Math. Struct. in Comp. Science (2010), *vol.* 20, *pp.* 639–654. © Cambridge University Press 2010
doi:10.1017/S0960129510000150 First published online 24 June 2010

Diagrammatic logic applied to a parameterisation process

CÉSAR DOMÍNGUEZ[†§] and DOMINIQUE DUVAL[‡]

[†]*Departamento de Matemáticas y Computación, Universidad de La Rioja, Edificio Vives, Luis de Ulloa s/n, E-26004 Logroño, La Rioja, Spain*
Email: cesar.dominguez@unirioja.es
[‡]*Laboratoire Jean Kuntzmann, Université de Grenoble, 51 rue des mathématiques, BP 53, F-38041 Grenoble Cédex 9, France*
Email: Dominique.Duval@imag.fr

Received 25 August 2009; revised 19 April 2010

This paper provides an abstract definition of a class of logics, called diagrammatic logics, together with a definition of morphisms and 2-morphisms between them. The definition of the 2-category of diagrammatic logics relies on category theory, mainly on adjunction, categories of fractions and limit sketches. This framework is applied to the formalisation of a parameterisation process. This process, which consists of 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 parameterised specification and an actual parameter is shown to be a 2-morphism of logics.

1. Introduction

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

We will present the framework of diagrammatic logics in Section 2. This framework originated in Duval (2003; 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 logics to deal with computational effects, and for morphisms for expressing the meaning of these effects in more traditional 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 related to that in Makkai (1997). Our point of view is more abstract than institutions (Goguen and Burstall 1984) – see Duval (2003) for a comparison. Note that the current paper does not depend on Duval (2003; 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). Previous analysis of the data types used in EAT and Kenzo (Lambán *et al.* 2003; Domínguez *et al.* 2007; Domínguez *et al.* 2006) shows that there are two layers of data structures in these systems: the first layer consists of the usual abstract data

§ Partially supported by Ministerio de Educación y Ciencia, project MTM2009-13842-C02-01.

types, like the type of integers, while the second layer deals with algebraic structures, like the structure of groups, which are implemented using the abstract data types in the first layer. In addition, this analysis made it clear that in EAT, groups (for instance) are not implemented individually, but are implemented through *parameterised* families of groups. Lambán *et al.* (2003) defined an operation called the *imp* construction – it was given this name because of its role in the implementation process in the system EAT. In fact, the *imp* construction is a *parameterisation process*: starting from a specification Σ in which some operations are labelled as 'pure' (Domínguez *et al.* 2006), the *imp* construction builds a new specification Σ_A with a distinguished sort A added to the domain of each non-pure operation. Then the *parameter passing process* derives an implementation of Σ from each implementation of Σ_A and each value in the interpretation of A. Moreover, the *exact parameterisation property* was proved in Lambán *et al.* (2003): the implementations of EAT algebraic structures are as general as possible in the sense that they are ingredients of terminal objects in some categories of models. The parameter set in the Kenzo and EAT systems 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 any explicit storage, so each particular instance of the record gives rise to an algebraic structure; more details can be found in Lambán *et al.* (2003).

Lambán *et al.* (2003) then reinterpreted these results for the EAT algebraic structures in terms of object-oriented techniques, like *hidden algebras* (Goguen and Malcolm 2000) or *coalgebras* (Rutten 2000). Domínguez *et al.* (2007) then extended these results in the algebraic framework of *institutions* (Goguen and Burstall 1984): the *imp* construction is represented through institution encodings in the sense of Tarlecki (2000). Goguen and Roşu (2002) provided a survey of the different notions of morphisms between institutions, where institution encodings are called forward institution morphisms. In fact, institution encodings are not that common in institution theories, where institution morphisms are preferred. Also, the results in Domínguez *et al.* (2007) were obtained in a simplified context where the pure part in the given specifications was not taken into account. In addition, the institutional machinery could not be used directly as it had to be stretched with new *ad hoc* institutions and with degenerated parts of other institutions.

A first attempt at using diagrammatic logics to formalise this parameterisation process was given in Domínguez *et al.* (2005). In Section 3 we present a simple formalisation of the parameterisation and parameter passing processes as a morphism and a 2-morphism of diagrammatic logics, respectively.

