Semantic Constructions for Hidden Algebra

Corina Cîrstea[*]

Oxford University Computing Laboratory
Wolfson Building, Parks Road, Oxford OX1 3QD, UK
Corina.Cirstea@comlab.ox.ac.uk

Abstract. Hidden algebra is a behavioural algebraic specification formalism for objects. It captures their constructional aspect, concerned with the initialisation and evolution of their states, as well as their observational aspect, concerned with the observable b ehaviour of such states. When attention is restricted to the observational aspect, final/cofree constructions provide suitable denotations for the specification techniques involved. However, when the constructional aspect is integrated with the observatio nal one, the presence of nondeterminism in specifications prevents the existence of final/cofree algebras. It is shown here that *final/cofree families of algebras* exist in this case, with each algebra in such a family resolving the nondetermi nism in a particular way. Existence of final/cofree families yields a canonical way of constructing algebras of structured specifications from algebras of the component specifications. Finally, a layered approach to specifying complex objects in hidden al gebra is presented, with the semantics still involving final/cofree families.

1 Introduction

The use of algebra in the semantics of computation goes back to the 1970s and the use of initial algebras as denotational semantics for data types [9]. The constructional nature of data types makes algebra particularly suitable for their specification – the emphasis is on *generating* the elements of data types by means of *constructor operations*, with minimal structures such as initial or free algebras providing suitable denotations for data type specifications. Recently, the theory of *coalgebras* (the formal duals of algebras) has been used for the specification of state-based systems in general [5], and of objects in particular [4]; here, the emphasis is on *observing* system states by means of *destructor operations*, and maximal structures such as final or cofree coalgebras, incorporating possible behaviours, are used as denotations.

Objects are characterised by a state together with an interface which provides (limited) access to this state. Specifically, the object interface can be used to initialise its state in a particular way, to perform certain changes on its current state, or to observe certain properties of this state. One can identify a constructional aspect of objects, concerned with the initialisation and evolution of their

[*] Research supported by an ORS Award and an Oxford Bursary.

J.L. Fiadeiro (Ed.): WADT'98, LNCS 1589, pp. 63–78, 1999.

states, and an observational aspect of objects, concerned with the observations that can be made about such states.

Hidden algebra [1] combines concepts from algebra and coalgebra in order to capture the two aspects of objects and the relationship between them. One can argue that hidden algebra lies at the intersection of algebra and coalgebra, as its syntax is (a restricted version of) the syntax of many-sorted algebra, while its semantics is behavioural (coalgebraic). Consequently, the behaviours specifiable in hidden algebra are, in a sense, both algebraic and coalgebraic.

The coalgebraic nature of hidden algebra, already observed in [1], has been further investigated in [8], where the relevance of final/cofree constructions to coalgebraic hidden specifications and their reuse along specification maps has been emphasised. *Final hidden algebras* have been shown to provide a characterisation of the abstract behaviours associated to such specifications, while *cofree hidden algebras* have been used as formal denotations for their reuse.

When arbitrary hidden specifications are considered, the nondeterminism arising from underspecifying the behaviour of the constructor operations prevents the existence of final/cofree hidden algebras. It has been suggested in [8] that in this case, *final/cofree families* of hidden algebras should be taken as denotations, since such constructions can be used to characterise the ways of resolving the nondeterminism involved. Final/cofree families generalise final/cofree objects in a category, while still retaining their universal properties. The present paper gives a detailed account of the existence of such families in hidden algebra, illustrating their suitability as semantic constructions for hidden specifications. Existence of final/cofree families also yields a canonical way of constructing algebras of structured specifications from algebras of the component specifications.

Due to restrictions on its syntax, triggered by the coalgebraic nature of the approach, hidden algebra only provides limited support for the specification of *complex objects* (objects having other objects as components). In particular, neither can previously specified objects be used to specify the construction of new objects, nor can such objects be passed as arguments to the destructor operations of other objects. This paper also integrates complex objects in the hidden algebra formalism.

The paper is structured as follows. After recalling some category-theoretic concepts that will be used later in the paper, Section 2 introduces the hidden algebra formalism and briefly summarises the results in [8] regarding the existence of final/cofree constructions in *coalgebraic hidden algebra* (a restricted version of hidden algebra used to specify coalgebraic behaviours). Section 3 focuses on the existence of final/cofree families of hidden algebras and on their suitability as denotations for hidden specifications and their reuse. Section 4 uses a generalisation of the category-theoretic notion of limit to define a canonical way of combining algebras of component specifications into algebras of structured specifications. Section 5 presents a layered approach to the specification of complex objects in hidden algebra, with final/cofree families still providing appropriate denotations for the specification techniques involved. Finally, Section 6 summarises the results presented and briefly outlines future work.

2 Preliminaries

The first part of this section introduces some categorical concepts that will be used later in the paper, while the second part gives an outline of the hidden algebraic approach to object specification and of some existing results regarding the existence of final/cofree constructions in coalgebraic hidden algebra.

2.1 Some Category-Theoretic Notions

A *final object* in a category C is an object F of C such that any other object of C has a unique arrow into F. The notion of *final family of objects* generalises the notion of final object by requiring the existence of a unique arrow from any other object of C into an object in the final family.

Definition 1. *Given a category C, a family $(F_j)_{j \in J}$ of C-objects is a* **final family of C-objects** *if and only if, for any C-object C, there exist unique $j \in J$ and C-arrow $f : C \to F_j$ in C.*

Remark 1. A final family of C-objects determines a partition of C into subcategories with final objects (given by objects in the family). For $j \in J$, C_j is given by the full subcategory of C whose objects have an arrow into F_j.

[3] presents a generalisation of the category-theoretic notion of limit, called a *multi-limit*, which enjoys a universal property similar to that of a limit.

Definition 2. *Given a diagram $\mathsf{d} : \mathsf{D} \to \mathsf{C}$ in a category C, a* **multi-limit for** d *consists of a family $(L^i, (l^i_D : L^i \to \mathsf{d}(D))_{D \in |\mathsf{D}|})_{i \in I}$ of cones for d, having the property that given any other cone $(C, (c_D)_{D \in |\mathsf{D}|})$ for d, there exist unique $i \in I$ and C-arrow $c : C \to L^i$ such that $l^i_D \circ c = c_D$ for each D-object D.*

Then, final families of objects appear as multi-limits of empty diagrams.

A *couniversal arrow* from a functor $\mathsf{U} : \mathsf{D} \to \mathsf{C}$ to a C-object C is a C-arrow of form $\epsilon_C : \mathsf{U}\bar{C} \to C$ for some D-object $\bar{C}$, having the property that given any D-ob ject D and C-arrow $f : \mathsf{U}D \to C$, there exists a unique factorisation of f through ϵ_C of form $f = \mathsf{U}\bar{f}; \epsilon_C$ with $\bar{f} : D \to \bar{C}$. ($\bar{C}$ is called a *cofree D-object over C w. r.t.* U.) The notion of *couniversal family of arrows* [3] generalises that of couniversal arrow as follows.

