Using limits of parchments to systematically construct institutions of partial algebras

Till Mossakowski

University of Bremen, Dept. of Computer Science, P.O.Box 33 04 40, D-28334 Bremen
Phone: +49-421-218-2935, E-mail: till@informatik.uni-bremen.de

1 Introduction

It has been widely accepted that partial algebras are an important semantical basis for the specification of software systems (cf. Sect. 2.1 of [3]). The motivation for this work grew out of a systematic study of institutions capturing partiality in different ways [13].

When constructing an institution of partial algebras, one usually redefines every concept for total algebras from scratch, and generalizes it to the partial case. This may become a tedious process. Moreover, often one has to choose between different generalizations, but the design decisions are scattered all over the definitions.

In this work, we try to systematize the combination of different institutions. Goguen and Burstall have argued [9, 5] that combinations can be done via (co-)limits. Now the category of institutions and institution morphisms [9] is complete [18], but limits in this category put together different concepts only *without* true interaction.

Therefore, we move to Goguen's and Burstall's parchments [8], which allow defining institutions using

- initial algebra semantics for the definition of sentence syntax,
- a special signature for the definition of models and
- a semantical algebra for the definition of semantical evaluation of sentences.

We introduce a notion of morphism between parchments, which allows us to combine different concepts *with* interaction, using limits.

Now there is the problem that, in many cases, the limit contains unwanted "junk" truth values and thus the space of truth values need not be preserved. This problem can be solved by imposing a congruence relation on the semantical algebra of the limit. The congruence can usually be generated from a small number of axioms, which state how different concepts interact.

Thus, we cannot expect a combination to be done automatically (as in the case of combining theories), but we have to use a two-step semi-automatic method: first specify the components to be combined, and then specify interaction of concepts.

These ideas are illustrated by generalizing several concepts from total to partial algebras. The role of the congruence relation here is to decide whether we want a two- or three-valued logic and to determine which concept of equality to use.

The paper is organized as follows: Section 2 introduces institutions and institution morphisms from [9], showing by an example that limits of institutions only allow combinations without interaction. Section 3 recalls parchments [8]. Then, in Sect. 4, parchment morphisms are defined. This is used for combination of parchments via limits in Sect. 5. Section 6 contains some conclusions and compares our work with related work.

2 Why Institution Morphisms Do Not Suffice

A specification language is usually based on some notion of signature, model, sentence and satisfaction. These are the ingredients of Goguen's and Burstall's abstract model theory [9]:

An *institution* $I = (\underline{Sign}^I, sen^I, \underline{Mod}^I, \models^I)$ consists of

1. a category $\underline{Sign}^I$ of *signatures*,
2. a functor $sen^I : \underline{Sign}^I \longrightarrow \underline{Set}$ giving the set of *sentences* over a given signature,
3. a functor $\underline{Mod}^I : (\underline{Sign}^I)^{op} \longrightarrow \underline{Classes}$ giving the class[1] of *models* over a given signature,
4. a satisfaction relation $\models^I_\Sigma \subseteq \underline{Mod}^I(\Sigma) \times sen^I(\Sigma)$ for each $\Sigma \in \underline{Sign}^I$

such that for each $\sigma : \Sigma \longrightarrow \Sigma'$ in $\underline{Sign}^I$ the *Satisfaction Condition*, stating that *truth is invariant under change of notation*, holds:

$$M' \models^I_{\Sigma'} sen^I(\sigma)(\varphi) \iff M'|_\sigma \models^I_\Sigma \varphi$$

We write $M'|_\sigma$ (the *reduct* of M' under σ) for $\underline{Mod}^I(\sigma)(M')$. If $\sigma : \Sigma \longrightarrow \Sigma'$ is an inclusion, we also write $M'|_\Sigma$.

Now there are different notions of map or morphisms between institutions (see [6] for an overview). We here use institution morphisms from [9], since they capture the intuition of building a more complex institution I over a simpler institution J. The idea is that J may contain some "basic" concepts, and I may add some further concepts to J.

An *institution morphism* $\mu = (\Phi, \alpha, \beta) : I \longrightarrow J$ consists of

- a functor $\Phi : \underline{Sign}^I \longrightarrow \underline{Sign}^J$,
- a natural transformation $\alpha : sen^J \circ \Phi \longrightarrow sen^I$ and
- a natural transformation $\beta : \underline{Mod}^I \longrightarrow \underline{Mod}^J \circ \Phi^{op}$

such that the following satisfaction invariant holds:

$$\beta_\Sigma(M') \models^J_{\Phi\Sigma} \varphi \iff M' \models^I_\Sigma \alpha_\Sigma(\varphi)$$

Theorem 1 (Tarlecki[18]). *The category* $\underline{Ins}$ *of institutions and institution morphisms is complete.* □

[1] There is no problem in allowing *categories* of models here, but model morphisms are entirely orthogonal to the subject of this paper, so we leave them out here.

Thus we can apply the idea of combining things via colimits to institutions themselves, with the special point that we have to take limits here instead of colimits. Taking limits in $\underline{\mathcal{CAT}}$ results in categories of "amalgamated objects", i. e. we put signatures and models together at the level of single objects. In contrast to this, sentences are combined with colimits in $\underline{Set}$ (due to the contravariant direction of the sentence component). That is, *sets of* sentences are combined. To show how this works, we introduce some well-known institutions and morphisms between them.

Example 1. The institution **MS** of many-sorted algebras (with trivial sentences only). Signatures $\Sigma = (S, OP)$ consist of (finite sets) of sort symbols $s \in S$ and total operation symbols $op\colon s_1, \ldots, s_n \longrightarrow s \in OP$. Signature morphisms map symbols of the appropriate kind to each other, respecting the sorting.

A Σ-model (or Σ-algebra) A consists of a family $|A| = (A_s)_{s \in S}$ of carrier sets and a family $(op_A\colon A_{s_1} \times \cdots \times A_{s_n} \longrightarrow A_s)_{op:s_1,\ldots,s_n \longrightarrow s \in OP}$ of total operations.

If $\Sigma = (S, OP)$, $\sigma\colon \Sigma \longrightarrow \Sigma'$ is a signature morphism and $A' \in |\underline{Mod}^{\mathbf{MS}}\ \Sigma'|$, then the reduct $A'|_\sigma$ is the Σ-algebra A with $A_s := A'_{\sigma(s)}$ $(s \in S)$ and $op_A := (\sigma(op))_{A'}$ $(op \in OP)$.