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 will not always distinguish between equivalent categories. The 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 the various kinds of structures with an underlying graph, our terminology will refer to: 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. As usual, a *span* in a category is a pair of morphisms with the same source, and a *cospan* is a pair of morphisms with the same target.

https://doi.org/10.1017/S0960129510000150 Published online by Cambridge University Press

2. Diagrammatic logics

We will define the 2-category of diagrammatic logics and its related notions in Sections 2.1, 2.2 and 2.3, and then describe the diagrammatic equational logic in Section 2.4.

2.1. *Limit sketches*

There are several definitions of limit sketches (also called projective sketches), but in all of them 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 $id_X : X \to X$ in $\mathbf{E}$, which becomes the identity morphism at the object X in $C(\mathbf{E})$. As another example, 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 notational simplicity, 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 that maps the potential features of $\mathbf{E}_1$ to potential features of $\mathbf{E}_2$. This forms the category of limit sketches. A *realisation* (or *loose model*) of a limit sketch $\mathbf{E}$ with values in a category $\mathbf{C}$ is a graph morphism that maps the potential features of $\mathbf{E}$ to real features of $\mathbf{C}$. A morphism of realisations is (an obvious generalisation of) a natural transformation. This gives rise to the category $Real(\mathbf{E}, \mathbf{C})$ of realisations of $\mathbf{E}$ with values in $\mathbf{C}$, which is simply denoted by $Real(\mathbf{E})$ 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 realisation* $\mathscr{Y}_{\mathbf{E}}$ of a limit sketch $\mathbf{E}$ takes its values in $Real(\mathbf{E})$. It is defined as $\mathscr{Y}_{\mathbf{E}}(E) = P(\mathbf{E})[E, -]$ 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 $\mathscr{Y}_{\mathbf{E}}$, up to contravariance, the limit sketch $\mathbf{E}$ can be identified with a part of $Real(\mathbf{E})$, which will be called the *elementary* part of $Real(\mathbf{E})$ (with respect to $\mathbf{E}$) and denoted $Real_{el}(\mathbf{E})$. This 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 realisation or morphism of realisations of $\mathbf{E}$ can be obtained by colimits and cotuples from $Real_{el}(\mathbf{E})$. Moreover, a fundamental theorem due to Ehresmann states that every morphism of limit sketches $\mathbf{e} : \mathbf{E}_1 \to \mathbf{E}_2$ gives rise to an adjunction $F_{\mathbf{e}} \dashv G_{\mathbf{e}}$, where the right adjoint $G_{\mathbf{e}}$ is the precomposition with $\mathbf{e}$, which maps every $X_2 : \mathbf{E}_2 \to Set$ to $X_2 \circ \mathbf{e} : \mathbf{E}_1 \to Set$ (Ehresmann 1968):

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

https://doi.org/10.1017/S0960129510000150 Published online by Cambridge University Press

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

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

A *locally presentable category* (Gabriel and Ulmer 1971) is a category $\mathbf{C}$ that is equivalent to the category of set-valued realisations of a limit sketch $\mathbf{E}$, and we say $\mathbf{E}$ is a limit sketch *for* the category $\mathbf{C}$. In addition, we define a *locally presentable functor* as a functor $F : \mathbf{C}_1 \to \mathbf{C}_2$ that is the left adjoint to the precomposition with some morphism of limit sketches $\mathbf{e}$ such that $\mathbf{C}_1$ and $\mathbf{C}_2$ are locally presentable categories. We say $\mathbf{e}$ is a morphism of limit sketches *for* the functor F.

2.2. *Diagrammatic logic: models and proofs*

The framework of diagrammatic logics comes from Duval (2003; 2007).

Definition 2.1. A *diagrammatic logic* is a locally presentable functor $L : \mathbf{S} \to \mathbf{T}$ such that its right adjoint $R : \mathbf{T} \to \mathbf{S}$ is full and faithful. The categories $\mathbf{S}$ and $\mathbf{T}$ are the category of *specifications* and the category of *theories*, respectively, of the diagrammatic logic L (they are also called L-specifications and L-theories, respectively). A specification Σ *presents* a theory Θ if Θ is isomorphic to $L(\Sigma)$. Two specifications are *equivalent* if they present the same theory.

