Comparing Meseguer's Rewriting Logic with the Logic CRWL*

Miguel Palomino Tarjuelo

Dpto. Sistemas Informáticos y Programación
Facultad de Matemáticas, Universidad Complutense de Madrid, Spain
miguelpt@eucmos.sim.ucm.es

Abstract. Meseguer's rewriting logic and the rewriting logic CRWL are two well known approaches to rewriting as logical deduction that, despite some clear similarities, were designed with different objectives. Here we study the relationships between them, both at a syntactic and at a semantic level. It is not possible to establish an entailment system map between them, but both can be naturally simulated in each other. Semantically, there is no embedding between the corresponding institutions. We also use the syntactic results to prove reflective properties of CRWL and to extend those already known for Meseguer's rewriting logic.

1 Introduction

The aim of this paper is to study in detail, and to clarify to some extent, the relationships between two well known approaches to rewriting as logical deduction, namely, José Meseguer's rewriting logic [12], and the constructor-based rewriting logic (CRWL) developed by Mario Rodríguez Artalejo's research group in Madrid [9].

The first of these was proposed as a logical framework wherein to represent other logics, and also as a semantic framework for the specification of languages and systems. The experience accumulated throughout the last years has come to support that original intention; in particular, it has been shown that rewriting logic is a very flexible framework in which many other logics, including first-order logic, intuitionistic logic, linear logic, Horn logic with equality, as well as any other logic with a sequent calculus, can be represented. An important characteristic of these representations that should be stressed is that they are usually quite simple and natural, so that their mathematical properties are often straightforward to derive.

On the other hand, the goal of the constructor-based rewriting logic is to serve as a logical basis for declarative programming languages involving lazy evaluation, offering support, in addition, to non-strict and possibly non-deterministic functions.

* Supported by a postgraduate fellowship from the Spanish Ministry for Education, Culture and Sports.

Despite these differences there is a clear resemblance between both logics, namely, the fact that logical deduction is based on rewriting. It seems natural, then, to ask about the relationships between deduction in these logics, and to extend the question so as to encompass whether the corresponding models are also related.

A suitable framework in which to carry out this study is the theory of *general logics* developed by Meseguer [10]. There, a logic is described in a very abstract manner and two separated components are distinguished in it: an *entailment system* and an *institution*, corresponding with the syntactic and the semantic parts of the logic, respectively.

We will begin by studying derivability and, for that, we will try to associate entailment systems to both logics. Unfortunately, it will be proven that there is none corresponding to deduction in CRWL, and so we will be forced to leave the formal framework and undertake more informal simulations of the logics in each other. Although such simulations could be possible by making use of suitable down-level encodings, relying on the analogies between both logics, our interest resides in finding natural and simple simulations which at least show that their expressive power is the same. In addition, these results will be used to study reflective properties of both logics.

After the comparison at the syntactic level, the next step is the study of the corresponding models. Now we will be able to associate an institution to each logic, so this study will take place within the formal framework of the theory of institutions. The main result we will obtain is that models in these logics bear no relation at all, and through its way we will clarify some subtle points regarding the definition of models in Meseguer's rewriting logic.

A detailed account of all the results presented here can be found in [15].

Notation

We will write RL for Meseguer's rewriting logic; more precisely, for the version which uses unsorted and unconditional equational logic as its underlying logic. A signature in RL is a pair (Σ, E) with Σ a ranked alphabet of function symbols and E a set of Σ-equations. Rewriting operates on equivalence classes of terms modulo the set of equations E. We denote by $T_\Sigma(\mathcal{X})$ the Σ-algebra of Σ-terms with variables in a set $\mathcal{X}$, and by $[t]_E$ or just $[t]$ the E-equivalence class of $t \in T_\Sigma(\mathcal{X})$. To indicate that $\{x_1, \ldots, x_n\}$ is the set of variables occurring in t we write $t(x_1, \ldots, x_n)$. Given $t(x_1, \ldots, x_n)$, and terms $u_1, \ldots, u_n$, $t(u_1/x_1, \ldots, u_n/x_n)$ denotes the term obtained from t by *simultaneously substituting* u_i for x_i, $i = 1, \ldots, n$. To simplify notation we denote a sequence of objects $a_1, \ldots, a_n$ by $\overline{a}$.

An RL-theory $\mathcal{R}$ is a 4-tuple $\mathcal{R} = (\Sigma, E, L, \Gamma)$, where (Σ, E) is a signature and Γ is a set of rewrite rules, *labelled* with elements of L, of the form

$$r : [t] \to [t'] \textit{ if } [a_1] \to [b_1] \wedge \ldots \wedge [a_m] \to [b_m].$$

For a complete exposition of RL we refer the reader to [12].

CRWL uses signatures with constructors $\Sigma = C_\Sigma \cup F_\Sigma$, where C_Σ and F_Σ are disjoint sets of constructor and defined function symbols, respectively. $\Sigma_\perp$ refers to the signature which is obtained from Σ by adding a new constructor $\perp$ of arity 0. Given a set $\mathcal{X}$ of variables, we will write $\mathit{Expr}(\Sigma, \mathcal{X})$ for the set of expressions which can be built with Σ and $\mathcal{X}$, and $\mathit{Term}(\Sigma, \mathcal{X})$ for those terms which only make use of C_Σ and $\mathcal{X}$. $\mathit{Expr}_\perp(\Sigma, \mathcal{X})$ and $\mathit{Term}_\perp(\Sigma, \mathcal{X})$, the sets of partial expressions and terms, are defined analogously using $\Sigma_\perp$.