There is only one sentence *true* for each signature, which holds in every model. □

Example 2. The institution **MS**(=) (**MS** with equality) inherits signatures, models and reducts from **MS** . A sentence over a signature $\Sigma = (S, OP)$ is a formula

$$\forall X.t_1 = t_2$$

where X is an S-sorted, pairwise disjoint system of variables and t_1, t_2 are Σ-terms in variables X. Now $A \models_\Sigma t_1 = t_2$ iff for all valuations $\nu\colon X \longrightarrow A$, we have $\nu^\#(t_1) = \nu^\#(t_2)$, where $\nu^\#\colon T_\Sigma(X) \longrightarrow A$ is the homomorphic extension of ν from X to $T_\Sigma(X)$. For details and the verification of the satisfaction condition, see [9].

Additionally, the sentence *true* holding in every model is kept from **MS** . □

Example 3. The institution **PMS** of partial many-sorted algebras (with trivial sentences only) has signatures of form $\Sigma = (S, OP, POP)$ consisting of sort symbols $s \in S$, total operation symbols $op\colon s_1, \ldots, s_n \longrightarrow s \in OP$ and partial operation symbols $pop\colon s_1, \ldots, s_n \longrightarrow s \in POP$, with signature morphisms defined as expected.

A Σ-model A consists of a (S, OP)-model in **MS** plus a family $(pop_A\colon A_{s_1} \times \cdots \times A_{s_n} \multimap A_s)_{pop:s_1,\ldots,s_n \longrightarrow s \in POP}$ of partial operations. Reducts are defined as in **MS** .

There is only one sentence *true* for each signature, which holds in every model. □

Example 4. The institution morphism $\mu^= = (Id, \alpha^=, Id)\colon \mathbf{MS}(=) \longrightarrow \mathbf{MS}$ is the identity on signatures and models, and for sentences, $\alpha^=_\Sigma$ is the inclusion. □

Example 5. The institution morphism $\mu^{\mathrm{P}} = (\Phi^{\mathrm{P}}, Id, \beta^{\mathrm{P}})\colon \mathbf{PMS} \longrightarrow \mathbf{MS}$ maps a signature $\Sigma = (S, OP, POP)$ to $\Phi^{\mathrm{P}}(\Sigma) = (S, OP)$. Now a Σ-model consists of a $\Phi^{\mathrm{P}}(\Sigma)$-model plus some family of partial operations, and $\beta^{\mathrm{P}}_\Sigma$ just forgets this family of partial operations. □

We are now prepared to consider the combination of $\mathbf{MS}\,(=)$ and $\mathbf{PMS}$ via the pullback

$$\begin{tikzcd}
\mathbf{PMS}(=) \arrow[r, dotted] \arrow[d, dotted] & \mathbf{PMS} \arrow[d, "\mu^{\mathrm{P}}"] \\
\mathbf{MS}(=) \arrow[r, "\mu^{=}"] & \mathbf{MS}
\end{tikzcd}$$

It turns out that signatures and models in $\mathbf{PMS}\,(=)$ are those of $\mathbf{PMS}$, while (S, OP, POP)-sentences are (S, OP)-sentences in $\mathbf{MS}\,(=)$. Thus all equations in $\mathbf{PMS}\,(=)$ are built up from total operation symbols, while a true interaction of concepts should allow us to write down equations containing partial operation symbols as well.

3 Parchments

The lack of interaction of concepts when taking limits of institutions shows that it is not enough to have a colimit (for example, a disjoint union) of sentence sets. We rather would like to unite *constructions*, or *operators* on sentence sets.

The idea behind the notion of parchment introduced by Goguen and Burstall [8] is to generate sentences by initial algebra semantics. For each signature Σ, an abstract syntax $Lang\,\Sigma$, given by some sorts and operations, then generates the Σ-sentences which are considered as terms over $Lang\,\Sigma$.

A combination of abstract syntaxes of sentences then causes a true interaction of different syntactical operators in the generation process. This is also reflected in the semantics. The semantics is given by

- a special signature Γ acting as a semantical universe, such that models are signature morphisms into Γ and
- a $Lang\,\Gamma$-algebra $\mathcal{G}$ specifying semantical evaluation.

Let $\underline{Sign}^{\mathbf{MS}\,*}$ be the category of many-sorted algebraic signatures $\underline{Sign}^{\mathbf{MS}}$ with a distinguished sort $*$ and a distinguished operation symbol $true : *$ and signature morphisms preserving $*$ and $true$.

A *parchment* $P = (\underline{Sign}^P, Lang^P, \Gamma^P, \mathcal{G}^P)$ consists of

- a category $\underline{Sign}^P$ of signatures,
- a functor $Lang^P : \underline{Sign}^P \longrightarrow \underline{Sign}^{\mathbf{MS}\,*}$,
- a signature Γ^P in $\underline{Sign}^P$ and
- a semantical algebra $\mathcal{G}^P$ in $\underline{Mod}^{\mathbf{MS}\,*}\,Lang^P\,\Gamma^P$.

$\mathcal{G}^P_*$ is the *space of truth values* of P. Usually, $\mathcal{G}^P_* = \mathbf{Bool}$ (with $\mathbf{Bool} = \{\,true;\ false\,\}$) is required, but for being able to construct limits later on, we drop this restriction.

Instead, we interpret $true_{\mathbb{G}}$ as true and all other values in $\mathbb{G}_*$ as false in the following:

A parchment $P = (\underline{Sign}^P, Lang^P, \Gamma^P, \mathbb{G}^P)$ yields an institution $I(P) = (\underline{Sign}^P, sen, \underline{Mod}, \models)$ where

- $sen\ \Sigma = (T_{Lang^P\ \Sigma})_*$, where $T_{Lang^P\ \Sigma}$ is the initial $Lang^P$ Σ-algebra
- $sen\ (\sigma\colon \Sigma \longrightarrow \Sigma') = (T_{Lang^P\ \sigma})_*$
  where $T_{Lang^P\ \sigma}\colon T_{Lang^P\ \Sigma} \longrightarrow T_{Lang^P\ \Sigma'}|_{Lang^P\ \sigma}$ is the initial homomorphism
- $\underline{Mod}\ \Sigma = \{\, m\colon \Sigma \longrightarrow \Gamma \in \underline{Sign}^P(\Sigma, \Gamma)\,\}$
- $\underline{Mod}\ \sigma(\Sigma' \xrightarrow{m} \Gamma) = \Sigma \xrightarrow{\sigma} \Sigma' \xrightarrow{m} \Gamma$
- $m \models_\Sigma \varphi$ iff $m_*^\natural(\varphi) = true_{\mathbb{G}^P}$
  where $m^\natural\colon T_{Lang^P\ \Sigma} \longrightarrow \mathbb{G}|_{Lang^P\ m}$ is the initial homomorphism.