Informally, this definition means that each specification Σ generates a theory $L(\Sigma)$, and that each theory Θ may be viewed as a specification $R(\Theta)$. The fact that R is full and faithful is equivalent to the fact that the counit natural transformation $\varepsilon : L \circ R \Rightarrow Id$ is an isomorphism. According to Gabriel and Zisman (1967), it is also equivalent to the fact that L is a *localisation*, up to an equivalence of categories: it consists of adding inverse morphisms for some morphisms to constrain them to become isomorphisms. Consider a diagrammatic logic L:

$$\mathbf{S} \underset{R}{\overset{L}{\underset{\longleftarrow}{\overset{\longrightarrow}{\perp}}}} \mathbf{T}\,.$$

Definition 2.1 also means that R defines an isomorphism from $\mathbf{T}$ to its image, which is a reflective subcategory of $\mathbf{S}$ (reflective subcategories are defined in Mac Lane (1998, Chapter 4), but will not be used any further in the current paper). The fact that R is full and faithful means that every theory Θ, when viewed as a specification $R(\Theta)$, presents itself. The next definition claims that every model of a specification takes its values in some theory.

Definition 2.2. A *(strict) model* M of a specification Σ in a theory Θ is a morphism of theories $M : L\Sigma \to \Theta$ or, equivalently (thanks to the adjunction), a morphism of specifications $M : \Sigma \to R\Theta$.

It follows that equivalent specifications have the same models. A model M of Σ in Θ is sometimes called an *oblique morphism*, and is denoted $M : \Sigma \to \Theta$. If, 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

https://doi.org/10.1017/S0960129510000150 Published online by Cambridge University Press

$\mathbf{T}[L\Sigma, \Theta]$ is the *category of models of* Σ *in* Θ, and is denoted $L[\Sigma, \Theta]$. Otherwise, $L[\Sigma, \Theta]$ is simply the discrete category with the models of Σ in Θ as objects.

Definition 2.3. 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 that are related by entailments are equivalent.

Definition 2.4. An *instance* ρ of a specification Σ in a specification Σ_1 is a cospan in $\mathbf{S}$ made up 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 is denoted by $\rho = \tau \backslash \sigma : \Sigma \to \Sigma_1$.

We can illustrate an instance $\rho = \tau \backslash \sigma$ of Σ in Σ_1 graphically by

$$\Sigma \xrightarrow{\ \sigma\ } \Sigma'_1 \xleftarrow[\ \tau\ ]{} \Sigma_1$$

This then easily gives us 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 $L\tau$:

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

Since the category $\mathbf{S}$ has colimits and 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 $\rho = \tau \backslash \sigma : \Sigma \to \Sigma_1$ in $\mathbf{S_2}$ and define $L\rho = (L\tau)^{-1} \circ L\sigma : L\Sigma \to L\Sigma_1$ 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 localisation, the *quotient* category of the bicategory $\mathbf{S_2}$ is equivalent to $\mathbf{T}$.

Definition 2.5. An *inference system* for a diagrammatic logic L is a morphism of limit sketches $\mathbf{e} : \mathbf{E}_S \to \mathbf{E}_T$ for the locally presentable functor L.

Thanks to the Yoneda contravariant realisation, 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 $\mathbf{e} : \mathbf{E}_S \to \mathbf{E}_T$ for L – see Duval (2007) for more details.

Definition 2.6. 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 is denoted by $r = s/t : H \to C$.

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

https://doi.org/10.1017/S0960129510000150 Published online by Cambridge University Press

Definition 2.7. Given an inference rule $r = s/t : H \to C$ and an instance $\kappa : \mathscr{Y}(H) \to \Sigma$ of the hypothesis $\mathscr{Y}(H)$ in a specification Σ, the corresponding *inference step* provides the instance $\kappa \circ \mathscr{Y}(r) : \mathscr{Y}(C) \to \Sigma$ of the conclusion $\mathscr{Y}(C)$ in Σ.

Definition 2.8. A *proof* (or *derivation*, or *derived rule*) is the description of a fraction in $\mathbf{S_2}$ in terms of inference rules (using 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 consisting of Σ_1 together with an equation $f = g$ consisting 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 2.9. A *morphism of logics* $F : L_1 \to L_2$ 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:

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

Using the Yoneda contravariant realisation, a morphism of logics $F : L_1 \to L_2$ 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 (such as the undecoration morphism in Section 3.1) while others are easier to describe at the logic level (such as the parameterisation morphism in Section 3.2). The next result is a straightforward application of adjunction.

