Finite Limits and Anti-unification in Substitution Categories

Wolfram Kahl

Finite Limits and Anti-Unification in Substitution Categories

Wolfram Kahl

McMaster University, Canada
kahl@cas.mcmaster.ca

Abstract. It is well-known that coequalisers and pushouts of substitutions correspond to solutions of unification problems, and therefore do not always exist. But how about equalisers and pullbacks? If the literature contains the answers, they are well-hidden.
We provide explicit details and proofs for these constructions in categories with substitutions as morphisms, and in particular work out the details of categorial products for which the universal arrow construction turns out to correspond exactly to anti-unification.

1 Introduction

Substitutions occur in the formal study of syntactic systems, and are mappings from "variables" to "terms" (or "expressions"). Terms may contain variables, and "application" of substitution to terms produces terms again. Rydeheard and Stell (1987) introduced (based on closely related ideas by Lawvere (1963)) a categorial treatment of substitutions, where objects are sets considered as sets of variables, and morphisms from V_1 to V_2 are substitutions that map each element of V_1 to a term containing only variables from V_2. They say: "In this case variables are *localized*." This is in contrast with most of the conventional literature on substitutions, were mostly a single global set of variables is assumed. For example, Eder (1985) investigates a "more general than" order on idempotent substitutions.

Rydeheard and Stell (1987) used their category-theoretic formulation in particular for the construction of a unification algorithm, where unification is defined as coequaliser of substitutions. Since unification problems are not always solvable, coequalisers do not always exist in substitution categories, but since unification problems are important, coequalisers (and pushouts which correspond to unification problems with disjoint variable sets) have received much attention in the literature. However, I have not been able to find explicit statements about equalisers, pullbacks, or products in substitution categories. Since substitution categories are a concrete instance of Kleisli categories, Szigeti's (1983) study on limits and colimits in Kleisli categories using adjunctions is related, but, in his own words: "Mention must be made that these results are powerless in concrete instances." Hosseini and Qasemi Nezhad (2016) tackle the problem of existence of equalisers in Kleisli categories for a number of more concrete monads, but still without covering the term monad.

One motivation for studying finite limits in substitution categories comes from the fact that substitutions can be components of homomorphisms of attributed graphs (Kahl, 2014, 2015), and the properties of the resulting categories are key to the applicability of categorial approaches to graph transformation (Ehrig et al., 2006) in the spirit of the "high level replacement (HLR) systems" introduced by Ehrig et al. (1991). In particular for the "adhesive categories" introduced by Lack and Sobociński (2004, 2005) as a useful abstraction for frequently-studied HLR properties, existence of pullbacks becomes a key property. Another important question in that context is whether pushouts along all monomorphisms exist, or whether at least useful classes of monomorphisms can be defined along which pushouts exist.

In this paper, we provide concrete constructions and detailed proofs for equalisers, products, and pullbacks in substitution categories. In summary, the instances of basic category-theoretic concepts for the substitution category for a fixed signature take the following shapes, where well-known general facts are included in parentheses for completeness:

1. Epimorphisms are those substitutions where all target variables occur in the image.
2. Monomorphisms are those substitutions for which the image of any variable does not result from applying that substitution to any term different from that variable, as previously shown in (Kahl, 2015).
3. Equalisers of two substitutions in the substitution category always exist, and are the variable subset injections for the subset on which the two substitutions have the same images — these are the equalisers of the two substitutions in *Set*, but the proof of the universal property is substitution-specific.
4. Regular monomorphisms are precisely the injective variable renamings.
5. (Coequalisers are most general unifiers, and do not always exist.)
6. Products have as objects the sets of all pairs of not-equally-headed terms; the construction of the universal morphism is essentially anti-unification. Products of finite (variable) sets are in general infinite.
7. (Coproducts are inherited from *Set* as for every monad.)
8. Pullbacks always exist (since they can be obtained from products and equalisers). Pullbacks of substitutions between finite (variable) sets have finite pullback objects.
9. (Pushouts correspond to most general unifiers of substitutions with disjoint ranges, and do not always exist.) Pushouts along regular monomorphisms do exist, and regular monomorphisms are stable under pushout.

We are working on a mechanisation of this theory in the dependently-typed programming language and proof assistant Agda (Norell, 2007); at the time of writing, proofs for items 2, 3, and 6 are already complete.

Overview

After some background in sections 2 and 3 about monads and the term monad, we will work through the list above, except for the items in parentheses, which are well-known. We devote Sect. 4 to epimorphisms, Sect. 5 to monomorphisms, Sect. 6 to equalisers and regular monomorphisms, Sect. 7 to products, Sect. 8 to pullbacks, and Sect. 9 to pushouts.

2 Notation and Background: Categories and Monads

We assume familiarity with the basics of category theory; for notation, we write "$f : A \to B$" to declare that morphism f goes from object A to object B, or we may refer to the source and target objects of f as $\mathsf{src}\, f = A$ and $\mathsf{trg}\, f = B$. We use ";" as the associative binary *forward composition* operator that maps two morphisms $f : A \to B$ and $g : B \to C$ to $(f ; g) : A \to C$. The identity morphism for object A is written $\mathbb{I}_A$. We assign ";" higher priority than other binary operators, and assign unary operators higher priority than all binary operators.

The category of sets and functions is denoted by Set. For a function $f : A \to B$ and an element $x : A$, we normally denote function application by juxtaposition "$f\ a$". (We may need to add parentheses to either subexpression, "$(f)\ a$" or "$f(a)$".) This juxtaposition has higher precedence than all visible operators.

A *functor* $\mathcal{F}$ from one category to another maps objects to objects and morphisms to morphisms respecting the structure that src, trg, $\mathbb{I}$, and composition constitute; we denote functor application by juxtaposition both for objects, $\mathcal{F}\ A$, and for morphisms, $\mathcal{F}\ f$.