With this definition, $\underline{Mod}$ always becomes continuous, but there is a slightly different formulation (dropping Γ) which also models non-continuous $\underline{Mod}$.

Example 1'. The institution of many-sorted algebras can be generated by a parchment $(\underline{Sign}, Lang, \Gamma, \mathbb{G})$, also denoted by **MS** .

- $\underline{Sign}$ is taken from the institution **MS**
- Γ has as sorts all sets $s \in |\underline{Set}|$, and as operation symbols $op\colon s_1, \ldots, s_n \longrightarrow s$ all functions $op\colon s_1 \times \cdots \times s_n \longrightarrow s \in |\underline{Set}|$[2]
- $Lang$ takes an **MS**-signature (S, OP) to the (overloaded) **MS** $*$-signature
  sorts $*$
  $Term(X, s)$ for X an S-sorted variable system, $s \in S$
  opns $true : *$
  $x\colon \longrightarrow Term(X, s)$ for $x : s \in X$
  $op : Term(X, s_1) \times \cdots \times Term(X, s_n) \longrightarrow Term(X, s)$
  for X S-sorted, $op\colon s_1, \ldots, s_n \longrightarrow s \in OP$
- $\mathbb{G}_* = \mathbf{Bool}$, $true_{\mathbb{G}} = true$
- $\mathbb{G}_{Term(X,s)} = [Env(X) \longrightarrow s]$
  where $Env(X)$ is the set of all valuations v of X in S with $v(x : s) \in s$
- $x_{\mathbb{G}} = \lambda v.v(x)$ for $x : s \in X$
- $op_{\mathbb{G}}(f_1, \ldots, f_n) = \lambda v.op(f_1(v), \ldots, f_n(v))$ for $op\colon s_1, \ldots, s_n \longrightarrow s \in \Gamma$ □

Example 2'. **MS** $(=)$ is $(\underline{Sign}, Lang, \Gamma, \mathbb{G})$ with

- $\underline{Sign} = \underline{Sign}^{\mathbf{MS}}$, $\Gamma = \Gamma^{\mathbf{MS}}$
- $Lang(S, OP) = Lang^{\mathbf{MS}}\ (S, OP) +$
  sorts $Formula(X)$ for X an S-sorted variable system
  opns $=\colon Term(X, s)^2 \longrightarrow Formula(X)$ for X S-sorted, $s \in S$
  $bind\colon Formula(X) \longrightarrow *$ for X S-sorted
- $\mathbb{G}|_{Lang^{\mathbf{MS}}\ \Gamma} = \mathbb{G}^{\mathbf{MS}}$

[2] For lack of space, we disregard foundational problems with this construction here.

- $\mathbb{G}_{Formula(X)} = [Env(X) \longrightarrow \mathbf{Bool}\,]$
- $=_{\mathbb{G}}\ (f_1, f_2) = \lambda v. \begin{cases} true, & \text{if } f(v) = g(v) \\ false, & \text{if } f(v) \neq g(v) \end{cases}$
- $bind_{\mathbb{G}}(f) = \begin{cases} true, & \text{if } \forall v \in Env(X) : f(v) = true \\ false, & \text{if } \exists v \in Env(X) : f(v) = false \end{cases}$ □

Example 3'. The partial many sorted parchment $\mathbf{PMS} = (\underline{Sign}, Lang, \Gamma, \mathbb{G})$:

- $\underline{Sign}$ is taken from the institution $\mathbf{PMS}$
- Γ has as sorts all sets, and as total (partial) operation symbols all total (partial) functions between the corresponding sets
- $Lang(S, OP, POP) = Lang^{\mathbf{MS}}\ (S, OP) +$
  opns $pop : Term(X, s_1) \times \cdots \times Term(X, s_n) \longrightarrow Term(X, s)$
  for X S-sorted, $pop: s_1, \ldots, s_n \longrightarrow s \in POP$
- $\mathbb{G}_* = \mathbf{Bool}$, $\quad true_{\mathbb{G}} = true$
- $\mathbb{G}_{Term(X,s)} = [Env(X) \multimap s]$ (the set of all partial maps from $Env(X)$ to s)
- $x_{\mathbb{G}}(v) = \begin{cases} v(x), & \text{if } v(x) \text{ is defined} \\ \text{undefined}, & \text{otherwise} \end{cases}$
- $op_{\mathbb{G}}(f_1, \ldots, f_n)(v) = \begin{cases} op(f_1(v), \ldots, f_n(v)), & \text{if } f_1(v), \ldots, f_n(v) \text{ def.} \\ \text{undefined}, & \text{otherwise} \end{cases}$
- $pop_{\mathbb{G}}(f_1, \ldots, f_n)(v) = \begin{cases} pop(f_1(v), \ldots, f_n(v)), & \text{if } f_1(v), \ldots, f_n(v) \text{ def. and} \\ & pop(f_1(v), \ldots, f_n(v)) \text{ def.} \\ \text{undefined}, & \text{otherwise} \end{cases}$

□

We can justify the overloading of names by the fact that $I(\mathbf{MS}) = \mathbf{MS}$, $I(\mathbf{MS}\,(=)) = \mathbf{MS}\,(=)$ and $I(\mathbf{PMS}) = \mathbf{PMS}$.

4 Parchment Morphisms

The intuition when setting up a parchment morphism $(\Phi, \alpha, g): P \longrightarrow P'$ is that P is more expressive than P', or built over P', adding some new concepts. Φ forgets the new parts of signatures, while α injects the old abstract syntax of sentences into the new one, and g injects the old semantics into the new one.

Definition 2. Given two parchments $P = (\underline{Sign}, Lang, \Gamma, \mathbb{G})$ and $P' = (\underline{Sign'}, Lang', \Gamma', \mathbb{G}')$, a *parchment morphism*[3] $(\Phi, \alpha, g): P \longrightarrow P'$ consists of

- a functor $\Phi: \underline{Sign} \longrightarrow \underline{Sign'}$ such that $\Phi\,\Gamma = \Gamma'$
- a natural transformation $\alpha: Lang' \circ \Phi \longrightarrow Lang$

[3] Andrzej Tarlecki [16] has a different notion of parchment morphism. This and its relation to our notion is discussed in a forthcoming joint paper.