Proposition 2.10. Given a morphism of logics $F : L_1 \to L_2$ 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 2.11. A *2-morphism of logics* $\ell : F \Rightarrow F' : L_1 \to L_2$ is a pair of natural transformations (ℓ_S, ℓ_T) where $\ell_S : F_S \Rightarrow F'_S : \mathbf{S}_1 \to \mathbf{S}_2$ and $\ell_T : F_T \Rightarrow F'_T : \mathbf{T}_1 \to \mathbf{T}_2$ 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.

https://doi.org/10.1017/S0960129510000150 Published online by Cambridge University Press

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 questions about theories, typically, whether some morphisms of theories are invertible.

2.4. *The diagrammatic equational logic*

Equational logic provides a fundamental example of a diagrammatic logic. As usual in categorical logic (see Pitts (2000)), *equational theories* are defined as the categories with chosen finite products; together with the functors that preserve the chosen finite products, they form a category $\mathbf{T}_{eq}$. Similarly (see Lellahi (1989), Barr and Wells (1999) and Wells (1993)), *equational specifications* are defined as finite product sketches, which means the limit sketches (as in Section 2.1) such that their potential limits are only potential products; together 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 be just 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:

$$\mathbf{S}_{eq} \underset{R_{eq}}{\overset{L_{eq}}{\underset{\longleftarrow}{\overset{\longrightarrow}{\bot}}}} \mathbf{T}_{eq} \ .$$

The category of sets with cartesian products as the 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. Building limit sketches for $\mathbf{T}_{eq}$ and $\mathbf{S}_{eq}$ is a classical exercise, and it is then easy to check that L_{eq} is a diagrammatic logic. We will now give a simplified description – 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):

$$\texttt{Type} \underset{\texttt{codom}}{\overset{\texttt{dom}}{\leftleftarrows}} \texttt{Term} \ .$$

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 consists of $\mathbf{E}_{gr}$ with potential limits and tuples, is equivalent to $\mathbf{E}_{gr}$. The point `Type` has been duplicated to aid 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 $X \mapsto id_X$.

https://doi.org/10.1017/S0960129510000150 Published online by Cambridge University Press

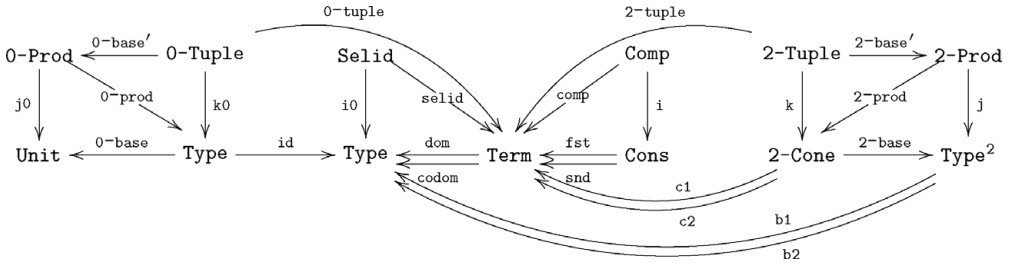


Fig. 1. The graph underlying $\mathbf{E}_{eq,S}$

— 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` 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` for the construction of the potential collapsing term $X \mapsto \langle\,\rangle_X : X \to 1$.

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, but we leave enlarging $\mathbf{E}_{eq,S}$ to get them as an exercise. 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 parameterisation process

In Section 3.1 we define several variants of the diagrammatic equational logic, which are related by morphisms. We then formalise the parameterisation and parameter passing processes in Sections 3.2 and 3.3, respectively.

3.1. *Some diagrammatic logics*

The theories of the *parameterised equational logic* L_A are the equational theories together with a distinguished type, called the *type of parameters*, and usually denoted A. The

https://doi.org/10.1017/S0960129510000150 Published online by Cambridge University Press

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

Fig. 2. Rules for the equational logic

specifications are the equational specifications with, possibly, a distinguished type A. The inclusion of limit sketches determines a morphism of logics $F_A : L_{eq} \to L_A$.

