* symmetry

ceq (\*[*t* -] = \*[*t'* -] if none in (Ω, *E*)) = true

if (\*[*t'* -] = \*[*t* -] if none in (Ω, *E*)) = true .

\*\*\* transitivity

ceq (\*[*t* -] = \*[*t'* -] if none in (Ω, *E*)) = true if parse(*t''*, Ω) = true

/\ (\*[*t* -] = \*[*t''* -] if none in (Ω, *E*)) = true

/\ (\*[*t''* -] = \*[*t'* -] if none in (Ω, *E*)) = true .

\*\*\* membership

ceq (\*[*t* -]:*s* if none in (Ω, *E*)) = true if parse(*u*, Ω) = true

/\ (\*[*t* -] = \*[*u* -] if none in (Ω, *E*)) = true

/\ (\*[*u* -]:*s* if none in (Ω, *E*)) = true .

\*\*\* implication introduction

\*\*\* where *Cond* = *A*1 *∧ ... ∧ A*n, with each *A*i an atomic formula. ceq (\*[*t* -] = \*[*t'* -] if *Cond* in (Ω,*E*)) = true

if (\*[*t* -] = \*[*t'* -] if none in

(addVarAsCnst(Ω, extVar(*t*, *t'*, *Cond* )), addEq(*E*, *Cond* ))) = true . ceq (\*[*t* -]:*s* if *Cond* in (Ω,*E*)) = true

if (\*[*t* -]:*s* if none in

(addVarAsCnst(Ω, extVar(*t*, *Cond* )), addEq(*E*, *Cond* ))) = true .

Fig. 1. The universal theory UMEqtl (fragment).

must consist of an application (possibly after a number of applications of the equations for decomposing/composing) of the “modus ponens” rule of [inference](#_bookmark11) using the equations in Figure 1. Notice also that: i) all equations but one in this figure are conditional; ii) their conditions include equalities that are either of the form (1), or can be reduced to that form by Proposition 3.3; and iii) these equalities must be proved before the conditional equations can

be used in an inference. Thus, each proof in UMEqtl of an equality of the form

(1) will have a certain depth, which we denote as the *conditional depth* of the proof [7]. The lemma is proved by induction on this conditional depth. Examples of similar proofs are given in detail in [7]. *✷*

As a corollary of the reflective result proved in this section, we will show in the next two sections the reflective nature of two other related logics: many- sorted equational logic and Horn logic with equality.

# Reflection in Many-Sorted Equational Logic

Many-sorted equational logic—in short, MSEqtl—is a sublogic of MEqtl, namely the sublogic obtained by making the set of sorts empty [11]; in par- ticular, for all theories *T* in MSEqtl, and sentences *φ* over the signature of *T* , it holds that *T ▶*MSEqtl *φ ⇐⇒ T ▶*MEqtl *φ*. But then, since we have only used kinds and conditional equations not involving any memberships in the definition of UMEqtl, we have that UMEqtl is a theory in MSEqtl, and therefore the following

**Theorem 4.1** UMEqtl *is a universal theory in* MSEqtl *for the class of ﬁnitely presentable theories having nonempty sorts.*

**Proof.** For all finitely presentable theories *T* in MSEqtl having nonempty sorts, and sentences *φ* over the signature of *T* , since, by definition, *T ▶ φ* is a sentence in MSEqtl,

*T ▶*MSEqtl *φ ⇐⇒ T ▶*MEqtl *φ*

*⇐⇒* UMEqtl *▶*MEqtl *T ▶ φ ⇐⇒* UMEqtl *▶*MSEqtl *T ▶ φ.*

*✷*

# Reflection in Horn Logic with Equality

In [11] it is shown that MEqtl is equivalent to many-sorted Horn logic with equality—in short, MSHorn. It is not surprising then that the reflective results about MEqtl can be translated straightforwardly to MSHorn.

