# Monads and Modular Term Rewriting

Christoph Lüth[1] and Neil Ghani[2]

[1] Bremen Institute for Safe Systems, FB 3, Universität Bremen
Postfach 330440, 28334 Bremen
cxl@informatik.uni-bremen.de
[2] The School Of Computer Science
University of Birmingham, Birmingham, England
nxg@cs.bham.ac.uk

**Abstract.** Monads can be used to model term rewriting systems by generalising the well-known equivalence between universal algebra and monads on the category **Set**. In [Lü96], the usefulness of this semantics was demonstrated by giving a purely categorical proof of the modularity of confluence for the disjoint union of term rewriting systems (Toyama's theorem). This paper provides further support for the use of monads in term rewriting by giving a categorical proof of the most general theorem concerning the modularity of strong normalisation. In the process, we also improve upon some technical aspects of the earlier work.

## 1 Introduction

Term rewriting systems (TRSs) are widely used throughout computer science as they provide an abstract model of computation while retaining a relatively simple syntax and semantics. Reasoning about large term rewriting systems can be very difficult and an alternative is to define structuring operations which build large term rewriting systems from smaller ones. Of particular interest is whether key properties are *modular*, that is, if the components of a structured term rewriting system satisfy a property, then does the term rewriting system as a whole?

Although most properties are not in general modular, there are numerous results in the literature providing conditions under which key properties such as confluence and strong normalisation are modular. Research originally focussed on the *disjoint union*, for which confluence is modular, whereas strong normalisation is not. Unfortunately generalisations of these results, e.g. for conditional term rewriting systems and unions which permit limited sharing of term constructors, are rather unsatisfactory and tend to require rather strong syntactic conditions. Overall, although many specific modularity results are known, what is lacking is a coherent framework which explains the underlying principles behind these results.

We believe that part of the problem is the concrete, syntactic nature of term rewriting and that a semantics is needed to abstract away from the syntactic details and concentrate on the underlying structure. *Abstract Reduction Systems* provide a semantics for term rewriting systems using relations, but relations do not posses enough structure to adequately model key term rewriting concepts such as *substitution, context, layer structure* etc. and hence problems involving these features cannot be properly addressed. Thus the relational model is used mainly as an organisational tool with the difficult results proved directly at the level of pure syntax.

Category theory has been used to provide a semantics for term rewriting systems at an intermediate level of abstraction between the actual syntax and the relational model, using structures such as *2-categories* [RS87,See87], *Sesqui-categories* [Ste94], *confluent categories* [Jay90] or *ordered categories* [Gha95]. However, despite some one-off results, these approaches have failed to make a lasting impact on term rewriting.

An alternative approach starts from the observation that the categorical treatment of universal algebra is based on the idea of a monad on the category **Set**. Since term rewriting systems can be regarded as a generalisation of universal algebra it is natural to model a term rewriting system by a monad over a more structured base category. The basic theory of monads over categories with more structure than **Set** has been developed by Kelly and Power [KP93] and forms the theoretical basis of this research.

The usefulness of the monadic semantics for term rewriting systems has been demonstrated by proving the modularity of confluence (Toyama's theorem) [Lü96]. This proof offers a general methodology for using monads to study modular term rewriting:

1) Prove that the semantics preserves the structuring operations (i.e. is *compositional*). For the disjoint union of term rewriting systems, this means proving that if $\Theta$ is a term rewriting system and $\mathsf{T}_\Theta$ is its semantics, then
$$\mathsf{T}_{\Theta_1+\Theta_2} \cong \mathsf{T}_{\Theta_1} + \mathsf{T}_{\Theta_2} \tag{1}$$
2) Express the action of the monad representing the combined term rewriting system pointwise as a colimit over the base category. For disjoint unions, this means
$$\mathsf{T}_{\Theta_1+\Theta_2}(X) = \mathit{colim}\, \mathcal{D}_X$$
where $\mathcal{D}_X$ is a diagram in the base category indexed by $X$.
3) Prove that if the objects of $\mathcal{D}_X$ satisfy the desired property, then so does $\mathit{colim}\, \mathcal{D}_X$.

This methodology is particularly pleasing as the diagram $\mathcal{D}_X$ abstractly represents the fundamental concept in modular term rewriting of the *layer structure* on terms of the combined term rewriting system. In addition, the conditions on the use of variables which appear in modular term rewriting arise naturally as conditions on the units of the component monads required to ensure that $\mathit{colim}\mathcal{D}_X$ satisfies the property of interest. This proof strategy is entirely categorical since we prove abstract theorems concerning monads from which modularity results can be deduced. This is in contrast to some other applications of category theory to term rewriting where the category theory only plays an organisational role and the difficult problems are solved by translating back into the syntax.

We use this methodology to prove that strong normalisation is modular for the disjoint union of term rewriting systems which are *strongly normalising under non-deterministic collapses*. This theorem is the most general result concerning strong normalisation for disjoint unions which occurs in the literature. In addition, we improve some technical aspects of the earlier proof of the modularity of confluence.

The paper is divided as follows. Section 2 motivates the use of monads as models of term rewriting systems by recalling the equivalence between universal algebra and finitary monads on **Set**. Section 3 formally introduces term rewriting systems, and Section 4 defines the monadic semantics for term rewriting systems. Section 5 shows how disjoint unions of term rewriting systems are treated semantically while section 6 contains the actual modularity results. We finish with some conclusions and directions for further research.

We assume a working knowledge of term rewriting systems, and category theory as gained from the first five chapters of [Mac71] (the notation and terminology of which will be used here, and to which we will often refer). Although this work implicitly involves enriched categories, no knowledge of them is either assumed or even necessary for a basic understanding of what follows; a gentler introduction into enriched category theory than the somewhat demanding standard text [Kel82] is [Bor94, Chapter 6]. We would like to thank Don Sannella and Stefan Kahrs for many stimulating discussions. Get well soon Alan. Glory, Glory to the Hibees!

## 2 Universal Algebra and Monads

**Definition 1 (Monad).** A *monad* $\mathsf{T} = \langle T, \eta, \mu \rangle$ on a category $\mathcal{C}$ is given by an endofunctor $T : \mathcal{C} \to \mathcal{C}$, called the *action*, and two natural transformations, $\eta : \mathbf{1}_{\mathcal{C}} \Rightarrow T$, called the *unit*, and $\mu : TT \Rightarrow T$, called the *multiplication* of the monad, satisfying the *monad laws*: $\mu \cdot T\eta = \mathbf{1}_{\mathcal{C}} = \mu \cdot \eta_T$, and $\mu \cdot T\mu = \mu \cdot \mu_T$.

The monadic approach to term rewriting generalises the well known equivalence between (finitary) monads on the category **Set** and universal algebra. Thus, in order to motivate our constructions, we begin with a brief account of this equivalence.

Every algebraic theory defines a monad on **Set** whose action maps a set to the free algebra over this set. The unit maps a variable to the associated term, while the multiplication describes the process of substitution. The monad laws ensure that substitution behaves correctly, i.e. substitution is associative and the variables are left and right units. Thus monads can be thought of as being an abstract calculus for equational reasoning where *variables, substitution* and *term algebra* (represented by the unit, multiplication and action of the monad) are taken as primitive operations. We shall now make these ideas more precise. However, since this material is standard category theory, we omit most proofs and instead refer the reader to the standard references ([Man76], [Rob94] and [Mac71, Section VI]).

### 2.1 Modelling Signatures

**Definition 2 (Signature).** A (single-sorted) *signature* consists of a function $\Sigma : \mathbb{N} \to \mathbf{Set}$. The set of *n-ary operators* of $\Sigma$ is defined $\Sigma_n \stackrel{def}{=} \Sigma(n)$

**Definition 3 (Term Algebra).** Given a signature $\Sigma$ and a set of variables $X$, the *term algebra* $T_\Sigma(X)$ is defined inductively:

$$\frac{x \in X}{'x \in T_\Sigma(X)}$$

$$\frac{f \in \Sigma_n \quad t_1, \ldots t_n \in T_\Sigma(X)}{f(t_1, \ldots, t_n) \in T_\Sigma(X)}$$

Note that we have used quotes to distinguish a variable $x \in X$ from the term $'x \in T_\Sigma(X)$. This will be important when analysing the layer structure on terms formed from the disjoint union of two signatures. A term of the form $'x$ will be called a *term variable* while all other terms are called *compound terms*. A element of $T_\Sigma(X)$ will be called a term built over $X$.

**Lemma 4.** *Given a signature $\Sigma$, the map $X \mapsto T_\Sigma(X)$ defines a monad $\mathsf{T}_\Sigma$ on* **Set**.

*Proof.* The action of $\mathsf{T}_\Sigma$ on sets is given in definition 3, while on functions it maps $\phi : X \to Y$ to its *lifting* $T_\Sigma(\phi) : T_\Sigma(X) \to T_\Sigma(Y)$ which replaces the variables in $T_\Sigma(X)$ by their image under $\phi$. The unit $\eta_{\Sigma,X} : X \to T_\Sigma(X)$ maps each variable $x \in X$ to the term variable $'x \in T_\Sigma(X)$. An element of $T^2_\Sigma(X)$ can be thought of as a substitution, that is a term whose variables are themselves terms, and the multiplication performs this substitution, thereby obtaining an element of $T_\Sigma(X)$. Naturality and the monad laws are easily verified. □

As an aside, this framework can also deal with many-sorted signatures: If $S$ is a set of sorts, then an $S$-sorted signature defines a monad on $\mathbf{Set}^S$ — see [Rob94, Section 3] for details.