A CRWL-theory is a pair (Σ, Γ), where Σ is a signature with constructors, and Γ is a set of conditional rewrite rules of the form

$$f(t_1, \ldots, t_n) \to r \Leftarrow a_1 \bowtie b_1, \ldots, a_n \bowtie b_n \quad (m \geq 0),$$

with f a function symbol, and $t_1, \ldots, t_n \in \mathit{Term}(\Sigma, \mathcal{X})$. From a given theory two kinds of sentences are derived: reduction statements of the form $a \to b$, and joinability statements $a \bowtie b$ (meaning that there exists a total term to which both a and b reduce). Again, we refer to [9] for a presentation of the CRWL-calculus and its model-theoretic semantics. (Note that in [9] the names "term" and "constructor term" are used instead of "expression" and "term".)

2 Entailment Systems for RL and CRWL

We begin by studying the notion of derivation in RL and CRWL. This notion is abstractly captured by Meseguer's *entailment systems*, which are given by a category **Sign** of signatures together with, for each signature Σ, a set of sentences, and an entailment relation $\vdash_\Sigma$ between sets of sentences and sentences, satisfying some conditions. Among those conditions is transitivity: if $\Gamma \vdash_\Sigma \varphi_i$ for all $i \in I$, and $\Gamma \cup \{\varphi_i \mid i \in I\} \vdash_\Sigma \psi$, then $\Gamma \vdash_\Sigma \psi$. Our goal is to associate entailment systems to both RL and CRWL, and then to relate them by maps of entailment systems.

Assigning an entailment system to RL is a relatively straightforward task: either we restrict to unconditional rewrite rules and define $\vdash_\Sigma$ by means of derivation in the RL-calculus [12], or we also consider conditional rules, in which case the RL-calculus must be extended to be able to derive them [15]. The basic rewriting calculi for both logics are shown in Fig. 1.

At first sight, the same two possibilities hold for CRWL. However, a closer look reveals that derivation in the CRWL-calculus [9]. is not transitive. Consider, for example, a signature Σ with $c, d, h \in \Sigma$, function symbols of arities 0, 0, and 1, respectively. Then it can be proved that

$$\{\, c \to h(c), h(x) \to h(d), h(x) \to h(d) \Leftarrow x \bowtie x, \,\} \vdash_{\text{CRWL}} c \to h(d)$$
$$\{\, c \to h(c), h(x) \to h(d) \Leftarrow x \bowtie x \,\} \vdash_{\text{CRWL}} h(x) \to h(d),$$

but

$$\{\, c \to h(c), h(x) \to h(d) \Leftarrow x \bowtie x \,\} \not\vdash_{\text{CRWL}} c \to h(d).$$

The reason for this is that the CRWL-calculus is sound and complete with respect to validity in models *only* under totally defined valuations [9]. In particular, in

— For an RL-theory (Σ, E, L, Γ) —

$$\frac{}{[t]\to[t]} \qquad \frac{[t_1]\to[t'_1] \ \dots \ [t_n]\to[t'_n]}{[f(\overline{t})]\to[f(\overline{t'})]} \qquad \frac{[t]\to[t'] \ \ [t']\to[t'']}{[t]\to[t'']}$$

$$\frac{\begin{array}{c}[w_1]\to[w'_1] \ \dots \ [w_n]\to[w'_n] \\ [a_1(\overline{w}/\overline{x})]\to[b_1(\overline{w}/\overline{x})] \ \dots \ [a_m(\overline{w}/\overline{x})]\to[b_m(\overline{w}/\overline{x})]\end{array}}{[t(\overline{w}/\overline{x})]\to[t'(\overline{w'}/\overline{x})]} \ \text{if } r{:}[t]\to[t'] \ \textit{if} \ \ [a_1]\to[b_1]\wedge\ldots\wedge[a_n]\to[b_n]\in\Gamma$$

— For a CRWL-theory (Σ, Γ) —

$$\frac{}{e\to\bot} \qquad \frac{}{e\to e} \qquad \frac{e_1\to e'_1 \ \dots \ e_n\to e'_n}{h(\overline{e})\to h(\overline{e'})}$$

$$\frac{C}{l\to r} \ \text{for } l\to r\Leftarrow C \text{ an instance of a rule of } \Gamma \text{ with partial terms}$$

$$\frac{e\to e' \ \ e'\to e''}{e\to e''} \qquad \frac{a\to t \ \ b\to t}{a\bowtie b} \ \text{if } t \text{ is a total term}$$

Fig. 1. Rules of deduction of the RL and CRWL rewriting calculi

the second entailment above, $h(x) \to h(d)$ means that $h(t)$ rewrites to $h(d)$ just for those instances where a total term t is substituted for x. This proves that the relation $\vdash_{\mathrm{CRWL}}$ is not transitive and, therefore, we are not going to be able to build an entailment system based on the CRWL rewriting calculus.

3 Simulating CRWL in RL

As there does not exist an entailment system associated to the CRWL-calculus, we cannot define a map of entailment systems as intended. In the following we will be pleased just with presenting how entailment in CRWL can be simulated in RL. The set of labels of an RL-theory does not take part in the entailment process, and so it is omitted; the same convention will also be adopted in Sects. 4 and 5.

Of course, every CRWL-theory T can be trivially "simulated" in RL by means of an RL-theory T' with a constant c_t for each term (and each expression) t in T, and with axioms $c_t \to c_{t'}$ whenever $t \to t'$. But such a T' is not computable in general, and this is a property that will be required in Sect. 6 when we apply the simulation to the study of reflection in CRWL. And so we must look for another construction.