A *monad* on a category $\mathcal{C}$ consists is a functor $\mathcal{M} : \mathcal{C} \to \mathcal{C}$ for which there are two natural transformations ("polymorphic morphisms") $\mathsf{return}_A : A \to \mathcal{M}\ A$ and $\mathsf{join}_A : \mathcal{M}\ (\mathcal{M}\ A) \to \mathcal{M}\ A$ satisfying $\mathsf{return}_{\mathcal{M}\ A} ; \mathsf{join}_A = \mathbb{I}$ and $\mathcal{M}\ \mathsf{return}_A ; \mathsf{join}_A = \mathbb{I}$ and $\mathcal{M}\ \mathsf{join}_A ; \mathsf{join}_A = \mathsf{join}_{\mathcal{M}\ A} ; \mathsf{join}_A$. Important monads are the List monad, and the term monad $\mathcal{T}_\Sigma$ for any (algebraic) signature Σ. For the former, $\mathsf{join}_{\mathsf{List},A} : \mathsf{List}\ (\mathsf{List}\ A) \to \mathsf{List}\ A$ is the function that flattens (or concatenates) lists of lists.

Each monad $\mathcal{M}$ on $\mathcal{C}$ induces the so-called *Kleisli category* $\mathbb{K}_\mathcal{M}$ that has the same objects as $\mathcal{C}$, but $\mathcal{C}$-morphisms $A \to \mathcal{M}\ B$ as morphisms from A to B. Kleisli composition of $f : A \to \mathcal{M}\ B$ with $g : B \to \mathcal{M}\ C$ will be written $f \fatsemi g$; this is defined by $f \fatsemi g = f ; (\mathcal{M}\ g) ; \mathsf{join}_C$. In the Kleisli category, return is the identity for Kleisli composition, that is, for each Kleisli morphism f from A to B we have $\mathsf{return}_A \fatsemi f = f$ and $f = f \fatsemi \mathsf{return}_B$.

Note the different symbols ";" for composition in the base category and "$\fatsemi$" for composition in the Kleisli category! Both will occur frequently, and also together. They satisfy, among others, the equations $(f ; g) \fatsemi h = f ; (g \fatsemi h)$ and $f \fatsemi (g ; \mathsf{return}) = f ; \mathcal{M}\ g$ and $(f \fatsemi g) ; \mathcal{M}\ h = f \fatsemi (g ; \mathcal{M}\ h)$.

3 Substitution Categories as Kleisli Categories of Term Monads

Let $\mathcal{T}_\Sigma$ denote the term functor for signature Σ, that is, $\mathcal{T}_\Sigma\ X$ is the set of Σ-terms with elements of set X as variables. As usual, $\mathcal{T}_\Sigma\ X$ is defined inductively by the following:

- Each variable $v : X$ is a term, that is, $v \in \mathcal{T}_\Sigma\ X$.
- If $t_1, \ldots, t_n \in \mathcal{T}_\Sigma\ X$ are n terms, and f is an n-ary function symbol provided by Σ, then the resulting function symbol application is a term, too: $f(t_1, \ldots, t_n) \in \mathcal{T}_\Sigma\ X$.

$\mathcal{T}_\Sigma$ is an endofunctor on the category *Set*, and naturally extends to a monad, the *term monad*. Its "join" natural transformation, $\mathsf{join}_{\mathcal{T}_\Sigma}$, produces for each set A (of variables) the function $\mathsf{join}_{\mathcal{T}_\Sigma, A} : \mathcal{T}_\Sigma\ (\mathcal{T}_\Sigma\ A) \to \mathcal{T}_\Sigma\ A$ which "flattens" nested terms over variables in A (that is, terms over $\mathcal{T}_\Sigma\ A$ as their set of variables). The "return" natural transformation of the term monad, $\mathsf{return}_{\mathcal{T}_\Sigma}$, maps each variable v to the term v. (We will omit the subscript $\mathcal{T}_\Sigma$ for join and return.)

The Kleisli category of the term monad $\mathcal{T}_\Sigma$ will be denoted $\mathbb{T}_\Sigma$; its morphisms from set X to set Y are substitutions, that is, functions $X \to \mathcal{T}_\Sigma\ Y$, that is, the sets X and Y are "interpreted" as sets of *variables*.

The definition of Kleisli composition instantiated for the term monad $\mathcal{T}_\Sigma$ takes two arbitrary substitutions $F : X \to \mathcal{T}_\Sigma\ Y$ and $G : Y \to \mathcal{T}_\Sigma\ Z$ to their composed substitution

$$F \fatsemi G \quad = \quad F ; \mathcal{T}_\Sigma\ G ; \mathsf{join}_Z \ .$$

Conventionally, this would be described via "application" of substitutions to terms: We write $F \triangleright t$ for the application of substitution $F : X \to \mathcal{T}_\Sigma\ Y$ to term $t : \mathcal{T}_\Sigma\ X$, and the substitution composition $F \fatsemi G$ can alternatively be defined by

$$(F \fatsemi G)\ v \quad = \quad G \triangleright (F\ v) \ , \qquad \text{for all } v : X.$$

When starting from the monadic setting, application of substitutions can be defined as follows:

$$F \triangleright t = ((\mathcal{T}_\Sigma\ F) ; \mathsf{join}_Y)(t)$$

For most signatures Σ and most variable sets V, the set of terms $\mathcal{T}_\Sigma\ V$ is infinite, but there are a few exceptions, which are important to keep in mind:

- The set $\mathcal{T}_\Sigma\ \varnothing$ of ground terms (i.e., terms without variables) is empty iff Σ has no constant symbols (that is, no zero-ary function symbols).
- The set $\mathcal{T}_\Sigma\ \varnothing$ of ground terms has exactly n elements for $n > 0$ iff Σ has no function symbols with arity at least one, and exactly n constant symbols.
- For a one-element variable set $\mathbb{1}$, the set $\mathcal{T}_\Sigma\ \mathbb{1}$ has at most one element iff Σ has no symbols at all.

We shall need the following definitions:

Definition 3.1. For a substitution $\sigma : V_1 \to \mathcal{T}_\Sigma\ V_2$, its *range* $\mathsf{ran}\ \sigma : \mathbb{P}\ V_2$ is the set of all variables occurring in $\sigma\ v$ for some $v : V_1$. □

Definition 3.2. A *position* is a finite sequence of positive natural numbers, with ϵ denoting the empty sequence and $k.p$ denoting the sequence with first element k and tail sequence p.