### 2.2 Finitariness

Given a signature, the term algebra construction defines a monad on **Set**. This monad satisfies an important continuity condition, namely it is *finitary*. To understand this condition, consider the action of the term algebra monad on an infinite set of variables. In particular, given a signature $\Sigma$ and an infinite set $X$,

$$T_\Sigma(X) = \bigcup_{X_0 \subset X \text{ is finite}} T_\Sigma(X_0)$$

This equation holds because all the operators in $\Sigma$ have a finite arity and thus a term built over $X$ can only contain a finite number of variables — such terms are therefore built over a finite subset of $X$. Categorically this is expressed by saying the functor $T_\Sigma$ is *finitary*:

**Definition 5 (Finitary Monads).** A functor is *finitary* iff it preserves filtered colimits [Mac71, Section IX.1]. A monad is *finitary* iff its action is finitary.

**Lemma 6.** *If $\Sigma$ is a signature, then $T_\Sigma$ is finitary [Rob94, Lemma 1.7].*

One can consider signatures with operations of infinite arities in which case the associated monad satisfies a suitably generalised definition of finitariness. All monads we shall consider are finitary in the sense of definition 5 — an example of a monad which isn't finitary is the powerset monad on **Set** which forms powersets of sets of arbitrary large size and hence has "operations" of arbitrary large arity.

### 2.3 Equations

Every signature can be modelled by a finitary monad on the category **Set**. In fact monads can be used to model *algebraic theories* which consist of a signature and a set of equations on the derived terms.

**Definition 7 (Equations and Algebraic Theories).** Let $\Sigma$ be a signature. A *$\Sigma$-equation* is of the form $X \vdash t = s$ where $X$ is a set and $t, s \in T_\Sigma(X)$. An *algebraic theory* $\mathcal{A} = \langle \Sigma, E \rangle$ consists of a signature $\Sigma$ and a set $E$ of $\Sigma$-equations.

The term algebra construction generalises from signatures to algebraic theories by mapping a set $X$ to the term algebra quotiented by the equivalence relation generated from the equations, $T_\mathcal{A}(X)$. The map $X \mapsto T_\mathcal{A}(X)$ extends to a finitary monad on the category **Set**. One can even show that the category of algebras of this monad is equivalent to the category of models of $\mathcal{A}$, justifying the correctness of the monadic semantics: "universal algebra is the study of finitary monads over **Set**" [Man76].

### 2.4 Compositionality of Monadic Semantics

We have seen how an algebraic theory $\mathcal{A}$ can be modelled by a finitary monad $\mathsf{T}_\mathcal{A}$ on the category **Set**. To be useful, this semantics should be *compositional* in that the semantics of a structured algebraic theory should be determined by the semantics of its components. In this

section, we will show that the monadic semantics is compositional for the disjoint union of algebraic theories

$$\mathsf{T}_{\mathcal{A}_1+\mathcal{A}_2} \cong \mathsf{T}_{\mathcal{A}_1} + \mathsf{T}_{\mathcal{A}_2} \tag{2}$$

where $\mathcal{A}_1$ and $\mathcal{A}_2$ are algebraic theories, $\mathcal{A}_1 + \mathcal{A}_2$ is their disjoint union and $\mathsf{T}_1 + \mathsf{T}_2$ is the coproduct of two monads $\mathsf{T}_1$ and $\mathsf{T}_2$. [1] This compositionality property is established by showing that every finitary monad arises from an algebraic theory. This algebraic theory is called the *internal language* of the monad, drawing on the analogy between the simply typed $\lambda$-calculus and cartesian closed categories.

**Definition 8 (Internal Signature).** The *internal signature* of a finitary monad $\mathsf{S} = \langle S, \eta, \mu \rangle$ on **Set** is given by

$$\Sigma_S(n) \stackrel{def}{=} \bigcup_{card(X)=n} S(X)$$

One can construct a map $\varepsilon_{S,X} : T_{\Sigma_S}(X) \to SX$ which interprets terms from $T_{\Sigma_S}(X)$ in $SX$. We say that a monad $S$ *admits* an equation $(X, l, r)$ where $l, r \in T_{\Sigma_S}(X)$, written $S \models_X l = r$, if $\varepsilon_{S,X}(l) = \varepsilon_{S,X}(r)$. The set of equations admitted by $S$, written $\mathcal{E}_S$, is defined as

$$\mathcal{E}_S \stackrel{def}{=} \{(X, l, r) \mid S \models_X l = r\}$$

**Definition 9 (Internal Language).** The internal language of a finitary monad $\mathsf{S}$ on **Set** is given by $\mathcal{L}_S \stackrel{def}{=} \langle \Sigma_S, \mathcal{E}_S \rangle$.

Crucially, these constructions are adjoint:[2]

$$\mathbf{AlgTh} \underset{\mathcal{L}}{\overset{\mathsf{T}}{\underset{\longleftarrow}{\overset{\longrightarrow}{\bot}}}} \mathbf{Mon}_{Fin}(\mathbf{Set})$$

where the categories **AlgTh** of algebraic theories and $\mathbf{Mon}_{Fin}(\mathbf{Set})$ of finitary monads on **Set** are appropriately defined [BW85], with the evaluation $\varepsilon_{S,X}$ being the counit of the adjunction — see [Lü97] for the exact definitions and the proof. Since the functor which maps an algebraic theory to its semantics is left adjoint, the semantics preserve colimits, and equation 2 holds.

In summary, monads provide a semantics for algebraic theories with the concepts of term-algebra, variable and substitution taken as primitive. This semantics is compositional, allowing us to reason about the disjoint union of algebraic theories in terms of the component theories.

## 3 Term Rewriting Systems

We now briefly review the theory of term rewriting systems — further details may be found in [Klo92]. First, fix a countably infinite set $V$ of variables.

**Definition 10 (Rewrite Rules and Term Rewriting Systems).** Let $\Sigma$ be a signature. A *$\Sigma$-rewrite rule* is of the form $r : t \to s$ where $t, s \in T_\Sigma(V)$. A *term rewriting system* $\Theta = \langle \Sigma, R \rangle$ consists of a signature $\Sigma$ and a set $R$ of $\Sigma$-rewrite rules.

[1] Of course, one also has to show that this coproduct exist, as we will do in Section 5.

[2] Modulo the size problems arising when considering the category of monads over **Set**.

A rewriting rule gives rise to the *one-step reduction* relation $C[\sigma(t)] \to_r C[\sigma(s)]$, where $C[\,]$ is a context and $\sigma$ is a substitution. The one-step reduction relation $\to_R$ of a term rewriting system $\Theta = \langle \Sigma, R \rangle$ is defined as the union of $\{\to_r\}_{r \in R}$, while the *many-step reduction relation*, denoted $\twoheadrightarrow_R$, is the transitive-reflexive closure of the one-step reduction relation. The two key properties of term rewriting systems which are of most interest are confluence and strong normalisation.

**Definition 11 (Confluence and Strong Normalisation).** A term rewriting system $\Theta = \langle \Sigma, R \rangle$ is *confluent* iff

$$\forall x, y_1, y_2.\, x \twoheadrightarrow_R y_1 \wedge x \twoheadrightarrow_R y_2 \; \exists z.\, y_1 \twoheadrightarrow_R z \wedge y_2 \twoheadrightarrow_R z$$

$\Theta$ is *strongly normalising* (SN, terminating, Noetherian) iff there is no infinite sequence

$$x_1 \to_R x_2 \to_R x_3 \to_R \ldots$$

A term rewriting system which is both confluent and SN is called *complete*. Complete systems are important as they decide the equality generated by the rewriting rules: every term reduces to a unique normal form, and two terms are equivalent iff their normal forms are equal.

A rewriting rule $r : t \to s$ is called *expanding* if $t$ is a variable, and *collapsing* if $s$ is a variable. It is said to *introduce variables* if there are is a variable occuring in $s$ which does not occur in $t$, and be *duplicating* if a variables occurs more often in $s$ than in $t$. In traditional definitions of term rewriting, rewriting rules are not to allowed to be expanding or variable-introducing, but from a semantic point these restrictions are unnatural and hence omitted.

Modular term rewriting studies how properties of large term rewriting systems are inherited from their component systems. The key definitions are

**Definition 12 (Disjointness and Modularity).** Given term rewriting systems $\Theta_1 = \langle \Sigma_1, R_1 \rangle$ and $\Theta_2 = \langle \Sigma_2, R_2 \rangle$ their *disjoint union* is defined as $\langle \Sigma_1 + \Sigma_2, R_1 + R_2 \rangle$. A property $P$ is *modular* if the disjoint union of $\Theta_1$ and $\Theta_2$ satisfies $P$ iff $\Theta_1$ satisfies $P$ and $\Theta_2$ satisfies $P$.

As shown by Toyama [Toy87], confluence is a modular property. He also gave a counterexample for strong normalisation not being modular. Given the two systems

$$R_1 \stackrel{def}{=} \{\mathtt{H(A,B,'}x\mathtt{)} \to \mathtt{H('}x\mathtt{,'}x\mathtt{,'}x\mathtt{)}\}\,,\; R_2 \stackrel{def}{=} \{\mathtt{G('}x\mathtt{,'}y\mathtt{)} \to \mathtt{'}x, \mathtt{G('}x\mathtt{,'}y\mathtt{)} \to \mathtt{'}x\}$$

the term $\mathtt{H(A,B,G(A,B))}$ is not strongly normalising. However, strong normalisation is modular under a variety conditions, such as both systems are not collapsing (i.e. contain no collapsing rules) [Rus87], both systems are not duplicating [Rus87], one system is neither duplicating nor collapsing [Mid89], or both systems are *simplifying* [KO92].

## 4 Monads as Models of Term Rewriting Systems

We will now show that every term rewriting system can be modelled by a finitary monad over a base category with more structure than **Set**. Kelly and Power [KP93] have shown how algebraic theories can be generalised to categories other than **Set** in such a way that the theory of section 2 can be developed at this more abstract level. Since term rewriting systems can be regarded as a generalised algebraic theory containing not only term constructors but also rewrite constructors (i.e. the rewrite rules), we therefore deduce a monadic semantics for term rewriting systems as an example of this general theory.