The idea is to associate to every CRWL-theory T a theory T' in RL (whose set of equational axioms will be empty) in which all the operations in T, together with a new constant $\bot$, are available, plus one rule for each axiom in T and, perhaps, some more rules coping with the rules of deduction of the CRWL-calculus. Since rules in CRWL can only be instantiated with terms and not expressions, we introduce a unary relation *pterm* (technically, a unary function) and a constant *true* to distinguish them in RL. One immediate rule defining *pterm* is $\mathit{pterm}(\bot) \to \mathit{true}$; however, how to express that variables are also partial terms? The obvious rule $\mathit{pterm}(x) \to \mathit{true}$ is clearly not valid: everything would be a partial term! This means that we must add the CRWL variables to the signature of T' as constants, and use a new set $\mathcal{X}$ of variables for RL. This way, assuming variables in CRWL belong to a set $\mathcal{V}$, the rules defining *pterm* will be

$$
\begin{aligned}
&pterm(\perp) \to true\\
&pterm(v_i) \to true \quad (\forall v_i \in \mathcal{V})\\
&pterm(h(x_1,\dots,x_n)) \to true\\
&\qquad \mathit{if}\ \ pterm(x_1) \to true \wedge \dots \wedge pterm(x_n) \to true \quad (\forall h \in C^n_\Sigma, n \in \mathbb{N}),
\end{aligned}
$$

In a similar way, two more predicates, *pexpr* and *tterm*, dealing with partial expressions and total terms, are defined [15].

As a side effect, rewriting in CRWL can no longer be simulated in RL directly through the rewriting relation. Consider, for example, the theory of natural numbers in CRWL, with 0 a constructor and + a function symbol. In RL, $pterm(0+0)$ should not rewrite to *true*; however, with the usual definitions, $0+0 \to 0$ and by congruence $pterm(0+0) \to pterm(0)$, and this last term must reduce to *true*. Therefore, a rewrite in CRWL will be simulated through a binary relation R so that $e \to e'$ in CRWL if and only if $R(e, e') \to true$ in RL. In a similar way, strict equalities $a \bowtie b$ will be simulated through a binary relation $\bowtie$.

It just remains to translate the rules of deduction of the CRWL-calculus, which is straightforward. For example, the bottom rule stating that every expression is reducible to $\perp$ is written

$$R(x, \perp) \to true \ \ \mathit{if}\ pexpr(x) \to true,$$

whereas the joinability rule

$$\frac{a \to t \qquad b \to t}{a \bowtie b} \quad \text{if } t \text{ is a total term}$$

results in

$$x \bowtie y \to true \ \ \mathit{if}\ R(x,z) \to true \wedge R(y,z) \to true \wedge tterm(z) \to true.$$

Finally, to every rule $l(\overline{v}) \to r(\overline{v}) \Leftarrow a_1(\overline{v}) \bowtie b_1(\overline{v}), \dots, a_m(\overline{v}) \bowtie b_m(\overline{v})$ in the CRWL-theory, we associate the following rule in RL

$$
\begin{aligned}
&R(l(\overline{x}), r(\overline{x})) \to true\\
&\qquad \mathit{if}\ \ a_1(\overline{x}) \bowtie b_1(\overline{x}) \to true \wedge \dots \wedge a_m(\overline{x}) \bowtie b_m(\overline{x}) \to true \wedge\\
&\qquad\quad pterm(x_1) \to true \wedge \dots \wedge pterm(x_n) \to true,
\end{aligned}
$$

where each CRWL variable v_i (a constant in the RL-theory) has been substituted by the variable x_i. This last rule corresponds with the reduction rule in the CRWL-calculus, and the condition that program rules in CRWL can only be instantiated with terms is taken care of by demanding $pterm(x) \to true$ for all the variables appearing in it.

We will write $\alpha(T)$ for the RL-theory associated to a CRWL-theory T. The following proposition ensures us that the translation is correct.

Proposition 1. *Given a CRWL-theory* $T = (\Sigma, \Gamma)$ *such that* $\alpha(T) = (\Sigma', \emptyset, \Gamma')$, *then, if* $l, r, a, b \in T_{\Sigma'}(\mathcal{X})$:

$$
\begin{aligned}
&l, r \in Expr_\perp(\Sigma, \mathcal{V}) \ \mathit{and}\ T \vdash_{\mathrm{CRWL}} l \to r \iff \alpha(T) \vdash_{\mathrm{RL}} R(l,r) \to true;\\
&a, b \in Expr_\perp(\Sigma, \mathcal{V}) \ \mathit{and}\ T \vdash_{\mathrm{CRWL}} a \bowtie b \iff \alpha(T) \vdash_{\mathrm{RL}} a \bowtie b \to true.
\end{aligned}
$$

4 Simulating RL in CRWL

We now embark ourselves on findind the converse simulation of RL in CRWL. We are still interested in a computable and simple translation, and the idea for this is very similar to that of the previous section. Now, however, there are no terms and expressions to distinguish, and so predicates such as *pterm* are no longer necessary; as a consequence, we will be able to use the same set $\mathcal{X}$ of variables for both logics. The fact that only joinability statements are allowed to appear in the condition of a rewrite rule in CRWL forces us to represent, as in Sect. 3, the rewriting relation in RL through a binary relation R in CRWL, so that $t \to t'$ in RL if and only if $R(t, t') \to \mathit{true}$ in CRWL; rewriting modulo a set of equations will be handled by transforming each equation $t = t'$ into the rewrites $t \to t'$ and $t' \to t$.

More precisely, given a signature (Σ, E) in RL we associate to it a CRWL-theory over the signature Σ' with $C_{\Sigma'} = \Sigma \cup \{\mathit{true}\}$ and $F_{\Sigma'} = \{R\}$, with *true* and R of arities 0 and 2, respectively. The rules in the theory are

$$\begin{aligned} R(x_1, x_2) &\to \mathit{true} \Leftarrow x_1 \bowtie x_2, \\ R(x, y) &\to \mathit{true} \Leftarrow R(x, z) \bowtie \mathit{true}, R(z, y) \bowtie \mathit{true}, \end{aligned}$$

and, for each $f \in \Sigma_n$, $n \in \mathbb{N}$,

$$R(f(x_1, \ldots, x_n), f(y_1, \ldots, y_n)) \to \mathit{true} \Leftarrow R(x_1, y_1) \bowtie \mathit{true}, \ldots, R(x_n, y_n) \bowtie \mathit{true},$$

mimicking the reflexivity, transitivity, and congruence rules in the RL-calculus, together with

$$R(t, t') \to \mathit{true} \qquad \text{and} \qquad R(t', t) \to \mathit{true},$$

for every $t = t' \in E$. The goal of the condition in the rule corresponding to reflexivity is to avoid instantiating it with terms containing $\perp$, which have no meaning in RL.