– a $Lang'\,\Gamma'$-homomorphism $g\colon \mathbb{G}' \longrightarrow \mathbb{G}|_{\alpha_\Gamma}$

This gives a category $\underline{Parch}$ of parchments and parchment morphisms.

If $g_* = Id$ then (Φ, α, g) *preserves truth.* (We always have that $g_*(true_{\mathbb{G}'}) = true_{\mathbb{G}}$, but falsehood need not be preserved.) Let $\underline{TruthParch}$ be the category of parchments and parchment morphisms preserving truth. □

Proposition 3. *I can be extended to a functor* $I\colon \underline{TruthParch} \longrightarrow \underline{Ins}$.

Proof:
Put $I((\underline{Sign}, Lang, \Gamma, \mathbb{G}) \xrightarrow{(\Phi,\alpha,g)} (\underline{Sign}', Lang', \Gamma', \mathbb{G}')) = I(\underline{Sign}, Lang, \Gamma, \mathbb{G}) \xrightarrow{(\Phi,\bar{\alpha},\beta)} I(\underline{Sign}', Lang', \Gamma', \mathbb{G}')$ with

– $\bar{\alpha}_\Sigma = init_*$, where $T_{Lang'\,\Phi\,\Sigma} \xrightarrow{init} T_{Lang\,\Sigma}|_{\alpha_\Sigma}$ is the initial homomorphism

– $\beta_\Sigma(\Sigma \xrightarrow{m} \Gamma) = \Phi\,\Sigma \xrightarrow{\Phi\,m} \Phi\,\Gamma \xrightarrow{id} \Gamma'$

Now $\beta_\Sigma(m) \models_{\Phi\,\Sigma} \varphi'$ iff $(\Phi\,m)^\natural_*(\varphi') = true_{\mathbb{G}'}$ iff $g_*((\Phi\,m)^\natural_*(\varphi')) = true_{\mathbb{G}}$ iff $(m)^\natural_*(init_*(\varphi'))) = true_{\mathbb{G}}$ (by the diagram below) iff $m \models_\Sigma \bar{\alpha}_\Sigma\,\varphi'$.

$$\begin{tikzcd}
T_{Lang'\,\Phi\,\Sigma} \arrow[rr, "(\Phi\,m)^\natural"] \arrow[d, "init"'] & & \mathbb{G}|_{Lang'\,\Phi\,m} \arrow[d, "g\,|_{Lang'\,\Phi\,m}"] \\
T_{Lang\,\Sigma}|_{\alpha_\Sigma} \arrow[r, "(m)^\natural\,|_{\alpha_\Sigma}"'] & \mathbb{G}|_{Lang\,m \circ \alpha_\Sigma} \arrow[r, "Id"'] & \mathbb{G}|_{\alpha_\Gamma \circ Lang'\,\Phi\,m}
\end{tikzcd}$$

The diagram commutes because $T_{Lang'\,\Phi\,\Sigma}$ is initial. □

Now the institution morphisms $\mu^=$ and μ^{P} can be lifted to parchment morphisms (preserving truth):

Example 4'. Let $\mu^=\colon \mathbf{MS}\,(=) \longrightarrow \mathbf{MS}$ be (Id, α, Id), where α_Σ is the inclusion. □

Example 5'. Let $\mu^{\mathrm{P}}\colon \mathbf{PMS} \longrightarrow \mathbf{MS}$ be (Φ, α, g) with

– $\Phi\,(S, OP, POP) = (S, OP)$

– α_Σ is the inclusion

– $g_{Term(X,s)}$ is the inclusion of $[Env(X) \longrightarrow s]$ into $[Env(X) \multimap s]$, and g is the identity for the remaining sorts □

To combine $\mathbf{MS}(=)$ and $\mathbf{PMS}$, we would like to take a pullback now. But first, we have to prove completeness.

5 Putting Parchments Together Using Colimits

When combing parchments, a natural condition to require would be that the space of truth valued is preserved. That is, we want to take limits in $\underline{TruthParch}$. But $\underline{TruthParch}$ is *not* complete (see Proposition 11 below)! Therefore, we have to take limits in $\underline{Parch}$, at the risk of getting parchment morphisms *not* preserving truth, even when starting with a diagram in $\underline{TruthParch}$.

We now need some prerequisites. The following theorem is Theorem 2 in [19]:

Theorem 4. *If $C: Ind^{op} \longrightarrow \underline{\mathcal{CAT}}$ is an indexed category such that*

1. *Ind is cocomplete*
2. *For each $i \in |Ind|$, $C(i)$ is cocomplete*
3. *For each $\sigma: i \longrightarrow j \in Ind$, $C(\sigma): C(j) \longrightarrow C(i)$ has a left adjoint*

then $Flat(C)$ is cocomplete.

To be able to apply Theorem 4, we need four Lemmas: First 3.4(4) of [1]:

Lemma 5. *Let $\underline{Mod}^{\boldsymbol{MS}*}: \underline{Sign}^{\boldsymbol{MS}*} \longrightarrow \underline{\mathcal{CAT}}$ be the model functor for many sorted algebras, and let $\Sigma \in \underline{Sign}^{\boldsymbol{MS}*}$. Then $\underline{Mod}^{\boldsymbol{MS}*}\,\Sigma$ is cocomplete.* □

then 7.16 of [7] (or the very general results of [17]):

Lemma 6. *Let σ be a signature morphism in $\underline{Sign}^{\boldsymbol{MS}*}$. Then $\underline{Mod}^{\boldsymbol{MS}*}\,\sigma$ has a left adjoint, denoted by F_σ.* □

then Lemma 4.1 of [18]:

Lemma 7. *Let $\underline{T}$ be a category and let $INTO(\underline{T})$ be the category of functors $F: \underline{I} \longrightarrow \underline{T}$ into $\underline{T}$ with morphisms $(\Phi, \alpha): (F \xrightarrow{I} \underline{T}) \longrightarrow (F' \xrightarrow{I'} \underline{T})$ consisting of $\Phi: \underline{I} \longrightarrow \underline{I'}$ and $\alpha: F' \circ \Phi \longrightarrow F$. Then $INTO(\underline{T})$ is complete whenever $\underline{T}$ is cocomplete.* □

and finally Lemma 4.2 of [18]:

