Coalgebra Semantics for Hidden Algebra: Parameterised Objects and Inheritance

Corina Cîrstea*

Oxford University Computing Laboratory
Wolfson Building, Parks Road, Oxford OX1 3QD, UK

Abstract. The theory of hidden algebras combines standard algebraic techniques with coalgebraic techniques to provide a semantic foundation for the object paradigm. This paper focuses on the coalgebraic aspect of hidden algebra, concerned with signatures of destructors at the syntactic level and with finality and cofree constructions at the semantic level. Our main result shows the existence of cofree constructions induced by maps between coalgebraic hidden specifications. Their use in giving a semantics to parameterised objects and inheritance is then illustrated. The cofreeness result for hidden algebra is generalised to abstract coalgebra and a universal construction for building object systems over existing subsystems is obtained. Finally, existence of final/cofree constructions for arbitrary hidden specifications is discussed.

1 Introduction

Algebraic techniques have been intensively studied over the last decades. Their suitability for the specification of data types is due to the availability of effective definition and proof techniques based on induction. Recent work on coalgebras (the formal duals of algebras) [Rei95, Jac95, Jac96, Rut96, Jac97, JR97] suggests their suitability for the specification of dynamical systems. The theory of coalgebras provides a notion of observational indistinguishability as *bisimulation*, a characterisation of abstract behaviours as elements of *final coalgebras* and *coinduction* as a definition/proof principle for system behaviour.

Hidden algebra, introduced in [GD94] and further developed in [MG94, GM97] combines algebraic and coalgebraic techniques in order to provide a semantic foundation for the object paradigm. It is an extension of the theory of many sorted algebras that uses both *constructor* and *destructor* operations and a loose behavioural semantics over a fixed data universe for (the states of) objects. Its coalgebraic nature, emerging from the observational character of the approach, has already been exploited in [MG94] where (coinductive) proof techniques for behavioural satisfaction were developed. The present paper further investigates the relationship between hidden algebra and coalgebra, focusing on the semantic level and in particular on cofree constructions. Their suitability as semantics for the specification techniques used in hidden algebra is emphasised.

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

The structure of the paper is as follows. Section 2 gives a brief account of the theory of coalgebras as well as an outline of hidden algebra. Section 3 focuses on the coalgebraic aspect of hidden algebra: hidden algebras are mapped to coalgebras (by forgetting the constructors) in such a way that behavioural congruences correspond to bisimulation equivalences on the associated coalgebras. Consequently, coinduction can be used both as a definition principle for object behaviour and as a proof principle for behavioural equivalence. Existence of *final algebras* for coalgebraic hidden specifications is also obtained. The main result of the paper concerns the existence of *cofree hidden algebras* induced by maps between *coalgebraic* (destructor) hidden specifications. Such maps correspond to reusing specifications either horizontally by importation or vertically by refinement. In certain cases, the cofree construction corresponds to a reuse of implementations along the underlying reuse of specifications. A generalisation of a cofreeness result in [Rut96], concerning the existence of cofree object systems over given subsystems is sketched in the last part of Section 3. Section 4 illustrates the use of cofree constructions in giving semantics to the importation of coalgebraic hidden modules, parameterised modules and inheritance. Cofree constructions provide canonical ways to build implementations for more structured/specialised specifications from implementations of the specifications they are built on. Section 5 generalises the final/cofree coextension semantics in Section 3 by considering arbitrary hidden specifications. In this case, the semantics is given by final/cofree *families* of hidden algebras. Section 6 summarises the results presented and briefly outlines future work.

2 Preliminaries

This section gives an account of the basic ideas and concepts in coalgebraic specifications, emphasising their duality to algebraic specifications. A brief introduction to hidden algebra (a combination of algebraic and coalgebraic techniques intended as a specification framework for objects) is also given.

2.1 Algebra and Coalgebra

Algebra and its associated inductive techniques have been successfully used for the specification of data types. The emphasis there is on how the values of a data type are generated, using constructor operations going into the type. Data types are presented as **F-algebras**, i.e. tuples (A, α), with A an object and $\alpha : \mathsf{F}A \to A$ a morphism in some category C, with $\mathsf{F} : \mathsf{C} \to \mathsf{C}$. Among F-algebras, **initial** ones $\iota : \mathsf{F}I \to I$ (least fixed points of F) are most relevant – their elements denote closed programs. Initial algebras come equipped with an **induction principle** stating that no proper subalgebras exist for initial algebras. This principle constitutes the main technique used in algebraic specifications for both definitions and proofs: defining a function on the initial algebra by induction amounts to defining its values on all the constructors; and proving that two functions on the initial algebra coincide amounts to showing that they agree on all the constructors. **Free constructions** are also relevant for data types: they provide least extensions of algebras of a data type to algebras of another and have been used to give semantics to parameterised data types, see e.g. [EM85].

The theory of coalgebras [Rut96, JR97], having its roots in automata theory and transition system theory [Rut95] and concerned with dynamical systems, can be viewed as a dualisation of the theory of algebras. Object systems are coalgebraically defined by specifying how their states can be observed, using destructor operations going out of the object types. Object types appear as G-coalgebras, i.e. tuples (C, β), with C an object and $\beta : C \rightarrow \mathsf{G}C$ a morphism in some category C, with $\mathsf{G} : \mathsf{C} \rightarrow \mathsf{C}$. **Final** G-coalgebras $\zeta : Z \rightarrow \mathsf{G}Z$ (greatest fixed points of G) are in this case relevant – they incorporate all G-behaviours. The unique coalgebra homomorphism from a coalgebra to the final one maps object states to their behaviour. A **bisimulation** between two coalgebras is a relation on their carriers, carrying itself coalgebraic structure. Bisimulations relate states that exhibit the same behaviour. Final coalgebras come equipped with a **coinduction principle** stating that no proper bisimulations exist between a final coalgebra and itself; that is, two elements of a final coalgebra having the same behaviour coincide. Coinduction can be used both in definitions, to define functions into the final coalgebra by giving coalgebraic structure to their domains, and in proofs, to show equality of two elements of the final coalgebra by exhibiting a bisimulation that relates them. Finally, **cofree constructions** are relevant for object types as they provide least restrictive (co)extensions of coalgebras of an object type to coalgebras of another.

2.2 Hidden Algebra

This section provides an outline of hidden algebra. For a detailed presentation of the approach the reader is referred to [GM97].

Hidden algebra extends many sorted algebra to support the specification of objects with hidden states, only accessible through specified interfaces. The fundamental distinction between data values and object states is reflected in the use of visible sorts/operations with standard semantics for data and of hidden sorts/operations with loose behavioural semantics for objects.

A fixed data universe, given by an algebra D (the **data algebra**) of a many sorted signature (V, Ψ) (the **data signature**) is assumed, with the additional constraint that each element of D is named by a constant in Ψ. For convenience, we take $D_v \subseteq \Psi_{[],v}$ for each $v \in V$.

Definition 1. A **(hidden) signature over** (V, Ψ, D) 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 $w \in V^*$, $v \in V$ and (ii) **monadicity:** for $\sigma \in \Sigma_{w,s}$, at most one sort appearing in w (by convention, the first one) is hidden.