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

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. We can build $\mathbf{E}_{dec,T}$ from $\mathbf{E}_{eq,T}$, in a way that reflects the meaning of the word 'decoration' as follows – a smaller choice for $\mathbf{E}_{dec,T}$ can be found in Domínguez and Duval (2009). The decorations in this context simply consist of the 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 realisation Δ of $\mathbf{E}_{eq,T}$ with values in the category of equational theories, as follows. We will first describe the set-valued realisation Δ_0 of $\mathbf{E}_{eq,T}$ underlying Δ. The set $\Delta_0(\texttt{Type})$ consists of one type D and the set $\Delta_0(\texttt{Term})$ of two terms p and g, so

$$\begin{aligned}\Delta_0(\texttt{Cons}) &= \{(p,p),(p,g),(g,p),(g,g)\}\\ \Delta_0(\texttt{2-Cone}) &= \{(p,p),(p,g),(g,p),(g,g)\}\\ \Delta_0(\texttt{Type}^2) &= \{(D,D)\},\end{aligned}$$

https://doi.org/10.1017/S0960129510000150 Published online by Cambridge University Press

and we use the denotation

$$\Delta_0(\texttt{Unit}) = \{\star\}.$$

Then:

— $\Delta_0(\texttt{selid})$ maps D to p;
— $\Delta_0(\texttt{comp})$ maps (p,p) to p and everything else to g;
— $\Delta_0(\texttt{2-prod})$ maps (D,D) to (p,p);
— $\Delta_0(\texttt{2-tuple})$ maps (p,p) to p and everything else to g;
— $\Delta_0(\texttt{0-prod})$ maps $\star$ to p;
— $\Delta_0(\texttt{0-tuple})$ maps D to p.

The structure of the equational theory on each set $\Delta_0(E)$ is induced by a monomorphism $p \to g$ in $\Delta(\texttt{Term})$. Then $\mathbf{E}_{dec,T}$ is the *sketch of elements* (which is similar to the more usual *category of elements*) of the realisation 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 $\texttt{c} : \texttt{Term.p} \to \texttt{Term.g}$ over $\texttt{id}_{\texttt{Term}}$, which is a potential monomorphism for the conversion of pure terms to general terms.

Clearly, by forgetting the decorations, we get a morphism of diagrammatic logics F_{und} from L_{dec} to L_{eq}, which is 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 from L_{eq} to L_{dec} such that $F_{und} \circ F_p = id_{L_{eq}}$.

3.2. *The parameterisation process is a morphism of logics*

In this section we define a morphism of logics $F_{par} : L_{dec} \to L_A$. 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 parameterisation process starts from a decorated specification and returns a parameterised specification. Roughly speaking, it replaces every general feature in a decorated specification by a parameterised 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 $f : X \to Y$ is replaced by $f' : A \times X \to Y$ where A is the type of parameter. Figures 3 and 4 define the image of the elementary decorated specifications (pure terms are denoted using '$\rightsquigarrow$', and the projections $pr_X : A \times X \to A$ and $\varepsilon_X : A \times X \to X$ are often omitted): for each point E.x in $\mathbf{E}_{dec,S}$, the parameterisation process replaces the elementary decorated specification $\mathscr{Y}(\texttt{E.x})$ by the parameterised specification $F_{par}(\mathscr{Y}(\texttt{E.x}))$.

The morphisms between elementary decorated specifications are transformed in a straightforward way. For instance, the image of the morphism $\mathscr{Y}(\texttt{c})$, where $\texttt{c} : \texttt{Term.p} \to \texttt{Term.g}$ is the conversion arrow, maps $f' : A \times X \to X$ in $F_{par}(\mathscr{Y}(\texttt{Term.g}))$ to $f \circ \varepsilon_X : A \times X \to Y$ in $F_{par}(\mathscr{Y}(\texttt{Term.p}))$, or, more precisely, in a parameterised specification equivalent to $F_{par}(\mathscr{Y}(\texttt{Term.p}))$. This provides a graph morphism $F_{par} : Real_{el}(\mathbf{E}_{dec,S}) \to Real(\mathbf{E}_{A,S})$.

https://doi.org/10.1017/S0960129510000150 Published online by Cambridge University Press

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

Fig. 3. The parameterisation morphism on elementary decorated (pure) specifications

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

$$F_{par} : L_{dec} \to L_A,$$

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 *parameterisation* morphism.