Lemma 8. *Let $\underline{A}$, $\underline{B}$ and $\underline{C}$ be categories and $F: \underline{A} \longrightarrow \underline{C}$, $G: \underline{B} \longrightarrow \underline{C}$ be functors. If $\underline{A}$ and $\underline{B}$ are cocomplete and F is cocontinuous, then the comma category $(F \downarrow G)$ is cocomplete.* □

With this, we can prove

Lemma 9. *Let $Proj: INTO(\underline{Sign}^{\boldsymbol{MS}*}) \longrightarrow \underline{\mathcal{CAT}}$ map a functor to its domain category. Let $\underline{1}$ be the category with one object and one morphism. Then the comma category $(\underline{1} \downarrow Proj)$ is complete.*

Proof:
$\underline{Sign}^{\mathbf{MS}}$ is shown to be cocomplete in [19]. Let Σ_* be the signature consisting of $true : *$. Then $\underline{Sign}^{\mathbf{MS}*}$ is the comma category $(\Sigma_* \downarrow \underline{Sign}^{\mathbf{MS}})$, and it is cocomplete by Lemma 8. By Lemma 7, $INTO(\underline{Sign}^{\mathbf{MS}*})$ is complete. By the dual of Lemma 8, $(\underline{1} \downarrow Proj)$ is complete. □

Theorem 10. *$\underline{Parch}$ is complete.*

Proof:

Let *Apply* be the functor indicated by the following diagram:

$$(\underline{1} \downarrow Proj) \xrightarrow{\quad Apply \quad} \underline{Sign}^{\mathbf{MS}^{*\,op}} \xrightarrow{\underline{Mod}^{\mathbf{MS}^{*}}} \underline{\mathcal{CAT}}$$

$$\begin{array}{ccc}
(\Gamma, Lang) & \longmapsto & Lang\,\Gamma \\
\Big\downarrow{\scriptstyle (\Phi,\alpha)} & & \Big\uparrow{\scriptstyle \alpha_\Gamma} \\
(\Gamma', Lang') & \longmapsto & Lang'\,\Gamma' = Lang'\,\Phi\,\Gamma
\end{array}$$

$\underline{Mod}^{\mathbf{MS}^{*}} \circ Apply\colon ((\underline{1} \downarrow Proj)^{op})^{op} \longrightarrow \underline{\mathcal{CAT}}$ is an indexed category with $Flat(\underline{Mod}^{\mathbf{MS}^{*}} \circ Apply)$ equivalent to $\underline{Parch}^{op}$.

Now $(\underline{1} \downarrow Proj)^{op}$ is cocomplete by the dual of Lemma 9. $\underline{Mod}^{\mathbf{MS}^{*}}\, Apply(\Gamma, Lang) = \underline{Mod}^{\mathbf{MS}^{*}}\, Lang\,\Gamma$ is cocomplete by Lemma 5, and $\underline{Mod}^{\mathbf{MS}^{*}}\, Apply(\Phi, \alpha) = \underline{Mod}^{\mathbf{MS}^{*}}\, \alpha_\Gamma$ has a left adjoint by Lemma 6.

Thus, by Theorem 4, $Flat(\underline{Mod}^{\mathbf{MS}^{*}} \circ Apply) \cong \underline{Parch}^{op}$ is cocomplete. $\square$

Since the proof of Theorem 10 proceeds on a rather abstract level, let us spell out some details of the construction of Theorem 4 for the case of pullbacks

$$\begin{array}{ccc}
(\underline{Sign}', Lang', \Gamma', \mathcal{G}') & \xrightarrow{(\Phi^3, \alpha^3, g^3)} & (\underline{Sign}_2, Lang_2, \Gamma_2, \mathcal{G}_2) \\
\Big\downarrow{\scriptstyle (\Phi^4, \alpha^4, g^4)} & & \Big\downarrow{\scriptstyle (\Phi^2, \alpha^2, g^2)} \\
(\underline{Sign}_1, Lang_1, \Gamma_1, \mathcal{G}_1) & \xrightarrow{(\Phi^1, \alpha^1, g^1)} & (\underline{Sign}, Lang, \Gamma, \mathcal{G})
\end{array}$$

in $\underline{Parch}$. The first step is to take a pullback in $\underline{\mathcal{CAT}}$ ...

$$\begin{array}{ccc}
\underline{Sign}' & \xrightarrow{\Phi^3} & \underline{Sign}_2 \\
\Big\downarrow{\scriptstyle \Phi^4} & & \Big\downarrow{\scriptstyle \Phi^2} \\
\underline{Sign}_1 & \xrightarrow{\Phi^1} & \underline{Sign}
\end{array} \qquad \Gamma' = \Gamma_1 \underset{\Gamma}{+} \Gamma_2$$

...then take a pushout in $\underline{Sign}^{\mathbf{MS}^{*}}$ ...

$$
\begin{tikzcd}
Lang\,\Phi^1\,\Phi^4\,\Sigma \arrow[r, "\alpha^1_{\Phi^4\,\Sigma}"] \arrow[d, "Id"'] & Lang_1\,\Phi^4\,\Sigma \arrow[dd, dotted, "\alpha^4_{\Sigma}"'] \\
Lang\Phi^2\,\Phi^3\,\Sigma \arrow[d, "\alpha^2_{\Phi^3\,\Sigma}"'] & \\
Lang_2\,\Phi^3\,\Sigma \arrow[r, dotted, "\alpha^3_{\Sigma}"] & Lang'\,\Sigma
\end{tikzcd}
$$

...and finally take a pushout in $\underline{Mod}^{\mathbf{MS}} * Lang'\,\Gamma'$:

$$
\begin{tikzcd}
F_{\alpha^4_{\Gamma'}}\,F_{\alpha^1_{\Gamma_1}}\,\mathbb{G} \arrow[r, "F_{\alpha^4_{\Gamma'}}\,g^{1\#}"] \arrow[d, "Iso"'] & F_{\alpha^4_{\Gamma'}}\,\mathbb{G}_1 \arrow[dd, dotted, "g^{4\#}"'] \\
F_{\alpha^3_{\Gamma'}}\,F_{\alpha^2_{\Gamma_2}}\,\mathbb{G} \arrow[d, "F_{\alpha^3_{\Gamma'}}\,g^{2\#}"'] & \\
F_{\alpha^3_{\Gamma'}}\,\mathbb{G}_2 \arrow[r, dotted, "g^{3\#}"] & \mathbb{G}'
\end{tikzcd}
$$

The latter pushout can be constructed as follows: Let

$$
\begin{aligned}
LangPO \;=\; & Lang'\,\Gamma' + \;\alpha^4_{\Gamma'}(Diag(\mathbb{G}_1)) + \alpha^3_{\Gamma'}(Diag(\mathbb{G}_2)) + \\
& \textbf{axioms } c^{g^1_s(a)} = c^{g^2_s(a)} : \alpha^4_{\Gamma'}(\alpha^1_{\Gamma_1}(s)) \quad \text{for } a \in \mathbb{G}_s, s \in S(Lang\,\Gamma)
\end{aligned}
$$

Then, $\mathbb{G}' = T_{LangPO}\,|_{Lang'\,\Gamma'}$. That is, we combine diagrams of $\mathbb{G}_1$ and $\mathbb{G}_2$, equate the images of $\mathbb{G}$ and interpret the result freely.