$\Sigma \setminus \Psi$-operations having exactly one hidden-sorted argument are called **destructors**, while those having only visible-sorted arguments are called **constructors**.

Definition 2. 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 $\phi\lceil_{(V,\Psi)} = id_{(V,\Psi)}$ and $\phi(H) \subseteq H'$. A **(hidden) signature morphism** is a hidden signature map such that if $\sigma' \in \Sigma'_{h'w',s'}$ with $h' \in \phi(H)$, then $\sigma' = \phi(\sigma)$ for some $\sigma \in \Sigma$.

Signature maps specify arbitrary (vertical) structure, while signature morphisms specify horizontal structure (importation of hidden modules). Imported hidden sorts are protected by signature morphisms, in that no new destructor operations are added for them by the target signature.

Definition 3. A **(hidden) Σ-algebra** is a many sorted $(V \cup H, \Sigma)$-algebra A such that $A\lceil_{\Psi} = D$. A **(hidden) Σ-homomorphism** between Σ-algebras A and B is a many sorted Σ-homomorphism $f : A \to B$ such that $f_v = id_{D_v}$ for $v \in V$. Σ-algebras and Σ-homomorphisms form a category $\mathsf{HAlg}(\Sigma)$. Hidden signature maps $\phi : \Sigma \to \Sigma'$ induce reduct functors $\mathsf{U}_\phi : \mathsf{HAlg}(\Sigma') \to \mathsf{HAlg}(\Sigma)$.

Hidden algebra takes a behavioural approach to objects: their states can only be observed through experiments; indistinguishability of states by experiments is captured by behavioural equivalence.

Definition 4. Given a 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 defined by: $a \sim_{A,s} a'$ iff $c_A[a] = c_A[a']$ for all contexts c for s, with $s \in V \cup H$ and $a, a' \in A_s$.

Satisfaction of equations is also behavioural – one only requires the two sides of an equation to look the same under any observation rather than coincide.

Definition 5. A **(hidden) specification** is a triple (H, Σ, E) with (H, Σ) a hidden signature and E a set of Σ-equations. A Σ-algebra A **behaviourally satisfies** a (conditional) 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 $\theta : X \to A$, $\bar{\theta}(l) \sim_A \bar{\theta}(r)$ whenever $\bar{\theta}(l_i) \sim_A \bar{\theta}(r_i)$, $i = 1, \ldots, n$. Given sets E and E' of Σ-equations, we write $E \models\!\equiv_\Sigma E'$ if $A \models\!\equiv_\Sigma E$ implies $A \models\!\equiv_\Sigma E'$ for any Σ-algebra A.

[MG94] gives a characterisation of behavioural equivalence as greatest *behavioural congruence* (congruence which coincides with equality on visible sorts) and uses it to obtain a coinductive-like proof technique for behavioural equivalence.

We restrict our attention to specifications whose equations have visible-sorted conditions only. To each such specification (Σ, E) one can associate another specification $(\Sigma, \overline{E})$ (by letting $\overline{E} = \{c[e] \mid e \in E,\ c \in T_\Sigma[z]$ appropriate for $e\}$), such that $A \models\!\equiv_\Sigma E$ iff $A \models\!\equiv_\Sigma \overline{E}$ iff $A \models_\Sigma \overline{E}$.

Definition 6. Let (Σ, E) and (Σ', E') be hidden specifications. A hidden signature map $\phi : \Sigma \to \Sigma'$ defines a **specification map** $\phi : (\Sigma, E) \to (\Sigma', E')$ if and only if $E' \models\!\equiv_{\Sigma'} \phi(\overline{E})$. A specification map whose underlying signature map is a signature morphism is called a **specification morphism**.

Given a specification map $\phi : (\Sigma, E) \to (\Sigma', E')$, the functor U_ϕ induced by $\phi : \Sigma \to \Sigma'$ maps hidden (Σ', E')-algebras to hidden (Σ, E)-algebras.

Theorem 7. *The category* Spec *of hidden specifications and specification maps is finitely cocomplete. Pushouts in* Spec *preserve specification morphisms.*

We note in passing that the constraint on hidden signature morphisms is used in [GD94] to obtain an institution of hidden algebras. Moreover, specification morphisms $\phi : (\Sigma, E) \to (\Sigma', E')$ satisfy $E' \models_{\Sigma'} \phi(E)$, i.e. they are the theory morphisms of this institution. A different institution may be obtained by considering hidden signature maps and a slightly different notion of sentence, given by a Σ-equation together with a subsignature of Σ for the contexts under which the equation is expected to hold. This is the institution that underlies our treatment of parameterisation in Section 4.1.

3 Coalgebra and Hidden Algebra

This section focuses on the coalgebraic nature of hidden algebra. First we illustrate how viewing hidden algebras as coalgebras provides both a characterisation of abstract behaviours by means of final coalgebras and a coalgebraic definition of behavioural equivalence as greatest bisimulation. Next, we prove the existence of cofree constructions induced by maps between coalgebraic hidden specifications. Such constructions provide canonical ways to (co)extend algebras along specification maps by restricting the behaviour as little as possible. Finally, we present a generalisation of a result in [Rut96] concerned with cofree object systems over given subsystems.

3.1 Basic Results

A closer look at the definition of behavioural equivalence reveals that only destructor operations are relevant. Hence, in investigating the coalgebraic aspect of hidden algebra we can restrict our attention to signatures of destructors.

Definition 8. A hidden signature Σ is a **coalgebraic/destructor signature** if all $\Sigma \setminus \Psi$-operations are destructors.

Proposition 9. *Let Δ be the destructor subsignature of Σ. Then Σ-behavioural equivalence is the greatest behavioural Δ-congruence.*

Proof. By monadicity together with the data algebra being fixed.

Proposition 10. *For a coalgebraic signature Δ,* $\mathsf{HAlg}(\Delta) \simeq \mathsf{G}_\Delta\text{-}\mathsf{Coalg}$, *where*

$$\mathsf{G}_\Delta : \mathsf{Set}^H \to \mathsf{Set}^H, \quad \mathsf{G}_\Delta(X)_h = \prod_{\delta \in \Delta_{hw,s}} X_s^{D_w}, \quad h \in H \ \ (\text{with } X_v = D_v \text{ if } v \in V)$$

Proof. Δ-algebras A correspond to G_Δ-coalgebras $\alpha : C \to \mathsf{G}_\Delta C$ with $C_h = A_h$ for $h \in H$ and α_h mapping $a \in A_h$ and $\delta \in \Delta_{hw,s}$ to $\delta_A(a, \_) : D_w \to A_s$. Also, Δ-homomorphisms $f : A \to A'$ correspond to G_Δ-homomorphisms $g : C \to C'$ with $g_h = f_h$ for $h \in H$. Moreover, the above is a one-to-one correspondence.