A conditional rewrite rule

$$[l] \to [r] \ \mathit{if} \ [a_1] \to [b_1] \wedge \ldots \wedge [a_m] \to [b_m]$$

over (Σ, E) in RL is then translated to

$$R(l, r) \to \mathit{true} \Leftarrow R(a_1, b_1) \bowtie \mathit{true}, \ldots, R(a_m, b_m) \bowtie \mathit{true},$$

where l, r, a_i, b_i are arbitrary members of $[l]$, $[r]$, $[a_i]$ and $[b_i]$, respectively.

In fact, the previous definitions must be slightly modified due to some technical details. In a conditional rewrite rule $l \to r \Leftarrow C$ in CRWL, l must be linear, and it is obvious that with the above definitions this property is not ensured for the translation of equations and rewrite rules; therefore, those rules must be "linearised" (see [1, 15]).

In what follows, we write $\beta(T)$ for the CRWL-theory associated to a RL-theory T. The next proposition guarantees the correctness of the translation.

Proposition 2. *Given any RL-theory $T = (\Sigma, E, \Gamma)$, and $l, r \in T_\Sigma(\mathcal{X})$:*

$$\begin{aligned} T \vdash_{\mathrm{RL}} [l] \to [r] &\iff (\exists l' \in [l], \exists r' \in [r]) \ \ \beta(T) \vdash_{\mathrm{CRWL}} R(l', r') \to \mathit{true} \\ &\iff (\forall l' \in [l], \forall r' \in [r]) \ \ \beta(T) \vdash_{\mathrm{CRWL}} R(l', r') \to \mathit{true}. \end{aligned}$$

5 Simulating Rewriting Logic over Membership Equational Logic in CRWL

The results presented in Sect. 4 can be extended to the case in which the underlying equational logic for Meseguer's rewriting logic is membership equational logic [13]; we write MERL for it. As there, a binary function symbol R can be used to simulate the rewriting relation in CRWL: the goal in mind is having $t \to t'$ in MERL if and only if $R(t, t') \to \mathit{true}$ in CRWL. Now, in addition, a binary function symbol I is needed to represent equality in membership equational logic ($t = t' \iff I(t, t') \to \mathit{true}$), as well as another one M to simulate memberships ($t : s \iff M(t, s) \to \mathit{true}$).

A further distinction concerns the treatment of variables. CRWL has no types so, given the function symbols of a membership equational signature, we could use them to build more terms than those that we would have in the original logic and, moreover, the use of such terms would enlarge the set of provable statements in a theory. So it will be necessary to distinguish somehow the well-typed terms and, for that, we are forced again to represent variables in MERL as constants in CRWL. The task of recognising well-typed terms will be carried out by a binary function symbol wtterm in such a way that $\mathit{wtterm}(t, k) \to \mathit{true}$ if and only if the term t has kind k. We can use $\mathcal{V}$ for the set of variables in MERL and $\mathcal{X}$ for the variables in CRWL.

Let then $((K, \Sigma, S), E)$ be a MERL signature, i.e., a membership equational logic theory. We associate to it a CRWL-theory with signature Σ' such that $C_{\Sigma'} = \Sigma \cup \{\mathit{true}\} \cup K \cup S \cup \mathcal{V}$ (the elements of K, S and $\mathcal{V}$ are constants) and $F_{\Sigma'} = \{\mathit{wtterm}, R, I, M\}$. The rules defining wtterm are:

$$\begin{array}{ll} \mathit{wtterm}(v, k) \to \mathit{true} & (\forall v \in \mathcal{V} \text{ of kind } k \in K), \\ \mathit{wtterm}(c, k) \to \mathit{true} & (\forall c : k \in \Sigma), \end{array}$$

and, for all $f : k_1 \dots k_n \to k \in \Sigma$,

$$\begin{array}{l} \mathit{wtterm}(f(x_1, \dots, x_n), k) \to \mathit{true} \\ \qquad \Leftarrow \mathit{wtterm}(x_1, k_1) \bowtie \mathit{true}, \dots, \mathit{wtterm}(x_n, k_n) \bowtie \mathit{true}. \end{array}$$

The translation of the rules of deduction of the equational and rewriting calculus goes along the same lines as that of Sect. 4. For example, modus ponens for an equation

$$\begin{array}{rl} (\forall \mathcal{W})\ t(\overline{v}) = t'(\overline{v}) \Leftarrow & a_1(\overline{v}) = b_1(\overline{v}) \wedge \dots \wedge a_m(\overline{v}) = b_m(\overline{v}) \wedge \\ & w_1(\overline{v}) : s_1 \wedge \dots \wedge w_p(\overline{v}) : s_p, \end{array}$$

with $\mathcal{W} \subseteq \mathcal{V}$ containing all variables in $\overline{v}$, is simulated by means of (the linearised version of) the rule

$$\begin{array}{rl} I(t(\overline{x}), t'(\overline{x})) \to \mathit{true} \Leftarrow & I(a_1(\overline{x}), b_1(\overline{x})) \bowtie \mathit{true}, \dots, I(a_m(\overline{x}), b_m(\overline{x})) \bowtie \mathit{true}, \\ & M(w_1(\overline{x}), s_1) \bowtie \mathit{true}, \dots, M(w_p(\overline{x}), s_p) \bowtie \mathit{true}, \\ & \mathit{wtterm}(x_1, k_1) \bowtie \mathit{true}, \dots, \mathit{wtterm}(x_n, k_n) \bowtie \mathit{true}, \end{array}$$

where k_i is the kind of v_i; and similarly (replacing I with M) for memberships.