With this, we can compute the pullback

$$
\begin{tikzcd}
\mathbf{PMS} \underset{\mathbf{MS}}{+} \mathbf{MS}\,(=) \arrow[r, dotted, "\nu^{=}"] \arrow[d, dotted, "\nu^{\mathrm{P}}"'] & \mathbf{PMS} \arrow[d, "\mu^{\mathrm{P}}"] \\
\mathbf{MS}\,(=) \arrow[r, "\mu^{=}"] & \mathbf{MS}
\end{tikzcd}
$$

Let $\mathbf{PMS} \underset{\mathbf{MS}}{+} \mathbf{MS}\,(=)$ be $(\underline{Sign}, Lang, \Gamma, \mathbb{G})$. Then

- $\underline{Sign} = \underline{Sign}^{\mathbf{PMS}}$
- $\Gamma = \Gamma^{\mathbf{PMS}}$

- $Lang\,\Sigma = Lang^{\mathbf{PMS}}\,\Sigma \cup Lang^{\mathbf{MS}\,(=)}\,\Sigma$
- $\mathbb{G}_{Term(X,s)} = [Env(X) \multimap s]$
- $[Env(X) \longrightarrow \mathbf{Bool}\,] \subset \mathbb{G}_{Formula(X)}$ is a proper subset, because $=_{\mathbb{G}}\ (f_1, f_2)$ generates new data for partial maps f_1, f_2
- $\mathbf{Bool}\ \subset \mathbb{G}_*$ is a proper subset, because $bind_{\mathbb{G}}(=_{\mathbb{G}}\ (f_1, f_2))$ generates new data for partial maps f_1, f_2

We are now one step further: there is some interaction of concepts, we have equations over partial operation symbols in $\mathbf{PMS} \underset{\mathbf{MS}}{+} \mathbf{MS}\,(=)$! But their semantical interpretation is a bit funny. The problem is that there is no hint how to interpret equations over partial operation symbols. Thus the pullback construction generates a "free" interaction of concepts, yielding a "junk" truth value for each possible equation involving partial term evaluation. Though it happens that $I(\mathbf{PMS} \underset{\mathbf{MS}}{+} \mathbf{MS}\,(=))$ is $\mathbf{PMS}\,(\overset{\mathrm{e}}{=})$, the institution $\mathbf{PMS}$ plus existence equations (see below), the presence of so many truth values causes problems when using $\mathbf{PMS} \underset{\mathbf{MS}}{+} \mathbf{MS}\,(=)$ for further combinations. For example, adding negation or implication does not behave in the expected way: a formula containing an undefined term always is non-true (i. e. false, under the functor I), so $t \overset{\mathrm{e}}{=} t$ and $\neg t \overset{\mathrm{e}}{=} t$ would both be false in case of undefinedness of t.

The solution to this problem: We have to specify how equality is interpreted on partial term evaluation! This can be done by defining a congruence relation on $\mathbb{G}$. For example, define a congruence $\overset{e}{\equiv}$ on $\mathbb{G}$ generated by

$$=_{\mathbb{G}}\ (f_1, f_2) \overset{e}{\equiv} \quad \lambda v. \begin{cases} true, & \text{if } f_1(v), f_2(v) \text{ are defined and equal} \\ false, & \text{otherwise} \end{cases}$$

Then the quotient $\mathbb{G} \xrightarrow{_/\overset{e}{\equiv}} \mathbb{G}/\overset{e}{\equiv}$ leads to a parchment morphism

$$\mathbf{PMS}\,(\overset{\mathrm{e}}{=}) \xrightarrow{(Id, Id, _/\overset{e}{\equiv})} \mathbf{PMS} \underset{\mathbf{MS}}{+} \mathbf{MS}\,(=)$$

The resulting quotient reduces the junk, and preservation of the truth value space is restored. And $\mathbf{PMS}\,(\overset{\mathrm{e}}{=})$ is the parchment of many-sorted partial algebras with existence equations [4]! Note that taking a quotient semantical algebra is a special case of limit of parchments.

Similarly, we can define congruences $\overset{w}{\equiv}$ and $\overset{s}{\equiv}$ leading to parchments $\mathbf{PMS}\,(\overset{\mathrm{W}}{=})$ (weak equations), and $\mathbf{PMS}\,(\overset{\mathrm{S}}{=})$ (strong equations, [4, 12]).

We also can get a three-valued logic.

Let $t_1 \overset{3}{=} t_2$ be $\begin{cases} true, & \text{if } t_1, t_2 \text{ are defined and equal} \\ false, & \text{if } t_1, t_2 \text{ are defined and not equal} \\ -, & \text{otherwise} \end{cases}$

Identify a three-valued map $f\colon Env(X) \longrightarrow \{\,true; false; -\,\}$ with $=_{\mathbb{G}}\ (f_1, f_2)$ for some f_1, f_2 satisfying $f(v) = f_1(v) \overset{3}{=} f_2(v)$ and let $\overset{3}{\equiv}$ be the congruence relation

generated by $=_{\not\subseteq}(f_1, f_2) \overset{3}{\equiv} \lambda v.f_1(v) \overset{3}{=} f_2(v)$. Then

$$\mathbf{PMS}\,(\overset{3}{=}) \xrightarrow{(Id, Id, _/\overset{3}{\equiv})} \mathbf{PMS} \underset{\mathbf{MS}}{+} \mathbf{MS}\,(=)$$

yields partial algebras with a three valued logic.