Proof. It is straightforward to check 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. It is then easy to check the equality $F_{par} \circ F_p = F_A$ on elementary specifications. □

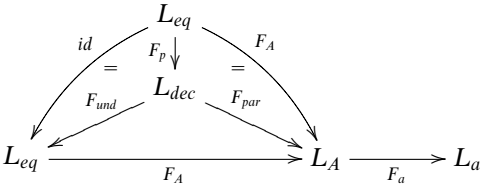
To sum up, we have built the following logics and morphisms of logics, where F_A, F_a and F_p come from inclusions of limit sketches. The morphism F_{und} is the identity on the pure part of L_{dec}. The morphisms F_{und}, F_{par} and F_A form a (non-commutative) triangle,

https://doi.org/10.1017/S0960129510000150 Published online by Cambridge University Press

	point E.x	$\mathcal{Y}$(E.x)	$F_{par}(\mathcal{Y}(\mathtt{E.x}))$
term	Term.g	$X \xrightarrow{f} Y$	$A \times X \xrightarrow{f'} Y$
composition	Comp.g	$X \xrightarrow{f} Y \xrightarrow{g} Z$, $g \circ f$	$A \times X \xrightarrow{\langle pr_X, f' \rangle} A \times Y \xrightarrow{g'} Z$, $g' \circ \langle pr_X, f' \rangle$
pairing	2-Tuple.g	$X \xrightarrow{\langle f,g \rangle} Y_1 \times Y_2$	$A \times X \xrightarrow{\langle f',g' \rangle} Y_1 \times Y_2$

Fig. 4. The parameterisation morphism on elementary decorated (general) specifications

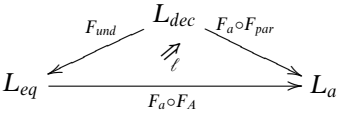
which becomes commutative when restricted to the pure part of L_{dec}:



The parameterisation morphism F_{par} formalises the parameterisation process. The span made of F_{und} and F_{par} formalises 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 parameterised specification $\Sigma_A = F_{par}(\Sigma_{dec})$.

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

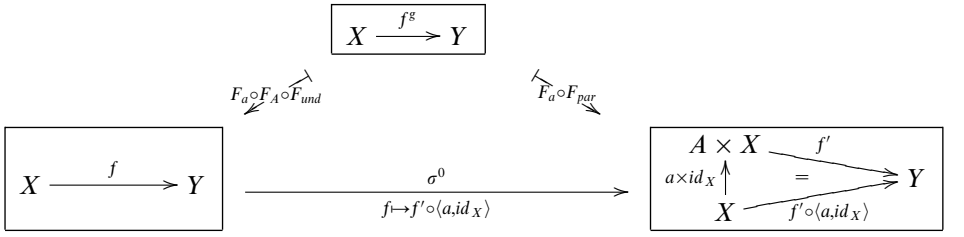
In this section, we define a 2-morphism of logics $\ell : F_a \circ F_A \circ F_{und} \Rightarrow F_a \circ F_{par}$:



In order to explain how a 2-morphism of logics may provide a formalisation of the parameter passing process, we will focus on the decorated specification $\Sigma^0_{dec} = \mathcal{Y}(\mathtt{Term.g})$ (where g means 'general', as above). This decorated specification Σ^0_{dec} consists of a term $f^g : X \to Y$ decorated as 'general' (the superscript 'g' emphasises this decoration), and its pure part consists of X and Y. On the one hand, the L_{eq}-specification $\Sigma^0_{eq} = F_{und}(\Sigma^0_{dec})$ is $f : X \to Y$, as well as the L_A-specification $F_A(F_{und}(\Sigma^0_{dec}))$ and the L_a-specification $\Sigma^0_{eq,a} = F_a(F_A(F_{und}(\Sigma^0_{dec})))$. On the other hand, as in Figure 4, the L_A-specification $\Sigma^0_A = F_{par}(\Sigma^0_{dec})$ is $f' : A \times X \to Y$ (which is shorthand for A, X, Y, the product $A \times X$ with its projections,

https://doi.org/10.1017/S0960129510000150 Published online by Cambridge University Press

and $f' : A \times X \to Y$), hence the L_a-specification $\Sigma_a^0 = F_a(F_{par}(\Sigma_{dec}^0))$ is also $f' : A \times X \to Y$. Up to entailment in the logic L_a, we may add to Σ_a^0 the terms $a : 1 \to A$, $a \times id_X : X \to A$ (where X is identified with $1 \times X$ for readability) and $f' \circ \langle a, id_X \rangle : X \to Y$. Obviously, there is a morphism of L_a-specifications $\sigma^0 : \Sigma_{eq,a}^0 \to \Sigma_a^0$ that maps f to $f' \circ \langle a, id_X \rangle$.