The translation of a rewrite rule

$$[l(\overline{v})] \to [r(\overline{v})] \textit{ if } [a_1(\overline{v})] \to [b_1(\overline{v})] \wedge \ldots [a_m(\overline{v})] \to [b_m(\overline{v})]$$

is

$$\begin{aligned} R(l(\overline{x}), r(\overline{x})) \to \textit{true} \Leftarrow\ & R(a_1(\overline{x}), b_1(\overline{x})) \bowtie \textit{true}, \ldots, R(a_m(\overline{x}), b_m(\overline{x})) \bowtie \textit{true}, \\ & \textit{wtterm}(x_1, k_1) \bowtie \textit{true}, \ldots, \textit{wtterm}(x_n, k_n) \bowtie \textit{true}, \end{aligned}$$

with k_i the kind of v_i.

In what follows, we will write $\delta(T)$ for the CRWL-theory associated to a MERL-theory T. We have the following two main results.

Proposition 3. *Let T be an RL-theory with signature $((K, \Sigma, S), E)$, $\delta(T) = (\Sigma', \Gamma')$, and let $t, t', s \in \mathit{Expr}_\perp(\Sigma', \mathcal{X})$:*

$$\begin{aligned} \delta(T) \vdash_{\mathrm{CRWL}} I(t, t') \to \textit{true} &\Longleftrightarrow (\exists k \in K)\ t, t' \in T_\Sigma(\mathcal{V})_k \textit{ and } E \vdash (\forall \mathcal{V})\ t = t'; \\ \delta(T) \vdash_{\mathrm{CRWL}} M(t, s) \to \textit{true} &\Longleftrightarrow (\exists k \in K)\ t \in T_\Sigma(\mathcal{V})_k, s \in S_k \textit{ and } E \vdash (\forall \mathcal{V})\ t : s. \end{aligned}$$

Proposition 4. *Given any RL-theory T with signature $((K, \Sigma, S), E)$ and set of rules Γ, and given $l, r \in T_\Sigma(\mathcal{V})$, the following are equivalent:*

1. $T \vdash_{\mathrm{MERL}} [l] \to [r]$;
2. $(\exists l' \in [l], \exists r' \in [r])\ \delta(T) \vdash_{\mathrm{CRWL}} R(l', r') \to \textit{true}$;
3. $(\forall l' \in [l], \forall r' \in [r])\ \delta(T) \vdash_{\mathrm{CRWL}} R(l', r') \to \textit{true}$.

6 Reflection in CRWL and in RL

Intuitively, a reflective logic is a logic in which important aspects of its metatheory, such as the concepts of theory and entailment, can be represented and reasoned about *in* the logic. A general axiomatic notion of reflective logic was proposed by Clavel and Meseguer [3, 5]. The notion is itself expressed in terms of the notion of an entailment system.

Given an entailment system $\mathcal{E}$ and a nonempty set of theories $\mathcal{C}$ in it, a theory U is $\mathcal{C}$-universal if there is a function, called a representation function,

$$\overline{(\_ \vdash \_)} : \bigcup_{T \in \mathcal{C}} (\{T\} \times \mathit{sen}(T)) \to \mathit{sen}(U),$$

such that for each $T \in \mathcal{C}$, $\varphi \in \mathit{sen}(T)$,

$$T \vdash \varphi \quad \text{iff} \quad U \vdash \overline{T \vdash \varphi}.$$

If, in addition, $U \in \mathcal{C}$, then the entailment system $\mathcal{E}$ is called $\mathcal{C}$-reflective. Finally, a reflective logic is a logic whose entailment system is $\mathcal{C}$-reflective for $\mathcal{C}$ the class of all finitely presentable theories in the logic. The condition that the representation function be computable and injective has also been required recently [6].

RL has been proved to be reflective in [3, 5, 6] and we will use this result to obtain an analogous one for CRWL. Strictly speaking, the notion of reflection does not apply to CRWL as it was observed in Sect. 3 that CRWL does not have an associated entailment system. Even in the case of RL, except for the original result in [3], subsequent generalizations do not immediately fit within the formal definition of reflection as they allow conditional sentences on the left of the entailment relation but not on the right. For this reason, in what follows we consider a "loose" definition of reflection (making use of some kind of "weak" entailment system) general enough to encompass all the cases just discussed.

Let $\mathcal{U}$ be a universal theory for RL; we will use it to prove that CRWL is also reflective. In Sects. 3 and 4 we defined mappings α and β that map a theory T in CRWL, respectively in RL, to a theory in RL, respectively in CRWL, which simulates the behaviour of T. By abuse of notation we will also use α and β to name the translation of sequents defined in those sections. (Recall that $\alpha(T)$ is *not* obtained by simply applying α to every $\varphi \in T$, and analogously for β.) Then, given an arbitrary CRWL-theory T and a statement φ, we have the following chain of equivalences:

$$\begin{aligned} T \vdash_{\mathrm{CRWL}} \varphi &\iff \alpha(T) \vdash_{\mathrm{RL}} \alpha(\varphi) \\ &\iff \mathcal{U} \vdash_{\mathrm{RL}} \overline{\alpha(T) \vdash \alpha(\varphi)} \\ &\iff \beta(\mathcal{U}) \vdash_{\mathrm{CRWL}} \beta(\overline{\alpha(T) \vdash \alpha(\varphi)}), \end{aligned}$$

which proves that $\beta(\mathcal{U})$ is universal in CRWL (and, since β was quite simple, no much more complicated than the original $\mathcal{U}$).