Adding relations can be done using $\mathbf{MSR}\,(R)$ which is $\mathbf{MS}$ augmented by relation symbols in the signatures, relations in the models and applications of relation symbols to terms in the sentences.

$$\begin{tikzcd}
\mathbf{PMS}\,(\overset{e}{=} R) \arrow[r, "_/\overset{R}{\equiv}"] & \mathbf{PMS}\,(\overset{e}{=}) \underset{\mathbf{MS}}{+} \mathbf{MSR}\,(R) \arrow[r, dotted] \arrow[d, dotted] & \mathbf{PMS}\,(\overset{e}{=}) \arrow[d] \\
 & \mathbf{MSR}\,(R) \arrow[r] & \mathbf{MS}
\end{tikzcd}$$

where $\overset{R}{\equiv}$ is the congruence relation generated by

$$R(X)_{\not\subseteq}(f_1, \ldots, f_n) \overset{R}{\equiv} \lambda v. \begin{cases} true, & \text{if } (f_1(v), \ldots, f_n(v)) \text{ def. and } \in R \\ false, & \text{otherwise} \end{cases}$$

Similarly, we get $\mathbf{PMS}\,(\overset{3}{=} R)$.

Adding partiality to first order logic can be done with the parchment $\mathbf{MSFOL}\ (= R)$ (total first order logic), where universal quantification is modeled by an overloaded operation $\forall(Y)\colon Formula(X \cup Y) \longrightarrow Formula(X)$ (cf. [15]):

$$\begin{tikzcd}
\mathbf{PMSFOL}\ (\overset{e}{=} R) \arrow[r, dotted] \arrow[d, dotted] & \mathbf{PMS}\,(\overset{e}{=} R) \arrow[d] \\
\mathbf{MSFOL}\ (= R) \arrow[r] & \mathbf{MS}
\end{tikzcd}$$

In the three valued case,

$$\begin{tikzcd}
\mathbf{PMSFOL}\ (\overset{3}{=} R) \arrow[r, "_/\overset{FOL3}{\equiv}"] & \mathbf{PMS}\,(\overset{3}{=} R) \underset{\mathbf{MS}}{+} \mathbf{MSFOL}\ (= R) \arrow[r, dotted] \arrow[d, dotted] & \mathbf{PMS}\,(\overset{3}{=} R) \arrow[d] \\
 & \mathbf{MSFOL}\ (= R) \arrow[r] & \mathbf{MS}
\end{tikzcd}$$

we have to specify a congruence $\overset{FOL3}{\equiv}$ defining three-valued logical connectives and quantifiers. For example, we could use Kleene's definitions [12] here, which are also used in VDM [11].

Due to lack of space, we cannot consider parchments for institutions which are built using not only partial term evaluation, but also partial *variable valuations* (see [14, 2]).

With help of the examples, we are now ready to prove

Proposition 11. *$\underline{TruthParch}$ is* not *complete!*

Proof:
Suppose that $(P, \rho^{=}, \rho^{\mathrm{P}})$ is a pullback of $(\mu^{\mathrm{P}}, \mu^{=})$ in $\underline{TruthParch}$. By the universal property of $\mathbf{PMS} \underset{\mathbf{MS}}{+} \mathbf{MS}(=)$, there exists a unique $\mu: P \longrightarrow \mathbf{PMS} \underset{\mathbf{MS}}{+} \mathbf{MS}(=)$ in $\underline{Parch}$ with $\nu^{=} \circ \mu = \rho^{=}$ and $\nu^{\mathrm{P}} \circ \mu = \rho^{\mathrm{P}}$. For $x \in \{e, s, w\}$, we have constructed $(Id, Id, g^x): \mathbf{PMS}(\overset{\mathrm{X}}{=}) \longrightarrow \mathbf{PMS} \underset{\mathbf{MS}}{+} \mathbf{MS}(=)$ with $g^x = (_/\overset{x}{\equiv})$ above. Now $\nu^{=} \circ (Id, Id, g^x)$ and $\nu^{\mathrm{P}} \circ (Id, Id, g^x)$ both preserve truth. Thus the outer diagram commutes in $\underline{TruthParch}$. By the universal property of P, there is a $\nu^x: \mathbf{PMS}(\overset{\mathrm{X}}{=}) \longrightarrow P$. By the universal property of $\mathbf{PMS} \underset{\mathbf{MS}}{+} \mathbf{MS}(=)$ it follows that $(Id, Id, g^x) = \mu \circ \nu^x$. Now taking $\mu = (\Phi, \alpha, g)$, $\nu^x = (\Phi^x, \alpha^x, h^x)$, $\mathcal{G}^{+} :=$

$$\begin{tikzcd}
\mathbf{PMS}(\overset{\mathrm{X}}{=}) \arrow[dr, "\nu^x"] \arrow[drr, "(Id,Id,g^x)"] \arrow[drrr, "(Id,Id,g^x)\circ\nu^{=}"] \arrow[ddr, "(Id,Id,g^x)\circ\nu^{\mathrm{P}}"'] & & & \\
& P \arrow[rr, "\rho^{=}"] \arrow[dr, "\mu"] \arrow[dd, "\rho^{\mathrm{P}}"] & & \mathbf{PMS} \arrow[dd, "\mu^{\mathrm{P}}"] \\
& & \mathbf{PMS} \underset{\mathbf{MS}}{+} \mathbf{MS}(=) \arrow[ur, "\nu^{=}"] \arrow[dl, "\nu^{\mathrm{P}}"] & \\
& \mathbf{MS}(=) \arrow[rr, "\mu^{=}"] & & \mathbf{MS}
\end{tikzcd}$$

$\mathcal{G}^{\mathbf{PMS} \underset{\mathbf{MS}}{+} \mathbf{MS}(=)}$ and $\mathcal{G}^{x} := \mathcal{G}^{\mathbf{PMS}(\overset{\mathrm{X}}{=})}$, let $f \in [Env(X) \multimap s]$ be the totally undefined function. Let $a = g_*(bind_{\mathcal{G}^{+}}(=_{\mathcal{G}^{+}}(f, f)))$. Then $h^x_*(a) =$
$= h^x_*(g_*(bind_{\mathcal{G}^{+}}(=_{\mathcal{G}^{+}}(f, f))))$
$= g^x(bind_{\mathcal{G}^{+}}(=_{\mathcal{G}^{+}}(f, f))) = bind_{\mathcal{G}^{x}}(=_{\mathcal{G}^{x}}(g^x_{Term(X,s)}(f), g^x_{Term(X,s)}(f)))$
$= bind_{\mathcal{G}^{x}}(=_{\mathcal{G}^{x}}(f, f)) = \begin{cases} true, & \text{if } x \in \{s, w\} \\ false, & \text{if } x = e \end{cases}$

Thus, regardless if $a = true$ or $a = false$, h^x cannot preserve truth for all $x \in \{ e, s, w \}$, and therefore, $(P, \rho^{=}, \rho^{\mathrm{P}})$ is not a pullback in $\underline{TruthParch}$. $\square$

6 Conclusion and Comparsion with Related Work

We have argued that the category of institutions and institution morphisms only allows combinations of institutions *without true interaction*. But the notion of parchment introduced by Goguen and Burstall [8] can be equipped with a notion of parchment morphism. The category of parchments turns out to be complete, and parchments combine via limits *with* true interaction.