Corollary 11. *There exists a final Δ-algebra F_Δ, having hidden carriers:*

$$F_{\Delta,h} = \prod_{v \in V} [L_\Delta[z_h]_v \to D_v], \quad h \in H$$

(with $L_\Delta[z_h]$ consisting of "local" Δ-contexts for sort h, i.e. contexts containing only one occurrence of the hidden variable) and Δ-operations:

- $\delta_{F_\Delta}((s_v)_{v\in V}, \bar{d}) = s_{v'}(\delta(z_h, \bar{d}))$ *for* $\delta \in \Delta_{hw,v'}$
- $\delta_{F_\Delta}((s_v)_{v\in V}, \bar{d}) = (s'_v)_{v \in V}$ *with* $s'_v(c) = s_v(c[\delta(z_h, \bar{d})])$ *for* $\delta \in \Delta_{hw,h'}$

Moreover, behavioural equivalence on a Δ-algebra coincides with bisimilarity on its associated coalgebra.

The elements of F_Δ correspond to *abstract behaviours* (functions mapping experiments to data values); the unique homomorphism from an arbitrary Δ-algebra to F_Δ maps hidden states to their behaviour; two hidden states are behaviourally equivalent if and only if they are mapped to the same element of F_Δ.

We note that signature maps $\phi : (H, \Delta) \to (H', \Delta')$ induce natural transformations $\eta : \mathsf{U} \circ \mathsf{G}_{\Delta'} \Rightarrow \mathsf{G}_\Delta \circ \mathsf{U}$ (where $\mathsf{U} : \mathsf{Set}^{H'} \to \mathsf{Set}^H$ is the reindexing functor induced by $\phi : H \to H'$) given by: $(\eta_X)_h((f_{\delta'})_{\delta' \in \Delta'_{\phi(h)w',s'}}) = (f_{\phi(\delta)})_{\delta \in \Delta_{hw,s}}$ for $f_{\delta'} \in X_s^{D_w}$, $h \in H$. This observation will be used in Section 3.3.

In algebraic specifications, equations induce relations on the carriers of algebras and quotients of algebras by *least congruences* containing such relations are of interest. Dually, in coalgebra one is interested in *greatest invariants* (subcoalgebras) contained in given predicates on the carriers of coalgebras [Jac97]. Such predicates can be specified in hidden algebra using *state equations*, i.e. equations in one hidden variable – the induced predicates consist of those states for which the equations are behaviourally satisfied.

Definition 12. A hidden specification (H, Δ, E) is **coalgebraic** if (H, Δ) is coalgebraic and all the equations in E are state equations.

3.2 Cofree Coextensions

In algebraic specifications, free constructions provide least extensions of algebras along morphisms between data type specifications. Dually, in coalgebraic specifications *cofree constructions* are of interest – they provide least restrictions of coalgebras along maps between coalgebraic hidden specifications.

Given categories C and D and a functor $\mathsf{U} : \mathsf{D} \to \mathsf{C}$, a **cofree construction** w.r.t. U on a C-object C consists of a D-object C^* and a C-morphism $\epsilon_C : \mathsf{U}C^* \to C$ which is *couniversal*: given any D-object D and C-morphism $f : \mathsf{U}D \to C$, there exists a unique D-morphism $\bar{f} : D \to C^*$ such that $\epsilon_C \circ \mathsf{U}\bar{f} = f$. If C^* and ϵ_C exist for each C-object C, the mapping $C \mapsto C^*$ extends to a functor $\mathsf{F} : \mathsf{C} \to \mathsf{D}$ in such a way that the C-morphisms ϵ_C define a natural transformation $\epsilon : \mathsf{U} \circ \mathsf{F} \to \mathsf{Id}_{\mathsf{C}}$. Moreover, F is a right adjoint to U with counit ϵ.

This section proves the existence of cofree hidden algebras w.r.t. forgetful functors induced by coalgebraic specification maps. [Rut96] formulates a similar result in an abstract setting where C and D are categories of coalgebras of endofunctors on **Set** and U is induced by a natural transformation between such endofunctors. Here we extend this result to the case when the underlying categories of C and D are distinct. This extension appears as a generalisation of the cofreeness result for hidden algebra and provides a canonical way of building structured systems over existing subsystems.

The cofree construction for hidden algebra dualises, to a certain extent, the free construction [TWW82] for many sorted algebra. When cofreely coextending a (Δ, E)-algebra A along a specification map $\phi : (\Delta, E) \to (\Delta', E')$, instead of using the elements of A to *generate* the elements of a Δ'-algebra, one views them as information that can be *extracted* from elements of a Δ'-algebra (finality replaces initiality). Also, quotienting by least congruences is replaced by taking greatest invariants. The construction amounts to:

1. first, building the final algebra F'_A of an enriched signature Δ'_A containing destructor operations that give A-states as result
2. next, taking the greatest Δ'_A-invariant of F'_A for which the above destructors agree with the Δ-structure of A
3. finally, taking the greatest Δ'_A-invariant induced by the equations $\overline{E'}$.

2 ensures that the Δ-reduct of the cofree coextension has a Δ-homomorphism into A, 1 ensures that the cofree coextension is final among all Δ'-algebras having this property, while 3 ensures behavioural satisfaction of E'.

Theorem 13 Cofreeness. *Let $\phi : (\Delta, E) \to (\Delta', E')$ be a coalgebraic specification map. The reduct functor $\mathsf{U}_\phi : \mathsf{HAlg}(\Delta', E') \to \mathsf{HAlg}(\Delta, E)$ has a right adjoint $\mathsf{C}_\phi : \mathsf{HAlg}(\Delta, E) \to \mathsf{HAlg}(\Delta', E')$.*

Proof. We first define the action of C_ϕ on objects. Let $A \models_\Delta E$. In order to temporarily view the A-states as data, a visible sort $\bar{h}$ is added to Ψ for each $h \in H$, resulting in a data signature $\Psi^\oplus$; also, operations $s_h : h \to \bar{h}$ and $s_h : \phi(h) \to \bar{h}$ are added to Δ and Δ' respectively, resulting in signatures $\Delta^\oplus$ and $\Delta'^\oplus$ with inclusions $\iota_A : (\Psi^\oplus, \Delta, D_A) \hookrightarrow (\Psi^\oplus, \Delta^\oplus, D_A)$ and $\iota'_A : (\Psi^\oplus, \Delta', D_A) \hookrightarrow (\Psi^\oplus, \Delta'^\oplus, D_A)$ (where D_A denotes the extension of D to a $\Psi^\oplus$-algebra interpreting each $\bar{h}$ as A_h). Then $\phi : (\Psi, \Delta, D) \to (\Psi, \Delta', D)$ extends to $\phi_A : (\Psi^\oplus, \Delta^\oplus, D_A) \to (\Psi^\oplus, \Delta'^\oplus, D_A)$ by letting $\phi_A\restriction_\Delta = \phi$, $\phi_A(s_h) = s_h$ for each $h \in H$.

