Open access • Journal Article • DOI:10.1017/S0960129510000150

Diagrammatic logic applied to a parameterisation process — Source link

César Domínguez, Dominique Duval

Institutions: University of La Rioja, University of Grenoble

Published on: 01 Aug 2010 - Mathematical Structures in Computer Science (Cambridge University Press)

Topics: T-norm fuzzy logics, Monoidal t-norm logic, Diagrammatic reasoning, Category theory and Adjunction

Diagrammatic logic applied to a parameterization process

César Domínguez * Dominique Duval †

August 25., 2009

Abstract. This paper provides an abstract definition of some kinds of logics, called diagrammatic logics, together with a definition of morphisms and of 2-morphisms between diagrammatic logics. The definition of the 2-category of diagrammatic logics rely on category theory, mainly on adjunction, categories of fractions and limit sketches. This framework is applied to the formalization of a parameterization process. This process, which consists in adding a formal parameter to some operations in a given specification, is presented as a morphism of logics. Then the parameter passing process, for recovering a model of the given specification from a model of the parameterized specification and an actual parameter, is seen as a 2-morphism of logics.

1 Introduction

This paper provides an introduction to the framework of diagrammatic logics with an application to the formalization of a parameterization process.

The framework of diagrammatic logics is presented in section 2. It stems from [Duval 2003, Duval 2007], where the aim was to get an abstract definition of logics, with relevant notions of models and proofs, together with a good notion of morphism between logics: we were looking for kinds of logics for dealing with computational effects and for morphisms for expressing the meaning of the effects into more usual logics. This work is based on adjunction [Kan 1958] and categories of fractions [Gabriel and Zisman 1967] with an additional level of abstraction provided by limit sketches [Ehresmann 1968], which leads to a notion of entailment apparented to [Makkai 1997]. Our point of view is more abstract than the institutions [Goguen and Burstall 1984], see [Duval 2003] for a comparison. This new paper does not depend on [Duval 2003, Duval 2007].

On the other hand, the EAT and Kenzo software systems have been developed by F. Sergeraert for symbolic computation in algebraic topology [Rubio et al. 2007, Dousson et al. 1999]. The data types used in EAT and Kenzo have been specified through a parameterization process in [Domínguez et al. 2006, Domínguez et al. 2007], which is described in [Lambán et al. 2003] in terms of object-oriented technologies like hidden algebras [Goguen and Malcolm 2000] or coalgebras [Rutten 2000]. The parameterization process consists in adding a formal parameter to some operations in a given specification. It is followed by the parameter passing process, which recovers a model of the given specification from any model of the parameterized specification and any actual parameter. A first attempt to use diagrammatic logics in order to formalize this parameterization process is given in [Domínguez et al. 2005]. In section 3 we present a simple formalization of the parameterization and parameter passing processes as a morphism and a 2-morphism of diagrammatic logics, respectively. The focus in this application is on the models, but in [Dumas et al. 2009] another kind of application is studied, where proofs in a diagrammatic logic play an important role.

Most categorical notions used in this paper can be found in [Mac Lane 1998] or [Barr and Wells 1999]. For simplicity, we omit most size issues and we do not always distinguish between equivalent categories. The

*Departamento de Matemáticas y Computación, Universidad de La Rioja, Edificio Vives, Luis de Ulloa s/n, E-26004 Logroño, La Rioja, Spain, cesar.dominguez@unirioja.es.

†Laboratoire Jean Kuntzmann, Université de Grenoble, 51 rue des mathématiques, BP 53, F-38041 Grenoble Cédex 9, France, Dominique.Duval@imag.fr.

class of morphisms from X to Y in a category $\mathbf{C}$ is denoted $\mathbf{C}[X,Y]$. A *graph* means a directed multigraph, and in order to distinguish between various kinds of structures with an underlying graph we speak about the *objects* and *morphisms* of a category, the *types* and *terms* of a theory or a specification and the *points* and *arrows* of a limit sketch. The diagrammatic logics which are considered in this paper are the equational logic and several apparented logics. However diagrammatic logics can be much richer, for instance first-order logic as well as simple lambda calculus and logics with induction or coinduction can be seen as diagrammatic logics.

2 Diagrammatic logics

The 2-category of diagrammatic logics and its related notions are defined in sections 2.1, 2.2 and 2.3, then the diagrammatic equational logic is described in section 2.4.

2.1 Limit sketches

There are several definitions of limit sketches (also called projective sketches), all of them are such that a limit sketch generates a category with limits [Coppey and Lair 1984, Barr and Wells 1999]. While a category with limits is a graph with identities, compositions, limit cones and tuples, satisfying a bunch of axioms, we define a *limit sketch* $\mathbf{E}$ as a graph with *potential* identities, compositions, limit cones and tuples, which become real features in the generated category with limits $C(\mathbf{E})$. For instance a point X in $\mathbf{E}$ may have a potential identity, this is an arrow $id_X\colon X \to X$ in $\mathbf{E}$ which becomes the identity morphism at the object X in $C(\mathbf{E})$. As another instance, a diagram in $\mathbf{E}$ may have a potential limit cone, which becomes a limit cone in $C(\mathbf{E})$. Potential features are not required to satisfy any axiom in $\mathbf{E}$. In addition, for the simplicity of notations, we assume that each potential feature is unique: a point has at most one potential identity, a diagram has at most one potential limit cone, and so on.

A *morphism* of limit sketches $\mathbf{e} : \mathbf{E}_1 \to \mathbf{E}_2$ is a graph morphism which maps the potential features of $\mathbf{E}_1$ to potential features of $\mathbf{E}_2$. This forms the category of limit sketches. A *realization* (or *loose model*) of a limit sketch $\mathbf{E}$ with values in a category $\mathbf{C}$ is a graph morphism which maps the potential features of $\mathbf{E}$ to real features of $\mathbf{C}$. A morphism of realizations is (an obvious generalization of) a natural transformation. This gives rise to the category $Real(\mathbf{E},\mathbf{C})$ of realizations of $\mathbf{E}$ with values in $\mathbf{C}$, denoted simply $Real(\mathbf{E})$ when $\mathbf{C}$ is the category of sets. The category $Real(\mathbf{E})$ has colimits and we will use the fact that left adjoint functors preserve colimits.