=

=

=

A signature in MSHorn is a triple (*L,* Σ*,* Π), with *L* a set of sorts, Σ =

*{*Σw,l*}*(w,l)*∈*L*∗×*L a family of sets of function symbols, and Π = *{*Πw*}*w*∈*L*∗* a family of sets of predicate symbols. A signature Ω = (*K,* Σ*, S*) in MEqtl can then be mapped to a signature Ω٨ = (*K,* Σ*, S*٨) in MSHorn= by taking

٨ = *S*k for *k ∈ K*, and *S*٨ = *∅* for any *w ∈ K∗ \ K*. Thus, if we adopt

*S*

w

k

a postfix notation : *s* for each predicate in Ω٨, corresponding to a sort *s* in Ω, each sentence over Ω in MEqtl can be seen as a sentence over Ω٨ in MSHorn= . We will write (Ω*, E*)٨ for (Ω٨*, E*). Then, for all sentences *φ* over

Ω it holds that (Ω*, E*) *▶*MEqtl *φ ⇐⇒* (Ω*, E*)٨ *▶*

MSHorn

=

= *φ*. Moreover, [11]

defines a translation *α* mapping theories and sentences in MSHorn into MEqtl,

satisfying *T ▶* = *φ ⇐⇒ α*(*T* ) *▶*MEqtl *α*(*φ*). But then, we have the

following

=

MSHorn

## Theorem 5.1 U٨

MEqtl

*is a universal theory in* MSHorn *for the class of all*

*ﬁnitely presentable theories with nonempty sorts.*

**Proof.** For all finitely presentable theories with nonempty sorts *T* in MSHorn, and sentences *φ* over *T* ,

=

*T ▶* = *φ ⇐⇒ α*(*T* ) *▶*MEqtl *α*(*φ*)

MSHorn

*⇐⇒* UMEqtl *▶*MEqtl *α*(*T* ) *▶*MEqtl *α*(*φ*)

MEqtl *▶*

*⇐⇒* U

٨

= *α*(*T* ) *▶*MEqtl *α*(*φ*)

*✷*

MSHorn

# Reflection in Rewriting Logic

* 1. *Rewriting Logic*

Rewriting logic—in short, Rwl—is parameterized with respect to an under- lying equational logic; here we use MEqtl as this underlying logic. Given a MEqtl signature Ω = (*K,* Σ*, S*), the *sentences* of rewriting logic are “se- quents” of the form *t −→ t'*, where *t* and *t'* are Ω-terms of the same kind possibly involving some variables from a *K*-kinded set of variables *X*.

A *rewrite theory T* is a 3-tuple *T* = (Ω*, E, R*) where (Ω*, E*) is a MEqtl theory, and *R* is a set of *(conditional) rewrite rules* of the form

*t −→ t' if* ^(*u*j : *s*j) *∧* ^(*v*k = *v'* ) *∧* ^(*w*h *−→ w'* ) *,*

k h

j k h

where *t, t'* are Σ-terms of the same kind, each *u*j is a Σ-term of the kind of

the sort *s*j, *v*k*, v'*

k

are Σ-terms of the same kind, and *w*h*, w'*

are also terms of

the same kind. As before, we assume that the underlying MEqtl theory has nonempty kinds.

h

* + 1. *The Rules of* Rwl

We now introduce the rules of deduction of Rwl. Our formulation is slightly different from, but generalizes, that in [10], in that we give an explicit rule of *E*-equality for rewrite sequents *t −→ t'*, instead of absorbing such a rule in sequents [*t*] *−→* [*t'*] between *E*-equivalence classes. In addition, our underlying equational logic is now MEqtl.

Given a rewrite theory *T* = (Ω*, E, R*), we say that *T entails* a sequent *t −→ t'* and write *T ▶ t −→ t'* if and only if *t −→ t'* can be obtained by finite application of the following *rules of deduction* :

* + - 1. **Reflexivity**. For each *t ∈ T*Σ(*X*), *T ▶ t −→ t.*
      2. **Replacement**. For each rewrite rule (*t −→ t' if C*mb *∧ C*eq *∧ C*rl) in

*R*, with *t*, *t'* of kind *k*, context *C*k *∈* Cı (*X*), and substitution *σ*, where

Σ

*C*mb , (*u*1 : *s*1 *∧ ... ∧ u*j : *s*j), *C*eq , (*v*1 = *v' ∧ ... ∧ v*k = *v'* ), and