Positions are used to select subterms:

$$f(t_1, \ldots, t_n)|_{k.p} = t_k|_p \qquad \text{if } f \text{ has arity } n \text{, and } 1 \leq k \leq n; \qquad t|_\epsilon = t \ . \qquad \square$$

4 Epimorphisms in Substitution Categories

For every monad over category $\mathcal{C}$ we have that, if f is epi in $\mathcal{C}$, then $f \mathbin{;} \mathsf{return}$ is epi in the Kleisli category. Substitutions of shape $f \mathbin{;} \mathsf{return}$ only map to variables. However, not all epis in $\mathbb{T}_\Sigma$ are of this shape:

Theorem 4.1. A substitution $\sigma : V_1 \to \mathcal{T}_\Sigma\ V_2$ is epi in $\mathbb{T}_\Sigma$ iff all variables in V_2 occur in the range of σ, that is, $\mathsf{ran}\ \sigma = V_2$.

Proof. Recall that σ is epi iff for all $\tau_1, \tau_2 : V_2 \to \mathcal{T}_\Sigma\ V_3$ we have that $\sigma \mathbin{⨾} \tau_1 = \sigma \mathbin{⨾} \tau_2$ implies $\tau_1 = \tau_2$.

Assume σ is epi. Choose $V_3 = \{x, y\}$, and define τ_1 and τ_2 as follows:

$$\tau_1(v) = x \qquad \text{and} \qquad \tau_2(v) = \begin{cases} x & \text{if } v \in \mathsf{ran}\ \sigma \\ y & \text{if } v \notin \mathsf{ran}\ \sigma \end{cases}$$

Then $\sigma \mathbin{⨾} \tau_1 = \sigma \mathbin{⨾} \tau_2$, and since σ is epi, also $\tau_1 = \tau_2$, which implies that $\mathsf{ran}\ \sigma = V_2$.

Conversely, if $\mathsf{ran}\ \sigma = V_2$, and any V_3 and any $\tau_1, \tau_2 : V_2 \to \mathcal{T}_\Sigma\ V_3$ with $\sigma \mathbin{⨾} \tau_1 = \sigma \mathbin{⨾} \tau_2$ are given, then for each variable v in V_2, there is at least one $u \in V_1$ and position p in the term $\sigma\ u$ such that $(\sigma u)|_p = v$; then

$$\tau_1\ v = \tau_1\ ((\sigma u)|_p) = (\tau_1 \triangleright (\sigma u))|_p = (\tau_2 \triangleright (\sigma u))|_p = \tau_2\ ((\sigma u)|_p) = \tau_2\ v \quad ,$$

and therefore $\tau_1 = \tau_2$. □

5 Monomorphisms in Substitution Categories

A first version of the following analysis of monomorphisms in substitution categories appeared in the appendix of (Kahl, 2015).

In any monad, if the "return" natural transformation produces monomorphisms (which it does for $\mathcal{T}_\Sigma$), then monomorphisms in the Kleisli category of this monad are also monomorphisms in the underlying category. Monomorphisms F of the underlying category that are preserved by the monad functor give rise to monomorphisms $F \mathbin{;} \mathsf{return}$ in the Kleisli category.

The term functor preserves all monomorphisms: An injective variable mapping $F : V_1 \to V_2$ gives rise to an injective term mapping $\mathcal{T}_\Sigma\ F : \mathcal{T}_\Sigma\ V_1 \to \mathcal{T}_\Sigma\ V_2$ that only renames variables. The resulting substitution $F \mathbin{;} \mathsf{return} : V_1 \to \mathcal{T}_\Sigma\ V_2$ is an injective variable renaming, which is therefore a mono in the category of substitutions, too — this also can easily be seen directly.

However, not all monos in $\mathbb{T}_\Sigma$ are of this simple shape.

For σ, being a monomorphism in the category of substitutions exactly means that *substitution application* of σ does not unify any two different terms. That is, σ is a monomorphism in the category of substitutions iff for any two terms t_1 and t_2 we have

$$\sigma \triangleright t_1 = \sigma \triangleright t_2 \qquad \text{implies} \qquad t_1 = t_2 \quad .$$

Due to the quantification over arbitrary terms t_1 and t_2, this condition is not easy to check directly.

It is easy to see that monomorphisms in the category of substitutions, as a consequence of this condition, cannot map any variables to ground terms. However, this by itself does not constitute a characterisation of monomorphisms.

Fortunately a much simpler condition is (necessary and) sufficient: We can show that monomorphisms in the $\mathbb{T}_\Sigma$ are those substitutions that do not identify variables with different terms:

Theorem 5.1. A substitution $\sigma : V_1 \to \mathcal{T}_\Sigma\ V_2$ is a monomorphism in the category of substitutions iff for every variable $v : V_1$ and every term $t : \mathcal{T}_\Sigma\ V_1$, we have:

$$\sigma \triangleright t = \sigma\ v \qquad \text{implies} \qquad t = v \ .$$

Proof. "⇒" follows directly by applying the monomorphism property to the two terms v and t.
"⇐": Assume that σ satisfies the given condition. To show that σ is a monomorphism in the category of substitutions it suffices to show that for any two terms $t, u : \mathcal{T}_\Sigma\ V_1$ with $\sigma \triangleright t = \sigma \triangleright u$, we have $t = u$. We show this by induction on the structure of t and u:

- If $t = v$ is a variable, then $\sigma\ v = \sigma \triangleright t = \sigma \triangleright u$, from which the given property yields $v = u$.
- The case where u is a variable is analogous.
- If $t = f(t_1, \ldots, t_n)$ and u is not a variable, then $\sigma \triangleright t = \sigma \triangleright u$ implies that there are terms $u_1, \ldots, u_n$ such that $u = f(u_1, \ldots, u_n)$ and $\sigma \triangleright t_i = \sigma \triangleright u_i$, from which the induction hypothesis yields $t_i = u_i$ for all i, implying $t = u$. □

For finite substitutions, the condition of Theorem 5.1 directly translates into a decision procedure that for each variable $v : V_1$ checks whether for any *different* variable $u : V_1$, its image $\sigma\ u$ occurs as a subterm in $\sigma\ v$. Due to the observation above, this can be further sped up for the negative case by first checking whether $\sigma\ v$ is ground, in which case σ cannot be a monomorphism.