The *Yoneda contravariant realization* $\mathcal{Y}_\mathbf{E}$ of a limit sketch $\mathbf{E}$ takes its values in $Real(\mathbf{E})$. It is defined as $\mathcal{Y}_\mathbf{E}(E) = P(\mathbf{E})[E,-]$ where $P(\mathbf{E})$ is the *prototype* of $\mathbf{E}$, which means, the category generated by $\mathbf{E}$ such that every potential feature of $\mathbf{E}$ becomes a real feature of $P(\mathbf{E})$. Thanks to $\mathcal{Y}_\mathbf{E}$, up to contravariance the limit sketch $\mathbf{E}$ can be identified to a part of $Real(\mathbf{E})$ which will be called the *elementary* part of $Real(\mathbf{E})$ (with respect to $\mathbf{E}$) and denoted $Real_{el}(\mathbf{E})$. It is a graph with *distinguished* features, defined as the identities, compositions, colimits and cotuples which are the images of the potential features of $\mathbf{E}$. A fundamental property is that the elementary part of $Real(\mathbf{E})$ is *dense* in $Real(\mathbf{E})$: every realization or morphism of realizations of $\mathbf{E}$ can be obtained by colimits and cotuples from $Real_{el}(\mathbf{E})$. Moeover, a fundamental theorem due to Ehresmann states that every morphism of limit sketches $\mathbf{e}\colon \mathbf{E}_1 \to \mathbf{E}_2$ gives rise to an adjunction $F_\mathbf{e} \dashv G_\mathbf{e}$ where the right adjoint $G_\mathbf{e}$ is the precomposition with $\mathbf{e}$ [Ehresmann 1968]:

$$Real(\mathbf{E}_1) \underset{G_\mathbf{e}}{\overset{F_\mathbf{e}}{\underset{\longleftarrow}{\overset{\longrightarrow}{\bot}}}} Real(\mathbf{E}_2)$$

Then the functor $F_\mathbf{e}$ *contravariantly extends* $\mathbf{e}$ via the Yoneda contravariant realizations, in the sense that there is a natural isomorphism:

$$F_\mathbf{e} \circ \mathcal{Y}_{\mathbf{E}_1} \cong \mathcal{Y}_{\mathbf{E}_2} \circ \mathbf{e} \,.$$

A *locally presentable category* [Gabriel and Ulmer 1971] is a category $\mathbf{C}$ which is equivalent to the category of set-valued realizations of a limit sketch $\mathbf{E}$, then $\mathbf{E}$ is called a limit sketch *for* the category $\mathbf{C}$. In addition, we define a *locally presentable functor* as a functor $F\colon \mathbf{C}_1 \to \mathbf{C}_2$ which is the left adjoint to the precomposition with some morphism of limit sketches $\mathbf{e}$, so that $\mathbf{C}_1$ and $\mathbf{C}_2$ are locally presentable categories. Then $\mathbf{e}$ is called a morphism of limit sketches *for* the functor F.

2.2 Diagrammatic logic: models and proofs

The framework of diagrammatic logics stems from [Duval 2003, Duval 2007].

Definition 2.1 A *diagrammatic logic* is a locally presentable functor L such that its right adjoint R is full and faithful.

The fact that R is full and faithful is equivalent to the fact that the counit natural transformation $\varepsilon\colon L \circ R \Rightarrow Id$ is an isomorphism. According to [Gabriel and Zisman 1967], it is also equivalent to the fact that L is a *localization*, up to an equivalence of categories: it consists of adding inverse morphisms for some morphisms, constraining them to become isomorphisms. Let us consider a diagrammatic logic L:

$$\mathbf{S} \underset{R}{\overset{L}{\underset{\perp}{\rightleftarrows}}} \mathbf{T}$$

Definition 2.1 also means that R defines an isomorphim from $\mathbf{T}$ to its image, which is a reflective subcategory of $\mathbf{S}$.

Definition 2.2 The categories $\mathbf{S}$ and $\mathbf{T}$ are the category of *specifications* and the category of *theories*, respectively, of the diagrammatic logic L. A specification Σ *presents* a theory Θ if Θ is isomorphic to $L(\Sigma)$. Two specifications are *equivalent* if they present the same theory.

The fact that R is full and faithful means that every theory Θ, when seen as a specification $R(\Theta)$, presents itself. With the next definition, we claim that every model of a specification takes its values in some theory.

Definition 2.3 A *(strict) model* M of a specification Σ in a theory Θ is a morphism of theories $M\colon L\Sigma \to \Theta$ or equivalently (thanks to the adjunction) a morphism of specifications $M\colon \Sigma \to R\Theta$.

It follows that equivalent specifications have the same models. A model M of Σ in Θ is sometimes called an *oblique morphism*, it is denoted $M\colon \Sigma \to \Theta$. Whenever in addition $\mathbf{S}$ and $\mathbf{T}$ are 2-categories with a natural isomorphism between $\mathbf{T}[L\Sigma, \Theta]$ and $\mathbf{S}[\Sigma, R\Theta]$, then $\mathbf{T}[L\Sigma, \Theta]$ is the *category of models of* Σ *in* Θ, denoted $L[\Sigma, \Theta]$. Otherwise, $L[\Sigma, \Theta]$ is simply the discrete category with the models of Σ in Θ as objects.

Definition 2.4 An *entailment* is a morphism τ in $\mathbf{S}$ such that $L\tau$ is invertible in $\mathbf{T}$.

A similar notion can be found in [Makkai 1997]. Two specifications which are related by entailments are equivalent.

Definition 2.5 An *instance* ρ of a specification Σ in a specification Σ_1 is a cospan in $\mathbf{S}$ made of a morphism $\sigma : \Sigma \to \Sigma'_1$ and an entailment $\tau : \Sigma_1 \to \Sigma'_1$. It is also called a *fraction* with *numerator* σ and *denominator* τ, and it is denoted $\rho = \tau \backslash \sigma : \Sigma \to \Sigma_1$.