1

k

*C*rl , (*w*1 *−→ w' ∧ ... ∧ w*h *−→ w'* ),

1 h

(Ω*, E*) *▶ u*1*σ* : *s*1 *···* (Ω*, E*) *▶ u*j*σ* : *s*j (Ω*, E*) *▶ v*1*σ* = *v' σ ···* (Ω*, E*) *▶ v*k*σ* = *v' σ*

1 k

*T ▶ w*1*σ −→ w' σ ··· T ▶ w*h*σ −→ w' σ*

1 h

*.*

*T ▶ C*[*tσ*] *−→ C*[*t'σ*]

## Transitivity.

*T ▶ t −→ t' T ▶ t' −→ t'' T ▶ t −→ t'' .*

## Equality.

(Ω*, E*) *▶ t* = *u* (Ω*, E*) *▶ t'* = *u' T ▶ t −→ t'*

*T ▶ u −→ u' .*

* 1. *A Universal Theory for* Rwl

Here, we introduce the universal theory URwl and a representation function ( *▶* ) that encodes pairs consisting of a theory *T* and a sentence over its signature, as a sentence in URwl. We also sketch the proof of the universality of URwl. The key observation is that URwl is an extension of UMEqtl, so that we can use the universality of UMEqtl in the proof of the universality of URwl. In what follows, we will be dealing with finitely presentable theories in Rwl.

* + 1. *The Signature of* URwl

The signature of the theory URwl is an extension of the signature of UMEqtl. To represent (possibly conditional) rules, the signature of URwl includes the constructor:

op \_=>\_if\_ : [Term] [Term] [RuleCondition] -> [Rule] . In addition, the signature of URwl contains a Boolean function op \_=>\_in\_ : [Term] [Term] [Theory] -> [Bool] .

to represent provability of sentences in a given rewrite theory; the main axioms of URwl define this operation.

* + 1. *The Representation Function*

We next define the representation function ( *▶* ). For all finitely presentable rewrite theories with nonempty kinds *T* , and sentences *t −→ t'* over the sig-

nature of *T* ,

*T ▶ t −→ t'* , (\*[*t* -] => \*[*t'* -] in *T* ) *−→* true *.*

where ( ) is an extension of the representation function defined for UMEqtl.

* + 1. *The Axioms of* URwl

Finally, we define the axioms of URwl, which include rules that correspond to the inference rules of Rwl, along with equations to compose/decompose a term and all the equations in UMEqtl. Due to space limitations, we only show here, in Figure 2, the rules in URwl that correspond to the inference rules of Rwl; the same remarks as in footnote 7 apply to our notation for these rules. As with UMEqtl, we have the following

**Proposition 6.1** *For T* = (Ω*, E, R*) *a ﬁnitely presentable rewrite theory with nonempty kinds, with* Ω = (*K,* Σ*, S*)*, for all terms t, t', u, u' ∈ T*Σ(*X*)*, contexts C ∈* Cı (*X*)*, and substitutions σ ∈ S*(Σ,X)*, it holds that, if t* = *C*[*uσ*] *and t'* = *C*[*u'σ*]*, then*

Σ

U *▶* (\*[*t* -] => \*[*t'* -] in *T* ) *−→* (*C*[*u σ*] => *C*[*u'σ*] in *T* ) *.*

Rwl

* 1. *The Correctness of the Universal Theory* UMEqtl

We sketch now the proof of the correctness of the universal theory URwl.

**Theorem 6.2** *For all ﬁnitely presentable rewrite the*[*ories*](#_bookmark10) *with nonempty kinds*

*T* = (Ω*, E, R*)*, with* Ω = (*K,* Σ*, S*)*, and terms t, t' in T*Σ(*X*)

Rwl

*T ▶ t −→ t' ⇐⇒* U *▶* (\*[*t* -] => \*[*t'* -] in *T* ) *−→* true *.*

**Proof.** The proof is analogous to that of Theorem 3.4, using in key steps of the proof the fact that URwl is an extension of UMEqtl. *✷*

# Comparison with Previous Results

The work discussed in this paper generalizes and extends our previous work on reflection in rewriting logic [3,7,12]. The results presented here *generalize* our previous results on reflection in rewriting logic to its more general variant, namely, to the case of conditional rewrite theories whose underlying equational specifications are theories in membership equational logic. To simplify the correctness proof of the universal theory, we have, however, adopted a different approach for defining the universal theory. Essentially, in [3,7], an entailment

of the form *T ▶ t −→ t'* was reflected as *U ▶ ⟨T, t⟩ −→ ⟨T, t'⟩*. Accordingly, the “transitivity” rule of deduction did not have to be explicitly reified in the universal theory, since, if *T ▶ t*1 *−→ t*3 was proved by transitivity from *T ▶ t*1 *−→ t*2 and *T ▶ t*2 *−→ t*3, then, of course, *U ▶ ⟨T, t*1*⟩ −→ ⟨T, t*3*⟩*,

including UMEqtl .