As it has been pointed out previously, the reflective results about RL have been proved only for special cases. The one we need here is that in which the underlying equational logic is unsorted and the rules are conditional [6]. Moreover, the results presented here allow us to extend the reflective results to the full power of MERL. For that we can consider that α yields theories in MERL, because unsorted equational logic is naturally embeddable in membership equational logic, and let δ be the function defined in Sect. 5 such that, for T a MERL-theory, $T \vdash_{\mathrm{MERL}} \varphi \iff \delta(T) \vdash_{\mathrm{CRWL}} \delta(\varphi)$. Then, for $\mathcal{W}$ any universal theory in CRWL (like $\beta(\mathcal{U})$, for example),

$$\begin{aligned} T \vdash_{\mathrm{MERL}} \varphi &\iff \delta(T) \vdash_{\mathrm{CRWL}} \delta(\varphi) \\ &\iff \mathcal{W} \vdash_{\mathrm{CRWL}} \overline{\delta(T) \vdash \delta(\varphi)} \\ &\iff \alpha(\mathcal{W}) \vdash_{\mathrm{RL}} \alpha(\overline{\delta(T) \vdash \delta(\varphi)}), \end{aligned}$$

which proves that $\alpha(W)$ is universal in MERL.

In Maude, a specification and programming language based on RL, reflection is exploited systematically to extend the language with program transformation methods and internal strategies [4]. In particular, a flexible and robust module algebra incorporating parameterisation and object-oriented features into the language has been built in [8]. In $\mathcal{TOY}$, an experimental language and system that implements the CRWL paradigm, up til now reflection has played no role at all. A possible reason could be that reflection on its own were not a sufficient condition for many results, but required the existence of other logical properties. On the other hand, it could simply be that more attention has to be paid to it, so that a closer look may reveal some possible applications.

7 An Institution for CRWL

In this section we leave behind our study of the entailment relation and turn our attention to models and satisfaction. Our interest consists in associating suitable institutions to both CRWL and RL and, thereafter, to relate them via maps of institutions with "good" properties.

A first step towards this direction is taken in [14], where an institution is associated to CRWL. This institution is defined with the goal of providing a basis for the semantics of modules in CRWL, and restricts its attention to a class of particular term algebras. Since our objective is more general, we do not place such a limitation and define $\mathcal{I}_{\text{CRWL}} = (\mathbf{Sign}, sen, \mathbf{Mod}, \models)$ as follows:

- **Sign**: the category of signatures with constructors and signature morphisms (for the definition of signature morphisms, see [14]);
- sen : **Sign** $\rightarrow$ **Set** the functor assigning to each signature Σ the set of all conditional rewrite rules over it, and to each signature morphism, its extension to rewrite rules;
- **Mod** : $\mathbf{Sign}^{\text{op}} \rightarrow$ **Cat** the functor assigning to each signature the category of CRWL-algebras and homomorphisms over it, and to each $\sigma : \Sigma \rightarrow \Sigma'$ the forgetful functor taking $\mathcal{A}' \in |\mathbf{Mod}(\Sigma')|$ to the CRWL-algebra $\mathcal{A}'_\sigma$ with the same underlying poset and such that $h^{\mathcal{A}'_\sigma} = \sigma(h)^{\mathcal{A}'}$ for all $h \in \Sigma$, and which is the identity over homomorphisms;
- $\models$ the satisfaction relation in CRWL.

Proposition 5. *$\mathcal{I}_{\text{CRWL}}$ is an institution.*

It can be proved that the category $\mathbf{Mod}(T)$ has products for every CRWL-theory T; it is not complete, however, as in Sect. 9 it is shown that, in general, $\mathbf{Mod}(T)$ does not have equalizers. $\mathcal{I}_{\text{CRWL}}$ is also a semiexact institution [15].

8 An Institution for RL

The task of assigning an institution to RL is harder than expected. The first and more natural idea is to define the category of signatures **Sign** as the category of equational theories and theory morphisms, and the functor sen to map any such theory to the set of conditional rewrite rules over it. Since there are also notions of model and satisfaction in RL [12], the desired institution seems to be at hand. However, when one tries to put together the various components of the institution, problems start to arise. In the first place, the notion of satisfaction in RL is defined only for unconditional rewrite rules, so our first task must be to extend its definition so as to encompass the conditional ones. Although there are at least two possible ways in which this could be done, this is a relatively minor problem which can be solved by mirroring the definition of $\mathcal{R}$-systems (see [15]).

A far more serious problem is posed by the functor $\mathbf{Mod} : \mathbf{Sign}^{\mathrm{op}} \to \mathbf{Cat}$, mapping signatures to models. The difficulty resides in the fact that, in RL, models are assigned directly to RL-theories instead of signatures, as it is customary in other logics. One obvious solution would be to consider a signature (Σ, E) as a theory $\mathcal{R} = (\Sigma, E, \emptyset, \emptyset)$ with empty set of axioms (and labels), and to map (Σ, E) to the category $\mathcal{R}$-$\mathbf{Sys}$ of models of $\mathcal{R}$ [12]. But this approach presents an important drawback. Up to this point in the paper, we have omitted any explicit mention of the set of labels of an RL-theory. Although this was a safe convention when talking about deduction, it is not longer the case when our interest shifts to models. Due to the set of labels L in an RL-theory $\mathcal{R} = (\Sigma, E, L, \Gamma)$, the elements of Γ become special, *labelled* rewrite rules. These rules force $\mathcal{R}$-systems to have a certain internal structure: not only $\mathcal{R}$-systems must satisfy them, but also must associate to them a *distinguished* interpretation (natural transformation) that must be preserved by homomorphisms[1]. When considering a signature as a theory with empty sets of axioms, we are not taking into account labelled rewrite rules. This way, homomorphisms are not subjected to preserve any rewrite rule and the categories $\mathbf{Mod}(\Gamma)$ and $\mathcal{R}$-$\mathbf{Sys}$ turn out to be different, in opposition to our original intention.

Keeping **Sign** as the category of equational theories, there are other possibilities as to how to define **Mod** (and even $\models$) but, since they cannot reflect the distinction between labelled rules belonging to RL-theories and unlabelled rules, all of them are bound to failure (see [15] for a detailed discussion). For this reason we are led to an institution in which the category **Sign** subsumes all the information of an RL-theory. More precisely, we define $\mathcal{I}_{\mathrm{RL}} = (\mathbf{Sign}, sen, \mathbf{Mod}, \models)$ where:

- **Sign** is the *discrete* category of RL-theories;
- $sen : \mathbf{Sign} \to \mathbf{Set}$ maps each RL-theory to its corresponding set of conditional rewrite rules;
- $\mathbf{Mod} : \mathbf{Sign}^{\mathrm{op}} \to \mathbf{Cat}$ maps an RL-theory $\mathcal{R}$ to the category $\mathcal{R}$-$\mathbf{Sys}$;
- $\models$ the satisfaction relation conveniently extended to conditional rewrite rules as discussed above.