Let us illustrate an instance $\rho = \tau \backslash \sigma$ of Σ in Σ_1 as:

$$\Sigma \xrightarrow{\ \sigma\ } \Sigma'_1 \underset{\tau}{\overset{\dashrightarrow}{\longleftarrow}} \Sigma_1$$

this provides easily a diagram in the category $\mathbf{S}$, by omitting the dotted arrow, and a diagram in the category $\mathbf{T}$, by making the dotted arrow a solid one, inverse to $L\tau$:

$$\text{in } \mathbf{S}\text{:} \quad \Sigma \xrightarrow{\ \sigma\ } \Sigma'_1 \xleftarrow[\ \tau\ ]{} \Sigma_1 \qquad \text{in } \mathbf{T}\text{:} \quad L\Sigma \xrightarrow{\ L\sigma\ } L\Sigma'_1 \underset{L\tau}{\overset{(L\tau)^{-1}}{\rightleftarrows}} L\Sigma_1$$

Since the category $\mathbf{S}$ has colimits and since the composition of entailments is an entailment, the instances can be composed in the usual way as cospans, thanks to pushouts. This forms the *bicategory of instances* of the logic, denoted $\mathbf{S_2}$. Let $\rho = \tau\backslash\sigma \colon \Sigma \to \Sigma_1$ in $\mathbf{S_2}$, then we define $L\rho = (L\tau)^{-1} \circ L\sigma \colon L\Sigma \to L\Sigma_1$ in $\mathbf{T}$. The instances are better suited than the morphisms of specifications for presenting the morphisms of theories, because for every morphism of theories $\theta : L\Sigma \to L\Sigma_1$ there is an instance ρ such that $L\rho = \theta$. Since L is a localization, the *quotient* category of the bicategory $\mathbf{S_2}$ is equivalent to $\mathbf{T}$.

Definition 2.6 An *inference system* for a diagrammatic logic L is a morphism of limit sketches $\mathbf{e} \colon \mathbf{E}_S \to \mathbf{E}_T$ for the locally presentable functor L.

Thanks to the Yoneda contravariant realization, the morphism $\mathbf{e}$ has properties similar to the functor L. In particular, $\mathbf{e}$ can be chosen so as to consist of adding inverse arrows for some collection of arrows in $\mathbf{E}_S$; see [Duval 2003, theorem 3.13] for a systematic construction of $\mathbf{e}$. The next definitions depend on the choice of an inference system $\mathbf{e} \colon \mathbf{E}_S \to \mathbf{E}_T$ for L; more details are given in [Duval 2007].

Definition 2.7 An *inference rule* r with *hypothesis* H and *conclusion* C is a span in $\mathbf{E}_S$, made of two morphisms $t : H' \to H$ and $s : H' \to C$ such that $\mathbf{e}(t)$ is invertible in $\mathbf{E}_T$. It is also called a *fraction* with *numerator* s and *denominator* t, and it is denoted $r = s/t : H \to C$.

With this definition we claim that an inference rule with hypothesis H and conclusion C can be seen, via the Yoneda contravariant realization, as an instance of $\mathcal{Y}(C)$ in $\mathcal{Y}(H)$. So, we can define an inference step simply as a composition of fractions, which means, as a pushout in the category $\mathbf{S}$.

Definition 2.8 Given an inference rule $r = s/t : H \to C$ and an instance $\kappa \colon \mathcal{Y}(H) \to \Sigma$ of the hypothesis $\mathcal{Y}(H)$ in a specification Σ, the corresponding *inference step* provides the instance $\kappa \circ \mathcal{Y}(r) \colon \mathcal{Y}(C) \to \Sigma$ of the conclusion $\mathcal{Y}(C)$ in Σ.

Definition 2.9 A *proof* (or *derivation*, or *derived rule*) is the description of a fraction in $\mathbf{S_2}$ in terms of inference rules (thanks to composition and cotuples).

Typically, by deriving $\rho = \tau\backslash id_\sigma$ for a given morphism $\tau : \Sigma_1 \to \Sigma$, we get the property that τ is an entailment. For instance, in equational logic, let τ be the inclusion of a given specification Σ_1 into the specification Σ made of Σ_1 together with an equation $f = g$ made of two terms f, g in Σ_1; then τ is an entailment if and only if the equation $f = g$ holds in the theory presented by Σ_1.

2.3 The 2-category of diagrammatic logics

Definition 2.10 A *morphism of logics* $F \colon L_1 \to L_2$ is a pair of locally presentable functors (F_S, F_T) together with a natural isomorphism $F_T \circ L_1 \cong L_2 \circ F_S$.

This means that there are inference systems $\mathbf{e}_1$ and $\mathbf{e}_2$ for L_1 and L_2 respectively, and morphisms of limit sketches $\mathbf{e}_S$ and $\mathbf{e}_T$ for F_S and F_T respectively, which form a commutative square of limit sketches:

$$
\begin{array}{c}
L_1 \\
\downarrow {\scriptstyle F} \\
L_2
\end{array}
\qquad\qquad
\begin{array}{ccc}
\mathbf{S}_1 & \xrightarrow{L_1} & \mathbf{T}_1 \\
{\scriptstyle F_S}\downarrow & \cong & \downarrow{\scriptstyle F_T} \\
\mathbf{S}_2 & \xrightarrow{L_2} & \mathbf{T}_2
\end{array}
\qquad\qquad
\begin{array}{ccc}
\mathbf{E}_{1,S} & \xrightarrow{\mathbf{e}_1} & \mathbf{E}_{1,T} \\
{\scriptstyle \mathbf{e}_S}\downarrow & = & \downarrow{\scriptstyle \mathbf{e}_T} \\
\mathbf{E}_{2,S} & \xrightarrow{\mathbf{e}_2} & \mathbf{E}_{2,T}
\end{array}
$$

Using the Yoneda contravariant realization, a morphism of logics $F \colon L_1 \to L_2$ can be determined by any graph morphism on $\mathbf{S}_{1,el}$ (the elementary part of $\mathbf{S}_1$ with respect to $\mathbf{E}_1$) with values in $\mathbf{S}_2$ preserving the distinguished features of $\mathbf{S}_{1,el}$ and the entailments of L_1. Some morphisms of logics are easier to describe at the sketch level (as the undecoration morphism in section 3.1) while others are easier to describe at the logic level (as the parameterization morphism in section 3.2). The next result is a straightforward application of adjunction.

Proposition 2.11 *Given a morphism of logics $F\colon L_1 \to L_2$ and the corresponding adjunctions $F_T \dashv G_T$ between theories and $F_S \dashv G_S$ between specifications, for each specification Σ_1 of L_1 and each theory Θ_2 of L_2 the adjunctions provide an isomorphism, natural in Σ_1 and Θ_2, between the categories of models:*

$$L_1[\Sigma_1, G_T(\Theta_2)] \cong L_2[F_S(\Sigma_1), \Theta_2] \ .$$

Definition 2.12 A *2-morphism of logics* $\ell\colon F \Rightarrow F'\colon L_1 \to L_2$ is a pair of natural transformations (ℓ_S, ℓ_T) where $\ell_S\colon F_S \Rightarrow F'_S\colon \mathbf{S}_1 \to \mathbf{S}_2$ and $\ell_T\colon F_T \Rightarrow F'_T\colon \mathbf{T}_1 \to \mathbf{T}_2$ are such that $\ell_T \circ L_1 = L_2 \circ \ell_S$.