Now let F_A and F'_A be the final $(\Psi^\oplus, \Delta^\oplus, D_A)$- and $(\Psi^\oplus, \Delta'^\oplus, D_A)$-algebras. A can also be made into a $(\Psi^\oplus, \Delta^\oplus, D_A)$-algebra by defining $(s_h)_A$ as id_{A_h}. By finality, there exist unique $(\Psi^\oplus, \Delta^\oplus, D_A)$-homomorphisms $g : \mathsf{U}_{\phi_A} F'_A \to F_A$ and $l : A \to F_A$. Moreover, l faithfully embeds A into F_A: $l_h(a_1) = l_h(a_2) \Rightarrow$

$(s_h)_{F_A}(l_h(a_1)) = (s_h)_{F_A}(l_h(a_2)) \Rightarrow (s_h)_A(a_1) = (s_h)_A(a_2) \Rightarrow a_1 = a_2$. Define $\mathsf{C}_\phi A$ to be the greatest $(\Psi^\oplus, \Delta'^\oplus, D_A)$-invariant of F'_A such that $g\lceil_{\mathsf{U}_{\phi_A}\mathsf{C}_\phi A}$ factors through l and such that $\mathsf{U}_{\iota'_A}\mathsf{C}_\phi A \models_{\Delta'} \overline{E'}$.

$$\begin{tikzcd}
\mathsf{C}_\phi A \arrow[d, hook] & \mathsf{U}_{\phi_A}\mathsf{C}_\phi A \arrow[d, hook] \arrow[drr, dashed, "\epsilon_A"] & & \\
F'_A & \mathsf{U}_{\phi_A}F'_A \arrow[r, "g"'] & F_A & A \arrow[l, hook', "l"]
\end{tikzcd}$$

The action of C_ϕ on a (Δ, E)-homomorphism $f : A \to B$ is defined as follows. First, f is used to make A and F_A into $(\Psi^\oplus, \Delta^\oplus, D_B)$-algebras and F'_A into a $(\Psi^\oplus, \Delta'^\oplus, D_B)$-algebra. Finality of F_B and F'_B gives unique $(\Psi^\oplus, \Delta^\oplus, D_B)$- and $(\Psi^\oplus, \Delta'^\oplus, D_B)$-homomorphisms $! : F_A \to F_B$ and $!' : F'_A \to F'_B$. It then follows by maximality of $\mathsf{C}_\phi B$ that $!'\lceil_{\mathsf{C}_\phi A}$ factors through the inclusion of $\mathsf{C}_\phi B$ into F'_B (since $g'\lceil_{\mathsf{U}_{\phi_B}!'(\mathsf{C}_\phi A)}$ factors through l' and $\mathsf{U}_{\iota'_B}!'(\mathsf{C}_\phi A) \models_{\Delta'} \overline{E'}$). Hence, $\mathsf{C}_\phi f$ can be defined as $!'\lceil_{\mathsf{C}_\phi A}$.

$$\begin{tikzcd}
\mathsf{C}_\phi A \arrow[r, hook] \arrow[d, dashed, "\mathsf{C}_\phi f"'] & F'_A \arrow[d, "!'"] & \mathsf{U}_{\phi_B}F'_A \arrow[r, "g"] \arrow[d, "\mathsf{U}_{\phi_B}!'"'] \arrow[dr, phantom, "="] & F_A \arrow[d, "!"'] \arrow[dr, phantom, "="] & A \arrow[l, "l"'] \arrow[d, "f"] \\
\mathsf{C}_\phi B \arrow[r, hook] & F'_B & \mathsf{U}_{\phi_B}F'_B \arrow[r, "g'"] & F_B & B \arrow[l, "l'"']
\end{tikzcd}$$

It is straightforward to check that C_ϕ is a functor.

Lemma 14 Adjunction. *C_ϕ is right adjoint to U_ϕ.*

Proof. For $A \models_\Delta E$, the A-component ϵ_A of the counit $\epsilon : \mathsf{U}_\phi \circ \mathsf{C}_\phi \Rightarrow \mathsf{Id}$ is the unique factorisation of $g\lceil_{\mathsf{U}_{\phi_A}\mathsf{C}_\phi A}$ through l (recall that l is faithful). Hence, ϵ_A is a Δ-homomorphism.

It remains to prove couniversality of ϵ_A. Given $B \models_{\Delta'} E'$, the unique extension of a Δ-homomorphism $f : \mathsf{U}_\phi B \to A$ to a Δ'-homomorphism $\bar{f} : B \to \mathsf{C}_\phi A$ is obtained by first using f to make B into a $(\Psi^\oplus, \Delta'^\oplus, D_A)$-algebra (with unique $(\Psi^\oplus, \Delta'^\oplus, D_A)$-homomorphism $f' : B \to F'_A$) and then observing that uniqueness of $(\Psi^\oplus, \Delta^\oplus, D_A)$-homomorphisms into F_A gives $(\mathsf{U}_{\phi_A} f'); g = f; l$, which implies that $g\lceil_{\mathsf{U}_{\phi_A}Im(f')}$ factors through l; also, $\mathsf{U}_{\iota'_A}Im(f') \models_{\Delta'} \overline{E'}$, since $B \models_{\Delta'} \overline{E'}$. Hence, by maximality of $\mathsf{C}_\phi A$, $Im(f')$ is a $(\Psi^\oplus, \Delta'^\oplus, D_A)$-invariant of $\mathsf{C}_\phi A$ and $\bar{f} : B \to \mathsf{C}_\phi A$ can be defined as f'. Then $f = (\mathsf{U}_\phi \bar{f}); \epsilon_A$ follows by uniqueness of $(\Psi^\oplus, \Delta^\oplus, D_A)$-homomorphisms into F_A. Also, uniqueness of $\bar{f}$ follows from uniqueness of $(\Psi^\oplus, \Delta'^\oplus, D_A)$-homomorphisms into a subalgebra of the final $(\Psi^\oplus, \Delta'^\oplus, D_A)$-algebra.

Theorem 13 now follows from Lemma 14.

Remark. [Jac96] presents a cofreeness result for categories of *behaviour coalgebras*. Objects of such a category $\mathsf{G\text{-}BCoalg}$ are coalgebras of an endofunctor $\mathsf{G} : \mathsf{Set} \to \mathsf{Set}$, while morphisms between them are given by functions that only commute with the coalgebra structure up to bisimulation. Because of this weaker notion of morphism, an isomorphism class in G-BCoalg is given by an isomorphism class in Set together with a function into the carrier of the final G-coalgebra. The cofree construction is also set-theoretic: given functors $\mathsf{G}, \mathsf{H} : \mathsf{Set} \to \mathsf{Set}$ together with a natural transformation $\eta : \mathsf{H} \Rightarrow \mathsf{G}$ (inducing a forgetful functor U_η : H-BCoalg $\to$ G-BCoalg), the right adjoint R_η : G-BCoalg $\to$ H-BCoalg to U_η is, up to isomorphism, determined by a pullback in Set: if $B \in$ G-BCoalg with $b : B \to F_{\mathsf{G}}$ as unique G-homomorphism into the final G-coalgebra, then $\mathsf{R}_\eta B$ is determined, up to isomorphism, by the pullback in Set of b along the unique G-homomorphism $! : \mathsf{U}_\eta F_{\mathsf{H}} \to F_{\mathsf{G}}$, while the counit is obtained by pulling back ! along b. The inclusion of categories G-Coalg $\hookrightarrow$ G-BCoalg preserves final objects, hence the two cofree constructions are isomorphic in G-BCoalg. The advantage of the construction in [Jac96] over the standard construction stands in reducing the number of bisimilar states (while still implementing the same behaviour). Moreover, the construction in [Jac96] supports the reuse of implementations (the G-structure of B is used in defining the G-structure of its cofree coextension). With our construction, this only happens for Δ-algebras that are extensional (behavioural equivalence is equality), case in which the two constructions coincide.