Definition 3. *Given a functor $\mathsf{U} : \mathsf{D} \to \mathsf{C}$ and a C-object C, a family of C-arrows $(\epsilon_{C,j} : \mathsf{U}\bar{C}_j \to C)_{j \in J}$ with $\bar{C}_j$ a D-object for each $j \in J$ is a* **couniversal family of arrows fr om U to** C *if and only if, for any D-object D and C-arrow $f : \mathsf{U}D \to C$, there exist unique $j \in J$ and D-arrow $\bar{f} : D \to \bar{C}_j$ such that $U\bar{f}; \epsilon_{C,j} = f$. The family $(\bar{C}_j)_{j \in J}$ is called a* **cofree family of D-objects over** C **w.r.t.** U*. If, for any C-object C, there exists a couniversal family of arrows from U to C, then U is said to have a* **right multi-adjoint***.*

Remark 2. The concepts of multi-limit and couniversal family of arrows can be subsumed under the ordinary concepts of limit and couniversal arrow by considering *categories of families*. Given a category C, one can define a category $\mathsf{Fam}(\mathsf{C})$ wh ose objects are indexed families $(C_i)_{i \in I}$ of C-objects, and whose arrows from $(C_i)_{i \in I}$ to $(D_j)_{j \in J}$ are given by a reindexing function $h : I \to J$ together with an I-indexed family $(f_i : C_i \to D_{h(i)})_{i \in I}$ of C-arrows. C has a canonical embedding into $\mathsf{Fam}(\mathsf{C})$ which regards C-objects/arrows as families of C-objects/arrows indexed by a one-element set. Then, multi-limits of C-diagrams correspond to limits in $\mathsf{Fam}(\mathsf{C})$ of the translations of these diagrams along the embedding of C into $\mathsf{Fam}(\mathsf{C})$, while couniversal families of arrows from $\mathsf{U} : \mathsf{D} \to \mathsf{C}$ to C-objects corresponds to couniversal arrows from $\mathsf{Fam}(\mathsf{U}) : \mathsf{Fam}(\mathsf{D}) \to \mathsf{Fam}(\mathsf{C})$ to the translations of these objects along the embedding of C into $\mathsf{Fam}(\mathsf{C})$ (where $\mathsf{Fam}(\mathsf{U})$ takes $(D_i)_{i \in I}$ to $(\mathsf{U}(D_i))_{i \in I}$, and $\langle h, (f_i)_{i \in I} \rangle : (D_i)_{i \in I} \to (D'_j)_{j \in J}$ to $\langle h, (\mathsf{U}(f_i))_{i \in I} \rangle$).

2.2 Hidden Algebra

This section recalls the underlying definitions of the hidden algebra formalism, as well as some earlier results regarding the existence of semantic constructions based on finality in coalgebraic hidden algebra.

The fundamental distinction between data values and object states is reflected in the syntax of hidden algebra in the use of visible sorts/operation symbols for the data, and of hidden sorts/operation symbols for the objects. A *data universe*, given by an algebra D (the **data algebra**) of a many-sorted signature (V, Ψ) (the **data signature**) is fixed beforehand, with the additional constraint that each element of D is named by some constant in Ψ. For convenience, we assume $D_v \subseteq \Psi_{[],v}$ for each $v \in V$.

The operations available for creating and accessing the states of objects are specified using *hidden signatures*, while translations from one signature to another are captured by *hidden signature maps*.

Definition 4. *A* **(hidden) signature over** (V, Ψ) *is a pair* (H, Σ) *with* H *a set of* **hidden sorts***, and* Σ *a* $V \cup H$*-sorted signature satisfying: (i)* $\Sigma_{w,v} = \Psi_{w,v}$ *for* $w \in V^*$ *and* $v \in V$*, and (ii) for* $\sigma \in \Sigma_{w,s}$*, at most one sort appearing in* w *(by convention, the first one) is hidden.* $\Sigma \setminus \Psi$*-operation symbols having exactly one hidden-sorted argument are called* **destructor symbols***, while those having only visible-sorted arguments are called* **constructor symbols***. A* **(hidden) signature map** $\phi : (H, \Sigma) \to (H', \Sigma')$ *is a many-sorted signature morphism* $\phi : (V \cup H, \Sigma) \to (V \cup H', \Sigma')$ *such that* $\phi\restriction_{(V,\Psi)} = 1_{(V,\Psi)}$ *and* $\phi(H) \subseteq H'$.

An *algebra* of a hidden signature agrees with the data algebra on the interpretation of the visible sorts/operation symbols and, in addition, provides interpretations for the hidden sorts/operation symbols.

Definition 5. *A* **(hidden)** Σ**-algebra** *is a many-sorted* $(V \cup H, \Sigma)$*-algebra* A *such that* $A\restriction_{\Psi} = D$*. A* **(hidden)** Σ**-homomorphism** *between* Σ*-algebras* A *and* B *is a many-sorted* Σ*-homomorphism* $f : A \to B$ *such that* $f_v = 1_{D_v}$ *for* $v \in V$*.*

Σ-algebras and Σ-homomorphisms form a category, denoted $\mathsf{Alg}(\Sigma)$. Hidden signature maps $\phi : \Sigma \to \Sigma'$ then induce reduct functors $\mathsf{U}_\phi : \mathsf{Alg}(\Sigma') \to \mathsf{Alg}(\Sigma)$. For a Σ'-algebra A' (Σ'-homomorphism f'), we write $A'\restriction_\Sigma$ (respectively $f'\restriction_\Sigma$) for $\mathsf{U}_\phi(A')$ (respectively $\mathsf{U}_\phi(f')$) whenever ϕ is clear from the context.

Definition 6. *Given a hidden signature map $\phi : (H, \Sigma) \to (H', \Sigma')$ and a Σ-algebra A, a Σ'-algebra A' is a* **coextension of** A **along** ϕ *if and only if there exists a Σ-homomorphism $f : \mathsf{U}_\phi(A') \to A$.*

Hidden algebra takes a behavioural approach to specifying objects – their states are only specified *up to observability*. State observations are formalised by *contexts*, while indistinguishability of states by observations is captured by *behavioural equivalence*.

Definition 7. *Given a hidden signature (H, Σ), a* **Σ-context** *for sort $s \in V \cup H$ is an element of $T_\Sigma(\{z\})_v$, with z an s-sorted variable and $v \in V$. Given a Σ-algebra A,* **behavioural equivalence on** A *(denoted $\sim_A$) is given by: $a \sim_{A,s} a'$ if and only if $c_A(a) = c_A(a')$ for all contexts c for s, $s \in V \cup H$ and $a, a' \in A_s$.*

One uses (many-sorted) equations to specify correctness properties of object behaviour. However, one only requires the two sides of an equation to be indistinguishable by observations, rather than to coincide.