Given a morphism of logics $F = (F_S, F_T)$ or a 2-morphism of logics $\ell = (\ell_S, \ell_T)$, we will usually omit the subscripts S and T.

The diagrammatic logics together with their morphisms and 2-morphisms form a 2-category. By focusing on theories we get a functor from the 2-category of diagrammatic logics to the 2-category of categories. The other parts of the logic (the category of specifications, the adjunction, and the inference system) provide a way to answer some issues about theories, typically whether some morphisms of theories are invertible.

2.4 The diagrammatic equational logic

The equational logic provides a fundamental example of a diagrammatic logic. As usual in categorical logic (see [Pitts 2000]), the *equational theories* are defined as the categories with chosen finite products; with the functors which preserve the chosen finite products they form a category $\mathbf{T}_{eq}$. Similarly (see [Lellahi 1989, Barr and Wells 1999, Wells 1993]), the *equational specifications* are defined as the finite product sketches, which means, the limit sketches (as in section 2.1) such that their potential limits are only potential products; with the morphisms of finite product sketches they form a category $\mathbf{S}_{eq}$. Since all finite products may be recovered from binary products and a terminal type, we restrict the arity of products to either 2 or 0. We will often omit the word "equational". Every theory Θ can be seen as a specification $R_{eq}\Theta$ and every specification Σ generates, or presents, a theory $L_{eq}\Sigma$. This corresponds to an adjunction:

$$\mathbf{S}_{eq} \underset{R_{eq}}{\overset{L_{eq}}{\underset{\perp}{\rightleftarrows}}} \mathbf{T}_{eq}$$

The category of sets with the cartesian products as chosen products forms an equational theory denoted *Set*. By default the models of an equational specification Σ are the models of Σ in *Set*, called the *set-valued models* of Σ. It is a classical exercise to build limit sketches for $\mathbf{T}_{eq}$ and $\mathbf{S}_{eq}$, then it is easy to check that L_{eq} is a diagrammatic logic. A simplified description is given now, see [Domínguez and Duval 2009] for a detailed construction. The starting point is the limit sketch for graphs $\mathbf{E}_{gr}$, where the points `Type` and `Term` stand for the sets of vertices (or types) and edges (or terms) and the arrows `dom` and `codom` for the functions source (or domain) and target (or codomain):

$$\mathtt{Type} \underset{\mathtt{codom}}{\overset{\mathtt{dom}}{\leftleftarrows}} \mathtt{Term}$$

Figure 1 presents the main part of the graph underlying $\mathbf{E}_{eq,S}$, in addition there are potential limits, including the specification of potential monomorphisms, and equalities of arrows. We have represented this graph in such a way that the bottom line, which is made of $\mathbf{E}_{gr}$ with potential limits and tuples, is equivalent to $\mathbf{E}_{gr}$. The point `Type` has been duplicated for readability, and the point `Unit` is a potential terminal type, interpreted as a singleton.

- The point `Comp` stands for the set of pairs of composable terms, the arrow `i` for the inclusion into the set of pairs of consecutive terms and `comp` for $(f, g) \mapsto g \circ f$.
- The point `Selid` stands for the set of types with a potential identity, the arrow `i0` for the inclusion and `selid` for $X \mapsto id_X$.

- The point `2-Prod` stands for the set of pairs of types with a potential binary product, the arrow `j` for the inclusion into the set of pairs of types and `2-prod` for $(Y_1, Y_2) \mapsto (pr_i : Y_1 \times Y_2 \to Y_i)_{i=1,2}$.
- The point `2-Tuple` stands for the set of binary cones with a potential binary tuple, the arrow `k` for the inclusion into the set of binary cones, `2-base`′ for recovering the base $(f_i : X \to Y_i)_{i=1,2} \mapsto (Y_1, Y_2)$, and `2-tuple` stands for the construction of the potential binary tuple $(f_i : X \to Y_i)_{i=1,2} \mapsto \langle f_1, f_2 \rangle : X \to Y_1 \times Y_2$.
- The point `0-Prod` stands for the set of potential terminal types, the arrow `j0` for the injection (ensuring that there is at most one terminal type) and `0-prod` for the selection of the potential terminal type (if any).
- The point `0-Tuple` stands for the set of types with a potential collapsing term (or nullary tuple), the arrow `k0` for the inclusion into the set of types, `0-base`′ for recovering the potential terminal type and `0-tuple` stands for the construction of the potential collapsing term $X \mapsto \langle\,\rangle_X : X \to 1$.

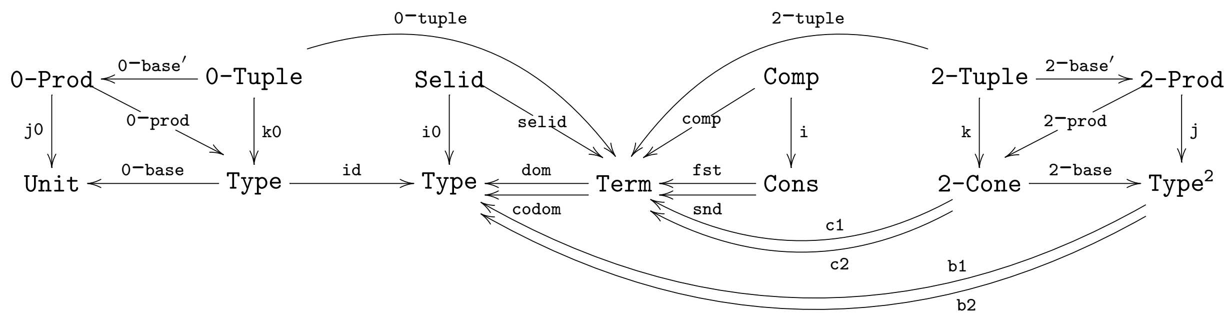


Figure 1: The graph underlying $\mathbf{E}_{eq,S}$

A limit sketch $\mathbf{E}_{eq,T}$ for equational theories is obtained from $\mathbf{E}_{eq,S}$ by choosing the entailments and mapping them to equalities, the corresponding morphism is the *diagrammatic equational logic* L_{eq}. Figure 2 provides the correspondence between the usual rules of equational logic and the diagrammatic inference rules, as fractions. Since only a part of $\mathbf{E}_{eq,S}$ is considered, some rules are missing, it is an exercise to enlarge $\mathbf{E}_{eq,S}$ so as to get them.