6 Equalisers

In any category $\mathcal{C}$, an equaliser of two parallel morphisms $f, g : A \to B$ is an object S together with a morphism $\zeta : S \to A$ such that $\zeta ; f = \zeta ; g$, and for every other candidate morphism h with $h ; f = h ; g$ there us a unique morphism u such that $h = u ; \zeta$. An equaliser morphism ζ is always mono.

In *Set*, an equaliser for two functions f and g is a subobject monomorphism selecting the subset of A on which f and g coincide.

Kleisli categories do not automatically inherit equalisers from the underlying category.

For two substitutions $\sigma, \tau : V_1 \to \mathcal{T}_\Sigma\ V_2$, let V_0 together with $\zeta : V_0 \to V_1$ be their equaliser in *Set*. We now show that V_0 together with $\zeta ; \mathsf{return}_{V_1}$ is an equaliser for σ and τ in $\mathbb{T}_\Sigma$.

Commutativity follows from the monad laws:

$$(\zeta \mathbin{;} \mathsf{return}_{V_1}) \fatsemi \sigma = \zeta \mathbin{;} \sigma = \zeta \mathbin{;} \tau = (\zeta \mathbin{;} \mathsf{return}_{V_1}) \fatsemi \tau$$

For the universal property, we need to resort to substitution-specific reasoning. Assume a substitution $h : Z \to V_1$ with $h \fatsemi \sigma = h \fatsemi \tau$.

$$\begin{tikzcd}
V_0 \arrow[r, "\zeta \mathbin{;} \mathsf{return}_{V_1}"] & V_1 \arrow[rr, bend left, "\sigma"] \arrow[rr, bend right, "\tau"'] & & V_2 \\
Z \arrow[u, dotted, "!u"] \arrow[ur, "h"] & & &
\end{tikzcd}$$

If $\mathsf{ran}\ h$ is empty, then we can define $u\ z = h\ z$ since the latter is a closed term, and since $\mathsf{ran}\ u$ is then also empty, we obtain $u \fatsemi (\zeta \mathbin{;} \mathsf{return}_{V_1}) = u \mathbin{;} \mathcal{T}_\Sigma\ \zeta = h$.

If $\mathsf{ran}\ h$ is non-empty, let $z : Z$ be a variable, and p a position such that $(h\ z)|_P = v$ for some variable $v : V_1$. Then

$$\begin{aligned}
\sigma\ v &= \sigma\ ((h\ z)|_P) = (\sigma \triangleright h\ z)|_P = ((h \fatsemi \sigma)\ z)|_P \\
&= ((h \fatsemi \tau)\ z)|_P = (\tau \triangleright h\ z)|_P = \tau\ ((h\ z)|_P) = \tau\ v\ ,
\end{aligned}$$

so v has to be in the range of ζ. Since ζ is a monomorphism in *Set*, that is, injective, there is exactly one variable $u_0 : V_0$ such that $\zeta\ u_0 = v$. Let $\nu : V_1 \to V_0$ be an arbitrary mapping such that $\zeta \mathbin{;} \nu = \mathbb{I}_{V_0}$ — such a mapping exists since V_0 is non-empty. Then we have:

$$(\nu \mathbin{;} \zeta)\ v = v \qquad \text{for all } v \in \mathsf{ran}\ \zeta. \tag{†}$$

We define $u = h \mathbin{;} \mathcal{T}_\Sigma\ \nu$, and obtain:

$$\begin{aligned}
& u \fatsemi (\zeta \mathbin{;} \mathsf{return}_{V_1}) \\
= \quad & \{ \text{Definition of } u \} \\
& (h \mathbin{;} \mathcal{T}_\Sigma\ \nu) \fatsemi (\zeta \mathbin{;} \mathsf{return}_{V_1}) \\
= \quad & \{ \text{Monad properties} \} \\
& h \mathbin{;} \mathcal{T}_\Sigma\ (\nu \mathbin{;} \zeta) \\
= \quad & \{ (\dagger) \text{ with } \mathsf{ran}\ h \subseteq \mathsf{ran}\ \zeta \} \\
& h
\end{aligned}$$

In both cases, u is uniquely determined due to the fact that $\zeta \mathbin{;} \mathsf{return}_{V_1}$ is a monomorphism in $\mathbb{T}_\Sigma$.

This shows that $\zeta \mathbin{;} \mathsf{return}_{V_1}$ is an equaliser for σ and τ in $\mathbb{T}_\Sigma$, and we have:

Theorem 6.1. $\mathbb{T}_\Sigma$ has equalisers: For two substitutions σ, τ from V_1 to V_2, an equaliser in $\mathbb{T}_\Sigma$ can be obtained as $\zeta \mathbin{;} \mathsf{return}_{V_1}$, where ζ is the equaliser in *Set* of σ and τ as functions in $V_1 \to \mathcal{T}_\Sigma\ V_2$. $\square$

If we consider any other $\mathbb{T}_\Sigma$-equaliser h of σ and τ, with $h : Z \to \mathcal{T}_\Sigma\ V_1$, then there must also be a substitution $q : V_0 \to \mathcal{T}_\Sigma\ Z$ such that $\zeta ; \mathsf{return}_{V_1} = q \mathbin{⨾} h$. This equation implies that in particular h must be of shape $h_0 ; \mathsf{return}_{V_1}$ for some (variable renaming) function $h_0 : Z \to V_1$.

Since *regular monomorphisms* are defined to be those that are equalisers of some pair of parallel morphisms, and since in *Set* all monomorphisms are regular, we now also have:

Theorem 6.2. The regular mononorphisms in $\mathbb{T}_\Sigma$ are precisely the morphisms obtained as $\zeta ; \mathsf{return}$ from some mononorphism ζ in *Set*. □

7 Products

For every monad, a coproduct (S, ι, κ) for A and B in the base category give rise to the coproduct $(S,\ \iota ; \mathsf{return} S,\ \kappa ; \mathsf{return} S)$ in the Kleisli category.

$$
\begin{tikzcd}
 & \mathcal{M}\ S & \\
A \arrow[ur, "\iota ; \mathsf{return}_S"] \arrow[r, "\iota"'] & S \arrow[u, "\mathsf{return}_S"'] & B \arrow[ul, "\kappa ; \mathsf{return}_S"'] \arrow[l, "\kappa"]
\end{tikzcd}
$$

For the term monad, coproducts are just disjoint unions of variable sets.