3.3 A Generalisation

In [Rut96], categories of coalgebras of arbitrary endofunctors $\mathsf{T}, \mathsf{S} : \mathsf{Set} \to \mathsf{Set}$ and forgetful functors U_η : S-Coalg $\to$ T-Coalg induced by natural transformations $\eta : \mathsf{S} \Rightarrow \mathsf{T}$ are considered (U_η maps an S-coalgebra $\gamma : C \to \mathsf{S}C$ to the T-coalgebra $\eta_C \circ \gamma : C \to \mathsf{T}C$) and existence of cofree coalgebras w.r.t. U_η is proved, under the assumption that for any set C, the endofunctor $\mathsf{S} \times C$ on Set (mapping a set X to the set $\mathsf{S}X \times C$) has a final coalgebra.

In the case of one-sorted specifications with no equations, our result can be viewed as an instance of the result in [Rut96] – according to a remark in Section 3.1, the signature map underlying ϕ induces a natural transformation between the endofunctors associated to Δ' and Δ. But our result also applies to specification maps whose underlying signature maps are not surjective on hidden sorts, suggesting a generalisation of the result in [Rut96] to the case when the categories underlying S and T are distinct. This generalisation involves a functor U between these categories and a natural transformation $\eta : \mathsf{U} \circ \mathsf{S} \Rightarrow \mathsf{T} \circ \mathsf{U}$. Existence of a cofree functor w.r.t. U_η is proved under similar assumptions.

Theorem 15. *Let* C *and* D *be categories with binary products and* $\mathsf{U} : \mathsf{D} \to \mathsf{C}$ *be a functor that preserves binary products and has a right adjoint right inverse* R. *Let* $\mathsf{T} : \mathsf{C} \to \mathsf{C}$, $\mathsf{S} : \mathsf{D} \to \mathsf{D}$ *be endofunctors and* $\eta : \mathsf{U} \circ \mathsf{S} \Rightarrow \mathsf{T} \circ \mathsf{U}$ *be a natural transformation (inducing a forgetful functor* U_η : S-Coalg $\to$ T-Coalg*). If the functors* $\mathsf{S} \times \mathsf{R}C$ *and* $\mathsf{T} \times C$ *have final coalgebras for any* C*-object* C*, then* U_η *has a right adjoint* C_η.

Proof. U_η maps an S-coalgebra $\gamma : D \to \mathsf{S}D$ to the T-coalgebra $\mathsf{U}\gamma;\eta_D : \mathsf{U}D \to \mathsf{TU}D$ (a T-subsystem $\mathsf{U}\gamma;\eta_D$ is extracted from the S-system γ). A canonical way to build S-systems over T-subsystems is given by the functor C_η, defined on a T-coalgebra $\gamma : C \to \mathsf{T}C$ as follows.

1. Let $\delta : F \to \mathsf{T}F \times C$ be the final $\mathsf{T} \times C$-coalgebra.
2. Let $! : \langle\gamma, id\rangle \to \delta$ be the unique $\mathsf{T} \times C$-homomorphism of $\langle\gamma, id\rangle$ into δ.
3. Let $\delta' : F' \to \mathsf{S}F' \times \mathsf{R}C$ be the final $\mathsf{S} \times \mathsf{R}C$-coalgebra. Then $\langle\eta_{F'}, id\rangle \circ \mathsf{U}\delta'$ is a $\mathsf{T} \times C$-coalgebra with $!' : \langle\eta_{F'}, id\rangle \circ \mathsf{U}\delta' \to \delta$ as unique $\mathsf{T} \times C$-homomorphism into δ.
4. Let $\gamma' : C' \to \mathsf{S}C' \times \mathsf{R}C$ be the greatest $\mathsf{S} \times \mathsf{R}C$-invariant of δ' such that $!' \restriction_{\mathsf{U}C'}$ factors through $!$ in $(\mathsf{T} \times C)$-Coalg and let $\epsilon_C : \mathsf{U}C' \to C$ be the unique factorisation (as $!$ is monic). Define $\mathsf{C}_\eta\gamma$ as $\pi_1 \circ \gamma'$.

The construction is illustrated in the diagram below.

$$\begin{tikzcd}
C' \arrow[r, "\gamma'"] \arrow[d, hook] & \mathsf{S}C' \times \mathsf{R}C \arrow[r, "\pi_1"] \arrow[d, hook] & \mathsf{S}C' \arrow[d, hook] \\
F' \arrow[r, "\delta'"] & \mathsf{S}F' \times \mathsf{R}C \arrow[r, "\pi_1"] & \mathsf{S}F'
\end{tikzcd}$$

$$\begin{tikzcd}
\mathsf{U}C' \arrow[r, "\mathsf{U}\gamma'"] \arrow[d, hook] \arrow[ddd, bend right, "\epsilon_C"'] & \mathsf{U}\mathsf{S}C' \times C \arrow[r, "\langle\eta_{C'}, id\rangle"] \arrow[d, hook] & \mathsf{T}\mathsf{U}C' \times C \arrow[r, "\pi_1"] \arrow[d, hook] & \mathsf{T}\mathsf{U}C' \arrow[ddd, "\mathsf{T}\epsilon_C"] \\
\mathsf{U}F' \arrow[r, "\mathsf{U}\delta'"] \arrow[d, "!'"] & \mathsf{U}\mathsf{S}F' \times C \arrow[r, "\langle\eta_{F'}, id\rangle"] & \mathsf{T}\mathsf{U}F' \times C \arrow[d] & \\
F \arrow[rr, "\delta"] & & \mathsf{T}F \times C & \\
C \arrow[u, "!"] \arrow[rr, "\langle\gamma, id\rangle"] & & \mathsf{T}C \times C \arrow[u] \arrow[r, "\pi_1"] & \mathsf{T}C
\end{tikzcd}$$

Then, C_η is right adjoint to U_η with counit ϵ: any T-homomorphism $f : \mathsf{U}_\eta\tau \to \gamma$ with $\tau : D \to \mathsf{S}D$ an S-coalgebra induces a $\mathsf{S} \times \mathsf{R}C$-structure on D such that f becomes a $\mathsf{T} \times C$-homomorphism. Uniqueness of $\mathsf{T} \times C$-homomorphisms into F together with maximality of γ' are then used to define an S-homomorphism $\bar{f} : \tau \to \mathsf{C}_\eta\gamma$ such that $\mathsf{U}_\eta\bar{f}; \epsilon_C = f$, in the same way as this was done in Theorem 13.

Remark. By letting $\mathsf{C} = \mathsf{Set}^H$, $\mathsf{D} = \mathsf{Set}^{H'}$, $\mathsf{R} : \mathsf{Set}^H \to \mathsf{Set}^{H'}$ with $(\mathsf{R}A)_{h'} = \prod_{h'=\phi(h)} A_h$, $\mathsf{T} = \mathsf{G}_\Delta$, $\mathsf{S} = \mathsf{G}_{\Delta'}$ and $\eta : \mathsf{U} \circ \mathsf{S} \Rightarrow \mathsf{T} \circ \mathsf{U}$ as in Section 3.1, we obtain Theorem 13 for the case when $E = E' = \emptyset$.