\*\*\* reflexivity

rl (\*[*t* -] => \*[*t* -] in (Ω, *E*, *R*)) => true.

\*\*\* replacement

\*\*\* where *R* = *{t −→ t' if C*mb *∧ C*eq *∧ C*rl*}∪ R'*, with

\*\*\* *C*mb = (*u*1 : *s*1 *∧ ... ∧ u*j :*s*j),

\*\*\* *C*eq = (*v*1 = *v' ∧ ... ∧ v*k = *v'* ), and

1 k

\*\*\* *C*rl = (*w*1 *−→ w' ∧ ... ∧ w*h *−→ w'* )

*'* 1    h

crl (*C*[*tσ*] => *C*[*t σ*] in (Ω, *E*, *R*)) => true

if (\*[*u*1*σ*]:*s*1 in (Ω, *E*)) = true

.

/\ (\*[*u*j*σ*]:*s*j in (Ω, *E*)) = true

/\ (\*[*v*1*σ*] = \*[*v' σ*] in (Ω, *E*)) = true

1

.

/\ (\*[*v*k*σ*] = \*[*v' σ*] in (Ω, *E*)) = true

/\ (\*[*w*1*σ*] => \*[*w' σ*] in (Ω, *E*, *R*)) => true

k

1

.

/\ (\*[*w*h*σ*] => \*[*w' σ*] in (Ω, *E*, *R*)) => true .

h

\*\*\* transitivity

crl (\*[*t* -] => \*[*t'* -] in (Ω, *E*, *R*)) => true if parse(*t''*, Ω) = true

/\ (\*[*t* -] => \*[*t''* -] in (Ω, *E*, *R*)) => true

/\ (\*[*t''* -] => \*[*t'* -] in (Ω, *E*, *R*)) => true .

\*\*\* equality

crl (\*[*u* -] => \*[*u'* -] in (Ω, *E*, *R*)) => true if parse(*t*, Ω) = true

/\ parse(*t'*, Ω) = true

/\ (\*[*t* -] = \*[*u* -] in (Ω, *E*)) = true

/\ (\*[*t'* -] = \*[*u'* -] in (Ω, *E*)) = true

/\ (\*[*t* -] => \*[*t'* -] in (Ω, *E*, *R*)) => true .

Fig. 2. The universal theory URwl (fragment).

would also be proved by transitivity from *U ▶ ⟨T, t*1*⟩ −→ ⟨T, t*2*⟩* and *U ▶*

*⟨T, t*2*⟩ −→ ⟨T, t*3*⟩*. In our current approach, however, an entailment of the form *T ▶ t −→ t'* is reflected as *U ▶* (*t ⇒ t' in T* ) *−→ true*, and the “transitivity” rule (and also the “symmetry” rule in the case of membership equational logic) has to be explicitly reflected in the universal theory. 8

In addition, the results presented here *extend* in a natural way our pre- vious results on reflection in rewriting logic to other related logics, namely, membership equational logic, many-sorted equational logic, and Horn logic with equality. The extensions are very natural, in the sense that the proposed

8 Since we have only given here sketches of the proofs of correctness for the different universal theories, we leave for another occasion a comparison between both approaches, showing the advantages of the current approach.

universal theories are themselves related.

# Conclusion

We have shown that the generalized variant of rewriting logic where the under- lying equational specifications are membership equational theories, and where the rules are conditional and can have equations, memberships and rewrites in the conditions is reflective. We have also shown that membership equational logic, many sorted equational logic, and Horn logic with equality are likewise reflective. These results provide logical foundations for reflective languages and tools based on these logics, and in particular for the Maude language itself. The results presented here can be further developed and generalized in several directions, including:

* giving proofs of reflection for other more restrictive but frequently used logics, such as Horn logic without equality;
* further extending the rewriting logic results to theories where some of the operators are *frozen*, so that no rewriting is allowed under them, and to theories where some kinds can be empty;
* developing adequate strategies to execute the universal theories of rewriting logic and of membership equational logic in Maude, so that proof objects can be associated to reflective proofs when desired.