The theory of section 2 requires the arity of operations, variables and term algebra to have the same structure (they are sets). This allows a uniform treatment of term formation by the multiplication of the monad. Thus, a consequence of introducing a reduction structure on the term algebra is that the arities and variables must share this structure. This underpins the use of *generalised rewrite rules* of definition 13, and technically this means our construction is *enriched* over the reduction structure in the sense of [Kel82].

The choice of this structure, and hence the choice of the base category, depends on the specific aspects of rewriting one is interested in. If one wants to reason about many-step reductions, this structure has to be transitive and reflexive suggesting the category **Pre** of pre-orders. To further distinguish between reductions with the same source and target ("named reductions") then **Cat** is appropriate. For one-step reductions (which we are not interested in this paper), one can consider **Rel** (binary relations) or **Grph** (graphs). In this paper we shall start by using **Pre** because definition 14 is notationally easier[3], although later (Section 4.1) we shall switch to **Cat**.

**Definition 13 (Generalised Rewrite Rules).** A *generalised rewrite rule* in a signature $\Sigma$ is given by a triple $(X, l, r)$, written as $(X \vdash l \to r)$, where $X = (X_0, \to_X)$ is a finite preorder and $l, r \in T_\Sigma(X_0)$ are terms.

Intuitively, a generalised rewrite rule $(X, l, r)$ has not only term variables given by the carrier set $X_0$ of $X$, but also *rewrite variables* given by the order structure of $X$. In order to instantiate a generalised rewrite rule, one must not only supply terms for the free variables of the rule, but these terms must have rewrites between them which conform to the order structure of $X$ — see rule [Inst] of Table 1. Note that traditional rewrite rules of definition 10 arise as a special case of definition 13 with discrete arities.

In universal algebra, each signature $\Sigma$ defines a functor $T_\Sigma$ whose action is to map a set $X$ to the set $T_\Sigma(X)$ of terms built using the operators of $\Sigma$ as term constructors and the elements of $X$ as variables. For an arbitrary signature $\Sigma$ over a category $\mathcal{C}$, we wish to define a functor $T_\Sigma : \mathcal{C} \to \mathcal{C}$ whose action is to map a $\mathcal{C}$-object a variables $X$ to the $\mathcal{C}$-object of $\Sigma$-terms constructed over $X$. For a term rewriting system, we call this the *term reduction algebra*:

**Definition 14 (Term Reduction Algebra).** Given a term rewriting system $\Theta = \langle \Sigma, R \rangle$ and a preorder $X = (X_0, \to_X)$, the *term reduction algebra* $T_\Theta(X)$ is the smallest preorder $\to_{T_\Theta(X)}$ on the terms $T_\Sigma(X_0)$ satisfying the inference rules of Table 1, where $t[t_1, \ldots, t_n]$ is the substitution of the $n$ variables in $t \in T_\Omega(Y)$ with terms $t_1, \ldots, t_n$.

So the term reduction-algebra $T_\Theta(X)$ has as objects the terms which can be built over $X$ and has as rewrites the transitive-reflexive closure of the union of the rewrites of $\Theta$ and the rewrites of $X$ when closed under the term constructors. As with the term algebra construction, the term reduction construction defines a monad:

**Lemma 15.** *The mapping $X \mapsto T_\Theta(X)$ extends to a finitary,* **Pre***-enriched monad $\mathsf{T}_\Theta$.*

*Proof.* The proof follows that for Lemma 4. See [Lü96] for more details. □

[3] The analogous term construction of Def. 14 is technically far more complicated, since one has to introduce a notation to distinguish between different reductions and then enforce some equations on them. Further, the finitely presentable objects in **Cat** are categories which are generated as the free category of finite graphs closed under coequalizers, which creates further notational inconveniences.

$$[\textsc{Var}]\quad \frac{x \to_X y}{{}'x \to_{T_\Theta(X)} {}'y}$$

$$[\textsc{Pre}]\quad \frac{t_1 \to_{T_\Theta(X)} s_1, \ldots, t_n \to_{T_\Theta(X)} s_n}{f(t_1, \ldots, t_n) \to_{T_\Theta(X)} f(s_1, \ldots, s_n)}\ f \in \Sigma_n$$

$$[\textsc{Inst}]\quad \frac{\begin{array}{l}(Y \vdash l \to r) \in R,\ Y = (\{y_1, \ldots, y_n\}, \to_Y) \\ \forall i = 1, \ldots, n\, \forall j = 1, \ldots n.\ y_i \to_Y y_j \ \Rightarrow\ t_i \to_{T_\Theta(X)} t_j\end{array}}{l[t_1, \ldots, t_n] \to_{T_\Theta(X)} r[t_1, \ldots, t_n]}\ t_1, \ldots, t_n \in T_\Sigma(X)$$

**Table 1.** Definition of the Reduction Preorder.

We wish to use the monadic semantics for term rewriting systems to study modularity and hence require the semantics to be compositional.

**Lemma 16.** *The monadic semantics for term rewriting systems is compositional.*

*Proof.* As in section 2.4, we construct an adjunction between the category of finitary monads and the category of term rewriting systems. See [Lü97] for the details. □

In the rest of this paper, we require monads to have the following technical properties, which will be needed for the construction of the coproduct in Section 5.

**Definition 17 (Regular Monads).** A monad $\mathsf{T} = \langle T, \eta, \mu \rangle$ is called *regular* if

1) the action $T$ preserves *weakly filtered colimits* (colimits of weakly filtered diagrams) where a diagram $D$ is weakly filtered if for all $i, j \in D$, there is a $k \in D$ and morphism $m : i \to k$, $n : j \to k$.
2) its unit $\eta$ is a monomorphism (i.e. every component of $\eta$ is a monomorphism);

**Lemma 18.** *For a term rewriting system $\Theta$, the monad $\mathsf{T}_\Theta$ is regular.*

*Proof.* That the unit is a monomorphism is easy to see. To show that $T_\Theta$ preserves weakly filtered diagrams, observe that weakly filtered diagrams are filtered diagrams which are allowed to have parallel pairs of arrows which are not coequalized by another arrow in the diagram. Hence if a functor $T$ preserves both filtered colimits and coequalizers it will preserve weakly filtered colimits. $T_\Theta$ preserves filtered colimits because the underlying monad on **Set** is finitary (see lemma 6); and it preserves coequalizers because it does not identify any terms. □

### 4.1 Enrichment and Choice of Base Category

The crucial insight behind the constructions of the previous section is the proper enrichment. In particular, the base category $\mathcal{A}$ has to be enriched over a closed monoidal category $\mathcal{V}$. Further, $\mathcal{A}$ and $\mathcal{V}$ have to be locally finitely presentable, i.e. they have a small set $\mathcal{N}$ of *generators*

representing isomorphism classes of *finitely presentable objects* [KP93]; for **Set**, the generators are the natural numbers and the finitely presentable objects are the finite sets.

In this enriched setting, a signature over $\mathcal{A}$ is a map $\Sigma : \mathcal{N} \to \mathcal{A}$, giving for any generator $n$ the operations of arity $n$, and the term algebra $T_\Sigma(X)$ is defined as the colimit in $\mathcal{A}$ of the chain $T_0(X) \hookrightarrow T_1(X) \hookrightarrow \ldots$ where $T_0(X) = X$ and

$$T_{n+1}(X) = X + \sum_{c \in \mathcal{N}} [c, T_n(X)] \otimes \Sigma(c) \tag{3}$$

Note how the closed structure over which $\mathcal{A}$ enriches occurs in equation 3. We think of $T_{n+1}(X)$ as the terms of depth $n+1$, constructed as operations of arity $c$ applied to $c$-tuples of terms of depth $n$. Our models of term rewriting systems arise when we take $\mathcal{A} = \mathcal{V} = \mathbf{Pre}$ with the usual cartesian closed structure providing the enrichment.[4] Each of the rules of Table 1 arises as a special case of equation 3. For instance the rule [Var] comes from the inclusion of $X$ in $T_1(X)$, while specialising equation 3 to the declaration of rewrite rules gives the following equivalent formulation of [Inst]

$$[\text{Inst'}]\ \frac{\theta \in \mathbf{Pre}(Y, T_\Theta(X)) \quad (Y \vdash l \to r) \in R}{\theta(l)\ \to_{T_\Theta(X)}\ \theta(r)}$$

We can also specialise equation 3 to the declaration of term constructors and hence obtain an equivalent formulation of rule [Pre].

Of course, one can vary not only the base category, but also the choice of the monoidal structure. **Rel** and **Grph** both have cartesian products, but are only closed with respect to a different monoidal product, which for two graphs $\mathcal{G}, \mathcal{H}$ has as edges not pairs of edges but pairs of either a vertex from $\mathcal{G}$ and an edge from $\mathcal{H}$ or vice versa. In **Pre** both monoidal structures coincide, and **Pre** is closed with respect to them, but in **Cat** they are different and **Cat** is closed with respect to both of them; categories enriched over the non-cartesian structure are called *Sesqui-categories* and have been used as models for term rewriting [Ste94] since they have a categorical notion of "length". For theoretical simplicity, and since we do not need this notion of length, we here choose cartesian enrichment of **Cat** (the familiar framework of 2-categories).

### 4.2 Monadic Versions of Rewriting Concepts

The value of any semantics lies in the fact that it simplifies existing proofs and helps to solve open problems. In the remainder of the paper, we give semantic proofs of modularity results for confluence and strong normalisation. The first step is to define confluence and strong normalisation for arbitrary monads, and show that these definitions coincide with the traditional definitions for term rewriting systems given in section 3.

**Definition 19 (Confluent Categories and Monads).** A category $\mathcal{C}$ is *confluent* if for any two morphisms $\alpha : x \to x_1, \beta : x \to x_2$ there are morphisms $\gamma : x_1 \to z, \delta : x_2 \to z$ such that $\gamma \cdot \alpha = \delta \cdot \beta$. A monad $\mathsf{T} = \langle T, \eta, \mu \rangle$ on **Cat** is *confluent* if $T\mathcal{X}$ is confluent whenever $\mathcal{X}$ is.