However, starting from a product (P, π, ρ) for A and B in the base category and trying the same construction does not produce a product in the substitution category:

$$
\begin{tikzcd}
A \arrow[d, "\mathsf{return}_A"'] & P \arrow[l, "\pi"'] \arrow[r, "\rho"] \arrow[dl, "\pi ; \mathsf{return}_A"] \arrow[dr, "\rho ; \mathsf{return}_B"'] & B \arrow[d, "\mathsf{return}_B"] \\
\mathcal{T}_\Sigma\ A & \mathcal{T}_\Sigma\ P & \mathcal{T}_\Sigma\ B \\
 & C \arrow[ul, "\sigma"] \arrow[u, "!\psi"] \arrow[ur, "\tau"'] &
\end{tikzcd}
$$

Given σ and τ, we would need to be able to construct a substitution ψ such that $\psi \mathbin{⨾} (\pi \mathbin{;} \mathsf{return}_A) = \sigma$ and $\psi \mathbin{⨾} (\rho \mathbin{;} \mathsf{return}_B) = \tau$. Let $v : C$ be a variable. Then $\psi\ v$ must be some term $u : \mathcal{T}_\Sigma\ P$ (over the variable set $P = A \times B$) for which $(\pi \mathbin{;} \mathsf{return}_A) \triangleright u = \sigma\ v$ and $(\rho \mathbin{;} \mathsf{return}_B) \triangleright u = \tau\ v$. Using the monad laws, these equations are equivalent to the following equations, where π and ρ are mapped over the variables of u:

$$(\mathcal{T}_\Sigma\ \pi)\ u = \sigma\ v \qquad \text{and} \qquad (\mathcal{T}_\Sigma\ \rho)\ u = \tau\ v \tag{$*$}$$

If u is not a variable, then the two left-hand sides of these equations will have the same outermost function symbol.

If we choose, for example, $\sigma\ v = a$ and $\tau\ v = b$ for different constant symbols a and b, then no appropriate u can be chosen, and no substitution ψ can be constructed.

The argument above, slightly modified, actually shows that for any product of A and B in $\mathbb{T}_\Sigma$, the choice $\sigma\ v = a$ and $\tau\ v = b$ for different constant symbols a and b implies that $\psi\ v$ must be a variable, and analogously in the more general case where $\sigma\ v$ and $\tau\ v$ have different outermost function symbols. We therefore define:

Definition 7.1. Given two (variable) sets V_1 and V_2, two terms $t_1 : \mathcal{T}_\Sigma\ V_1$ and $t_2 : \mathcal{T}_\Sigma\ V_2$ are called *strong-head-equal*, written $t_1 \doteq t_2$, if there are an (n-ary) function symbol f and terms $s_{1,1}, \dots, s_{1,n} : \mathcal{T}_\Sigma\ V_1$ and $s_{2,1}, \dots, s_{2,n} : \mathcal{T}_\Sigma\ V_2$ such that $t_1 = f(s_{1,1}, \dots, s_{1,n})$ and $t_2 = f(s_{2,1}, \dots, s_{2,n})$.

We will write $t_1 \not\doteq t_2$ for $\neg(t_1 \doteq t_2)$. □

The discussion above also shows that, since $\psi\ v$ is a variable w from the product set P, the information that $\sigma\ v = a$ and $\tau\ v = b$ must be contained in that variable.

One might consider to use $P = \mathcal{T}_\Sigma\ A\ \times\ \mathcal{T}_\Sigma\ B$, which makes the projection definitions easy: $\pi : P \to \mathcal{T}_\Sigma\ A$ with $\pi(t_1, t_2) = t_1$ and analogously for ρ. However, now we have several choices for $u = \psi\ v$ if $\sigma\ v = f(a)$ and $\tau\ v = f(b)$: We could set $u = f((a, b))$ or $u = (f(a), f(b))$. (Remember that "variables" in P are pairs of terms!) Both choices would satisfy the required equations $(*)$.

This shows that, more generally, pairs (t_1, t_2) with $t_1 \doteq t_2$ should not be "variables" in P.

We obtain that $\mathbb{T}_\Sigma$ always has products:

Theorem 7.2. For any to sets A and B, the set $P = \{(t_1, t_2) : \mathcal{T}_\Sigma\ A\ \times\ \mathcal{T}_\Sigma\ B \mid t_1 \not\doteq t_2\}$ together with the projections $\pi : P \to \mathcal{T}_\Sigma\ A$ with $\pi(t_1, t_2) = t_1$ and $\rho : P \to \mathcal{T}_\Sigma\ B$ with $\rho(t_1, t_2) = t_2$ forms a product in $\mathbb{T}_\Sigma$.

Proof. Let a set C be given, and two substitutions $\sigma : C \to \mathcal{T}_\Sigma\ A$ and $\tau : C \to \mathcal{T}_\Sigma\ B$.

We need to show that there is a unique substitution $\psi : C \to \mathcal{T}_\Sigma\ P$ such that $\psi \mathbin{⨾} \pi = \sigma$ and $\psi \mathbin{⨾} \rho = \tau$.

Recall that the *anti-unifier* of two terms t and u is the most specific generalisation g of the two terms, together with two substitutions ξ_1 and ξ_2 such that $\xi_1 \triangleright g = t$ and $\xi_2 \triangleright g = u$. We shall write:

$$(g, \xi_1, \xi_2) = \mathsf{antiUnif}(t, u)$$

Now let G be the variable set of g (that is, $g \in \mathcal{T}_\Sigma\ G$), and define $\xi : G \to P$ with $\xi\ z = (\xi_1\ z, \xi_2\ z)$ for every $z : G$; this is well-defined since $\xi_1\ z \nsim \xi_2\ z$ by the definition of anti-unification. We use this to define a variant of anti-unification

$$\begin{aligned}&\mathsf{AntiUnif} : (\mathcal{T}_\Sigma\ A \times \mathcal{T}_\Sigma\ B) \to \mathcal{T}_\Sigma\ P\\ &\mathsf{AntiUnif}(t, u) = (\mathcal{T}_\Sigma\ \xi)\ g\end{aligned}$$

that produces a single term with the structure of g over variables from P, which are pairs of terms in $\nsim$.