It should be noted that in this definition of the equational theories and specifications, the equations are identities of terms; a more subtle point of view, where the equations in a theory form a congruence, can be found in [Domínguez and Duval 2009].

3 A parameterization process

Several variants of the diagrammatic equational logic, related by morphisms, are defined in section 3.1. The parameterization process and the parameter passing process are formalized in sections 3.2 and 3.3, respectively.

3.1 Some diagrammatic logics

The theories of the *parameterized equational logic* L_A are the equational theories together with a distinguished type, called the *type of parameters* and usually denoted A. The specifications are the equational specifications with maybe a distinguished type A. The inclusion of limit sketches determines a morphism of logics $F_A : L_{eq} \to L_A$.

name	rule	fraction
composition	$\frac{f:X\to Y \quad g:Y\to Z}{g\circ f:X\to Z}$	$\mathtt{Cons} \underset{\mathtt{i}}{\overset{\dashrightarrow}{\longleftarrow}} \mathtt{Comp} \xrightarrow{\mathtt{comp}} \mathtt{Term}$
identity	$\frac{X}{id_X:X\to X}$	$\mathtt{Type} \underset{\mathtt{i0}}{\overset{\dashrightarrow}{\longleftarrow}} \mathtt{Selid} \xrightarrow{\mathtt{selid}} \mathtt{Term}$
binary product	$\frac{Y_1 \quad Y_2}{pr_i:Y_1\times Y_2\to Y_i}\,_{i=1,2}$	$\mathtt{Type}^2 \underset{\mathtt{j}}{\overset{\dashrightarrow}{\longleftarrow}} \mathtt{2\text{-}Prod} \xrightarrow{\mathtt{2\text{-}prod}} \mathtt{2\text{-}Cone}$
binary tuple	$\frac{f_1:X\to Y_1 \; f_2:X\to Y_2}{\langle f_1,f_2\rangle:X\to Y_1\times Y_2}$	$\mathtt{2\text{-}Cone} \underset{\mathtt{k}}{\overset{\dashrightarrow}{\longleftarrow}} \mathtt{2\text{-}Tuple} \xrightarrow{\mathtt{2\text{-}tuple}} \mathtt{Term}$
terminal type	$\frac{}{1}$	$\mathtt{Unit} \underset{\mathtt{j0}}{\overset{\dashrightarrow}{\longleftarrow}} \mathtt{0\text{-}Prod} \xrightarrow{\mathtt{0\text{-}prod}} \mathtt{Type}$
collapsing	$\frac{X}{\langle\,\rangle_X:X\to 1}$	$\mathtt{Type} \underset{\mathtt{k0}}{\overset{\dashrightarrow}{\longleftarrow}} \mathtt{0\text{-}Tuple} \xrightarrow{\mathtt{0\text{-}tuple}} \mathtt{Term}$

Figure 2: Rules for the equational logic

The theories of the *equational logic with a parameter* L_a are the parameterized equational theories together with a distinguished constant of type A, called the *parameter* and usually denoted $a\colon 1 \to A$. The specifications are the parameterized equational specifications with maybe a distinguished term $a\colon 1 \to A$. The inclusion of limit sketches determines a morphism of logics $F_a\colon L_A \to L_a$.

The theories of the *decorated equational logic* L_{dec} are the equational theories together with a wide subtheory called *pure* (*wide* means with the same types). The specifications are the equational specifications together with a wide subspecification. Here is a way to build $\mathbf{E}_{dec,T}$ from $\mathbf{E}_{eq,T}$ which reflects the meaning of the word "decoration", a smaller choice for $\mathbf{E}_{dec,T}$ can be found in [Domínguez and Duval 2009]. The decorations in this context are simply made of two keywords p for "pure" and g for "general"; some terms are pure, all terms are general, and there are rules for dealing with the decorations: identities and projections are always pure, and the compositions or tuples of pure terms are pure. This information can be encoded as a realization Δ of $\mathbf{E}_{eq,T}$ with values in the category of equational theories, as follows. First let us describe the set-valued realization Δ_0 of $\mathbf{E}_{eq,T}$ underlying Δ. The set $\Delta_0(\mathtt{Type})$ is made of one type D and the set $\Delta_0(\mathtt{Term})$ of two terms p and g, so that $\Delta_0(\mathtt{Cons}) = \{(p,p),(p,g),(g,p),(g,g)\}$, $\Delta_0(\mathtt{2\text{-}Cone}) = \{(p,p),(p,g),(g,p),(g,g)\}$ and $\Delta_0(\mathtt{Type}^2) = \{(D,D)\}$, and we denote $\Delta_0(\mathtt{Unit}) = \{\star\}$. Then $\Delta_0(\mathtt{selid})$ maps D to p, $\Delta_0(\mathtt{comp})$ maps (p,p) to p and everything else to g, $\Delta_0(\mathtt{2\text{-}prod})$ maps (D,D) to (p,p), $\Delta_0(\mathtt{2\text{-}tuple})$ maps (p,p) to p and everything else to g, $\Delta_0(\mathtt{0\text{-}prod})$ maps $\star$ to p and $\Delta_0(\mathtt{0\text{-}tuple})$ maps D to p. The structure of equational theory on each set $\Delta_0(E)$ is induced by a monomorphism $p \to g$ in $\Delta(\mathtt{Term})$. Then $\mathbf{E}_{dec,T}$ is the *sketch of elements* (similar to the more usual *category of elements*) of the realization Δ of $\mathbf{E}_{eq,T}$: the points of $\mathbf{E}_{dec,T}$ include one point `Type.D` over the point `Type` of $\mathbf{E}_{eq,T}$, two points `Term.p` and `Term.g` over the point `Term` of $\mathbf{E}_{eq,T}$, four points over `Cons`, and so on, and the arrows of $\mathbf{E}_{dec,T}$ include an arrow $\mathtt{c}\colon \mathtt{Term.p} \to \mathtt{Term.g}$ over $\mathtt{id}_{\mathtt{Term}}$ which is a potential monomorphism, for the conversion of pure terms to general terms.