This definition of a confluent category is different from Stell's [Ste94] which does not require the completions to form a commuting diagram. The definition of a confluent category is used

[4] In our construction above, the base category $\mathcal{A}$ and the enrichment category $\mathcal{V}$ happen to coincide, but there is actually no irrefutable reason for this. [Rob94] presents examples where the two differ.

by Jay [Jay90] but his confluent functors have a different intention and only require the identity and composition to be preserved up to having a common reduct.

For the following, recall that if $X$ is a pre-order and $\Theta = \langle \Sigma, R \rangle$ a term rewriting system, the term reduction algebra on $X$ is the union of the one-step reduction relation $\to_R$ and the closure of the variable rewrites in $X$ under application of operations called $\to_X$:

$$T_\Theta(X) = (\to_R \cup \to_X)^*$$

**Lemma 20.** *A TRS $\Theta = \langle \Sigma, R \rangle$ is confluent iff $T_\Theta$ is a confluent monad.*

*Proof.* Assume $T_\Theta$ is confluent and let $V$ be a discrete category. Then $V$ is confluent and hence $T_\Theta(V)$ is confluent. Since $V$ is discrete, the rewrites of $T_\Theta(V)$ are exactly the rewrites of $\Theta$ and hence $\Theta$ is confluent.

For the other direction, assume that $V$ is non-discrete. We can further assume that both $\to_V$ and $\twoheadrightarrow_R$ are confluent. The standard approach to show confluence of $(\to_R \cup \to_V)^*$ is to show that these relations commute. Here, this is not the case as variable rewrites can destroy redexes: consider the rewrite rule $(\{x\} \vdash \texttt{F('}x\texttt{, '}x\texttt{)} \to \texttt{G('}x\texttt{)})$, and the variables $\{x, y\}$ ordered as $x \to_V y$. The span in diagram 4 cannot be completed in one step each. However, we can

$$\begin{array}{ccccc} \texttt{F('}x\texttt{, '}x\texttt{)} & & \xrightarrow{\quad R \quad} & & \texttt{G('}x\texttt{)} \\ {\scriptstyle V}\downarrow & & & & \downarrow{\scriptstyle V} \\ \texttt{F('}x\texttt{, '}y\texttt{)} & \xrightarrow[V]{} & \texttt{F('}y\texttt{, '}y\texttt{)} & \xrightarrow[R]{} & \texttt{G('}y\texttt{)} \end{array} \tag{4}$$

show that the additional $V$-steps necessary to regain destroyed redexes do no destroy any more redexes, which is sufficient to prove confluence of the union and hence $T_\Theta(V)$. □

Strong normalisation for monads on **Cat** is formulated as follows:

**Definition 21 (Strong Normalisation for Categories and Monads).** A category $\mathcal{C}$ is *strongly normalising* (terminating, Noetherian), written $\mathcal{C} \models \mathrm{SN}$, if there is well-founded order $(X, >)$ and a map $w : |\mathcal{C}| \to X$ s.t.

$$\forall \alpha : x \to y \,.\, w(x) > w(y) \vee \alpha = \mathbf{1}_x \tag{5}$$

A monad $T$ on **Cat** is strongly normalising if $T\mathcal{X} \models \mathrm{SN}$ whenever $\mathcal{X} \models \mathrm{SN}$.

**Lemma 22.** *A TRS $\Theta$ is strongly normalising iff $\mathsf{T}_\Theta$ is a strongly normalising monad.*

*Proof.* If $\mathsf{T}_\Theta$ is strongly normalising then clearly so is $\Theta$. For the reverse, assume $\Theta$ is strongly normalising and let $X$ be a strongly normalising and confluent category. Then all variables have unique normal forms and so any infinite rewrite sequence in $\mathsf{T}_\Theta(X)$ can be mapped to an infinite series of $\Theta$-rewrites by replacing each variable with its normal form. If $X$ is only strongly normalising, then $X$ can be completed to a strongly normalising and confluent category $X^+$ and hence $\mathsf{T}_\Theta(X^+)$ is SN. Since $\mathsf{T}_\Theta(X)$ is a subcategory of a $\mathsf{T}_\Theta X^+$, it too is strongly normalising. □

Finally, collapsing and expanding rewrites are dual categorical concepts.

**Definition 23 (Non-Expanding/Collapsing Monads).** A functor $F : \mathcal{X} \to \mathcal{Y}$ is *non-expanding*, if for all objects $x \in \mathcal{X}$ and all morphisms $\alpha : Fx \to y'$ in $\mathcal{Y}$ there is a morphism $\beta : x \to y$ in $\mathcal{X}$ such that $F\beta = \alpha$. A monad $\mathsf{T} = \langle T, \eta, \mu \rangle$ on **Cat** is non-expanding if all components $\eta_{\mathcal{X}} : \mathcal{X} \to T\mathcal{X}$ of the unit are non-expanding, and the action preserves non-expanding functors, i.e. if $F : \mathcal{X} \to \mathcal{Y}$ is non-expanding, then so is $TF$.

A functor $F : \mathcal{X} \to \mathcal{Y}$ is *non-collapsing*, if $F^{op}$ is non-expanding. A monad $\mathsf{T} = \langle T, \eta, \mu \rangle$ on **Cat** is non-collapsing if all components $\eta_{\mathcal{X}} : \mathcal{X} \to T\mathcal{X}$ of the unit are non-collapsing, and the action preserves non-collapsing functors, i.e. if $F : \mathcal{X} \to \mathcal{Y}$ is non-collapsing, then so is $TF$.

One may easily verify that a term rewriting system $\Theta$ is non-expanding (non-collapsing) iff $\mathsf{T}_\Theta$ is non-expanding (collapsing).

## 5 A Monadic Approach to Modularity

We have given a semantics to term rewriting systems in terms of monads on **Cat**. By lemmas 20 and 22 we can reason about the disjoint union of $\Theta_1$ and $\Theta_2$ by reasoning about its semantics $\mathsf{T}_{\Theta_1+\Theta_2}$ which by lemma 16 is isomorphic to the coproduct of $\mathsf{T}_{\Theta_1}$ and $\mathsf{T}_{\Theta_2}$. This section gives a pointwise construction of the coproduct of two regular monads as the colimit of a diagram. Since this diagram is built solely from the component monads, we can reason about the coproduct monad in terms of the component monads.

To motivate this construction, consider two signatures $\Sigma_1, \Sigma_2$ which are modelled by the monads $\mathsf{T}_{\Sigma_1}$ and $\mathsf{T}_{\Sigma_2}$. The coproduct of these monads maps a set $X$ to the set of terms built from operations of $\Sigma_1$ and $\Sigma_2$. Such terms have an inherent notion of *layer*, that is one can decompose a term constructed from symbols in the union of two disjoint signatures into a term constructed from symbols in only one signature and strictly smaller subterms the top operation of which is from the other signature. For example, if $\mathtt{F} \in \Sigma_1, \mathtt{G} \in \Sigma_2$, then the term $\mathtt{F('G('}x\mathtt{), 'G('}y\mathtt{))}$ can be thought of as having a top layer consisting of the term $\mathtt{F('}x\mathtt{, '}y\mathtt{)}$ with the variables $x$ and $y$ replaced by the terms $\mathtt{G('}x\mathtt{)}$ and $\mathtt{G('}y\mathtt{)}$. Formally, the term $\mathtt{F('G('}x\mathtt{), 'G('}y\mathtt{))}$ is an object of the category $T_1T_2(X)$. Thus the terms which can be built from operations of $\Sigma_1 + \Sigma_2$ are contained in

$$\begin{aligned} T_{\Sigma_1+\Sigma_2}(X) = \; & X + T_1(X) + T_2(X) + \\ & T_1T_2(X) + T_2T_1(X) + \\ & T_1T_2T_1(X) + T_2T_1T_2(X) + \ldots \end{aligned}$$

However this disjoint union is too large. For instance the variables $X$ are included in the disjoint union from $X$, but also from $T_1(X)$ and $T_2(X)$ etc. Therefore we need equations to ensure the variables are included only once, and similarly by equations to ensure that substitution, i.e. the multiplication of each monad, is also respected. Thus we arrive at the definition of the action of the coproduct monad as the colimit of a diagram which has all the combinations of $\mathsf{T}_1$ and $\mathsf{T}_2$ as objects, and all morphisms which can be formed using the unit and multiplication of the two monads.

Formally, let $\mathcal{L} \stackrel{def}{=} \{1, 2\}$, and define $W \stackrel{def}{=} \mathcal{L}^*$ to be the words over $\mathcal{L}$. For every $w \in W$ define the functor $T^w : \mathbf{Cat} \to \mathbf{Cat}$ by $T^\varepsilon \stackrel{def}{=} 1_{\mathbf{Cat}}$ and $T^{jv} \stackrel{def}{=} T_jT^v$ where $j \in \mathcal{L}, v \in W$. As notational shortcuts, we also define the natural transformations $\eta^u_{i,v} \stackrel{def}{=} T^u(\eta_{i,T^v})$ and $\mu^u_{j,v} \stackrel{def}{=} T^u(\mu_{j,T^v})$ for $u, v \in W, j \in \mathcal{L}$. Further, from now on we will only consider regular monads without explicitly mentioning this.