For every variable $x : C$, we now define $\psi\ x = \mathsf{AntiUnif}(\sigma\ x, \tau\ x)$ and $(g_x, \xi_{x,1}, \xi_{x,2}) = \mathsf{antiUnif}(\sigma\ x, \tau\ x)$. Then:

$$\begin{aligned}
&(\psi \mathbin{⨾} \pi)\ x\\
=\ &\{\ \text{Definition of substitution composition}\ \}\\
&\pi \triangleright (\psi\ x)\\
=\ &\{\ \text{Definition of } \psi\ \}\\
&\pi \triangleright (\mathsf{AntiUnif}(\sigma\ x, \tau\ x))\\
=\ &\{\ \text{Definition of } \mathsf{AntiUnif}\ \}\\
&\pi \triangleright ((\mathcal{T}_\Sigma\ \xi_x)\ g_x)\\
=\ &\{\ \pi(\xi_x\ z) = \xi_{x,1}\ z\ \}\\
&\xi_{x,1} \triangleright g_x\\
=\ &\{\ \text{Definition of } \mathsf{antiUnif}\ \}\\
&\sigma\ x
\end{aligned}$$

and analogously $(\psi \mathbin{⨾} \rho)\ x = \tau\ x$.

Furthermore, if $\chi : C \to \mathcal{T}_\Sigma\ P$ is given such that $\chi \mathbin{⨾} \pi = \sigma$ and $\chi \mathbin{⨾} \rho = \tau$, then, for every variable $x : C$:

- If $\chi\ x = (t_1, t_2)$ is a variable from P, then $t_1 \nsim t_2$ and $\sigma\ x = (\chi \mathbin{⨾} \pi)\ x = t_1$ and $\tau\ x = (\chi \mathbin{⨾} \rho)\ x = t_2$, and therefore by definition of anti-unification also $\psi\ x = (t_1, t_2)$
- If $\chi\ x = f(s_1, \ldots, s_n)$, then

  $$\begin{aligned}\sigma\ x &= (\chi \mathbin{⨾} \pi)\ x = f(\pi \triangleright s_1, \ldots, \pi \triangleright s_n)\\ \tau\ x &= (\chi \mathbin{⨾} \rho)\ x = f(\rho \triangleright s_1, \ldots, \rho \triangleright s_n)\end{aligned}$$

  and therefore $\psi\ x = f\left(\mathsf{AntiUnif}(\pi \triangleright s_1, \rho \triangleright s_1), \ldots, \mathsf{AntiUnif}(\pi \triangleright s_n, \rho \triangleright s_n)\right)$.

By induction over the structure of terms we then obtain $\chi = \psi$. □

From the proof, it is obvious that P is infinite as soon as at least one of $\mathcal{T}_\Sigma\,A$ and $\mathcal{T}_\Sigma\,B$ is infinite. Therefore, we have:

Corollary 7.3.

1. Over the category of finite sets, a substitution category has all products only if the signature has no function symbols with arity at least one.
2. Over arbitrary sets, the substitution category $\mathbb{T}_\Sigma$ has all products. □

8 Pullbacks of Substitutions

Since $\mathbb{T}_\Sigma$ has products and equalisers, the standard definition of pullbacks from these can be used.

Let a cospan of substitutions $B \xrightarrow{\tau_1} D \xleftarrow{\tau_2} C$ in the Kleisli category of $\mathcal{T}_\Sigma$ be given, that is, two functions $\tau_1 : B \to \mathcal{T}_\Sigma\, D$ and $\tau_2 : C \to \mathcal{T}_\Sigma\, D$.

$$\begin{tikzcd}
 & D & \\
B \arrow[ur, "\tau_1"] & P \arrow[l, "\pi"'] \arrow[r, "\rho"] & C \arrow[ul, "\tau_2"'] \\
 & A \arrow[ul, "\sigma_1"] \arrow[u, "\zeta"] \arrow[ur, "\sigma_2"'] &
\end{tikzcd}$$

The equaliser of $\zeta : A \to \mathcal{T}_\Sigma\ P$ of $\pi \fatsemi \tau_1$ and $\pi \fatsemi \tau_2$ gives rise to the pullback $B \xleftarrow{\sigma_1} A \xrightarrow{\sigma_2} C$ with $\sigma_1 = \zeta \fatsemi \pi$ and $\sigma_2 = \zeta \fatsemi \rho$; the proof for this is a popular exercise in many introductions to category theory.

Corollary 8.1. For every signature, the resulting substitution category over *Set* has all pullbacks. □

The equaliser ζ selects those pairs (t_1, t_2) from P for which $\tau_1 \triangleright t_1 = \tau_2 \triangleright t_2$. Since $t_1 \not\equiv t_2$, at least one of t_1 and t_2 must be a variable, since if both were constructed from (necessarily different) function symbols, the equality $\tau_1 \triangleright t_1 = \tau_2 \triangleright t_2$ would not be possible.

For the case that t_1 is the variable v_1, we obtain:

$$\tau_2 \triangleright t_2 = \tau_1 \triangleright t_1 = \tau_1\ v_1$$

Since $\tau_1\ v_1$ is a finite term, there are only finitely many choices of t_2 satisfying this. Analogously, if t_2 is a variable, then there are only finitely many choices for t_1. Therefore, if the variable sets B, C, and D are all finite, then also A will be finite, even though P is (in general) infinite.

Theorem 8.2. For every signature, the resulting substitution category over the category of finite sets has all pullbacks. □

9 Pushouts

The question whether $V_1 \xleftarrow{\sigma} V_0 \xrightarrow{\tau} V_2$ has a pushout in $\mathbb{T}_\Sigma$ can be seen as a unification problem for two substitutions with disjoint variables, which corresponds to the standard construction of pushouts from coproducts and coequalisers. Therefore, obviously not all spans in $\mathbb{T}_\Sigma$ have pushouts.