Clearly by forgotting the decorations we get a morphism of diagrammatic logics $F_{und}\colon L_{dec} \to L_{eq}$, called the *undecoration* morphism. And by mapping every feature of $\mathbf{E}_{eq,T}$ to the corresponding pure feature of $\mathbf{E}_{dec,T}$ we get a morphism of diagrammatic logics $F_p\colon L_{eq} \to L_{dec}$ such that $F_{und} \circ F_p = id_{L_{eq}}$.

3.2 The parameterization process is a morphism of logics

In this section we define a morphism of logics $F_{par}\colon L_{dec} \to L_A$. We define F_{par} on specifications, its definition on theories follows easily. We will use the fact, which follows from the definition of a morphism of logics, that a specification may be replaced by an equivalent one whenever needed.

The parameterization process starts from a decorated specification and returns a parameterized specification. Roughly speaking, it replaces every general feature in a decorated specification by a parameterized one, in such a way that a pure feature does not really depend on the parameter. More precisely, types and pure terms are unchanged, while every general term $f\colon X \to Y$ is replaced by $f'\colon A \times X \to Y$ where A is the type of parameter. Figure 3 defines the image of the elementary decorated specifications (pure terms are denoted with "$\rightsquigarrow$" and the projections $pr_X\colon A \times X \to A$ and $\varepsilon_X\colon A \times X \to X$ are often omitted): for each point E.x in $\mathbf{E}_{dec,S}$, the parameterization process replaces the elementary decorated specification $\mathcal{Y}$(E.x) by the parameterized specification $F_{par}(\mathcal{Y}(\texttt{E.x}))$. The morphisms between elementary decorated specifications are transformed in a straightforward way. For instance, the image of the morphism $\mathcal{Y}$(c), where c: Term.p $\to$ Term.g is the conversion arrow, maps $f'\colon A \times X \to X$ in $F_{par}(\mathcal{Y}(\texttt{Term.g}))$ to $f \circ \varepsilon_X\colon A \times X \to Y$ in $F_{par}(\mathcal{Y}(\texttt{Term.p}))$, or more precisely in a parameterized specification equivalent to $F_{par}(\mathcal{Y}(\texttt{Term.p}))$. This provides a graph morphism $F_{par}\colon Real_{el}(\mathbf{E}_{dec,S}) \to Real(\mathbf{E}_{A,S})$.

	point E.x	$\mathcal{Y}$(E.x)	$F_{par}(\mathcal{Y}(\texttt{E.x}))$
type	Type.p	X	X
pure term	Term.p	$X \overset{f}{\rightsquigarrow} Y$	$X \xrightarrow{f} Y$
term	Term.g	$X \xrightarrow{f} Y$	$A\times X \xrightarrow{f'} Y$
pure composition	Comp.p	$X \overset{f}{\rightsquigarrow} Y \overset{g}{\rightsquigarrow} Z$; $X \overset{g\circ f}{\rightsquigarrow} Z$ (=)	$X \xrightarrow{f} Y \xrightarrow{g} Z$; $X \xrightarrow{g\circ f} Z$ (=)
composition	Comp.g	$X \xrightarrow{f} Y \xrightarrow{g} Z$; $X \xrightarrow{g\circ f} Z$ (=)	$A\times X \xrightarrow{\langle pr_X, f'\rangle} A\times Y \xrightarrow{g'} Z$; $A\times X \xrightarrow{g'\circ\langle pr_X,f'\rangle} Z$ (=)
selection of identity	Selid.p	$X \overset{id_X}{\rightsquigarrow} X$	$X \xrightarrow{id_X} X$
binary product	2-Prod.p	$Y_1 \overset{p_1}{\leftsquigarrow} Y_1\times Y_2 \overset{p_2}{\rightsquigarrow} Y_2$	$Y_1 \xleftarrow{p_1} Y_1\times Y_2 \xrightarrow{p_2} Y_2$
pure pairing	2-Tuple.p	$X \overset{f}{\rightsquigarrow} Y_1$, $X \overset{g}{\rightsquigarrow} Y_2$, $X \overset{\langle f,g\rangle}{\rightsquigarrow} Y_1\times Y_2$, $Y_1\times Y_2 \overset{p_1}{\rightsquigarrow} Y_1$ (=), $Y_1\times Y_2 \overset{p_2}{\rightsquigarrow} Y_2$ (=)	$X \xrightarrow{f} Y_1$, $X \xrightarrow{g} Y_2$, $X \xrightarrow{\langle f,g\rangle} Y_1\times Y_2$, $Y_1\times Y_2 \xrightarrow{p_1} Y_1$ (=), $Y_1\times Y_2 \xrightarrow{p_2} Y_2$ (=)
pairing	2-Tuple.g	$X \xrightarrow{f} Y_1$, $X \xrightarrow{g} Y_2$, $X \xrightarrow{\langle f,g\rangle} Y_1\times Y_2$, $Y_1\times Y_2 \overset{p_1}{\rightsquigarrow} Y_1$ (=), $Y_1\times Y_2 \overset{p_2}{\rightsquigarrow} Y_2$ (=)	$A\times X \xrightarrow{f'} Y_1$, $A\times X \xrightarrow{g'} Y_2$, $A\times X \xrightarrow{\langle f',g'\rangle} Y_1\times Y_2$, $Y_1\times Y_2 \xrightarrow{p_1} Y_1$ (=), $Y_1\times Y_2 \xrightarrow{p_2} Y_2$ (=)
terminal type	0-Prod.p	1	1
pure collapsing	0-Tuple.p	$X \overset{\langle\,\rangle_X}{\rightsquigarrow} 1$	$X \xrightarrow{\langle\,\rangle_X} 1$

Figure 3: The parameterization morphism on elementary decorated specifications

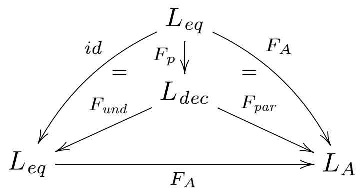
Theorem 3.1 *The graph morphism F_{par} defines a morphism of diagrammatic logics:*

$$F_{par}\colon L_{dec} \to L_A$$

which is the inclusion on the pure part of L_{dec}, in the sense that $F_{par} \circ F_p = F_A$. It is called the parameterization *morphism.*

Proof. It can be checked that this graph morphism preserves the distinguished features of $Real_{el}(\mathbf{E}_{dec,S})$ and the entailments of the decorated logic, so that it provides a morphism of diagrammatic logics. The equality $F_{par} \circ F_p = F_A$ is easily checked on elementary specifications. □

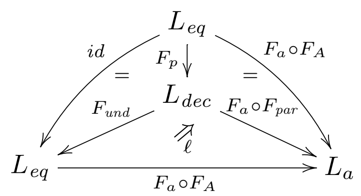
The morphisms of logics F_{und}, F_{par} and F_A form a (non-commutative) triangle, which becomes commutative when restricted to the pure part of L_{dec}:



The parameterization morphism F_{par} formalizes the parameterization process. The span made of F_{und} and F_{par} formalizes the process of starting from an equational specification Σ_{eq}, choosing a pure subspecification Σ_0 of Σ_{eq} so as to get a decorated specification Σ_{dec} such that $\Sigma_{eq} = F_{und}(\Sigma_{dec})$, then forming the parameterized specification $\Sigma_A = F_{par}(\Sigma_{dec})$.