**Definition 24 (The Colimit Diagram $\mathcal{D}_\mathcal{X}$).** For every category $\mathcal{X}$, we define the diagram $\mathcal{D}_\mathcal{X}$ to have as objects the categories $T^w(\mathcal{X})$ for $w \in W$ and the following edges.

$$(\eta^u_{i,v})_X : T^{uv}(X) \to T^{uiv}(X)$$
$$(\mu^u_{j,v})_X : T^{ujjv}(X) \to T^{ujv}(X)$$

**Lemma 25.** *The map on categories $\mathcal{X} \mapsto colim\, D_\mathcal{X}$ extends to a monad which is the coproduct of the monads $\mathsf{T}_1$ and $\mathsf{T}_2$.*

*Proof.* Functoriality follows from the universal property of the colimit, and by the fact that all arrows in the diagram are natural transformations. The unit is simply the inclusion of $\mathcal{X}$ into the colimiting object. The multiplication uses the fact that the diagram $D_\mathcal{X}$ is weakly filtered, and hence preserved by the two functors $T_1$, $T_2$. The monad laws and universal property follow from various diagram chases (see [Lü97] for details).[5] □

### 5.1 Analysing the Coproduct Monad

By the dual of Theorem 2 in [Mac71, pg. 109], every colimit can be expressed via coproducts and coequalizers. In particular, the colimit of $D_\mathcal{X}$ is given by the coequalizer of Diagram 6, where on

$$\coprod_{d:T^u\mathcal{X}\to T^v\mathcal{X}\in D_\mathcal{X}} T^u\mathcal{X} \underset{G}{\overset{F}{\rightrightarrows}} \coprod_{w\in W} T^w\mathcal{X} \tag{6}$$

the left side, we have for any morphism $d : T^u\mathcal{X} \to T^v\mathcal{X}$ in $D_\mathcal{X}$ (with $u, v \in W$) the component $T^u\mathcal{X}$ of the coproduct, and the two functors $F$ and $G$ are defined as $F(T^u\mathcal{X}) \stackrel{def}{=} \iota_u(T^u\mathcal{X})$, $G(T^u\mathcal{X}) \stackrel{def}{=} \iota_v(d(T^u\mathcal{X}))$ where $\iota_u$ and $\iota_v$ are the injections into the coproduct on the right. Thus in order to reason about the coproduct monad we need to reason about coequalizers in **Cat** (coproducts being trivial). The following lemma is taken from [Gra74, Chapter I.1]

**Lemma 26.** *Given two functors $F, G : \mathcal{X} \to \mathcal{Y}$, their coequalizer is a functor $Q : \mathcal{Y} \to \mathcal{Z}$, where $\mathcal{Z}$ is defined as follows:*

1) *The objects are the objects of $\mathcal{Y}$, quotiented by the equivalence closure $\equiv$ of the relation $\sim$ defined as $x \sim y \Leftrightarrow \exists z \in \mathcal{X}\,.\, Fz = x, Gz = y$.*
2) *The morphisms are sequences $\langle f_1, \ldots, f_n\rangle$ of morphisms $f_i$ in $\mathcal{Y}$ such that $\delta_s(f_i) \equiv \delta_t(f_{i-1})$ (where for a morphism $\alpha$, $\delta_s(\alpha)$ is its source, and $\delta_t(\alpha)$ its target), quotiented by the smallest equivalence relation $\equiv$ compatible with composition in $\mathcal{Y}$ such that $\langle f, g\rangle \equiv \langle g \cdot f\rangle$ if $f, g$ are composable in $\mathcal{Y}$, and $\langle Fh\rangle \equiv \langle Gh\rangle$ for all morphisms $h$ in $\mathcal{X}$.*

### 5.2 Deciding the Equivalence: Normal Forms

Terms in the disjoint union of two monads are equivalence classes of composed terms (elements from $T^w\mathcal{X}$). In this section, we improve upon the presentation of [Lü96] by introducing a pair of reduction systems which reduce the objects and morphisms of $\coprod_{w\in W} T^w\mathcal{X}$ to a unique normal form, hence deciding this equivalence. These constructions occur at the level of regular monads and nowhere do we use the fact that these monads arise from term rewriting systems.

[5] Actually, **Mon**(**Cat**) is a 2-category, so the coproduct has to satisfy another colimiting property on the 2-cells (so-called *modifications* between monad morphisms). It essentially does so because the diagram $\mathcal{D}_\mathcal{X}$ does not contain any 2-cells.

**Definition 27 (The Reduction System $\rightarrow_{Ob}$).** Define the following reduction systems on the objects of $\coprod_{w \in W} T^w\mathcal{X}$:

$$\begin{aligned} \rightarrow_\mu &\stackrel{def}{=} \{x \rightarrow_\mu \mu^u_{j,v}(x) \mid u, v \in W, j \in \mathcal{L}\} \\ \rightarrow_\eta &\stackrel{def}{=} \{\eta^u_{j,v}(x) \rightarrow_\eta x \mid u, v \in W, j \in \mathcal{L}\} \\ \rightarrow_{Ob} &\stackrel{def}{=} \rightarrow_\eta \cup \rightarrow_\mu \end{aligned}$$

We show that $\twoheadrightarrow_{Ob}$, the transitive-reflexive closure of $\rightarrow_{Ob}$, is complete and hence obtain a decision procedure for the associated equality. To do this, define the *rank* of a term $t \in T^w\mathcal{X}$ as $\mathrm{rank}(t) \stackrel{def}{=} |w|$ (where $|w|$ is the length of the word $w$).

**Lemma 28.** *$\twoheadrightarrow_{Ob}$ is complete.*

*Proof.* For SN, note that each reduction step $t \rightarrow_{Ob} u$ has the rank of $t$ strictly greater than that of $u$ and hence there can be no infinite reduction $x_1 \rightarrow_{Ob} x_2 \rightarrow_{Ob} x_3 \ldots$. For confluence, we refer to lemma 13 of [Lü96]. Clause (i) of that lemma implies confluence of $\rightarrow_\mu$, and clause (ii) implies confluence of $\rightarrow_\eta$. Clause (iii) implies that $\rightarrow_\eta$ and $\rightarrow_\mu$ commute and hence $\twoheadrightarrow_{Ob}$ is confluent. The cited proof also elucidates the necessity for the units $\eta_1$, $\eta_2$ to be monomorphisms. □

Since $\twoheadrightarrow_{Ob}$ is complete, every object in $t \in \coprod_{w \in W} T^w\mathcal{X}$ reduces to a unique normal form which we denote NF(t). This forms a decision procedure for the equivalence of the objects:

**Lemma 29.** *Given $t, t' \in \coprod_{w \in W} T^w\mathcal{X}$, $Qt = Qt'$ iff NF(t) = NF(t').*

*Proof.* $\mathrm{NF}(t) = \mathrm{NF}(t')$ iff $t$ and $t'$ are related is the equational theory on $\coprod_{w \in W} T^w\mathcal{X}$ generated by $\twoheadrightarrow_{Ob}$. This theory is clearly the same as that induced by the coequalizer of diagram 6. □

We now consider morphisms in the coequalizer of diagram 6. Since such morphisms are sequences of morphisms in $\coprod_{w \in W} T^w\mathcal{X}$, we start by considering the normal forms of morphisms in $\coprod_{w \in W} T^w\mathcal{X}$.

**Definition 30 (The Reduction System $\rightarrow_{Mor}$).** Define the reduction systems on the morphisms of $\coprod_{w \in W} T^w\mathcal{X}$:

$$\begin{aligned} \rightarrow_\mu &\stackrel{def}{=} \{\alpha \rightarrow_\mu \mu^u_{j,v}(\alpha) \mid u, v \in W, j \in \mathcal{L}\} \\ \rightarrow_\eta &\stackrel{def}{=} \{\eta^u_{j,v}(\alpha) \rightarrow_\eta \alpha \mid u, v \in W, j \in \mathcal{L}\} \\ \rightarrow_{Mor} &\stackrel{def}{=} \rightarrow_\eta \cup \rightarrow_\mu \end{aligned}$$

**Lemma 31.** *$\rightarrow_{Mor}$ is complete, and every $\alpha : x \rightarrow y$ in $T^w\mathcal{X}$ reduces to a unique normal form $NF(\alpha) : x' \rightarrow y'$ s.t. for all $\beta$, $Q\alpha = Q\beta$ iff $NF(\alpha) = NF(\beta)$.*

*Proof.* Analogously to lemma 28 and 29. □

The mapping of terms and morphisms to their normal form can not be extended to a functor, since the presence of non-expanding and non-collapsing rewrites means that the normal form need not preserve the source and target of a morphism. For example given a rewrite $\alpha : \texttt{'}x \rightarrow \texttt{G('}x\texttt{)}$ in $T_1(X)$, then $\mathrm{NF}(\alpha) = \alpha$ which is in $T_1(X)$ while $\mathrm{NF}(\texttt{'}x) = x$ which is in $X$.

**Definition 32.** Let $\alpha : s \to t$ be in $T^w\mathcal{X}$, and $\mathrm{NF}(\alpha) : s' \to t'$ its normal form. Then $\alpha$ is called *collapsing (expanding) in* $T_j$ if there are $u, v \in W, j \in \mathcal{L}$ and $y \in T^{uv}\mathcal{X}$ s.t. $t' = \eta^u_{j,v}(y)$ ($s' = \eta^u_{j,v}(y)$).

Note that a rewrite can be expanding or collapsing in both systems at the same time.

**Lemma 33.** *We can characterise expanding and collapsing rewrites as follows:*

*1)* $\alpha : s \to t$ *in* $T^w\mathcal{X}$ *is expanding iff for* $NF(\alpha) : s' \to t'$, $s' \neq NF(s)$.
*2)* $\alpha : s \to t$ *in* $T^w\mathcal{X}$ *is collapsing iff for* $NF(\alpha) : s' \to t'$, $t' \neq NF(t)$.