Definition 8. *A* **(hidden) specification** *is a triple (H, Σ, E) with (H, Σ) a hidden signature and E a set of (many-sorted) Σ-equations. A Σ-algebra A* **behaviourally satisfies** *a Σ-equation e of form $(\forall X)\ l = r$ if $l_1 = r_1, \ldots, l_n = r_n$ (written $A \models\!\equiv_\Sigma e$) if and only if, for any assignment $\theta : X \to A$ of values in A to the variables in X, $\bar{\theta}(l) \sim_A \bar{\theta}(r)$ whenever $\bar{\theta}(l_i) \sim_A \bar{\theta}(r_i)$ for $i = 1, \ldots, n$. Given a set E of Σ-equations and a Σ-equation e, one writes $E \models\!\equiv_\Sigma e$ if $A \models\!\equiv_\Sigma E$ implies $A \models\!\equiv_\Sigma e$ for any Σ-algebra A.*

The following properties of behavioural satisfaction will be used later.

Proposition 1. *Let A, B be Σ-algebras and $f : A \to B$ be a Σ-homomorphism.*

1. *$B \models\!\equiv_\Sigma e$ implies $A \models\!\equiv_\Sigma e$ for each Σ-equation e.*
2. *$A \models\!\equiv_\Sigma e$ implies $B \models\!\equiv_\Sigma e$ for each Σ-equation e in visible-sorted variables.*

We let $\mathsf{Alg}(\Sigma, E)$ denote the full subcategory of $\mathsf{Alg}(\Sigma)$ having Σ-algebras that behaviourally satisfy E as objects.

Proposition 2. *The category $\mathsf{Alg}(\Sigma, E)$ has pullbacks.*

Proof (sketch). Pullbacks in $\mathsf{Alg}(\Sigma, E)$ are constructed as pullbacks in the category of many-sorted Σ-algebras and Σ-homomorphisms.

We restrict our attention to specifications whose equations have visible-sorted conditions, if any. Given an equation e of form $(\forall \mathcal{V})\ l = r$ if $l_1 = r_1, \ldots, l_n = r_n$ such that $l_1, r_1, \ldots, l_n, r_n$ are visible- sorted, the *visible consequences* of e are of form: $(\forall \mathcal{V})\ c[l] = c[r]$ if $l_1 = r_1, \ldots, l_n = r_n$ ($c[e]$ for short), with $c \in T_\Sigma(\{z\})$ appropriate for l, r. Then, $A \models\!\equiv_\Sigma e$ if and on ly if $A \models_\Sigma c[e]$ for any $c \in T_\Sigma(\{z\})$.

Hidden algebra provides support for the reuse of specifications through the notion of *hidden specification map*.

Definition 9. *A hidden signature map $\phi : \Sigma \to \Sigma'$ defines a* **(hidden) specification map** $\phi : (\Sigma, E) \to (\Sigma', E')$ *if and only if* $E' \models_{\Sigma'} \phi(c[e])$ *for each* $e \in E$ *and each* Σ*-context* c *for* e.

Given a specification map $\phi : (\Sigma, E) \to (\Sigma', E')$, the reduct functor $\mathsf{U}_\phi : \mathsf{Alg}(\Sigma') \to \mathsf{Alg}(\Sigma)$ induced by the signature map $\phi : \Sigma \to \Sigma'$ takes hidden (Σ', E')-algebras to hidden (Σ, E)-algebras.

The following result allows a finite number of specifications related via specification maps to be combined in a canonical way.

Theorem 1. *The category* Spec *of hidden specifications and specification maps is finitely cocomplete.*

The rest of this section briefly recalls some existing results regarding the existence of semantic constructions based on finality in coalgebraic hidden algebra.

Definition 10. *A hidden signature* (H, Σ) *is a* **destructor signature** *if and only if* $\Sigma \backslash \Psi$ *consists of destructor symbols only. A hidden specification* (H, Σ, E) *is a* **destructor specification** *if and only if* (H, Σ) *is a destruct or signature, and each equation in* E *contains exactly one hidden-sorted variable.*

Any destructor specification admits a final algebra. This appears as a consequence of the existence of a one-to-one correspondence between hidden algebras of destructor signatures and coalgebras of endofunctors induced by such signatures on one hand, and of the equations in destructor specifications defining predicates on the carriers of algebras of the underlying signatures on the other. The elements of the final algebra of a destructor specification describe all possible behaviours under the specified d estructors that satisfy the constraints imposed by the equations.

The main result in [8] shows the existence of cofree constructions w.r.t. reduct functors induced by destructor specification maps. Given destructor specifications (Δ, E), (Δ', E') and a hidden specification map $\phi : (\Delta, E) \to (\Delta', E')$, the reduct functor $\mathsf{U}_\phi : \mathsf{Alg}(\Sigma', E') \to \mathsf{Alg}(\Sigma, E)$ is shown to have a right adjoint C_ϕ. The counit of the adjunction yields, for each (Δ, E)-algebra A, a couniversal arrow $\epsilon_A : \mathsf{U}_\phi(\mathsf{C}_\phi(A)) \to A$ from U_ϕ to A. That is, $\mathsf{C}_\phi(A)$ coextends A along ϕ and furthermore, the universal property of ϵ_A makes $\mathsf{C}_\phi(A)$ final (most general) among the (Δ', E')-coextensions of A along ϕ. $\mathsf{C}_\phi(A)$ is called a **cofree coextension of** A **along** ϕ.

3 Semantics with Final/Cofree Families

Due to the possibility of underspecifying constructor operations, existence of final/cofree hidden algebras does not generalise to arbitrary hidden specifications/specification maps. However, as already suggested in [8], final/cofree famil ies of hidden algebras can be used to characterise all possible ways of resolving the nondeterminism arising from underspecification. Here we prove the existence of such constructions in hidden algebra and emphasise their suitability as denotations for hi dden specifications/specification maps.

Theorem 2. *Let (Σ, E) denote a hidden specification. If each equation in E contains at most one hidden-sorted variable, then there exists a final family of hidden (Σ, E)-algebras.*

Proof. We define a relation $\sim$ on hidden (Σ, E)-algebras and use it to partition $\mathsf{Alg}(\Sigma, E)$ into subcategories. Next, we show that each of these subcategories has a final object. It then follows that $\mathsf{Alg}(\Sigma, E)$ has a final family of objects.

Given (Σ, E)-algebras A and B, we let $A \sim B$ if and only if there exist a (Σ, E)-algebra C and Σ-homomorphisms $f : C \to A$ and $g : C \to B$. Since $\mathsf{Alg}(\Sigma, E)$ has pullbacks (see Proposition 2), it follows that $A \sim B$ holds if and only if A and B are *connected* in $\mathsf{Alg}(\Sigma, E)$, i.e. there exists a *zigzag morphism* from A to B in $\mathsf{Alg}(\Sigma, E)$ (see [2], page 58). Hence, $\sim$ determines a partition $\mathcal{C}$ of $\mathsf{Alg}(\Sigma, E)$ into subcategories.