Since **Sign** is discrete, this trivially defines an institution. Admittedly, this restriction seems to be not justified. In fact, two types of morphisms of RL-theories are proposed in [11]. Basically, they are equational theory morphisms "preserving" the rules in the RL-theories; a complete study of the institution $\mathcal{I}_{\mathrm{RL}}$ extended with these morphisms will be undertaken in a future occasion. For our purposes, the present definition is general enough as it stands, and its extension would not modify the use we will make of it in the next section.

There exist other institutions associated to (varieties of) RL in the literature, e.g., [2, 7]; in these papers, the objects in the category of signatures are the sets

[1] In particular, the same rule can appear *twice* in an RL-theory $\mathcal{R}$ under two different labels. $\mathcal{R}$-systems are then forced to provide two, possibly different, interpretations for the same rule, each of them to be preserved by the homomorphisms.

of function symbols, without any rules. As a consequence of this simplicity and because of the reasons we have mentioned above, the general categorical models of RL must be somehow restricted, and the choice in these two papers is to require them to be preorders instead of general categories.

9 Searching for Embeddings

We would like to relate the institutions $\mathcal{I}_{\mathrm{CRWL}}$ and $\mathcal{I}_{\mathrm{RL}}$ by means of a map of institutions $(\Phi, \alpha, \beta) : \mathcal{I}_{\mathrm{CRWL}} \to \mathcal{I}_{\mathrm{RL}}$ having nice properties, in such a way that it indicated that $\mathcal{I}_{\mathrm{CRWL}}$ could be considered as a subinstitution of $\mathcal{I}_{\mathrm{RL}}$. The formal definition of subinstitution appeared originally in [10] and has been further generalized in subsequent articles. One of those extensions was introduced by Meseguer in [13], where it is called an *embedding*. The only requirement imposed on a map of institutions $(\Phi, \alpha, \beta) : \mathcal{I} \to \mathcal{I}'$ to be an embedding is that for each $T \in \mathbf{Th}_{\mathcal{I}}$, the functor $\beta_T : \mathbf{Mod}'(\Phi(T)) \to \mathbf{Mod}(T)$ must be an equivalence of categories.

We will show, however, that there is no embedding from $\mathcal{I}_{\mathrm{CRWL}}$ into $\mathcal{I}_{\mathrm{RL}}$. For that it will be enough to find a categorical property which is preserved by an equivalence of categories and a theory $T \in \mathbf{Th}_{\mathrm{CRWL}}$ such that $\mathbf{Mod}_{\mathrm{RL}}(\Phi(T))$, but not $\mathbf{Mod}_{\mathrm{CRWL}}(T)$, has the property.

Let Σ be a signature with constructors such that $C_\Sigma = \emptyset$ and F_Σ consists of just two constants f_1 and f_2, $\Gamma = \{f_2 \to x \Leftarrow f_1 \bowtie f_1\}$, and consider the CRWL-theory $T = (\Sigma, \Gamma)$. We define two CRWL-algebras over Σ: $\mathcal{A}$ given by the set $D^{\mathcal{A}} = \{\perp, a_1, a_2\}$ with partial order $\perp \sqsubseteq a_1 \sqsubseteq a_2$ and the cones $f_1^{\mathcal{A}} = \langle a_1 \rangle$ and $f_2^{\mathcal{A}} = \langle \perp \rangle$; and $\mathcal{B}$ with $D^{\mathcal{B}} = \{\perp, b_1\}$ and the cones $f_1^{\mathcal{B}} = f_2^{\mathcal{B}} = \langle \perp \rangle$. $\mathcal{A}, \mathcal{B} \in \mathbf{Mod}_{\mathrm{CRWL}}(T)$ trivially, because they do not satisfy the condition $f_1 \bowtie f_1$.

Let us now define two CRWL-homomorphisms $F, G : \mathcal{A} \to \mathcal{B}$, given by:

$$F(x) = \langle \perp \rangle \;\text{ and }\; G(x) = \begin{cases} \langle \perp \rangle \text{ if } x = \perp, a_1 \\ \langle b_1 \rangle \text{ if } x = a_2. \end{cases}$$

Clearly, F and G preserve both f_1 and f_2, so that they are actually homomorphisms. It can be shown that there does not exist the equalizer of F and G.

In contrast with what happens in CRWL, equalizers can be built in $\mathcal{R}$-**Sys** by mimicking the construction in **Cat**[15]. Since an equivalence of categories preserves limits, we have:

Proposition 6. *$\mathcal{I}_{\mathrm{CRWL}}$ is not embeddable in $\mathcal{I}_{\mathrm{RL}}$.*

What about the other way around? Can we embed $\mathcal{I}_{\mathrm{RL}}$ in $\mathcal{I}_{\mathrm{CRWL}}$? In order to prove that RL cannot be embedded in CRWL we have to find an RL-theory T such that $\mathbf{Mod}_{\mathrm{RL}}(T)$ has a categorical property that no category of models in CRWL has. For that, note that for any CRWL-theory T there exists a CRWL-algebra $\mathcal{A} \in |\mathbf{Mod}_{\mathrm{CRWL}}(T)|$ with an infinite number of automorphisms. Simply consider $\mathcal{A}$ given by $D^{\mathcal{A}} = \{\perp, a, b_1, b_2, \ldots\}$ with $\perp \sqsubseteq a$, $\perp \sqsubseteq b_1 \sqsubseteq b_2 \sqsubseteq \ldots$, the image of all functions associated to constructor symbols to be $\langle a \rangle$, and the corresponding one of defined function symbols to be $D^{\mathcal{A}}$. This way $\mathcal{A}$ is clearly a