*Proof.* We can apply $\to_\mu$ to a morphism $\alpha : x \to y$ iff we can apply it to its source $x$ and target $y$. It is feasible that we can apply $\to_\eta$ to either of $x$ or $y$ but not to $\alpha$; namely, if $x = \eta^u_{j,v}(x')$, but for all $\beta : x' \to y'$, $\eta^u_{j,v}(\beta) \neq \alpha$. This means that $\alpha \not\to_\eta \beta$, hence $\alpha$ is expanding. Note that this is also exactly the definition of $\eta$ being expanding, hence if $\eta$ is non-expanding, we have $s' = \mathrm{NF}(s)$ for all rewrites $\alpha : s \to t$ and their normal form$\mathrm{NF}(\alpha) : s' \to t'$. The second clause is proven analogously. □

We close this section by showing that given two monad morphisms $\kappa : \mathsf{T}_1 \to \mathsf{S}_1$, $\lambda : \mathsf{T}_2 \to \mathsf{S}_2$, their coproduct $\kappa + \lambda : \mathsf{T}_1 + \mathsf{T}_2 \to \mathsf{S}_1 + \mathsf{S}_2$ preserves the normal form with respect to the two reduction systems $\twoheadrightarrow_{Ob}, \twoheadrightarrow_{Mor}$ above. Intuitively $\kappa + \lambda$ replaces every $\mathsf{T}_1$ layer with its image in $\mathsf{S}_1$ under $\kappa$ and similarly for $\mathsf{T}_2$ layers. Formally the components of $\kappa + \lambda$ are constructed by defining the obvious cone over the diagram whose colimit defines $(\mathsf{T}_1 + \mathsf{T}_2)(X)$.

**Lemma 34.** *Given* $\kappa : \mathsf{T}_1 \to \mathsf{S}_1$, $\lambda : \mathsf{T}_2 \to \mathsf{S}_2$, *let* $M \stackrel{def}{=} \kappa + \lambda$. *Then* $M(NF(t)) = NF(M(t))$ *for* $t \in T^w\mathcal{X}$, *and* $M(NF(\alpha)) = NF(M(\alpha))$ *for* $\alpha : s \to t$ *in* $T^w\mathcal{X}$.

*Proof.* By induction on the derivations $t \twoheadrightarrow_{Ob} \mathrm{NF}(t)$, $\alpha \twoheadrightarrow_{Mor} \mathrm{NF}(\alpha)$. Essentially, whenever we can reduce $t \to_{Ob} t'$, then we can reduce $M(t) \to_{Ob} M(t')$ (by naturality of both unit and multiplication of $\mathsf{T}_1$ and $\mathsf{T}_2$, and $\kappa$ and $\lambda$ being monad morphisms.)

□

For sequences $\langle \alpha_1, \ldots, \alpha_n \rangle$ in the colimit, we do not really need to decide the equality on them, but merely want to reason about their length (in the light of lemma 43 below). Hence we introduce the notion of minimal length:

**Definition 35 (Minimal Length).** A sequence $A = \langle \alpha_1, \ldots, \alpha_n \rangle$ is of *minimal length* iff

1) All elements of the sequence are normal forms: $\forall i = 1, \ldots, n. \alpha_i = \mathrm{NF}(\alpha_i)$
2) No equivalent sequence is shorter: $B \equiv A \;\Rightarrow\; |A| \leq |B|$

In other words, a sequence $A = \langle \alpha_1, \ldots, \alpha_n \rangle$ is is of minimal length if all $\alpha_i$ are in normal form, and $\alpha_i$ and $\alpha_{i+1}$ cannot be composed in $T^w\mathcal{X}$. This is only the case if $\alpha_i$ contains a so-called collapsing rewrite which creates new redexes. For an example, consider the two monads given by the following two term rewriting systems (we omit the signatures, and also the contexts for the rewriting rules here):

$$R_1 = \{\mathtt{F(F('}x\mathtt{))} \to \mathtt{H('}x\mathtt{)}\}$$
$$R_2 = \{\mathtt{G('}y\mathtt{)} \to \mathtt{'}y\}$$

Then there are the following two reductions, $\alpha_1$ in $T_1T_2T_1(X)$, and $\alpha_2$ in $T_1(X)$

$$\alpha_1 : \texttt{F('G('F('}x\texttt{)))} \to \texttt{F(''F('}x\texttt{))}$$
$$\alpha_2 : \texttt{F(F('}x\texttt{))} \to \texttt{H('}x\texttt{)}$$

Since $\mathrm{NF}(\texttt{F(''F('}x\texttt{)))} = \texttt{F(F('}x\texttt{))}$, one can form a sequence $\langle\alpha_1, \alpha_2\rangle$ although $\alpha_1$ and $\alpha_2$ are not composable as they inhabit different components of $\coprod_{w\in W} T^w\mathcal{X}$. This situation is prototypical, and the following lemma gives a precise characterisation of the sequences which have minimal length.

**Lemma 36.** *A sequence $A = \langle\alpha_1, \ldots, \alpha_n\rangle$ is of minimal length iff for all $i = 1, \ldots, n-1$:*

*1) $\alpha_i$ is collapsing in $T_j$: $\delta_t(NF(\alpha_i)) = \eta^u_{j,v}(t)$ (where $u, v \in W, j \in \mathcal{L}, t \in T^{uv}\mathcal{X}$);*
*2) $\delta_s(NF(\alpha_{i+1})) = \mu^r_{i,w}(s)$ with $u = rj$ and $v = jw$, and $i \neq j$ (where $r, w \in W, j \in \mathcal{L}$, and $s \in T^{rjjw}\mathcal{X}$);*
*3) $\mu^r_{i,w}$ is expanding at $s$: for all $\beta : s \to s'$ in $T^{rjjw}\mathcal{X}$, $\mu^r_{i,w}(\beta) \neq NF(\alpha_{i+1})$;*

*Proof.* Consider any $\alpha_i : s_i \to t_i$, $\alpha_{i+1} : s_{i+1} \to t_{i+1}$ for $i < n$. If $\alpha_i : s_i \to t_i$ is not collapsing, then by lemma 33, $t_i = \mathrm{NF}(t_i) = \mathrm{NF}(s_{i+1}) = s_{i+1}$, and we could compose $\alpha_i$ and $\alpha_{i+1}$, obtaining an equivalent but shorter sequence $A' = \langle\alpha_1, \ldots, \alpha_{i+1}\cdot\alpha_i, \ldots, \alpha_n\rangle$.

If $\alpha_i$ is collapsing but $s_i \neq \mu^r_{i,w}(s)$, then we could compose $\alpha_i$ and $\eta^u_{j,v}(\alpha_{i+1})$, obtaining a shorter equivalent sequence. Similarly, if $s_i = \mu^r_{i,w}(s)$ but not $u = rj$ and $v = js$, but e.g. $u = rjj$ and $v = s$, then $\eta^{rjj}_{2,v}$ and $\mu^r_{j,v}(t')$ would by naturality commute and we can find a $\beta'$ s.t. $Q\beta' = Q\alpha_{i+1}$ and $\beta'$ is composable with $\alpha_i$; and if $j = 2$, then $\eta^u_{2,v}$ would have a left (or right) inverse.

Finally, if $\mu^r_{i,w}$ is non-expanding at $s$, there would be $\beta$ such that $\mu^r_{i,w}(\beta) = \mathrm{NF}(\alpha_{i+1})$ and again we could obtain a shorter equivalent sequence. □

We can compose sequences of minimal length to obtain sequences of minimal length:

**Lemma 37.** *Given two sequences $A = \langle\alpha_1, \ldots, \alpha_n\rangle$ and $B = \langle\beta_1, \ldots, \beta_m\rangle$ s.t. $\delta_s(\beta_1) = \delta_t(\alpha_n)$ is of minimal length. Then if $\langle\alpha_n, \beta_1\rangle$ is of minimal length so is $\langle\alpha_1, \ldots, \alpha_n, \beta_1, \ldots, \beta_m\rangle$, otherwise $\langle\alpha_1, \ldots, \beta_1\cdot\alpha_n, \ldots, \beta_m\rangle$ is of minimal length*

*Proof.* Trivial.

It is however not the case that a monad morphism $\kappa + \lambda$ preserves sequences of minimal length, since in general the three conditions of lemma 36 are not preserved.

## 6 Modularity Results

### 6.1 Modularity of Confluence

We now prove the modularity of confluence. The first step towards proving confluence is to find conditions under which a functor preserves confluence.

**Definition 38 (One-Step Completion).** Given a functor $Q : \mathcal{Y} \to \mathcal{Z}$, the category $\mathcal{Y}$ has the *one-step completion property with respect to $Q$*, written $\mathcal{Y} \models_Q \diamond$, if for all morphisms $\alpha : x \to x'$, $\beta : y \to y'$ in $\mathcal{Y}$ such that $Qx = Qy$ there are morphisms $\gamma : v \to v', \delta : w \to w'$ in $\mathcal{Y}$ such that $Q\gamma\cdot Q\alpha = Q\delta\cdot Q\beta$.

**Lemma 39.** *Let $Q : \mathcal{Y} \to \mathcal{Z}$ be the coequalizer of two functors $F, G : \mathcal{X} \to \mathcal{Y}$ in* **Cat**. *If $\mathcal{Y}$ is confluent and $\mathcal{Y} \models_Q \diamond$, then $\mathcal{Z}$ is confluent.*

*Proof.* Given two morphisms $\alpha = [\langle \alpha_1, \ldots, \alpha_n \rangle]$ and $\beta = [\langle \beta_1, \ldots, \beta_m \rangle]$ in $\mathcal{Z}$ with the same source (i.e. $Q(\delta_s(\beta_1)) = Q(\delta_s(\alpha_1))$). Then (since $\mathcal{Y} \models_Q \diamond$) there are $\beta'_1$, $\alpha'_1$ such that $Q(\beta'_1) \cdot Q(\alpha_1) = Q(\alpha'_1) \cdot Q(\beta_1)$. By induction on the length $n$ and $m$ of $\alpha$ and $\beta$, respectively, we obtain completions $\alpha' \stackrel{def}{=} [\langle \alpha_1^{(m)}, \ldots, \alpha_n^{(m)} \rangle]$, $\beta' \stackrel{def}{=} [\langle \beta_1^{(n)}, \ldots, \beta_m^{(n)} \rangle]$ such that $\beta' \cdot \alpha = \alpha' \cdot \beta$. □

To prove that the coequalizer of diagram 6 is confluent we show that $\coprod_{w \in W} T^w X \models_Q \diamond$ where $Q$ is the coequalising functor. In [Lü96], this was done using a *witness relation*. Here, the witnesses are replaced by the conceptually simpler normal forms. In the following, assume $\mathsf{T}_1$ and $\mathsf{T}_2$ to be confluent, non-expanding, regular monads.