We now show that each category C in $\mathcal{C}$ has a final object. We let Δ denote the destructor subsignature of Σ and let F_Δ denote a final Δ-algebra. We define a many-sorted subset F_C of F_Δ as follows: $F_{\mathsf{C},v} = D_v$ for $v \in V$, and $F_{\mathsf{C},h} = \{f \in F_{\Delta,h} \mid f = f_A(a) \text{ for some } A \in |\mathsf{C}| \text{ and } a \in A_h\}$ for $h \in H$ (where, for a Σ-algebra A, $f_A : A\restriction_\Delta \to F_\Delta$ denotes the unique Δ-homomorphism of its Δ-reduct into F_Δ). Then, F_C defines a Δ-subalgebra of F_Δ: given $f \in F_{\mathsf{C},h}$ with $f = f_A(a)$ for some $A \in |\mathsf{C}|$ and $a \in A_h$, and given $\delta \in \Delta_{hw,h'}$ with $h, h' \in H$ and $w \in V^*$, we have: $\delta_{F_\Delta}(f, \bar{d}) = f_A(\delta_A(a, \bar{d}))$, and hence $\delta_{F_\Delta}(f, \bar{d}) \in F_{\mathsf{C},h'}$ for each $\bar{d} \in D_w$. Moreover, F_C can be given the structure of a Σ-algebra by arbitrarily choosing $A \in |\mathsf{C}|$ and then letting $\gamma_{F_\mathsf{C}}(\bar{d}) = f_A(\gamma_A(\bar{d}))$ for each $\gamma \in \Sigma_{w,h}$ with $w \in V^*$ and $h \in H$, and each $\bar{d} \in D_w$. The definition of $\sim$ together with uniqueness of a Δ-homomorphism into a final Δ-algebra ensure that the definition of γ_{F_C} does not depend on the choice of A. Then, $F_\mathsf{C} \models\!\!\equiv_\Sigma E$ follows from each $e \in E$ containing at most one hidden-sorted variable: in this case, any assignment of values in F_C to the variables in e is obtained by post-composing a similar assignment into some $A \in |\mathsf{C}|$ with f_A; behavioural satisfaction of e in (a state f of) F_C then follows from its behavioural satisfaction in (a state a of) A, with $A \in |\mathsf{C}|$.

Hence, $F_\mathsf{C} \in |\mathsf{C}|$; furthermore, F_C is final in C: given A in $|\mathsf{C}|$, $A\restriction_\Delta$ has a unique Δ-homomorphism f_A into F_Δ which, by the definition of F_C, defines a Σ-homomorphism $f_A : A \to F_\mathsf{C}$. Uniqueness of such a Σ-homomorphism follows from uniqueness of a Δ-homomorphism into F_Δ.

It then follows that $(F_\mathsf{C})_{\mathsf{C}\in\mathcal{C}}$ is a final family of hidden (Σ, E)-algebras: given any (Σ, E)-algebra A, say $A \in |\mathsf{C}|$ for some $\mathsf{C} \in \mathcal{C}$, there exists a unique Σ-homomorphism $f_A : A \to F_\mathsf{C}$; also, for $\mathsf{C}' \neq \mathsf{C}$, there exists no Σ-homomorphism of A into $F_{\mathsf{C}'}$, as C and C' are disjoint.

The existence of a final family of (Σ, E)-algebras results in the existence of a final object in the category $\mathsf{Fam}(\mathsf{Alg}(\Sigma, E))$ (see Remark 2), given by $(F_\mathsf{C})_{\mathsf{C}\in\mathcal{C}}$.

The next result states an important property of final families.

Theorem 3. *Let (Σ, E) denote a hidden specification, $(F_i)_{i\in I}$ denote a final family of hidden (Σ, E)-algebras, and e denote an arbitrary Σ-equation. Then, e is behaviourally satisfied by all (Σ, E)-algebras if and only if e is behaviourally satisfied by each F_i, with $i \in I$.*

Proof. The **only if** direction follows by each F_i being a (Σ, E)-algebra. For the **if** direction, given an arbitrary (Σ, E)-algebra A, existence of a Σ-homomorphism from A to one of the F_is together with Proposition 1 and $F_i \models\!\equiv_\Sigma e$ yield $A \models\!\equiv_\Sigma e$.

The above result justifies the use of final families as denotations for hidden specifications satisfying the hypothesis of Theorem 2.

The proof of Theorem 2 also gives some information about how the algebras in the final family look like: for a hidden specification (Σ, E), the Δ-reduct of each algebra in the final family is a Δ-subalgebra of the final Δ-algebra (with Δ denoting the destructor subsignature of Σ). However, in most cases, the final family has a more concrete representation than the one above. Such cases correspond to *split specifications*.

Definition 11. *Given a hidden signature Σ with destructor subsignature Δ, a hidden specification (Σ, E) is called* **split** *if and only if $E = E_\Delta \cup E_\Sigma$, with E_Δ consisting of Δ-equations in one hidden-sorted var iable and E_Σ consisting of Σ-equations in no hidden-sorted variables.*

The intuition behind the above definition is that E_Δ constrains the behaviour of hidden states (by means of equations that use Δ-symbols only), while E_Σ constrains the interpretation of the constructor symbols in the state space defined by E_Δ, without imposing further constraints to this state space.

Proposition 3. *Let (Σ, E) denote a split hidden specification (with $E = E_\Delta \cup E_\Sigma$), let F_{Δ, E_Δ} denote a final (Δ, E_Δ)-algebra and let $\mathcal{F} = \{F \in \mathsf{Alg}(\Sigma) \mid F\restriction_\Delta = F_{\Delta, E_\Delta},\ F \models\!\equiv_\Sigma E_\Sigma\}$. Then, $\mathcal{F}$ defines a final family of hidden (Σ, E)-algebras.*

Proof. We must show that an arbitrary (Σ, E)-algebra A has exactly one Σ-homomorphism into an $F \in \mathcal{F}$. Any such homomorphism must extend the unique Δ-homomorphism $f_A : A\restriction_\Delta \to F_{\Delta, E_\Delta}$ resulting from $A\restriction_\Delta \models\!\equiv_\Delta E_\Delta$ on one hand, and must preserve the $\Sigma \setminus \Delta$-structure on the other. Hence, the only $F \in \mathcal{F}$ that A can have a Σ-homomorphism into has its $\Sigma \setminus \Delta$-structure induced by the $\Sigma \setminus \Delta$-structure of A: given $\gamma \in (\Sigma \setminus \Delta)_{w,h}$ with $w \in V^*$ and $h \in H$, $\gamma_F(\bar{d}) = f_A(\gamma_A(\bar{d}))$ for each $\bar{d} \in D_w$. Since all the equations in E_Σ are quantified over data only and since $A \models\!\equiv_\Sigma E_\Sigma$, it follows by Proposition 1 that $F \models\!\equiv_\Sigma E_\Sigma$. This concludes the proof.

Therefore, the carriers of all algebras in the final family of a split hidden specification coincide with the carrier of the final algebra of its destructor subspecification.

Finally, it is worth noting that given a hidden specification (Σ, E), the final family of (Σ, E)-algebras may be empty – this happens precisely when the specification (Σ, E) is *inconsistent*, i.e. there are no (Σ, E)-algebras.

In coalgebraic hidden algebra, cofree algebras provided suitable denotations for hidden specification maps. When specifications comprising both algebraic and coalgebraic structure are considered, the semantics involves *cofree families*.