CRWL-algebra, satisfies all conditional rewrite rules, and leaving $\perp$ and a fixed, and mapping all b_i to some fixed b_j we get an infinite family of automorphisms of $\mathcal{A}$. On the other hand, in RL, if $\mathcal{R}$ is the RL-theory given by $(\{c\}, \{x = c\}, \emptyset, \emptyset)$ then, for all $\mathcal{R}$-systems $\mathcal{S}$, the equality $id_{\mathcal{S}} = c_{\mathcal{S}}$, where $c_{\mathcal{S}}$ is a constant functor, forces $\mathcal{S}$ to be a category with just one object and one arrow, and no infinite family of homomorphisms can exist. Therefore (as an equivalence of categories is full and faithful), $\mathbf{Mod}_{\mathrm{RL}}(\mathcal{R})$ is not categorically equivalent to $\mathbf{Mod}_{\mathrm{CRWL}}(\Phi(\mathcal{R}))$, whatever Φ might be.

Proposition 7. *$\mathcal{I}_{\mathrm{RL}}$ is not embeddable in $\mathcal{I}_{\mathrm{CRWL}}$.*

Admittedly, although this counterexample formally solves the problem, it is so particular that it cannot be considered to truly reflect the real differences between RL and CRWL. Future work should concentrate on finding a more illustrative example.

10 Conclusions

The main outcome of the research carried out in this paper has been the clarification of the relationship between RL and CRWL. Both logics have been proved to be expressive enough to simulate deduction in each other in a simple way, though resorting to binary predicates. On the other hand, the results on institutions have shown that neither can RL be considered as a sublogic of CRWL, nor can CRWL with respect to RL.

During the preparation of this work we have been forced to take a close look at the notions of entailment system and institution, and the difficulties we have found have shown us that intuition can be misleading in this field. The conclusion we have reached is that it would be very convenient to develop some kind of generalization of these concepts. One reason supporting this claim is the fact that, although it seems clear that CRWL should fit within the frame of entailment systems, the lack of the transitivity property forbids it to be considered so. In addition, there have been several occasions wherein we have had to make a distinction between two types of sentences within the same logic. The most outstanding case was that of labelled and unlabelled rewrite rules in RL, but we should also emphasize that rules in CRWL-programs are a restricted class of the more general class of reduction statements, and that, when we talked about reflection, we had to "weaken" its definition in order to encompass some of the results about RL, due to conditional sentences not being treated like unconditional ones. What all these examples have in common is that sentences belonging to a theory are given a different treatment from the rest of sentences and, with the current definitions of entailment system and institution, there is no way of taking this distinction into account.

For future work, besides the generalizations just mentioned, it would be interesting to complete the definition of the institution $\mathcal{I}_{\mathrm{RL}}$ with signature morphisms (morphisms of RL-theories), as well as finding a more illustrative example showing the reasons why RL is not embeddable in CRWL.

Acknowledgments

The author is indebted to Narciso Martí-Oliet, José Meseguer, Mario Rodríguez-Artalejo and the anonymous referees, for their valuable comments.

References

1. P. Arenas-Sánchez and M. Rodríguez-Artalejo. A general framework for lazy functional logic programming with algebraic polymorphic types. *Theory and Practice of Logic Programming*, 1(2):185–245, Mar. 2001.
2. M. V. Cengarle. The rewriting logic institution. Technical Report 9801, Ludwig-Maximilians-Universität München, Institut für Informatik, May 1998.
3. M. Clavel. *Reflection in General Logics and in Rewriting Logic, with Applications to the Maude Language*. PhD thesis, Universidad de Navarra, Spain, Feb. 1998.
4. , M. Clavel, F. Durán, S. Eker, P. Lincoln, N. Martí-Oliet, J. Meseguer, and J. F. Quesada. Maude: Specification and programming in rewriting logic, *Theoretical Computer Science*, 2001. To appear.
5. M. Clavel. *Reflection in Rewriting Logic: Metalogical Foundations and Metaprogramming Applications*. CSLI Lecture Notes. CSLI Publications, 2000.
6. M. Clavel and J. Meseguer. Reflection in conditional rewriting logic. *Theoretical Computer Science*, 2001. To appear.
7. R. Diaconescu and K. Futatsugi. Logical foundations of CafeOBJ. *Theoretical Computer Science*, 2001. To appear.
8. F. Durán. *A Reflective Module Algebra with Applications to the Maude Language*. PhD thesis, Universidad de Málaga, Spain, Jun. 1999.
9. J. C. González-Moreno, M. T. Hortalá-González, F. J. López-Fraguas, and M. Rodríguez-Artalejo. An approach to declarative programming based on a rewriting logic. *Journal of Logic Programming*, 40:47–87, 1999.
10. J. Meseguer. General logics. In H.-D. Ebbinghaus, J. Fernández-Prida, M. Garrido, D. Lascar, and M. Rodríguez-Artalejo, editors, *Logic Colloquium'87*, pages 275–329. North-Holland, 1989.
11. J. Meseguer. Rewriting as a unified model of concurrency. Technical Report SRI-CSL-90-02, SRI International, Computer Science Laboratory, Feb. 1990. Revised June 1990.
12. J. Meseguer. Conditional rewriting logic as a unified model of concurrency. *Theoretical Computer Science*, 96(1):73–155, 1992.
13. 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.
14. J. M. Molina-Bravo. *Modularidad en Programación Lógico-Funcional de Primer Orden*. PhD thesis, Universidad de Málaga, Spain, Dec. 2000.
15. M. Palomino Tarjuelo. Relating Meseguer's rewriting logic with the constructor-based rewriting logic. Master's thesis, Facultad de Matemáticas, Universidad Complutense de Madrid, May 2001.