We claim that this morphism σ^0 corresponds to the parameter passing process. Indeed, if we look at set-valued models, a model M_A of Σ_A^0 consists of three sets $\mathbb{X}$, $\mathbb{Y}$, $\mathbb{A}$ for interpreting X, Y and A, respectively, and a function $\varphi' : \mathbb{A} \times \mathbb{X} \to \mathbb{Y}$ for interpreting f'. A model $M_{A,\alpha}$ of Σ_a^0 extending M_A is characterised by the *argument* α in $\mathbb{A}$, which interprets the *formal parameter* a. Then $M_{A,\alpha} \circ \sigma$ is the model of Σ_{eq}^0, which interprets f as $\varphi'(\alpha, -) : \mathbb{X} \to \mathbb{Y}$. So, when we focus on this decorated specification Σ_{dec}^0, the morphism of L_a-specifications σ^0 corresponds to the parameter passing process. More generally, the parameter passing process is now defined as a family of morphisms of L_a-specifications $\ell_{\Sigma_{dec}} : F_a(F_A(F_{und}(\Sigma_{dec}))) \to F_a(F_{par}(\Sigma_{dec}))$ with naturality conditions, which means, as a 2-morphism $\ell : F_a \circ F_A \circ F_{und} \Rightarrow F_a \circ F_{par}$.

Each decorated specification Σ_{dec}, with $\Sigma_{eq} = F_{und}(\Sigma_{dec})$, gives rise to two specifications with a 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, but, on the other hand, $\Sigma_a = F_a(F_{par}(\Sigma_{dec}))$, which is $F_{par}(\Sigma_{dec})$ viewed as a specification with a parameter. The morphism $\ell_{\Sigma_{dec}} : \Sigma_{eq,a} \to \Sigma_a$ is defined as follows. When Σ_{dec} is some $\mathscr{Y}(\mathtt{E.p})$ (where `p` means 'pure') it is easy to check that $\Sigma_{eq,a} = \Sigma_a$, so $\ell_{\Sigma_{dec}}$ is the identity. When $\Sigma_{dec} = \mathscr{Y}(\mathtt{Term.g})$ (where `g` means 'general'), we have $\ell_{\Sigma_{dec}} = \sigma^0$, as defined above. The definitions when $\Sigma_{dec} = \mathscr{Y}(\mathtt{Comp.g})$ and when $\Sigma_{dec} = \mathscr{Y}(\mathtt{2\text{-}Tuple.g})$ are similar.

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

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

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

Proof. We first extend the definition of $\ell_{\Sigma_{dec}}$ on the elementary decorated specifications to all specifications by colimits, and the result then follows. □

Theorem 3.2 has the expected consequence on models, which is stated as Proposition 3.3: given a set-valued model M_A of the paramererised specification Σ_A, each $\alpha \in M_A(A)$, which is called an *actual parameter* or an *argument*, gives rise to a model $\mathscr{M}(\alpha)$ of the equational specification Σ_{eq}. We introduce the following notation:

— For each set $\mathbb{A}$, let $Set_{\mathbb{A}}$ denote the object of $\mathbf{T}_A$ consisting of the equational theory of sets with $\mathbb{A}$ as the interpretation of A, so $R_A(Set_{\mathbb{A}}) = Set$.

https://doi.org/10.1017/S0960129510000150 Published online by Cambridge University Press

— For each set $\mathbb{A}$ and element $\alpha \in \mathbb{A}$, let $Set_{\mathbb{A},\alpha}$ denote the object of $\mathbf{T}_a$ consisting of the equational theory of sets with $\mathbb{A}$ and α as the interpretations of A and a, respectively, so $R_a(Set_{\mathbb{A},\alpha}) = Set_{\mathbb{A}}$.
— For each decorated specification $\Sigma_{dec} = (\Sigma_{eq}, \Sigma_0)$, consisting 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 $f : X \to Y$ in Σ_{eq} there is a $f' : A \times X \to Y$ 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 $M_A : \Sigma_A \to Set_{\mathbb{A}}$ in L_A, let $M_0 : \Sigma_{eq} \to Set$ denote the restriction of M_A to Σ_0. Then there is a function