Theorem 4. *Let $\phi : (H, \Sigma, E) \to (H', \Sigma', E')$ denote a hidden specification map. If each equation in E' contains at most one hidden-sorted variable, then the reduct functor $\mathsf{U}_\phi : \mathsf{Alg}(H', \Sigma', E') \to \mathsf{Alg}(H, \Sigma, E)$ has a righ t multi-adjoint.*

Proof. We let Δ and Δ' denote the destructor subsignatures of Σ and Σ' respectively, and $\phi_\Delta : \Delta \to \Delta'$ denote the restriction of the signature map $\phi : \Sigma \to \Sigma'$ to destructor subsignatures. We fix a (Σ, E)-algebra A and construct a cofree family of (Σ', E')-algebras over A. We let $\bar{A}$ denote the cofree coextension of $A\!\upharpoonright_\Delta$ along ϕ_Δ, with $\epsilon_A : \bar{A}\!\upharpoonright_\Delta \to A\!\upharpoonright_\Delta$ as the associated couniversal arrow.

The proof now follows the same line as the proof of Theorem 2. We consider a category $\mathsf{Alg}(\Sigma', E', A)$ whose objects correspond to (Σ', E')-coextensions of A along ϕ, and use a relation $\sim$ on its objects to partitio n it into subcategories with final objects. These final objects then yield a final family for $\mathsf{Alg}(\Sigma', E', A)$, which at the same time defines a cofree family of (Σ', E')-algebras over A.

$\mathsf{Alg}(\Sigma', E', A)$ is the category whose objects are pairs $\langle A', f \rangle$ with A' a (Σ', E')-algebra and $f : A'\!\upharpoonright_\Sigma \to A$ a Σ-homomorphism, and whose arrows from $\langle A'_1, f_1 \rangle$ to $\langle A'_2, f_2 \rangle$ are Σ'-homomorphisms $g : A'_1 \to A'_2$ such that $\bar{f}_1 = \bar{f}_2 \circ g\!\upharpoonright_{\Delta'}$ (where $\bar{f}_1 : A'_1\!\upharpoonright_{\Delta'} \to \bar{A}$ and $\bar{f}_2 : A'_2\!\upharpoonright_{\Delta'} \to \bar{A}$ denote the unique Δ'-homomorphisms satisfying $\epsilon_A \circ \bar{f}_1\!\upharpoonright_\Delta = f_1\!\upharpoonright_\Delta$, respectively $\epsilon_A \circ \bar{f}_2\!\upharpoonright_\Delta = f_2\!\upharpoonright_\Delta$). Given $\langle A'_1, f_1 \rangle$ and $\langle A'_2, f_2 \rangle$ in $\mathsf{Alg}(\Sigma', E', A)$, $\langle A'_1, f_1 \rangle \sim \langle A'_2, f_2 \rangle$ if and only if there exist $\langle A', f \rangle$ together with $g_1 : \langle A', f \rangle \to \langle A'_1, f_1 \rangle$, $g_2 : \langle A', f \rangle \to \langle A'_2, f_2 \rangle$ in $\mathsf{Alg}(\Sigma', E', A)$. One can easily show that $\mathsf{Alg}(\Sigma', E', A)$ has pullbacks, and therefore $\langle A'_1, f_1 \rangle \sim \langle A'_2, f_2 \rangle$ holds if and only if $\langle A'_1, f_1 \rangle$ and $\langle A'_2, f_2 \rangle$ are connected in $\mathsf{Alg}(\Sigma', E', A)$. Hence, $\sim$ determines a partition $\mathcal{C}$ of $\mathsf{Alg}(\Sigma', E', A)$ into subcategories. Furthermore, each subcategory C in $\mathcal{C}$ has a final object $\langle \bar{A}_\mathsf{C}, \epsilon_{A,\mathsf{C}} \rangle$. Its carriers are given by: $\bar{A}_{\mathsf{C},h} = \{a \in \bar{A}_h \mid a = \bar{f}(a') \text{ for some } \langle A', f \rangle \in |\mathsf{C}| \text{ and } a' \in A'_h\}$ for $h \in H$, with $\bar{f} : A'\!\upharpoonright_{\Delta'} \to \bar{A}$ denoting the unique Δ'-homomorphism satisfying $\epsilon_A \circ \bar{f}\!\upharpoonright_\Delta = f\!\upharpoonright_\Delta$. The Δ'-structure of $\bar{A}_\mathsf{C}$ coincides with the Δ'-structure of $\bar{A}$, while its $\Sigma' \setminus \Delta'$-structure is induced by the $\Sigma' \setminus \Delta'$-structure of (any of) the (Σ', E')-algebras in C. Also, $\bar{A}_\mathsf{C}$ behaviourally satisfies E', since each algebra in C does and since each equation in E' contains at most one hidden-sorted variable. Finally, the Δ-homomorphism $\epsilon_A : \bar{A}\!\upharpoonright_\Delta \to A$ defines a Σ-homomorphism $\epsilon_{A,\mathsf{C}} : \bar{A}_\mathsf{C}\!\upharpoonright_\Sigma \to A$. (The way $\Sigma' \setminus \Delta'$-operation symbols are interpreted in $\bar{A}_\mathsf{C}$ is used to prove this.) Hence, $\langle \bar{A}_\mathsf{C}, \epsilon_{A,\mathsf{C}} \rangle \in |\mathsf{C}|$.

It then follows easily that $\langle \bar{A}_\mathsf{C}, \epsilon_{A,\mathsf{C}} \rangle_{\mathsf{C} \in \mathcal{C}}$ defines a final $\mathsf{Alg}(\Sigma', E', A)$-family, while $(\epsilon_{A,\mathsf{C}} : \bar{A}_\mathsf{C}\!\upharpoonright_\Sigma \to A)_{\mathsf{C} \in \mathcal{C}}$ defines a couniversal family of arrows from U_ϕ to A.

Right multi-adjoints to the reduct functors induced by specification maps satisfying the hypothesis of Theorem 4 provide suitable denotations for specification steps given by such specification maps. Given an algebra A of the source specification, the right multi-adjoint yields a family of algebras of the target specification which coextend A; furthermore, each algebra in this family is *most general*, in that no algebra of the target specification which coextends A strictly extends it.

We conclude this section by noting that initial/free families of hidden algebras also exist (no restriction on the specifications involved is needed in this case). Although initial families do not satisfy properties similar to the ones stated in

Theorem 3, they are relevant for characterising behaviours which are reachable through ground Σ-terms. A consequence of the existence of both initial and final families of hidden specifications is the existence of a partition of the category of hidden algebras of such specifications into subcategories, with each subcategory corresponding to a particular behaviour for the constructor operations, and having an initial as well as a final representative.

4 Semantics with Multi-limits

Algebraic approaches to the specification of data types use colimit constructions to define canonical ways of combining specifications, and free extensions of algebras to define the semantics of combined specifications purely at the model level [10], [11]. In hidden algebra, colimits are used in a similar way at the specification level. However, at the model level the interest is in *coextending* (restricting) collections of behaviours, rather than in *extending* collections of values, and consequently dual constructions should be considered. Multi-limits are an obvious candidate for such constructions, as they define final solutions to categorically formulated constraints. Here we prove the existence of multi-limits in a general category of hidden algebras. This then yields a canonical construction for algebras of combined specifications from algebras of the component specifications.