**Lemma 40.** *If $\mathcal{X}$ is confluent, $\coprod_{w \in W} T^w \mathcal{X} \models_Q \diamond$*

*Proof.* First $\coprod_{w \in W} T^w \mathcal{X}$ is confluent since $\mathcal{X}$, $\mathsf{T}_1$ and $\mathsf{T}_2$ are confluent and coproducts in **Cat** trivially preserve confluence. Now, let $\alpha : x \to x'$ and $\beta : y \to y'$ in $\coprod_{w \in W} T^w \mathcal{X}$ such that $Qx = Qy$. By lemma 33, $\mathrm{NF}(\alpha) : \mathrm{NF}(x) \to x_0$ and $\mathrm{NF}(\beta) : \mathrm{NF}(y) \to y_0$. Further, by lemma 29, and since $Qx = Qy$, we also have $\mathrm{NF}(x) = \mathrm{NF}(y)$. Since $\coprod_{w \in W} T^w \mathcal{X}$ is confluent, there are completions $\gamma : x_0 \to z_0$ and $\delta : y_0 \to z_0$ s.t. $\gamma \cdot \mathrm{NF}(\alpha) = \delta \cdot \mathrm{NF}(\beta)$, and hence $Q\gamma \cdot Q\alpha = Q\delta \cdot Q\beta$. □

**Lemma 41.** *The coproduct of two confluent, non-expanding regular monads is confluent.*

*Proof.* By lemmas 40 and 39 if $\mathcal{X}$ is confluent then so is the colimit of diagram 6. Hence the coproduct monad is confluent. □

The modularity of confluence for TRSs follows easily:

**Theorem 42 (Toyama).** *Confluence is a modular property for non-expanding TRSs.*

*Proof.* Let $\Theta_1$ and $\Theta_2$ be confluent TRSs. By lemma 20 the monads $\mathsf{T}_{\Theta_1}$ and $\mathsf{T}_{\Theta_2}$ are confluent and so is their coproduct (lemma 41). By equation 1 this coproduct models the disjoint union of $\Theta_1$ and $\Theta_2$ and hence by lemma 20 this TRS is confluent. □

### 6.2 Modularity of Strong Normalisation

As we saw in Section 3, strong normalisation is *not* a modular property for the disjoint union of term rewriting systems. We will below find conditions under which the disjoint union of the two strongly normalising monads cannot be strongly normalising. From these conditions, several conditions under which the union is strongly normalising will be derived. This is an adaption of the minimal counterexamples technique in [Gra92]. To this end, we first need a criterion to determine when the disjoint union of two monads is not SN. This is the case if we can form arbitrarily long sequences of morphisms from $\coprod_{w \in W} T^w \mathcal{X}$ in the coequalizer.

**Lemma 43.** *Given monads $\mathsf{T}_1, \mathsf{T}_2$ on* **Cat** *s.t. $\mathsf{T}_1 \models SN, \mathsf{T}_2 \models SN$, then $\mathsf{T}_1 + \mathsf{T}_2 \not\models SN$ iff there is a sequence $A$ in normal form s.t. for all $n \in \mathbb{N}$, $|A| > n$.*

*Proof.* The main technical difficulty in the proof lies in constructing an appropriate well-founded order on the objects of *colim* $D_{\mathcal{X}}$. This is done by constructing a well-founded order on the normal forms of $\twoheadrightarrow_{Ob}$. See [Lü97] for the details. □

A sequence $A$ s.t. for all $n \in \mathbb{N}$, $|A| > n$ will be called an *infinite sequence* in the following, written $A = \langle \alpha_1, \ldots, \alpha_n, \ldots \rangle$. Recently, term rewriting has arrived at a rather general condition for the modularity of strong normalisation. A term rewriting system $\Theta$ is called *strongly normalising under deterministic collapses* (SNDC or $\mathcal{C}_{\mathcal{E}}$-terminating) [Ohl94,Gra92] if it is SN and the disjoint union $\Theta + \mathcal{C}_{\mathcal{E}}$ is SN, where $\mathcal{C}_{\mathcal{E}}$ is the term rewriting system $\mathcal{C}_{\mathcal{E}} \stackrel{def}{=} \{\mathtt{G('}x\mathtt{,'}y\mathtt{)} \to \mathtt{'}x, \mathtt{G('}x\mathtt{,'}y\mathtt{)} \to \mathtt{'}y\}$. The term rewriting proof is a rather intricate encoding construction (so much so that it was first only proven for a finitely-branching systems[Gra92]). In this setting, the proof is far simpler; we first find a monad $\mathsf{T}_\perp$ representing $\mathcal{C}_{\mathcal{E}}$ and then analyse its combination with $\mathsf{T}_1$. This combination will be obtained by a categorical property of $\mathsf{T}_\perp$.

**Definition 44 (A Monad called $\mathsf{T}_\perp$).** The monad $\mathsf{T}_\perp = \langle T_\perp, \eta_\perp, \mu_\perp \rangle$ on **Cat** is defined as follows: it maps a category $\mathcal{X}$ to the category $T_\perp(\mathcal{X})$, which has as objects $|T_\perp \mathcal{X}| \stackrel{def}{=} \{\perp\} + |\mathcal{X}|$ and as morphisms

$$T_\perp(\mathcal{X})(x,y) \stackrel{def}{=} \begin{cases} \{!_y\} & \text{if } x = \perp \\ \emptyset & \text{if } x \neq \perp, y = \perp \\ \mathcal{X}(x,y) & \text{otherwise} \end{cases}$$

with the evident composition (for $f : x \to y$, $f \cdot !_x = !_y$ etc.). For a functor $F : \mathcal{X} \to \mathcal{Y}$, $T_\perp(F)$ maps $\{\perp\}$ to $\{\perp\}$, and $x$ in $T_\perp(\mathcal{X})$ (with $x \in \mathcal{X}$) to $Fx$ in $T_\perp(\mathcal{Y})$, and similarly on the morphisms. The unit $\eta_{\perp,\mathcal{X}} : \mathcal{X} \to T_\perp \mathcal{X}$ is the injection of the category $\mathcal{X}$ into $T_\perp \mathcal{X}$, and the multiplication $\mu_{\perp,\mathcal{X}} : T_\perp T_\perp \mathcal{X} \to T_\perp \mathcal{X}$ identifies the two adjoined objects.

From the term rewriting point, this monad can be seen as representing the system $\mathcal{C}_{\mathcal{E}}$.[6] From the categorical point of view, this monad freely adjoins an initial object $\perp$ to a category— an example of a monad on **Cat** adding structure to a category. This monad is terminal amongst non-expanding monads on **Cat**, and hence there is monad morphism from $T_1 + T_2$ to $T_1 + T_\perp$ translating reductions in the coproduct of $T_1$ and $T_2$ to those in $T_1$ and $\mathcal{C}_{\mathcal{E}}$.

**Lemma 45.** *For any non-expanding monad $T$ on* **Cat**, *there is a unique monad morphism* $!_T : T \to T_\perp$.

*Proof.* For a category $\mathcal{X}$, $!_{T,\mathcal{X}} : T\mathcal{X} \to T_\perp \mathcal{X}$ is constructed as follows: on the objects,

$$!_{T,\mathcal{X}}(x) \stackrel{def}{=} \begin{cases} x_0 & \text{if } x = \eta(x_0) \\ \perp & \text{otherwise} \end{cases}$$

and on the morphisms

$$!_{T,\mathcal{X}}(f : x \to y) \stackrel{def}{=} \begin{cases} f_0 & \text{if } f = \eta_{f_0} \\ !_{y_0} & \text{if } y = \eta(y_0) \text{ and for all } x_0 \in \mathcal{X}, x \neq \eta(x_0) \\ \mathbf{1}_\perp & \text{if for all } g : x_0 \to y_0 \text{ in } \mathcal{X},\ \eta g \neq f \end{cases}$$

This is the only definition which makes $!_T$ a monad morphism (i.e. agree with the unit and multiplication of $T$). Note that $!_{T,\mathcal{X}}$ is only a functor if $T$ is non-expanding. □

[6] Although of course $T_\perp$ is not the monad $\mathsf{T}_{\mathcal{C}_{\mathcal{E}}}$ given by that system because of its multiplication.

By lemma 45, for two monads $T_1$ and $T_2$, there is a monad morphism $1+! : T_1+T_2 \to T_1+T_\perp$, substituting all compound terms from $T_2$ in $T_1 + T_2$ with the object $\perp$ from $T_\perp$.

**Definition 46 (SNDC).** A monad $T$ on **Cat** is called *strongly normalising under deterministic collapses*, $T \models \text{SNDC}$ if $T \models \text{SN} \wedge T + T_\perp \models \text{SN}$

We have to show that this definition is equivalent to the one used in term rewriting (quoted above):

**Lemma 47.** *The term rewriting system $\Theta$ is strongly normalising under deterministic collapses iff. the monad $\mathsf{T}_\Theta$ is.*

*Proof.* Using lemma 43, we have to show that there is an infinite reduction in $\Theta + \mathcal{C}_\mathcal{E}$ iff there is an infinite sequence of minimal length in $T_1 + T_\perp$. One direction is easy, since every non-identity rewrite $\alpha : s \to t$ in $T_1 + T_\perp$ gives rise to at least one rewrite step in $\Theta + \mathcal{C}_\mathcal{E}$. For the other directions, we can draw upon [Gra92, Lemma 2], which can be used to show that an infinite derivation in $\Theta + \mathcal{C}_\mathcal{E}$ contains infinitely many rewrites which satisfy the criteria of lemma 36[7]. □

The proof of our main result will roughly proceed as follows: since $T_1 + T_2$ is not SN, there is an infinite sequence of minimal length, $A$. We consider the image of $A$ under the monad morphism $M \stackrel{def}{=} 1+! : T_1 + T_2 \to T_1 + T_\perp$. From lemma 36, we know exactly when a sequence has minimal length. We will show that the monad morphism preserves these properties, so there will be an infinite sequence of minimal length in $T_1 + T_\perp$ as well, showing that $T_1 + T_\perp$ is not SNDC.