An important special case are pushouts along monomorphisms, or even along monomorphisms belonging to a particular class. We have seen in Sect. 5 that in $\mathbb{T}_\Sigma$, monomorphisms may map variables to non-variable terms, and since independent monomorphisms σ and τ may map the same variable to non-unifiable terms, just restricting to monomorphisms does not help here.

We therefore have to restrict our attention to monomorphisms that map variables only to variables, that is, regular monomorphisms in $\mathbb{T}_\Sigma$, which are according to Theorem 6.2 the monomorphisms of shape $m \mathbin{;} \mathsf{return}$ where m is a monomorphism in the base category, that is, an injective variable mapping.

Let a substitution $\sigma : V_0 \to \mathcal{T}_\Sigma\ V_1$ and an injective function $\iota_2 : V_0 \to V_2$ be given. Since this is in *Set*, we can see ι_2 as the first injection of a coproduct $V_0 \xrightarrow{\iota_2} V_2 \xleftarrow{\kappa_2} U$, where U needs to be isomorphic to the set $V_2 - \mathsf{ran}\ \iota_2$. Then define V_3 via another coproduct $V_1 \xrightarrow{\iota_3} V_3 \xleftarrow{\kappa_3} U$ in *Set*.

Now we define $\tau : V_2 \to \mathcal{T}_\Sigma\ V_3$ as a universal morphism associated with the coproduct V_2, namely:

$$\tau = [\sigma \mathbin{;} \mathcal{T}_\Sigma\ \iota_3, \quad \kappa_3 \mathbin{;} \mathsf{return}_{V_3}]$$

Then the following diagram is a pushout in $\mathbb{T}_\Sigma$:

$$\begin{tikzcd}
V_0 \arrow[r, "\sigma"] \arrow[d, "\iota_2 \mathbin{;} \mathsf{return}_{V_2}"'] & V_1 \arrow[d, "\iota_3 \mathbin{;} \mathsf{return}_{V_3}"] \\
V_2 \arrow[r, "\tau"] & V_3
\end{tikzcd}$$

Commutativity follows easily:

$$\begin{aligned}
& (\iota_2 \mathbin{;} \mathsf{return}_{V_2}) \mathbin{⨾} \tau \\
= \quad & \{\ \text{Monad properties}\ \} \\
& \iota_2 \mathbin{;} \tau \\
= \quad & \{\ \text{Definition of } \tau\ \} \\
& \iota_2 \mathbin{;} [\sigma \mathbin{;} \mathcal{T}_\Sigma\ \iota_3, \quad \kappa_3 \mathbin{;} \mathsf{return}_{V_3}] \\
= \quad & \{\ \text{Coproduct properties}\ \} \\
& \sigma \mathbin{;} \mathcal{T}_\Sigma\ \iota_3 \\
= \quad & \{\ \text{Monad properties}\ \} \\
& \sigma \mathbin{⨾} (\iota_3 \mathbin{;} \mathsf{return}_{V_3})
\end{aligned}$$

Assume another cospan $V_1 \xrightarrow{\varphi} V_4 \xleftarrow{\psi} V_2$ in $\mathbb{T}_\Sigma$ with

$$\sigma \mathbin{⨾} \varphi = (\iota_2 \mathbin{;} \mathsf{return}_{V_2}) \mathbin{⨾} \psi \ . \tag{‡}$$

Then define $\chi : V_3 \to \mathcal{T}_\Sigma\ V_4$ as a universal morphism associated with the coproduct V_3, namely $\chi = [\varphi,\ \kappa_2 \mathbin{;} \psi]$. The resulting triangles commute:

$$
\begin{array}{ll}
 & (\iota_3 \mathbin{;} \mathsf{return}_{V_3}) \mathbin{⨾} \chi \\
= & \{ \text{Monad properties, definition of } \chi \} \\
 & \iota_3 \mathbin{;} [\varphi,\ \kappa_2 \mathbin{;} \psi] \\
= & \{ \text{Coproduct properties} \} \\
 & \varphi
\end{array}
$$

and:

$$
\begin{array}{ll}
 & \tau \mathbin{⨾} \chi \\
= & \{ \text{Definition of } \tau\text{, coproduct properties} \} \\
 & [(\sigma \mathbin{;} \mathcal{T}_\Sigma\ \iota_3) \mathbin{⨾} \chi, \quad (\kappa_3 \mathbin{;} \mathsf{return}_{V_3}) \mathbin{⨾} \chi] \\
= & \{ \text{Monad properties} \} \\
 & [\sigma \mathbin{⨾} (\iota_3 \mathbin{;} \chi), \quad \kappa_3 \mathbin{;} \chi] \\
= & \{ \text{Definition of } \chi\text{, coproduct properties} \} \\
 & [\sigma \mathbin{⨾} \varphi, \quad \kappa_2 \mathbin{;} \psi] \\
= & \{ (‡)\text{, monad properties} \} \\
 & [\iota_2 \mathbin{;} \psi, \quad \kappa_2 \mathbin{;} \psi] \\
= & \{ \text{Coproduct properties} \} \\
 & \psi
\end{array}
$$

Furthermore, for every other substitution $\xi : V_3 \to \mathcal{T}_\Sigma\ V_4$ with $(\iota_3 \mathbin{;} \mathsf{return}_{V_3}) \mathbin{⨾} \xi = \varphi$ and $\tau \mathbin{⨾} \xi = \psi$ we have:

$$
\begin{array}{ll}
 & \chi \\
= & \{ \text{Definition of } \chi \} \\
 & [\varphi, \quad \kappa_2 \mathbin{;} \psi] \\
= & \{ \text{Assumptions about } \xi \} \\
 & [(\iota_3 \mathbin{;} \mathsf{return}_{V_3}) \mathbin{⨾} \xi, \quad \kappa_2 \mathbin{;} (\tau \mathbin{⨾} \xi)] \\
= & \{ \text{Monad properties} \} \\
 & [(\iota_3 \mathbin{;} \xi, \quad (\kappa_2 \mathbin{;} \tau) \mathbin{⨾} \xi)] \\
= & \{ \text{Definition of } \tau\text{, coproduct properties} \} \\
 & [(\iota_3 \mathbin{;} \xi, \quad (\kappa_3 \mathbin{;} \mathsf{return}_{V_3}) \mathbin{⨾} \xi)] \\
= & \{ \text{Monad properties} \} \\
 & [(\iota_3 \mathbin{;} \xi, \quad \kappa_3 \mathbin{;} \xi)] \\
= & \{ \text{Coproduct properties} \} \\
 & \xi
\end{array}
$$

Altogether, this shows:

Theorem 9.1. $\mathbb{T}_\Sigma$ has pushouts along regular monomorphisms. $\square$

Such a pushout just adds the variables of V_2 outside the range of m as additional "unused" target variables to σ.