It is shown in [3] that the standard results regarding the existence of limits (see e.g. [2]) generalise to multi-limits. In particular, existence of finite multi-limits in a category is a consequence of the existence of a final f amily of objects and of multi-pullbacks. We have already seen that final families of (Σ, E)-algebras exist, provided that the equations in E contain at most one hidden-sorted variable. Also, multi-pullbacks exist in $\mathsf{Alg}(\Sigma, E)$, since st andard pullbacks exist (see Proposition 2). Hence, we immediately obtain the following result.

Theorem 5. *Let (Σ, E) denote a hidden specification such that each equation in E contains at most one hidden-sorted variable. Then, the category $\mathsf{Alg}(\Sigma, E)$ has finite multi-limits. Furthermore, if (Σ, E) is a destructor specification, then f inite multi-limits coincide with finite limits.*

Theorem 5 will be used to prove a similar result for a general category Alg, whose objects are hidden algebras and whose arrows correspond to coextension relations between their source and target. One can also consider a subcategory CoAlg of Alg whose objects are hidden algebras of destructor specifications. CoAlg will be shown to have finite limits, while Alg will be shown to have finite multi-limits.

Theorem 6. *Let* Alg *denote the category having:*

- *objects: pairs $\langle P, A\rangle$, with P a hidden specification whose equations contain at most one hidden-sorted variable, and A a P-algebra*
- *arrows from $\langle P', A'\rangle$ to $\langle P, A\rangle$: pairs $\langle \phi, f\rangle$, with $\phi : P \to P'$ a hidden specification map, and $f : A'\restriction_P \to A$ a P-homomorphism.*

Also, let CoAlg *denote the full subcategory of* Alg *whose objects are such that their first component is a destructor specification. Then,* CoAlg *has finite limits, while* Alg *has finite multi-limits.*

Proof (sketch). Existence of finite limits in CoAlg follows from a general result in [6] regarding the existence of limits in the structure category of a fibration. This result states that if both the base category of a fibration and each of its fibres have a certain kind of limits, and if the reindexing functors between the fibres preserve that kind of limits, then the structure category also has those limits; furthermore, the limit of a diagram in the structure category is computed by first computing the limit of the underlying diagram in the base category, then lifting the initial diagram to the fibre over this limit, and finally computing the limit of the resulting diagram in this fibre. All the hypotheses of the result in [6] are satisfied by the fibration $\mathsf{CoSp} : \mathsf{CoAlg} \to \mathsf{Spec}^{\mathsf{op}}$ mapping $\langle P, A\rangle$ to P and $\langle \phi, f\rangle : \langle P', A'\rangle \to \langle P, A\rangle$ to $\phi : P \to P'$. First, the fact that CoSp is a fibration is an immediate consequence of the existence of cofree constructions along destructor specification maps – cartesian liftings are given by the couniversal arrows. Then, the first two hypotheses are guaranteed by Theorem 1 and respectively Proposition 2 together with the existence of final algebras of destructor specifications, while preservation of finite limits by the reindexing functors (the right adjoints to the reduct functors induced by the underlying specification maps) follows from the limit-preservation property of right adjoints.

The functor $\mathsf{Sp} : \mathsf{Alg} \to \mathsf{Spec}^{\mathsf{op}}$ defined similarly to CoSp is not a fibration, since cofree constructions do not exist for arbitrary specification maps. However, one can use a similar strategy to construct multi-limits of diagrams in Alg: given such a diagram $\mathcal{D}$, its multi-limit is obtained by first constructing the limit P of $\mathsf{Sp} \circ \mathcal{D}$ in $\mathsf{Spec}^{\mathsf{op}}$, then cofreely coextending each algebra in $\mathcal{D}$ to a family of P-algebras, and finally computing some multi-limits in $\mathsf{Alg}(P)$. The couniversal property of the multi-limit then follows from the couniversal properties of limits in $\mathsf{Spec}^{\mathsf{op}}$, cofree families, and respectively multi-limits in $\mathsf{Alg}(P)$.

Limits in CoAlg / multi-limits in Alg provide canonical ways of constructing algebras of structured specifications from algebras of the component specifications.

We conclude this section by illustrating the pullback construction in Coalg. Given destructor specification maps $\phi_i : P_0 \to P_i$ with $i = 1, 2$, P_i-algebras A_i with $i = 0, 1, 2$ and P_0-homomorphisms $f_i : A_i\restriction_{P_0} \to A_0$ with $i = 1, 2$ (defining a V-shaped diagram d in Coalg), a pullback for d is obtained by:

1. constructing the pushout $(P, (\phi'_i : P_i \to P)_{i=1,2})$ of ϕ_1, ϕ_2 in Spec
2. cofreely coextending A_0 along $\phi'_1 \circ \phi_1 = \phi'_2 \circ \phi_2$ to A'_0 and A_i along ϕ'_i to A'_i, $i = 1, 2$ (with $\epsilon_i : A'_i\restriction_{P_i} \to A_i$ as couniversal arrows, $i = 0, 1, 2$)
3. taking the pullback of f'_1, f'_2 in $\mathsf{Alg}(P)$, with $f'_i : A'_i \to A'_0$ denoting the unique P'_i-homomorphism satisfying $f_i \circ \epsilon_i\restriction_{P_0} = \epsilon_0 \circ f'_i\restriction_{P_0}$, $i = 1, 2$.

Then, $\langle \phi'_1, f'_1\rangle$, $\langle \phi'_2, f'_2\rangle$ define a pullback for $\langle \phi_1, f_1\rangle$, $\langle \phi_2, f_2\rangle$.

5 Complex Objects in Hidden Algebra

The coalgebraic nature of hidden algebra comes as a consequence of all the operations specified by hidden signatures taking at most one argument of hidden sort. This means that only data values can be passed as arguments to either constructors or destruct ors, becoming particularly restrictive for specifying objects that have other objects as components. This section outlines a way to overcome this problem, with final/cofree families still providing appropriate denotations for the specification techniques involved.

There are at least two alternatives for solving the problem described above. One of them is to simply drop the restriction regarding the presence of at most one argument of hidden sort in operations. Such an approach is taken in [7], where be havioural equivalence is defined only in terms of operations taking one argument of hidden sort, while its preservation by the remaining operations becomes a requirement on algebras. (The satisfaction of this requirement by algebras can either appear as a consequence of the satisfaction of the equations in the specification, or be imposed as a restriction on algebras.)

The other alternative is to retain (a version of) the restriction regarding the arity of operations, by making it *relative* to the already defined object types. Syntactically, this amounts to generalising the importation of the data signature into a hidden signature to the importation of a number of hidden signatures into another hidden signature, with similar restrictions on the kinds of importations allowed. Semantically, this involves fixing implementations for existing specifications befor e importing them into more structured specifications, in the same way in which a data algebra was fixed before importing the data signature into a hidden signature. However, complex objects should only have limited access to their component objects (whose implementation details should be abstracted away). That is, the operations of complex objects should use, for hidden arguments other than the object itself, abstractions (w.r.t behavioural equivalence) of the corresponding hidden carriers, rather than th e carriers themselves.