But it can happen that this combination introduces new, freely generated truth-values. This indicates the further need for specification of semantical evaluation in the combination, which cannot be automatically inferred from the components in all cases. Instead, the semantical interaction has to be specified explicitly using a congruence relation on the semantical algebra of the parchment, generated by a small number of axioms. These axioms specify how combined things behave semantically.

Thus design decisions are not scattered all over the definitions, but are made at a clearly singled-out point. This can be the basis for a semi-automatic process of combination of institutions.

We have illustrated these ideas by introducing partiality into different institutions ranging from equational to first-order logic. Further examples of combinations will appear in a longer version of this paper.

Hendrik Hilberdink's work on generating syntax through ends of purely model-theoretic functors [10] has some formal similarities to parchments. In his equation

$$T_{\Sigma}(X)_s = \int_m Set(Set^{T(S)}(X, \mathcal{E}_{(S,\Sigma)}(m)), \mathcal{E}_{(S,\Sigma)}(m)(s))$$

$Set^{T(S)}(X, \mathcal{E}_{(S,\Sigma)}(m))$ corresponds to our $Env(X)$ and $Set(Set^{T(S)}(X, \mathcal{E}_{(S,\Sigma)}(m)), \mathcal{E}_{(S,\Sigma)}(m)(s))$ corresponds to our $[Env(X) \longrightarrow s]$ (viewed as part of the $Lang\, m$-reduct of $\mathfrak{C}$). But there are two main differences:

1. While the end $\int_m \ldots$ *generates* term syntax and term evaluation from model categories, parchments allow to *specify* term syntax (via the $Lang$-functor) and term evaluation (via the semantical algebra $\mathfrak{C}$) in a modular, "global" way leading to the possibility of well-structured combinations. *Generating* term syntax out of model categories does not free us from *specifying* it: In many situations, we cannot cope with the realm of all possible operations consistent with model morphisms, but we have to single out particular ones. It is not clear to me what this difference means for the combination process, since it is even not clear to me how Hilberdink's approach allows combinations and if there is a completeness theorem. But perhaps both approaches are complementary.
2. For institutions of partial algebras, we have to deal with partial variable valuations and term evaluation. In Hilberdink's approach, therefore Set has to re-

placed by a category of partial maps, and it is not clear if the results generalize for this case.

Acknowledgement. I wish to thank Andrzej Tarlecki for fruitful discussions.

References

[1] J. Adámek, J. Rosický. *Locally Presentable and Accessible Categories.* Cambridge University Press, 1994.

[2] E. Astesiano, M. Cerioli. Non-strict don't care algebras and specifications. *Mathematical Structures in Comupter Science*, 1996. To appear.

[3] M. Bidoit, H.-J. Kreowski, P. Lescannes, F. Orejas, D. Sannella. *Algebraic System Specification and Development. A Survey and Annotated Bibliography, Lecture Notes in Computer Science* **501**. Springer Verlag, 1991.

[4] P. Burmeister. *A model theoretic approach to partial algebras.* Akademie Verlag, Berlin, 1986.

[5] R. M. Burstall, J. A. Goguen. Putting theories together to make specifications. In *Proceedings of the 5th International Joint Conference on Artificial Intelligence*, 1045–1058. Cambridge, 1977.

[6] M. Cerioli. *Relationships between Logical Formalisms.* PhD thesis, TD-4/93, Università di Pisa-Genova-Udine, 1993.

[7] H. Ehrig, B. Mahr. *Fundamentals of Algebraic Specification 1.* Springer Verlag, Heidelberg, 1985.

[8] J. A. Goguen, R. M. Burstall. A study in the foundations of programming methodology: Specifications, institutions, charters and parchments. In D. Pitt et al., ed., *Category Theory and Computer Programming, LNCS* **240**, 313–333. Springer Verlag, 1985.

[9] J. A. Goguen, R. M. Burstall. Institutions: Abstract model theory for specification and programming. *Journal of the Association for Computing Machinery* **39**, 95–146, 1992. Predecessor in: LNCS 164(1984):221–256.

[10] H. Hilberdink. The end of syntax. The end of algebra. Manuscripts, St. Peter's College, Oxford, 1995.

[11] C. B. Jones. *Systematic Software Development Using VDM.* Prentice Hall, 1990.

[12] S.C. Kleene. *Introduction to Metamathematics.* North Holland, 1952.

[13] T. Mossakowski. Equivalences among various logical frameworks of partial algebras. In H. Kleine Büning et al., ed., *Computer Science Logic. 9th Workshop, CSL'95. Paderborn, Germany, September 1995.* Springer LNCS, 1996. To appear.

[14] D. S. Scott. Identity and existence in intuitionistic logic. In M.P. Fourman, C.J. Mulvey, D.S. Scott, eds., *Application of Sheaves, Lecture Notes in Mathematics* **753**, 660–696. Springer Verlag, 1979.

[15] P. Stefaneas. The first order parchment. Report PRG-TR-16-92, Oxford University Computing Laboratory, 1992.

[16] A. Tarlecki. Working with multiple logical systems. Unpublished manuscript. See `http://zls.mimuw.edu.pl/~tarlecki/book/index.html`.

[17] A. Tarlecki. On the existence of free models in abstract algebraic institutions. *Theoretical Computer Science* **37**, 269–304, 1985.

[18] A. Tarlecki. Bits and pieces of the theory of institutions. In D. Pitt, S. Abramsky, A. Poigné, D. Rydeheard, eds., *Proc. Intl. Workshop on Category Theory and Computer Programming, Guildford 1985, LNCS* **240**, 334–363. Springer-Verlag, 1986.

[19] A. Tarlecki, R. M. Burstall, J. A. Goguen. Some fundamentals algebraic tools for the semantics of computation. Part 3: Indexed categories. *Theoretical Computer Science* **91**, 239–264, 1991.

This article was processed using the LaTeX macro package with LLNCS style