4 Semantics by Cofree Constructions

In this section, cofree functors are used to give semantics to parameterisation and inheritance in coalgebraic hidden algebra.

4.1 Parameterisation

Cofree functors C_ϕ induced by specification morphisms $\phi : P \to T$ provide an appropriate semantics for the importation of coalgebraic hidden modules: supplied with a P-algebra A, the cofree construction provides the most general T-algebra that exhibits the P-behaviour of A. A theory of parameterised modules with cofree constructions as semantics can be developed for coalgebraic hidden algebra in the same style as this was done for data types [EM85] using free constructions. Moreover, a semantic characterisation of correctness of parameter passing in terms of persistence of the cofree functors can be given.

Definition 16. A **coalgebraic parameterised specification** is a specification morphism $\phi : P \hookrightarrow T$ with both P and T coalgebraic. A **parameter passing morphism** for ϕ is a specification map $\psi : P \to P'$ with P' coalgebraic. The **instantiation** of P with ψ in T is given by the pushout (**parameter passing diagram**) $\phi' : P' \to T'$, $\psi' : T \to T'$ of $\phi : P \to T$, $\psi : P \to P'$ in Spec.

The semantics of parameter passing diagrams is given by pairs $(\mathsf{C}_\phi, \mathsf{C}_{\phi'})$ of cofree functors induced by the specification morphisms ϕ and ϕ' (see Theorem 7). As in the case of parameterised data types, correctness of parameter passing is defined by requiring (i) the protection of the actual parameter in the result specification and (ii) that the semantics of ϕ' extends the semantics of ϕ. However, the actual conditions we use are stronger than (the duals of) the ones in [EM85], because there, any P-algebra could be viewed as an initial P'-algebra for some P', whereas in our case, due to the data signature being fixed, not any P-algebra is isomorphic to a final P'-algebra.

Definition 17. Given a parameter passing diagram as above, parameter passing is **correct w.r.t.** ψ if and only if (i) $\mathsf{U}_{\phi'} \circ \mathsf{C}_{\phi'} \simeq \mathsf{Id}$, and (ii) $\mathsf{C}_\phi \circ \mathsf{U}_\psi \simeq \mathsf{U}_{\psi'} \circ \mathsf{C}_{\phi'}$. Parameter passing is **correct** if and only if it is correct w.r.t. any ψ.

Standard compositionality results use **amalgamations** to define the semantics of combined specifications purely on the semantic level [EM85]. Existence of amalgamations in hidden algebra amounts to pushouts in Spec being transformed by the functor $\mathsf{HAlg} : \mathsf{Spec} \to \mathsf{Cat}^{\mathsf{op}}$ into pullbacks in $\mathsf{Cat}^{\mathsf{op}}$.

Lemma 18. *Hidden algebra has amalgamations.*

Proof. By pushouts in Spec being pushouts of the underlying many sorted specifications, together with many sorted amalgamations preserving hidden algebras.

Definition 19. A parameterised specification ϕ is **persistent** if and only if C_ϕ is persistent ($\mathsf{U}_\phi \circ \mathsf{C}_\phi \simeq \mathsf{Id}$).

Lemma 20. *Given a parameter passing diagram as above, if ϕ is persistent then ϕ' is persistent.*

Proof. A consequence of amalgamations being pullbacks is that the functor $\mathsf{Id} \oplus_{\mathsf{U}_\psi} (\mathsf{C}_\phi \circ \mathsf{U}_\psi) : \mathsf{HAlg}(P') \to \mathsf{HAlg}(T')$ (with $\oplus$ denoting amalgamation) is right adjoint to $\mathsf{U}_{\phi'}$ with identity as counit. The conclusion then follows by any two right adjoints being naturally isomorphic.

Theorem 21. *Parameter passing is correct for ϕ if and only if ϕ is persistent.*

Proof. If ϕ is persistent then, by Lemma 20, ϕ' is persistent, hence (i) of Definition 17 holds. (ii) follows from $\mathsf{C}_{\phi'}$ being isomorphic to $\mathsf{Id} \oplus_{\mathsf{U}_\psi} (\mathsf{C}_\phi \circ \mathsf{U}_\psi)$, which gives $\mathsf{U}_{\psi'} \circ \mathsf{C}_{\phi'} \simeq \mathsf{C}_\phi \circ \mathsf{U}_\psi$. The converse follows by taking ψ the identity.

Example 1 Channels. Channels consisting of a sender and a receiver can be specified by parameterising the receiver by the sender. A sender is simply a stream that uses its `send` method to send values `vals`. An alternating sender is a sender that alternates the values it sends. A receiver receives values from a sender `sen` using its `rec` method and stores them in `valr`. The pushout semantics of instantiating `REC` with `ASEN` is a specification denoted `REC[ASEN]` which consists of `REC` together with the equation for alternating streams.

```
obj SEN is pr NAT .
  sort Sen .
  op vals : Sen -> Nat .
  op send : Sen -> Sen .
endo
obj ASEN is using SEN .
  var S : Sen .
  eq vals(send(send(S)) = vals(S) .
endo
```

```
th REC[X :: SEN] is
  sort Rec .
  op valr : Rec -> Nat .
  op sen : Rec -> Sen .
  op rec : Rec -> Rec .
  var R : Rec .
  eq sen(rec(R)) = send(sen(R)) .
  eq valr(rec(R))=vals(sen(R)) .
endth
```

Now consider a `SEN`-algebra `A` implementing alternating streams: $\mathtt{Sen_A} = \mathbb{N} \times \mathbb{N}$, $\mathtt{vals_A(n_1,n_2)} = \mathtt{n_1}$, $\mathtt{send_A(n_1,n_2)} = \mathtt{(n_2,n_1)}$. In constructing its cofree coextension $\mathtt{A}^*$ along $\mathtt{SEN} \hookrightarrow \mathtt{REC}$ we follow the three steps outlined in Section 3.2. First, we build the final $\mathtt{REC} \cup \{\mathtt{s} : \mathtt{Sen} \to \mathtt{Sen_A}\}$-algebra `A1`, having carriers $\mathtt{Sen_{A1}} = \{\mathtt{f} \mid \mathtt{f} : \{\mathtt{send}\}^* \to \mathbb{N} \times \mathtt{Sen_A}\}$, $\mathtt{Rec_{A1}} = \{(\mathtt{g},\mathtt{h}) \mid \mathtt{g} : \{\mathtt{rec}\}^* \to \mathbb{N}, \mathtt{h} : \{\mathtt{rec}\}^*\mathtt{sen}\{\mathtt{send}\}^* \to \mathbb{N} \times \mathtt{Sen_A}\}$. A sender state `f` assigns a sender value and a $\mathtt{Sen_A}$-state to each experiment consisting of a finite number of `send`s. Similarly, a receiver state `(g,h)` assigns a receiver value to each experiment consisting of a finite number of `rec`s, as well as a sender value and a $\mathtt{Sen_A}$-state to each experiment consisting of a finite number of `rec`s followed by `sen` and then by a finite number of `send`s. Second, the greatest subalgebra of `A1` for which examining the $\mathtt{Sen_A}$-state commutes with the `SEN`-operations is taken, resulting in a `REC`-algebra `A2` having carriers $\mathtt{Sen_{A2}} = \mathtt{Sen_{A1}}$ (the second component of `f` on the empty sequence of `send`s uniquely determines `f`) and $\mathtt{Rec_{A2}} = \{(\mathtt{g},\mathtt{h}) \mid \mathtt{g} : \{\mathtt{rec}\}^* \to \mathbb{N}, \mathtt{h} : \{\mathtt{rec}\}^* \to \mathtt{Sen_A}\}$. Finally, imposing the `REC`-equations results in a `REC`-algebra $\mathtt{A}^*$ having carriers: $\mathtt{Sen_{A^*}} = \mathtt{Sen_A}$, $\mathtt{Rec_{A^*}} = \mathbb{N} \times \mathtt{Sen_A}$ (the values of `g` and `h` on the empty sequence of `rec`s uniquely determine `g` and `h`) and operations: $\mathtt{vals_{A^*}} = \mathtt{vals_A}$, $\mathtt{send_{A^*}} = \mathtt{send_A}$, $\mathtt{valr_{A^*}(n,n_1,n_2)} = \mathtt{n}$, $\mathtt{sen_{A^*}(n,n_1,n_2)} = \mathtt{(n_1,n_2)}$, $\mathtt{rec_{A^*}(n,n_1,n_2)} = \mathtt{(n_1,n_2,n_1)}$. `A*` uses the implementation provided by `A` for its sender part.