Then, object specification becomes layered, with the first layer consisting of a specification for the data, the next layer consisting of specifications for simple objects, and the upper layers consisting of specifications of increasingly complex objects, that use the specifications situated at previous layers. This approach has the advantage of preserving the coalgebraic nature of the approach, including the existence of semantic constructions based on finality.

We let $\mathsf{Sign}^{\hookrightarrow}$ denote the category of many-sorted signatures and inclusion signature morphisms. The operations provided by a layered object system are specified using *system signatures*.

Definition 12. *A* **system signature** *is a finite partial order diagram* $\Sigma : \mathsf{I} \to \mathsf{Sign}^{\hookrightarrow}$*, with* $\Sigma(i) = (S_i, \Sigma_i)$*, such that if* $\Sigma(i)$ *includes* $\Sigma(j_1), \ldots, \Sigma(j_n)$*, then: (i)* Σ_i *adds no operation symbols to* $\bigcup \Sigma_{j_k}$ *whose argument/result sorts are all in* $\bigcup S_{j_k}$*, and (ii) the operation symbols of* Σ_i *have at most one argument sort not from* $\bigcup S_{j_k}$*.* $\Sigma_i \setminus \bigcup \Sigma_{j_k}$*-operation symbols having an argument of a sort not from* $\bigcup S_{j_k}$ *are called* Σ_i**-destructor symbols***, while the remaining operation symbols are called* Σ_i**-constructor symbols***.*

Ordinary hidden signatures (H, Σ) over (V, Ψ) then correspond to system signatures $(V, \Psi) \hookrightarrow (V \cup H, \Sigma)$.

An *algebra of a system signature* provides interpretations for the sorts and operation symbols at each layer, by abstracting away the implementation details of sorts from lower layers. *Behavioural equivalence* on such an algebra is defined as indistinguishability by contexts that use operation symbols from the current layer. The only difference w.r.t. the standard notion of behavioural equivalence is that variables of imported sorts are now allowed in contexts, since reachability of data values by constants of the data signature does not generalise to arbitrary layers.

Definition 13. *Given a system signature* $\mathbf{\Sigma}$*, a* **$\mathbf{\Sigma}$-algebra** *is an* I*-indexed family* $(A_i)_{i \in \mathsf{I}}$*, with* A_i *a* Σ_i*-algebra for each* $i \in \mathsf{I}$*, such that:*

1. *If* Σ_i *does not include any other signature, then behavioural equivalence on* A_i *coincides with equality:* $\sim_{A_i,s} = =_{A_{i,s}}$ *for* $s \in S_i$*.*
2. *If* Σ_i *includes* $\Sigma_{j_1}, \ldots, \Sigma_{j_n}$*, and* $s \in S_{j_k}$ *with* $k \in \{1, \ldots, n\}$*, then* $A_{i,s} = A_{j_k,s}/_{\sim_{A_{j_k},s}}$*. Behavioural equivalence on* A_i *is defined as follows:*
   (a) *for* $s \in \bigcup S_{j_k}$*,* $\sim_{A_i,s} = =_{A_{i,s}}$*.*
   (b) *for* $s \in S_i \setminus \bigcup S_{j_k}$ *and* $a, a' \in A_s$*,* $a \sim_{A_i,s} a'$ *if and only if* $c_{A_i}(\overline{x}, a) = c_{A_i}(\overline{x}, a')$ *for each* Σ_i*-context* $c \in T_{\Sigma_i}(X)(\{z\})_{s_l}$*, with* z *an* s *-sorted variable,* X *a* $\bigcup S_{j_k}$*-sorted set of variables, and* $s_l \in S_{j_l}$ *for some* $l \in \{1, \ldots, n\}$*, and each assignment of values* $\overline{x}$ *in* A_i *to the variables in* X*.*

Behavioural equivalence on A *is given by* $(\sim_{A_i})_{i \in \mathsf{I}}$*. Given* $\mathbf{\Sigma}$*-algebras* $A = (A_i)_{i \in \mathsf{I}}$ *and* $B = (B_i)_{i \in \mathsf{I}}$*, a* **$\mathbf{\Sigma}$-homomorphism** *from* A *to* B *is an* I*-indexed mappi ng* $f = (f_i)_{i \in \mathsf{I}}$ *with* $f_i : A_i \to B_i$ *a* Σ_i*-homomorphism for each* $i \in \mathsf{I}$*, such that* $f_i = 1_{A_i}$ *for each* $i \in \mathsf{I}$ *which is not maximal.*

That is, for $i \in \mathsf{I}$, $\sim_{A_i}$ coincides with equality for sorts imported by $\mathbf{\Sigma}(i)$, and with indistinguishability by contexts with variables and result of imported sorts, for sorts that are specific to $\mathbf{\Sigma}(i)$. Also, $\mathbf{\Sigma}$-homomorphisms only relate $\mathbf{\Sigma}$-algebras that coincide on all layers which are not maximal.

It is worth noting that, by considering abstractions of the lower-layer behaviours at the upper-layers, there is no need for additional constraints to ensure that behavioural equivalence is a congruence relation – the upper layers simply can not distinguish between behaviourally equivalent states at lower layers.

We now let $\mathsf{Spec}^{\hookrightarrow}$ denote the category of many-sorted specifications and inclusion specification morphisms, and $\mathsf{U} : \mathsf{Spec}^{\hookrightarrow} \to \mathsf{Sign}^{\hookrightarrow}$ denote the functor taking specifications to their underlying signatures.

Definition 14. *A* **system specification** *is a finite diagram* $\mathsf{P} : \mathsf{I} \to \mathsf{Spec}^{\hookrightarrow}$*, such that* $\mathsf{U} \circ \mathsf{P}$ *is a system signature. For* $i \in \mathsf{I}$*, we let* $\mathsf{P}(i) = (\Sigma_i, E_i)$*.*

A $\mathbf{\Sigma}$*-algebra* A **behaviourally satisfies** *a system specification* P *if and only if* $A_i \models\!\!\!\equiv_{\Sigma_i} e_i$ *for each* $e_i \in E_i$ *and each* $i \in \mathsf{I}$*.*

Ordinary hidden specifications (H, Σ, E) correspond to system specifications $(V, \Psi, E_D) \hookrightarrow (V \cup H, \Sigma, E_D \cup E)$, with $E_D = \{e \mid D \models_\Psi e\}$.

All the techniques for proving behavioural satisfaction (see [1]) apply to system specifications, the only difference being in the use, in contexts, of variables of imported sorts. Furthermore, behavioural satisfaction of an equation at a given layer implies its (standard) satisfaction at the upper layers.