Already if m is not a monomorphism, pushouts for $V_1 \xleftarrow{\sigma} V_0 \xrightarrow{m\,;\,\mathsf{return}} V_2$ in $\mathbb{T}_\Sigma$ will not exist if there are two variables $u, v : V_0$ with $m\ u = m\ v$, but for which $\sigma\ u$ and σv are not unifiable.

In the context of the proof above, if the $\mathbb{T}_\Sigma$-cospan $V_1 \xrightarrow{\varphi} V_4 \xleftarrow{\psi} V_2$, too, is a $\mathbb{T}_\Sigma$-pushout for $V_1 \xleftarrow{\sigma} V_0 \xrightarrow{\iota_2\,;\,\mathsf{return}} V_2$, then $\chi : V_3 \to \mathcal{T}_\Sigma\ V_4$ has to be an isomorphism between the two pushout objects V_3 and V_4, with the inverse satisfying in particular $\varphi \fatsemi \chi^{-1} = \iota_3\,;\,\mathsf{return}_{V_3}$. Due to this equality, the image of φ can only contain variable terms, and since $\varphi = (\iota_3\,;\,\mathsf{return}_{V_3}) \fatsemi \chi$ is a composition of a monomorphism with an isomorphism, it altogether has to be a regular monomorphism, so we have:

Theorem 9.2. In $\mathbb{T}_\Sigma$, regular monomorphisms are stable under pushout. $\square$

10 Conclusion and Outlook

For categories with variable sets as objects and substitutions as morphisms, we provided explicit characterisations and constructions of monomorphisms, epimorphisms, equalisers, regular monomorphisms, products, pullbacks, and of pushouts along regular monomorphisms. These can be useful in contexts where categories of substitutions become building blocks of more complicated categories, as for example in transformation of symbolically attributed graphs.

While in settings with global variable sets, such as that of Eder (1985), anti-unification appears as dual to unification, we identified the construction of the universal morphisms for products as corresponding to anti-unification. This, interestingly, is not at all categorially-dual to the coequalisers or pushouts that correspond to unification.

Acknowledgement: I would like to express my gratitude to the anonymous referees for their constructive comments, which helped significantly to improve the presentation.

References

E. Eder. Properties of substitutions and unifications. *J. Symbolic Comput.*, 1: 31–46, 1985. https://doi.org/10.1016/S0747-7171(85)80027-4.

H. Ehrig, A. Habel, H.-J. Kreowski, F. Parisi-Presicce. From graph grammars to high level replacement systems. In H. Ehrig, H.-J. Kreowski, G. Rozenberg (eds.), *GraGra '90*, LNCS, vol. 532, pp. 269–287. Springer, 1991. https://doi.org/10.1007/BFb0017372.

H. Ehrig, K. Ehrig, U. Prange, G. Taentzer. *Fundamentals of Algebraic Graph Transformation.* Springer, 2006. https://doi.org/10.1007/3-540-31188-2.

S. Hosseini, Y. Qasemi Nezhad. Equalizers in Kleisli categories. *Cahiers de Topologie ed Géométrie Différentielle Catégoriques*, 57(1):51–76, 2016. URL http://cahierstgdc.com/index.php/volumes/volume-lii-2016/.

W. Kahl. Categories of coalgebras with monadic homomorphisms. In M. Bonsangue (ed.) CMCS 2014, LNCS, vol. 8446, pp. 151–167. Springer, 2014. https://doi.org/10.1007/978-3-662-44124-4_9. Agda theories at http://RelMiCS.McMaster.ca/RATH-Agda/.

W. Kahl. Graph transformation with symbolic attributes via monadic coalgebra homomorphisms. *ECEASST*, 71:5.1–5.17, 2015. https://doi.org/10.14279/tuj.eceasst.71.999.

S. Lack, P. Sobociński. Adhesive categories. In I. Walukiewicz (ed.), *FOSSACS 2004*, LNCS, vol. 2987, pp. 273–288, 2004. https://doi.org/10.1007/b95995.

S. Lack, P. Sobociński. Adhesive and quasiadhesive categories. *RAIRO Inform. Théor. Appl.*, 39(3):511–545, 2005. https://doi.org/10.1051/ita:2005028.

F. W. Lawvere. Functorial semantics of algebraic theories. *Proc. Nat. Acad. Sci. USA*, 50:869–872, 1963. https://doi.org/10.2307/2272673.

U. Norell. *Towards a Practical Programming Language Based on Dependent Type Theory.* PhD thesis, Dept. Comp. Sci. and Eng., Chalmers Univ. of Technology, 2007. See also http://wiki.portal.chalmers.se/agda/pmwiki.php.

D. E. Rydeheard, J. G. Stell. Foundations of equational deduction: A categorical treatment of equational proofs and unification algorithms. In D. H. Pitt, A. Poigné, D. E. Rydeheard (eds.), *Category Theory and Computer Science, CTCS 1987*, LNCS, vol. 283, pp. 114–139. Springer, 1987. https://doi.org/10.1007/3-540-18508-9_23.

J. Szigeti. On limits and colimits in the Kleisli category. *Cahiers de Topologie ed Géométrie Différentielle Catégoriques*, 24(4):381–391, 1983. URL http://www.numdam.org/item?id=CTGDC_1983__24_4_381_0.