4.2 Inheritance

Class inheritance (with non-monotonic overriding) can be specified in hidden algebra using (partial) specification maps. Here we use a specification of bank accounts to emphasise the suitability of cofree constructions as a semantics for inheritance.

Example 2 Bank Accounts. Bank accounts `ACC` are specified using a `bal`(ance) attribute and methods for `dep`(ositing)/`with`(drawing) a given amount. More specialised accounts – a history account that maintains a `his`(tory) of the transactions made into the account and a savings account from which withdrawals are only allowed if the account is not in saving state – are then derived from `ACC`. The former specialisation corresponds to inheritance with monotonic overriding, while the latter non-monotonically overrides the `with` method[2].

```
obj ACCSIG is pr INT .
  sort Acc .
  op bal : Acc -> Int .
  ops dep, with : Acc Nat -> Acc .
endo
obj ACC is pr ACCSIG .
  var N : Nat .
  var A : Acc .
  eq bal(dep(A,N)) = bal(A) + N .
  eq bal(with(A,N)) = bal(A) - N .
endo
obj HACC is pr LIST[INT] .
  ex ACC * (sort Acc to HAcc) .
  op his : HAcc -> List .
  var N : Nat .
  var H : HAcc .
  *** monotonic overriding
  eq his(dep(H,N)) = N;his(H) .
  eq his(with(H,N)) = (-N);his(H) .
endo
```

```
obj SACC is
  ex ACCSIG * (sort Acc to SAcc) .
  op start, end : SAcc -> SAcc .
  op sav? : SAcc -> Bool .
  var N : Nat .
  var S : SAcc .
  *** monotonic overriding
  eq bal(dep(S,N)) = bal(S) + N .
  eq sav?(dep(S,N)) = sav?(S) .
  *** non-monotonic overriding
  ceq bal(with(S,N)) = bal(S) - N
   if sav?(S) == false .
  ceq bal(with(S,N)) = bal(S)
   if sav?(S) == true .
  eq sav?(with(S,N)) = sav?(S) .
  eq bal(start(S)) = bal(S) .
  eq sav?(start(S)) = true .
  eq bal(end(S)) = bal(S) .
  eq sav?(end(S)) = false .
endo
```

The semantics of the inheritance relation between `HACC` and `ACC` is given by the cofree functor induced by the specialisation of `ACC` to `HACC`. For the inheritance relation between `SACC` and `ACC`, the semantics is given by the composition of the forgetful functor induced by hiding the non-monotonically overridden operation `with` with the cofree functor induced by the specialisation of `ACC` without the `with` method to `SACC`.

[2] In general, only *defined* operations should be non-monotonically overridden. Given a coalgebraic specification (Δ, E), the operations in $\Delta' \subseteq \Delta$ are **defined** if in any (Δ, E)-algebra, behavioural $\Delta \setminus \Delta'$-equivalence is a Δ-congruence. A similar approach is taken in [Jac96], where in addition to a "core" part, a class specification may contain "definable" functions which do not contribute to the meaning of the specification and can therefore be arbitrarily overridden.

Now consider an ACC-algebra A given by: $\mathtt{Acc}_\mathtt{A}$ = Int, $\mathtt{bal}_\mathtt{A}$(I) = I, $\mathtt{dep}_\mathtt{A}$(I, J) = I + J, $\mathtt{with}_\mathtt{A}$(I, J) = I − J. Its cofree coextensions to a HACC-algebra HA and a SACC-algebra SA are given below.

```
HAcc_HA = Acc_A × IntList
bal_HA(I,L) = I
his_HA(I,L) = L
dep_HA((I,L),J) = (I + J, J;L)
with_HA((I,L),J) = (I − J,(−J);L)
```

```
SAcc_SA = Acc_A × {true,false}
bal_SA(I,B) = I
sav?_SA(I,B) = B
dep_SA((I,B),J) = (I + J,B)
with_SA((I,false),J) = (I − J,false)
with_SA((I,true),J) = (I,true)
start_SA(I,B) = (I,true)
end_SA(I,B) = (I,false)
```

The counit of the adjunction provides coercion operations that map states in the subclass to states in the superclass. In both of the above cases, the coercions are projections extracting the superclass attributes. Also in both cases, the superclass implementation is reused by the subclass.

5 Combining Algebra with Coalgebra

We have illustrated the relevance of final/cofree constructions to coalgebraic hidden specifications and maps between them. Not surprisingly, the existence of final/cofree hidden algebras does not generalise to arbitrary hidden specifications – there is no universal way of interpreting the constructors in either a final or a cofree algebra. However, final/cofree *families* of hidden algebras exist.

The notion of *final family of objects* generalises the notion of final object: given a category C, a family $(F_j)_{j \in J}$ of C-objects is **final** if and only if, for any C-object C, there exist unique $j \in J$ and C-morphism $f : C \to F_j$. Similarly, the notion of *couniversal family of morphisms* [Die79] generalises the notion of couniversal morphism: given a functor $\mathsf{U} : \mathsf{D} \to \mathsf{C}$ and a C-object C, a family of C-morphisms $\epsilon_{C,j} : \mathsf{U}C^*_j \to C$ with C^*_j an object of D for each $j \in J$ is a **couniversal family of morphisms from U to C** if and only for any D-object D and C-morphism $f : \mathsf{U}D \to C$, there exist unique $j \in J$ and D-morphism $\bar{f} : D \to C^*_j$ such that $U\bar{f};\epsilon_{C,j} = f$. If for every C there exists a couniversal family of morphisms from U into C, then U_ϕ is said to have a **right multiadjoint**.

Now let Σ denote a hidden signature with $\Sigma = \Gamma \cup \Delta$ as splitting into hidden subsignatures of constructors and destructors respectively and observe that signature maps preserve such splittings. Also, let F_Δ denote the final hidden Δ-algebra and I_Γ denote the initial hidden Γ-algebra (given by the free many sorted Γ-algebra over D). Finally, let $\mathsf{Set}_D^{V \cup H}$ denote the category of $V \cup H$-sorted sets with $(D_v)_{v \in V}$ as V-components and $V \cup H$-sorted functions with $(id_v)_{v \in V}$ as V-components.