3.3 The parameter passing process is a 2-morphism of logics

The diagram of logics in section 3.2 composed with the inclusion $F_a\colon L_A \to L_a$, which adds the parameter $a\colon 1 \to A$, provides another diagram with in addition a 2-morphism ℓ as described below:



Each decorated specification Σ_{dec}, with $\Sigma_{eq} = F_{und}(\Sigma_{dec})$, gives rise to two specifications with parameter: on the one hand $\Sigma_{eq,a} = F_a(F_A(\Sigma_{eq}))$, which is simply Σ_{eq} seen as a specification with a parameter, and on the other hand $\Sigma_a = F_a(F_{par}(\Sigma_{dec}))$. Let us define the morphism $\ell_{\Sigma_{dec}}\colon \Sigma_{eq,a} \to \Sigma_a$. When Σ_{dec} is some $\mathcal{Y}(\texttt{E.p})$ (where $\texttt{p}$ means "pure") it is easy to check that $\Sigma_{eq,a} = \Sigma_a$; then $\ell_{\Sigma_{dec}}$ is the identity. When $\Sigma_{dec} = \mathcal{Y}_{dec}(\texttt{Term.g})$ (where $\texttt{g}$ means "general") , then $\ell_{\Sigma_{dec}}$ is defined by $\ell_{\Sigma_{dec}}(f) = f' \circ \langle a, id_X \rangle\colon X \to Y$ (where $1 \times X$ is identified with X). The definitions when $\Sigma_{dec} = \mathcal{Y}_{dec}(\texttt{Comp.g})$ and when $\Sigma_{dec} = \mathcal{Y}_{dec}(\texttt{2-Tuple.g})$ are similar.

Theorem 3.2 *The morphisms $\ell_{\Sigma_{dec}}\colon \Sigma_{eq,a} \to \Sigma_a$ define a 2-morphism of diagrammatic logics:*

$$\ell\colon F_a \circ F_A \circ F_{und} \Rightarrow F_a \circ F_{par}\colon L_{dec} \to L_a$$

which is the identity on the pure part of L_{dec}. It is called the parameter passing *2-morphism.*

Proof. The definition of $\ell_{\Sigma_{dec}}$ on the elementary decorated specifications is extended to all specifications by colimits, and the result follows. □

Theorem 3.2 has the expected consequence on models, stated as proposition 3.3: given a set-valued model M_A of the paramererized specification Σ_A, each $\alpha \in M_A(A)$, called an *actual parameter* or an *argument*, gives rise to a model $\mathcal{M}(\alpha)$ of the equational specification Σ_{eq}. Let us introduce some notations. For each set $\mathbb{A}$, let $Set_{\mathbb{A}}$ denote the object of $\mathbf{T}_A$ made of the equational theory of sets with $\mathbb{A}$ as the interpretation of A, so that $R_A(Set_{\mathbb{A}}) = Set$. For each set $\mathbb{A}$ and element $\alpha \in \mathbb{A}$, let $Set_{\mathbb{A},\alpha}$ denote the object of $\mathbf{T}_a$ made of the equational theory of sets with $\mathbb{A}$ and α as the interpretations of A and a respectively, so that $R_a(Set_{\mathbb{A},\alpha}) = Set_{\mathbb{A}}$. For each decorated specification $\Sigma_{dec} = (\Sigma_{eq}, \Sigma_0)$, made of an equational specification Σ_{eq} and a wide subspecification Σ_0, and for each set-valued equational model M_0 of Σ_0, let $L_{eq}[\Sigma_{eq}, Set]|_{M_0}$ denote the set of models of Σ_{eq} extending M_0. Let $\Sigma_A = F_{par}(\Sigma_{dec})$, the definition of F_{par} is such that Σ_0 is also a subspecification of Σ_A and for each $f\colon X \to Y$ in Σ_{eq} there is a $f'\colon A \times X \to Y$ in Σ_A, with $f' = f \circ \varepsilon_X$ when f is pure.

Proposition 3.3 *Let $\Sigma_{dec} = (\Sigma_{eq}, \Sigma_0)$ be a decorated specification and let $\Sigma_A = F_{par}(\Sigma_{dec})$. For each set $\mathbb{A}$ and each set-valued model $M_A\colon \Sigma_A \to Set_{\mathbb{A}}$ in L_A, let $M_0\colon \Sigma_{eq} \to Set$ denote the restriction of M_A to Σ_0. Then there is a function:*

$$\mathcal{M}\colon \mathbb{A} \to L_{eq}[\Sigma_{eq}, Set]|_{M_0}$$

which maps each $\alpha \in \mathbb{A}$ to the model $\mathcal{M}(\alpha)$ of Σ_{eq} extending M_0 and such that $\mathcal{M}(\alpha)(f) = M_A(f')(\alpha, -)$ for each $f\colon X \to Y$ in Σ_{eq}.

Proof. Let $\Sigma_{eq,a} = F_a(F_A(\Sigma_{eq}))$ and $\Sigma_a = F_a(F_{par}(\Sigma dec))$. The precomposition with the morphism $\ell_{\Sigma_{dec}}\colon \Sigma_{eq,a} \to \Sigma_a$ gives rise to a functor $L_a[\Sigma_a, Set_{\mathbb{A},\alpha}] \to L_a[\Sigma_{eq,a}, Set_{\mathbb{A},\alpha}]$. Proposition 2.11 provides the isomorphisms $L_a[\Sigma_a, Set_{\mathbb{A},\alpha}] \cong L_A[\Sigma_A, Set_{\mathbb{A}}]$ and $L_a[\Sigma_{eq,a}, Set_{\mathbb{A},\alpha}] \cong L_{eq}[\Sigma_{eq}, Set]$. So, for each $\alpha \in \mathbb{A}$ we get a functor $L_A[\Sigma_A, Set_{\mathbb{A}}] \to L_{eq}[\Sigma_{eq}, Set]$. Let $M_{A,\alpha}$ denote the image of M_A, because of the definition of $\ell_{\Sigma_{dec}}$ it extends M_0 and satisfies $M_{A,\alpha}(f) = M_A(f')(\alpha, -)$ for each $f\colon X \to Y$ in Σ_{eq}. Now, when M_A is fixed, the result follows by defining $\mathcal{M}(\alpha) = M_{A,\alpha}$. □

The function $\mathcal{M}$ is not a bijection in general. However this may happen, under the conditions of proposition 3.4: this is the *exact parameterization* property from [Lambán et al. 2003], which is also proved in [Domínguez and Duval 2009].