**Lemma 48.** *Given a sequence $A = \langle \alpha, \beta \rangle$ of minimal length where $\alpha$ is collapsing in $T_2$, then $\langle M\alpha, M\beta \rangle$ is of minimal length as well.*

*Proof.* We first need some notation. Let $V \stackrel{def}{=} \{1, \perp\}^*$, and $T^v\mathcal{X}$ for $v \in V$ defined the obvious way. Then there is a map $\overline{(-)} : W \to V$, defined as $\overline{1v} \stackrel{def}{=} 1\overline{v}, \overline{2v} \stackrel{def}{=} \perp\overline{v}$ and $\overline{\varepsilon} \stackrel{def}{=} \varepsilon$. Further, there is a cone morphism $\nu_w : T^w\mathcal{X} \to T^{\overline{w}}\mathcal{X}$ defined as above (before lemma 34) inducing the monad morphism $M$ above.

We now show that the three conditions in lemma 36 characterising minimal length sequences are preserved by $M$. We can then conclude that $\langle M\alpha, M\beta \rangle$ is of minimal length as well. By lemma 34, $M(\text{NF}(\alpha)) = \text{NF}(M\alpha)$. Since $\nu_w$ is a cone morphism, we have

$$\nu_{u2v} \cdot \eta^u_{2,v} = \eta^{\overline{u}}_{2,\overline{v}} \cdot \nu_{uv}$$
$$\nu_{r1w} \cdot \mu^r_{1,w} = \mu^{\overline{r}}_{1,\overline{w}} \cdot \nu_{r11w}$$

Hence, $M(\alpha)$ will be collapsing in $T_\perp$ (the first condition), and $\delta_s(M(\text{NF}(\alpha_{i+1}))) = \nu_w(\mu^r_{1,w}(s)) = \mu^{\overline{r}}_{1,\overline{w}}(\nu_{r11w}(s))$ (the second condition).

It remains to show that $\mu^{\overline{u}}_{1,\overline{w}}$ is expanding at $\nu_{r11w}(s)$; this follows from the above equation and the fact that all $\beta$ in $T^{\overline{r}11\overline{w}}\mathcal{X}$ are in the image of $\nu_{r11w}$ (because 1 and ! are both surjective on objects).

[7] Namely, they are destructive at level 2.

**Theorem 49.** *Given two regular, non-expanding monads* $\mathsf{T}_1$ *and* $\mathsf{T}_2$ *on* **Cat** *s.t.* $\mathsf{T}_1 \models SN, \mathsf{T}_2 \models SN$ *, if* $\mathsf{T}_1 + \mathsf{T}_2 \not\models SN$*, then either* $\mathsf{T}_1 \not\models SNDC$ *and* $\mathsf{T}_2$ *is collapsing or vice versa.*

*Proof.* By lemma 43, there is an infinite sequence $A = \langle \alpha_1, \ldots, \alpha_n \ldots \rangle$ of minimal length in $\mathsf{T}_1 + \mathsf{T}_2$, and by lemma 36, all $\alpha_i$ in $A$ have to be collapsing. Hence, at least one of $T_1$ or $T_2$ is collapsing.

Further, since every $\alpha_i$ is collapsing, it means that there are infinitely many rewrites collapsing in $T_1$, or infinitely many rewrites collapsing in $T_2$. We may wolg. assume the latter. Now consider the sequence $MA = \langle M\alpha_1, \ldots, M\alpha_n, \ldots \rangle$ in $T_1 + T_\perp$. If we compose all $M\alpha_i$ and $M\alpha_{i+1}$ which can be composed, we obtain a sequence $A'$ of minimal length which is equivalent to $MA$. Since rewrites which are collapsing in $T_1$ will remain incomposable (by lemma 48, where lemma 37 guarantees that we can compose sequences of minimal length), $A'$ will be infinite as well. Hence, by lemma 43, $T_1 + T_\perp$ is not strongly normalising, so $T_1 \not\models$ SNDC. □

Theorem 49 has a whole host of interesting corollaries. The following list is not exhaustive:

**Corollary 50.** *The following modularity results follow from Theorem 49:*

*1) SN is modular for non-collapsing systems.*
*2) SN is modular for non-duplicating systems.*
*3) SN is modular provided one system is both non-collapsing and non-duplicating.*
*4) SN is modular for simplifying systems.*

*Proof.* The first is obvious. Non-duplicating and simplifying systems are strongly normalising under deterministic collapses [Gra92], showing the other three. □

Hence, all of the conditions listed after Definition 12 follow as corollaries from Theorem 49. [Ohl94] contains further derived criteria.

## 7 Conclusions and Further Work

We have shown how monads can be used to give a semantics to term rewriting systems by generalising the well-known equivalence between universal algebra and finitary monads on **Set**. Monads seem particularly suited to the study of modular term rewriting because the key concepts in this subject have concise monadic formulations. We believe this paper provides ample justification for these claims.

We propose to use monads to tackle more general modularity problems where the results in the literature remain unsatisfactory. In particular there are two principle directions for further research: firstly, monads can be used to model more general notions of term rewriting than that studied in this paper. In particular we hope to be able to study modularity for conditional and equational forms of term rewriting (i.e. rewriting modulo a set of equations). Secondly, non-disjoint unions which permit limited forms of sharing are modelled by push-outs which again have a compositional semantics. This enables us to study modularity of confluence and normalisation for these more general structuring operations.

## References

[Bor94] Francis Borceux. *Handbook of Categorical Algebra 2: Categories and Structures.* Number 51 in Encyclopedia of Mathematics and its Applications. Cambridge University Press, 1994.

[BW85] M. Barr and C. Wells. *Toposes, Triples and Theories.* Number 278 in Grundlehren der mathematischen Wissenschaften. Springer Verlag, 1985.

[Gha95] Neil Ghani. *Adjoint Rewriting.* PhD thesis, University of Edinburgh, 1995.

[Gra74] John W. Gray. *Formal Category Theory: Adjointness for 2-Categories.* Number 391 in Lecture Notes in Mathematics. Springer Verlag, 1974.

[Gra92] B. Gramlich. Generalized sufficient conditions for modular termination of rewriting. In *Proceedings of the Third International Conference on Algebraic and Logic Programming*, number 632 in Lecture Notes in Computer Science, pages 53–68. Springer Verlag, 1992.

[Jay90] C. B. Jay. Modelling reductions in confluent categories. In *Proceedings of the Durham Symposium on Applications of Categories in Computer Science*, 1990.

[Kel82] G. M. Kelly. *Basic Concepts of Enriched Category Theory*, volume 64 of *London Mathematical Society Lecture Note Series.* Cambridge University Press, 1982.

[Klo92] J. W. Klop. Term rewriting systems. In S. Abramsky, Dov M. Gabbay, and T. S. E. Maibaum, editors, *Handbook of Logic in Computer Science*, volume 2 (Background: Computational Structures), pages 1–116. Oxford University Press, 1992.

[KO92] M. Kurihara and A. Ohuchi. Modularity of simple termination of term rewriting systems with shared constructors. *Theoretical Computer Science*, 103:273– 282, 1992.

[KP93] G. M. Kelly and A. J. Power. Adjunctions whose counits are coequalizers, and presentations of finitary monads. *Journal for Pure and Applied Algebra*, 89:163– 179, 1993.

[Lü96] C. Lüth. Compositional term rewriting: An algebraic proof of Toyama's theorem. In H. Ganzinger, editor, *Rewriting Techniques and Applications*, number 1103 in Lecture Notes in Computer Science, pages 261– 275, New Brunswick, USA, July 1996. Springer Verlag.

[Lü97] Christoph Lüth. *Compositional Categorical Term Rewriting in Structured Algebraic Specifications.* PhD thesis, University of Edinburgh, 1997. Forthcoming.

[Mac71] S. Mac Lane. *Categories for the Working Mathematician*, volume 5 of *Graduate Texts in Mathematics.* Springer Verlag, 1971.

[Man76] Ernest G. Manes. *Algebraic Theories*, volume 26 of *Graduate Texts in Mathematics.* Springer Verlag, 1976.

[Mid89] A. Middeldorp. A sufficient condition for the termination of the direct sum of term rewriting systems. In *Fourth Annual Symposium on Logic in Computer Science*, pages 396–401. IEEE, Computer Society Press, June 1989.

[Ohl94] Enno Ohlebusch. On the modularity of termination of term rewriting systems. *Theoretical Computer Science*, 136:333– 360, 1994.

[Rob94] E. Robinson. Variations on algebra: monadicity and generalisations of equational theories. Technical Report 6/94, Sussex Computer Science, 1994.

[RS87] D. E. Rydeheard and J. G. Stell. Foundations of equational deduction: A categorical treatment of equational proofs and unification algorithms. In *Category Theory and Computer Science*, number 283 in Lecture Notes in Computer Science, pages 114– 139. Springer Verlag, 1987.

[Rus87] M. Rusinowitch. On the termination of the direct sum of term-rewriting systems. *Information Processing Letters*, 26(2):65–70, 1987.

[See87] R. A. G. Seely. Modelling computations: A 2-categorical framework. In *Proceedings of the Second Annual Symposium on Logic in Computer Science*, pages 65–71, 1987.

[Ste94] John G. Stell. Modelling term rewriting systems by Sesqui-categories. Technical Report TR94-02, Keele Unversity, January 1994.

[Toy87] Y. Toyama. On the Church-Rosser property for the direct sum of term rewriting systems. *Journal of the ACM*, 34(1):128–143, 1987.