Horizontal specification steps involve extending the module structure, while protecting the existing modules. They correspond to inclusions of diagrams in $\mathsf{Spec}^{\hookrightarrow}$ of the following kind:

$$\begin{tikzcd}
 & P_3 & & & & P_3 & & P_4 \\
P_1 \arrow[ur, hook] & & P_2 \arrow[ul, hook'] & \subseteq & P_1 \arrow[ur, hook] & & P_2 \arrow[ul, hook'] \arrow[ur, hook] & \\
 & \Psi \arrow[ul, hook'] \arrow[ur, hook] & & & & \Psi \arrow[ul, hook'] \arrow[ur, hook] & &
\end{tikzcd}$$

and are formally captured by the notion of *system specification morphism.*

Definition 15. *A* **system specification morphism** *from* $\mathsf{P} : \mathsf{I} \to \mathsf{Spec}^{\hookrightarrow}$ *to* $\mathsf{P}' : \mathsf{I}' \to \mathsf{Spec}^{\hookrightarrow}$ *is an inclusion of partial orders (i.e. an inclusion* $\iota : \mathsf{I} \hookrightarrow \mathsf{I}'$ *such that* $\mathsf{P}'(\iota(i)) = \mathsf{P}(i)$ *for each* $i \in \mathsf{I}$*), with* I *and* I' *additionally satisfying* $\mathsf{I}'(\_, \iota(i)) \simeq \mathsf{I}(\_, i)$ *(with* $\mathsf{I}(\_, i)$ *consisting of all* I*-arrows with codomain* i*).*

Final families of hidden algebras yield suitable denotations for system specification morphisms: given a system specification morphism $\iota : \mathsf{P} \to \mathsf{P}'$ and a P-algebra A, a final family of P'-algebras over A can be obtained b y combining the final families of $\mathsf{P}'(i')$-algebras over A, with $i' \notin \iota(\mathsf{I})$.

Vertical specification steps involve specialising some of the existing modules, and are formally captured by the notion of *system specification map.*

Definition 16. *Given system specifications* $\mathsf{P} : \mathsf{I} \to \mathsf{Spec}^{\hookrightarrow}$ *and* $\mathsf{P}' : \mathsf{I} \to \mathsf{Spec}^{\hookrightarrow}$ *with* $\mathsf{P}(i) = \mathsf{P}'(i)$ *for each* $i \in \mathsf{I}$ *which is not maximal in* I*, a* **system specification map** *from* P *to* P' *is given by a family of many-sorted signature morphisms* $\phi_i : \Sigma_i \to \Sigma'_i$*, with* i *maximal in* I*, such that* $E'_i \models_{\Sigma'_i} \phi_i(c[e_i])$ *for each* $e_i \in E_i$*, each context* c *for* e_i*, and each* i *maximal in* I*.*

That is, only modules which are maximal w.r.t. the partial order structure can be specialised. The following depicts a system specification map:

$$\begin{tikzcd}
 & P_3 \arrow[rrrr, "\phi_3"] & & & & P'_3 & \\
P_1 \arrow[ur, hook] & & P_2 \arrow[ul, hook'] & = & P_1 \arrow[ur, hook] & & P_2 \arrow[ul, hook'] \\
 & \Psi \arrow[ul, hook'] \arrow[ur, hook] & & = & & \Psi \arrow[ul, hook'] \arrow[ur, hook] &
\end{tikzcd}$$

System specification maps $\phi : \mathsf{P} \to \mathsf{P}'$ induce reduct functors $\mathsf{U}_\phi : \mathsf{Alg}(\mathsf{P}') \to \mathsf{Alg}(\mathsf{P})$, mapping $(A'_i)_{i \in \mathsf{I}}$ to $(A'_i \upharpoonright_{\Sigma_i})_{i \in \mathsf{I}}$. Cofree f amilies of algebras w.r.t. the maximal components of ϕ yield suitable denotations for system specification maps: they induce right multi-adjoints to the reduct functors U_ϕ.

One can alternate horizontal and vertical steps to specify arbitrarily complex objects: while horizontal steps correspond to importing existing specifications along protecting inclusion morphisms, vertical steps protect the lower layers of system specifications and specialise the upper-most layers.

We conclude this section by commenting on the relationship between the approach taken here and the one of [7]. In a sense, our approach is less general, as it does not support algebraic operations of arbitrary form (each operation can only take one argument of a non-imported sort). However, the approach here is more suitable for the specification of complex objects. It is often the case that the destructors associated to such objects take other objects as arguments; in this case, the notion of behavioural equivalence should depend on such destructors. While this is captured here (by layering the specification), in [7] it is prevented by the requirement that any operation which takes more than one hidden argument preserves behavioural equivalence, rather than influences it.

6 Conclusions and Future Work

Hidden specifications comprising both algebraic and coalgebraic structure and maps between such specifications have been considered, and final/cofree families of hidden algebras have been shown to provide appropriate denotations for them. A canonical way of combining algebras of component specifications into algebras of structured specifications has also been derived. Finally, an extension of hidden algebra that supports the specification of arbitrarily complex objects, with the semantics still given by final/cofree families, has been presented.

The use of an algebraic syntax together with a coalgebraic semantics restricts the form of constructors/destructors one can specify in hidden algebra. Other ways of combining algebra and coalgebra for objects should also be investigated, possibly by making the separation between their algebraic and coalgebraic aspects more explicit, in order to allow the specification of more general behaviours.

Acknowledgement I would like to thank my supervisor, Dr Grant Malcolm, for his continuous advice and encouragement.

References

1. J. Goguen and G. Malcolm. A hidden agenda. Technical Report CS97-538, UCSD, 1997. 64, 64, 76
2. F. Borceux. *Handbook of Categorical Algebra*, volume I. Cambridge University Press, 1994. 69, 72
3. Y. Diers. Familles universelles de morphismes. *Annales de la Société Scientifique de Bruxelles*, 93(3), 1979. 65, 65, 72
4. B. Jacobs. Objects and classes, coalgebraically. In B. Freitag, C. B. Jones, C. Lengauer, and H.-J. Schek, editors, *Object Orientation with Parallelism and Persistence*. Kluwer Academic Publishers, 1996. 63
5. J. Rutten. Universal coalgebra: a theory of systems. Technical Report CS-R9652, CWI, 1996. 63

6. C. Hermida. On fibred adjunctions and completeness for fibred categories. In H. Ehrig and F. Orejas, editors, *Recent Trends in Data Type Specification*, volume 785 of *LNCS*. Springer, 1993. 73, 73
7. R. Diaconescu. Behavioural coherence in object-oriented algebraic specification. Technical Report IS-RR-98-0017F, Japan Adv. Instit. for Sci. and Tech., 1998. 74, 77, 77
8. C. Cîrstea. Coalgebra semantics for hidden algebra: parameterised objects and inheritance. In F. Parisi-Presicce, editor, *Recent Trends in Algebraic Development Techniques*, volume 1376 of *LNCS*. Springer, 1998. 64, 64, 64, 68, 68
9. J. Goguen, J. Thatcher, and E. Wagner. An initial algebra approach to the specification, correctness, and implementation of abstract data types. In R. Yeh, editor, *Current Trends in Programming Methodology*, volume 4. Prentice-Hall, 1978. 63
10. H. Ehrig and B. Mahr. Fundamentals of algebraic specification 1: Equations and initial semantics. In W. Brauer, G. Rozenberg, and A. Salomaa, editors, *EATCS Monographs on TCS, Volume 6*. Springer, 1985. 72
11. H. Ehrig, M. Baldamus, and F. Orejas. New concepts for amalgamation and extension in the framework of specification logics. In M. Bidoit and C. Choppy, editors, *Recent Trends in Data Type Specification*, volume 655 of *LNCS*. Springer, 1993. 72