Proposition 3.4 *With the specifications Σ_{eq}, Σ_0 and Σ_A as in proposition 3.3, let M_0 be a model of Σ_0 and M_A a terminal model of Σ_A extending M_0. Then the function $\mathcal{M}$ from proposition 3.3 is a bijection:*

$$M_A(A) \cong L_{eq}[\Sigma_{eq}, Set]|_{M_0} \ .$$

It follows from [Rutten 2000] and [Hensel and Reichel 1995] that there is a terminal model of Σ_A over M_0. Proposition 3.4 corresponds to the way algebraic structures are implemented in the systems Kenzo/EAT. In these systems the parameter set is encoded by means of a record of Common Lisp functions, which has a field for each operation in the algebraic structure to be implemented. The pure terms correspond to functions which can be obtained from the fixed data and do not require an explicit storage. Then, each particular instance of the record gives rise to an algebraic structure.

References

[Barr and Wells 1999] Barr, M. and Wells, C. (1999) Category Theory for Computing Science. *Centre de Recherches Mathématiques (CRM) Publications*, 3rd Edition.

[Coppey and Lair 1984] Coppey, L. and Lair, C. (1984) Leçons de Théorie des Esquisses. *Diagrammes* **12**.

[Domínguez and Duval 2009] Domínguez, C. and Duval, D. (2009) A parameterization process as a categorical construction.

[Domínguez et al. 2005] Domínguez, C., Duval, D., Lambán, L. and Rubio, J. (2005) Towards diagrammatic specifications of symbolic computation systems. In: Mathematics, Algorithms, Proofs. Coquand, T., Lombardi, H. and Roy, M. (Eds.) *Dagstuhl Seminar* **05021**. `http://drops.dagstuhl.de/portals/index.php?semnr=05021`.

[Domínguez et al. 2007] Domínguez, C., Lambán, L. and Rubio, J. (2007) Object-oriented institutions to specify symbolic computation systems. *Rairo - Theoretical Informatics and Applications* **41** 191–214.

[Domínguez et al. 2006] Domínguez, C., Rubio, J. and Sergeraert, F. (2006) Modeling Inheritance as coercion in the Kenzo system. *Journal of Universal Computer Science* **12** (12) 1701–1730.

[Dousson et al. 1999] Dousson, X., Sergeraert, F. and Siret, Y. (1999) The Kenzo program. Institut Fourier, Grenoble. `http://www-fourier.ujf-grenoble.fr/~sergerar/Kenzo`.

[Dumas et al. 2009] Dumas, J.G., Duval, D. and Reynaud, J.C. (2009) Cartesian effect categories are Freyd-categories. `arXiv:0903.3311v3`.

[Duval 2003] Duval, D. (2003) Diagrammatic specifications. *Mathematical Structures in Computer Science* **13** 857–890.

[Duval 2007] Duval, D. (2007) Diagrammatic inference. arXiv:0710.1208v1.

[Ehresmann 1968] Ehresmann, C. (1968) Esquisses et types de structures algébriques. *Bull. Instit. Polit. Iaşi* **XIV**.

[Gabriel and Ulmer 1971] Gabriel, P. and Ulmer, F. (1971) *Lokal präsentierbare Kategorien.* Springer Lecture Notes in Mathematics **221**.

[Gabriel and Zisman 1967] Gabriel, P. and Zisman, M. (1967) *Calculus of Fractions and Homotopy Theory.* Springer.

[Goguen and Burstall 1984] Goguen, J. A. and Burstall, R. M. (1984) *Introducing Institutions*. Springer Lecture Notes in Computer Science **164** 221–256.

[Goguen and Malcolm 2000] Goguen, J. and Malcolm, G. (2000) A hidden agenda. *Theoretical Computer Science* **245** (1) 55–101.

[Hensel and Reichel 1995] Hensel, U. and Reichel, H. (1995) Defining equations in terminal coalgebras. In: *Recent Trends in Data Type Specifications*, Springer Lecture Notes in Computer Science **906** 307–318.

[Kan 1958] Kan, D.M. (1958) Adjoint Functors. *Transactions of the American Mathematical Society* **87** 294–329.

[Lambán et al. 2003] Lambán, L., Pascual, V. and Rubio, J. (2003) An object-oriented interpretation of the EAT system. *Applicable Algebra in Engineering, Communication and Computing* **14** (3) 187–215.

[Lellahi 1989] Lellahi, S.K. (1989) Categorical abstract data type (CADT). *Diagrammes* **21**, SKL1-SKL23.

[Mac Lane 1998] Mac Lane, S. (1998) *Categories for the Working Mathematician.* Springer, 2th edition.

[Makkai 1997] Makkai, M. (1997) Generalized sketches as a framework for completeness theorems (I). *Journal of Pure and Applied Algebra* **115** 49–79.

[Pitts 2000] Pitts, A.M. (2000) Categorical Logic. Chapter 2 of Abramsky, S., Gabbay, D.M. and Maibaum, T.S.E. (Eds.) *Handbook of Logic in Computer Science* **5**. Algebraic and Logical Structures. Oxford University Press.

[Rubio et al. 2007] Rubio, J., Sergeraert, F. and Siret, Y. (2007) EAT: Symbolic Software for Effective Homology Computation. Institut Fourier, Grenoble. `ftp://fourier.ujf-grenoble.fr/pub/EAT`.

[Rutten 2000] Rutten, J.J.M.M. (2000) Universal coalgebra: a theory of systems. *Theoretical Computer Science* **249** (1) 3–80.

[Wells 1993] Wells, C. (1993) Sketches: Outline with References. `http://www.cwru.edu/artsci/math/wells/pub/papers.html`.