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

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

$$\ell_{\Sigma_{dec}} : \Sigma_{eq,a} \to \Sigma_a$$

gives rise to a functor

$$L_a[\Sigma_a, Set_{\mathbb{A},\alpha}] \to L_a[\Sigma_{eq,a}, Set_{\mathbb{A},\alpha}].$$

Proposition 2.10 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. The definition of $\ell_{\Sigma_{dec}}$ means that $M_{A,\alpha}$ extends M_0 and satisfies

$$M_{A,\alpha}(f) = M_A(f')(\alpha, -)$$

for each $f : X \to Y$ in Σ_{eq}. When M_A is fixed, the result now follows by defining $\mathscr{M}(\alpha) = M_{A,\alpha}$. □

The function $\mathscr{M}$ is not a bijection in general, but under the conditions of Proposition 3.4 it may be: this is the *exact parameterisation* property of Lambán *et al.* (2003, Corollary 2), which is also proved in Domínguez and Duval (2009, Corollary 3.8).

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 be a terminal model of Σ_A extending M_0. Then the function $\mathscr{M}$

https://doi.org/10.1017/S0960129510000150 Published online by Cambridge University Press

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 and EAT, as described in Lambán *et al.* (2003).

4. Conclusion

In this paper we have defined the framework of diagrammatic logics and its application to both a parameterisation process and its associated parameter passing process. Our focus here has been on the models; Dumas *et al.* (2009) studied another kind of application where proofs in a diagrammatic logic play an important role. In this paper we have only considered equational logic and some related logics. However, building on the theories of limit sketches and locally presentable categories, it should be possible to build a large variety of diagrammatic logics.

References

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

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

Domínguez, C. and Duval, D. (2009) A parameterization process as a categorical construction. Available at `arXiv:0908.3634`.

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

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

Dumas, J. G., Duval, D. and Reynaud, J. C. (2009) Cartesian effect categories are Freyd-categories. (Available at `arXiv:0903.3311`.)

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

Duval, D. (2007) Diagrammatic inference. (Available at `arXiv:0710.1208`.)

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

Gabriel, P. and Ulmer, F. (1971) Lokal präsentierbare Kategorien. *Springer-Verlag Lecture Notes in Computer Science* **221**.

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

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

https://doi.org/10.1017/S0960129510000150 Published online by Cambridge University Press

Goguen, J. and Malcolm, G. (2000) A hidden agenda. *Theoretical Computer Science* **245** (1) 55–101.
Goguen, J. A. and Roşu, G. (2002) Institution morphisms. *Formal Aspects of Computing* **13** 274–307.
Hensel, U. and Reichel, H. (1995) Defining equations in terminal coalgebras. In: Recent Trends in Data Type Specifications. *Springer-Verlag Lecture Notes in Computer Science* **906** 307–318.
Kan, D. M. (1958) Adjoint Functors. *Transactions of the American Mathematical Society* **87** 294–329.
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, S. K. (1989) Categorical abstract data type (CADT). *Diagrammes* **21** SKL1–SKL23.
Mac Lane, S. (1998) *Categories for the Working Mathematician*, 2nd Edition Springer-Verlag.
Makkai, M. (1997) Generalized sketches as a framework for completeness theorems (I). *Journal of Pure and Applied Algebra* **115** 49–79.
Pitts, A. M. (2000) Categorical Logic. In: Abramsky, S., Gabbay, D. M. and Maibaum, T. S. E. (eds.) Algebraic and Logical Structures. *Handbook of Logic in Computer Science* **5** Chapter 2, Oxford University Press.
Rubio, J., Sergeraert, F. and Siret, Y. (2007) EAT: Symbolic Software for Effective Homology Computation. Institut Fourier, Grenoble. (Available at `http://www-fourier.ujf-grenoble.fr/~sergerar/Kenzo/#Eat`.)
Rutten, J. J. M. M. (2000) Universal coalgebra: a theory of systems. *Theoretical Computer Science* **249** (1) 3–80.
Tarlecki, A. (2000) Towards heterogeneous specifications. In: Gabbay, D. M. and de Rijke, M. (eds.) *Frontiers of Combining Systems (FroCos'98)*, Studies in Logic and Computation **7** Research Studies Press/Wiley 337–360.
Wells, C. (1993) Sketches: Outline with References. (Available at `http://www.cwru.edu/artsci/math/wells/pub/papers.html`.)

https://doi.org/10.1017/S0960129510000150 Published online by Cambridge University Press