Theorem 22. *For any hidden signature Σ there exists a final family of hidden Σ-algebras.*

Proof. Let $I, F \in \mathsf{Set}_D^{V \cup H}$ be the carriers of I_Γ and F_Δ respectively and let $J = \{j \mid j : I \to F \text{ in } \mathsf{Set}_D^{V \cup H}\}$. Each $j \in J$ uniquely induces a Σ-structure F_j on F such that $F_j\restriction_\Delta = F_\Delta$ and such that j defines a Γ-homomorphism from I_Γ to $F_j\restriction_\Gamma$. Then $(F_j)_{j \in J}$ is a final family of hidden Σ-algebras.

Therefore, the category of hidden Σ-algebras can be sliced into subcategories C_j, $j \in J$, with each C_j having a final object F_j. This justifies using the family $(F_j)_{j \in J}$ as final-like semantics for Σ.

Theorem 23. *Let $\phi : \Sigma \to \Sigma'$ be a hidden signature map. The functor U_ϕ : $\mathsf{HAlg}(\Sigma') \to \mathsf{HAlg}(\Sigma)$ has a right multiadjoint.*

Proof. Let $\phi_\Delta : \Delta \to \Delta'$ denote the restriction of ϕ to destructor subsignatures. For a hidden Σ-algebra A, let $(A\restriction_\Delta)^*$ denote the cofree coextension of $A\restriction_\Delta$ along ϕ_Δ and let J_A denote the family of Σ'-algebras A_j^* such that $A_j^*\restriction_{\Delta'} = (A\restriction_\Delta)^*$ and such that the function underlying $\epsilon_{A\restriction_\Delta} : \mathsf{U}_{\phi_\Delta}(A\restriction_\Delta)^* \to A\restriction_\Delta$ defines a Σ-homomorphism $\epsilon_{A,j} : \mathsf{U}_\phi A_j^* \to A$. Then, the family $(\epsilon_{A,j})_{j \in J_A}$ is a couniversal family of morphisms from U_ϕ to A.

Theorems 22 and 23 can be extended from hidden signatures to *split* hidden specifications. A hidden specification (Σ, E) is called **split** if and only if $E = E_\Delta \cup E_\Sigma$ with E_Δ consisting of state Δ-equations and E_Σ consisting of Σ-equations with visible-sorted variables only. Final families of (Σ, E)-algebras exist for any split specification (Σ, E) – the sub-family $J' \subseteq J$ consisting only of those F_js which behaviourally satisfy E is considered. Also, if (Σ, E) and (Σ', E') are split hidden specifications and $\phi : (\Sigma, E) \to (\Sigma', E')$ is a specification map such that $\phi\restriction_{(\Delta, E_\Delta)}: (\Delta, E_\Delta) \to (\Delta', E'_{\Delta'})$ is also a specification map, then U_ϕ has a right multiadjoint – for each Σ-algebra A, the sub-family $J'_A \subseteq J_A$ consisting only of those Σ'-algebras A_j^* which behaviourally satisfy E' is considered.

6 Conclusions and Future Work

We have investigated the coalgebraic nature of hidden algebra, concentrating on semantical aspects such as finality and cofree constructions. We have proved the existence of cofree hidden algebras along maps between coalgebraic hidden specifications and emphasised their relevance in giving semantics to parameterisation and inheritance. Also, we have sketched a possible generalisation of a cofreeness result from [Rut96]. Finally, the final/cofree semantics has been lifted from coalgebraic to arbitrary hidden algebra.

With the current definition of hidden signatures, hidden constants (operations from visible sorts to hidden sorts) are the only constructor operations allowed. In practice however, new objects can be created by putting together existing objects (e.g. by tupling), suggesting a generalisation of the theory of hidden algebras that allows arbitrary constructors. One expects to still be able to reason coalgebraically about behavioural equivalence, hence Proposition 9 must hold for generalised hidden signatures (preservation of Δ-behavioural equivalence by constructors can be achieved either by imposing it as a constraint on algebras or by fully specifying the Δ-behaviour of the constructors). The extension of the results in this paper to generalised hidden algebra remains to be studied.

The integration of the algebraic and coalgebraic aspects of hidden algebra also deserves further study, perhaps along the lines of [Mal96] where objects are viewed as *algebra-coalgebra pairs*, or [TP97] where a similar notion called *bi-algebra* is considered.

Acknowledgements I would like to thank my supervisor, Dr Grant Malcolm, for his guidance and his comments on several drafts of this paper.

References

[Die79] Y. Diers. Familles universelles de morphismes. *Annales de la Société Scientifique de Bruxelles*, 93(3):175–195, 1979.

[EM85] H. Ehrig and B. Mahr. Fundamentals of algebraic specification 1: Equations and initial semantics. In *EATCS Monographs on TCS*. Springer, 1985.

[GD94] J. Goguen and R. Diaconescu. Towards an algebraic semantics for the object paradigm. In H. Ehrig and F. Orejas, editors, *Recent Trends in Data Type Specification*, number 785 in LNCS. Springer, 1994.

[GM97] J. Goguen and G. Malcolm. A hidden agenda. to appear, 1997.

[Jac95] B. Jacobs. Mongruences and cofree coalgebras. In V.S. Alagar and M. Nivat, editors, *Algebraic Methods and Software Technology*, number 936 in LNCS. Springer, 1995.

[Jac96] B. Jacobs. Inheritance and cofree constructions. In P. Cointe, editor, *European Conference on Object-Oriented Programming*, number 1098 in LNCS. Springer, 1996.

[Jac97] B. Jacobs. Invariants, bisimulations and the correctness of coalgebraic refinements. Technical Report CSI-R9704, University of Nijmegen, 1997.

[JR97] B. Jacobs and J. Rutten. A tutorial on (co)algebras and (co)induction. *Bulletin of the EATCS*, 62:222–259, 1997.

[Mal96] G. Malcolm. Behavioural equivalence, bisimilarity and minimal realisation. In M. Haveraaen, O. Owe, and O.-J. Dahl, editors, *Recent Trends in Data Type Specifications*, number 1130 in LNCS. Springer, 1996.

[MG94] G. Malcolm and J. Goguen. Proving correctness of refinement and implementation. Technical Monograph PRG-114, Oxford University, 1994.

[Rei95] H. Reichel. An approach to object semantics based on terminal coalgebras. *Mathematical Structures in Computer Science*, 5, 1995.

[Rut95] J. Rutten. A calculus of transition systems (towards universal coalgebra). Technical Report CS-R9503, CWI, 1995.

[Rut96] J. Rutten. Universal coalgebra: a theory of systems. Technical Report CS-R9652, CWI, 1996.

[TP97] D. Turi and G. Plotkin. Towards a mathematical operational semantics. In *Proceedings LICS*, 1997.

[TWW82] J. Thatcher, E. Wagner, and J. Wright. Data type specification: Parameterization and the power of specification techniques. *ACM Transactions on Programming Languages and Systems*, 4(4), 1982.