This work is one step further within a broader effort, whose first results appeared in [6], to develop a general theory of reflection for logics and declar- ative languages. In this regard, the results presented in this paper raise the issue of how the universal theories of related logics are themselves related. We expect that the metalogical foundations provided by the theory of gen- eral logics [9], which are at the base of our axiomatic approach to the study of reflection, will provide the concepts needed to address in a precise and formalism-independent way this issue.

# Acknowledgments

We thank Narciso Mart´ı-Oliet for many discussions on the topic of reflection in rewriting logic, and for his detailed and very helpful comments on earlier drafts of this paper.

# References

1. D. Basin, M. Clavel, and J. Meseguer. Rewriting logic as a metalogical framework. In S. Kapoor and S. Prasad, editors, *Twentieth Conference on the Foundations of Software Technology and Theoretical Computer Science, New Delhi, India, December 13–15, 2000, Proceedings*, volume 1974 of *Lecture Notes in Computer Science*, pages 55–80. Springer-Verlag, 2000.
2. D. Basin, M. Clavel, and J. Meseguer. Reflective metalogical frameworks. Submitted for publication, 2002.
3. M. Clavel. *Reflection in Rewriting Logic: Metalogical Foundations and Metaprogramming Applications*. CSLI Publications, 2000.
4. M. Clavel, F. Dur´an, S. Eker, P. Lincoln, N. Mart´ı-Oliet, J. Meseguer, and J. F. Quesada. Maude: Specification and programming in rewriting logic. Manual distributed as documentation of the Maude system, Computer Science Laboratory, SRI International. [http://maude.csl.sri.com/manual,](http://maude.csl.sri.com/manual) Jan. 1999.
5. [M. Clavel, F. Dura´n, S. Eker, P. Lincoln,](http://www.elsevier.nl/locate/entcs/volume36.html) N. Mart´ı-Oliet, J. Meseguer, and J. F. Quesada. Towards Maude 2.0. In K. Futatsugi, editor, *Proceedings Third International Workshop on Rewriting Logic and its Applications, WRLA 2000, Kanazawa, Japan, September 18–20, 2000*, volume 36 of *Electronic Notes in Theoretical Computer Science*, pages 297–318. Elsevier, 2000. [http://www.](http://www/) elsevier.nl/locate/entcs/volume36.html.
6. M. Clavel and J. Meseguer. Axiomatizing reflective logics and languages. In G. Kiczales, editor, *Proceedings of Reflection’96, San Francisco, California, April 1996*, pages 263–288, 1996.
7. M.Clavel and J.Meseguer. Reflection in conditional rewriting logic. *Theoretical Computer Science*, 285(2), Aug. 2002.
8. J. Goguen and J. Meseguer. Completeness of many-sorted equational logic.

*Houston Journal of Mathematics*, 11(3):307–334, 1985.

1. J. Meseguer. General logics. In H.-D. Ebbinghaus et al., editor, *Logic Colloquium’87*, pages 275–329. North-Holland, 1989.
2. J. Meseguer. Conditional rewriting logic as a unified model of concurrency.

*Theoretical Computer Science*, 96(1):73–155, 1992.

1. J. Meseguer. Membership algebra as a logical framework for equational specification. In F. Parisi-Presicce, editor, *Recent Trends in Algebraic Development Techniques, 12th International Workshop, WADT’97, Tarquinia, Italy, June 3–7, 1997, Selected Papers*, volume 1376 of *Lecture Notes in Computer Science*, pages 18–61. Springer-Verlag, 1998.
2. M. Palomino Tarjuelo. Comparing Meseguer’s rewriting logic with the logic CRWL. In M. Hanus, editor, *Proceedings WFLP 2001, International Workshop on Functional and (Constraint) Logic Programming*, volume 64 of *Electronic Notes in Theoretical Computer Science*. Elsevier, 2001.