Calculating invariants as coreflexive bisimulations

Luís S. Barbosa[1], José N. Oliveira[1], and Alexandra M. Silva

[1] Dep. Informática, Universidade do Minho, 4700-320 Braga, Portugal,
[2] Centrum voor Wiskunde en Informatica (CWI), Kruislaan 413, NL-1098 SJ Amsterdam

Abstract. Invariants, bisimulations and assertions are the main ingredients of coalgebra theory applied to computer systems engineering. In this paper we reduce the first to a particular case of the second and show how both together pave the way to a theory of coalgebras which regards invariant predicates as types. An outcome of such a theory is a calculus of invariants' proof obligation discharge, a fragment of which is presented in the paper.

The approach has two main ingredients: one is that of adopting relations as "first class citizens" in a pointfree reasoning style; the other lies on a synergy found between a relational construct, Reynolds' *relation on functions* involved in the *abstraction theorem* on parametric polymorphism and the coalgebraic account of bisimulation and invariants.

In this process, we provide an elegant proof of the equivalence between two different definitions of bisimulation found in coalgebra literature (due to B. Jacobs and Aczel & Mendler, respectively) and their instantiation to the classical Park-Milner definition popular in process algebra.

Keywords: coalgebraic reasoning; proof obligations; pointfree transform; program calculation.

1 Introduction

The widest application of computer systems in our days is that of supporting business, the most dynamic aspect of modern society. This means that computer systems are subject to the permanent stress of ever changing business rules, which materialize into (either static or dynamic) properties of the underlying code.

Computer scientists regard business rules as examples of invariant properties. The word "invariant" captures the idea that such desirable properties are to be maintained *invariant*, that is, unharmed across all transactions which are embodied in the system's functionality.

Changing business rules has a price: the code needs to be upgraded so as to ensure that changes are properly taken into account. How does one check this? Testing is the most widely used technique for this purpose. But (as is well known) it does not ensure correctness and it is highly costly. Ideally, one should be able to *formally verify* that the new invariants are enforced without running the (new) code at all.

This calls for a general theory of invariant preservation upon which one could base such an extended static checking mechanism. And this theory requires a broad view of computer systems able to take into account data persistence and continued interaction.

Coalgebra theory, widely acknowledged as the *mathematics of state-based systems* [20], provides an adequate modeling framework for such systems. The basic insight in coalgebraic modelling is that of representing state-based systems by functions of type

$$p : X \longrightarrow \mathsf{F}\,X \tag{1}$$

which, for every state $x \in X$, describe the observable effects of an elementary step in the evolution of the system (*i.e.*, a state transition). The possible outcomes of such steps are captured by notation $\mathsf{F}\,X$, where functor F acts as a *shape* for the system's interface.

Jacobs [10] identifies three cornerstones in the theory of coalgebras: *invariants*, *bisimilarity* and *assertions*. About the first he writes: *an important aspect of formally establishing the safety of systems is to prove that certain crucial predicates are actually invariants.*

In this paper we develop a theory of invariant preservation whose novelty resides in explicitly expressing invariants as *bisimulations*. (See section 3 and its follow up.) The third cornerstone, assertions, is addressed in section 6. Altogether, there is a calculational flavour in the way we state and prove facts, which stems from the explicit use of relational techniques, a body of knowledge often referred to as the *algebra of programming* [7].

Our starting point is Jacobs definition of an invariant for a given coalgebra [10]:

Definition 1. *Let* $\mathsf{F} : Sets \to Sets$ *be a polynomial functor. An **invariant** for a coalgebra* $c : X \to \mathsf{F}(X)$ *is a predicate* $P \subseteq X$ *satisfying for all* $x \in X$,

$$x \in P \Rightarrow c(x) \in Pred(\mathsf{F})(P). \tag{2}$$

$Pred(\mathsf{F})(P)$ stands for the *lifting* of predicate P via functor F. (We will spell out the meaning of this construct very soon.)

Our approach will be to reason about (2) via the PF-transform [16] — a transformation of first order predicate formulæ into *pointfree* binary relation formulæ [7] which will enable us to blend the concept of invariant with that of bisimulation in a quite handy way. (In fact, we will show the former is a particular case of the latter.) By *pointfree* we mean formulæ which are free of quantifiers and variables (points) such as x above.

Structure of the paper. The paper starts by PF-transforming definition (2), in section 2, to conclude that invariants are a special case of bisimulations (section 3). Section 4 recasts bisimulations in terms of Reynolds' arrow combinator and resorts to its calculational power to provide elegant proofs of equivalence between three most common definitions of bisimulation. The development of (a fragment of) the theory of invariants is pursued in section 5, upon a category of "predicates as types". Moving on, section 6 illustrates how the approach proposed in this paper can be also of use to reason about modal assertions over coalgebras. Finally, section 7 concludes and gives pointers to related and future work.

2   Invariants PF-transformed

Our first step is to convert definition (2) into a binary relational formula. The principle is that of PF-transforming universally quantified formulæ by applying, from right to

left, the definition of relational inclusion which follows,

$$R \subseteq S \;\equiv\; \langle \forall\, y, x \;::\; y\, R\, x \Rightarrow y\, S\, x \rangle \tag{3}$$

for R, S two binary relations [3]. In the case of (2), this means that R will have to capture set (predicate) P and S will have to do the same for set $Pred(\mathsf{F})(P)$. One of the standard ways of encoding a set X as a binary relation is as follows: one defines a relation Φ_X such that

$$y\, \Phi_X\, x \;\equiv\; y = x \,\wedge\, x \in X \tag{4}$$

Relations of this kind are referred to as *coreflexives* because they are fragments of the identity relation id: $\Phi_X \subseteq id$. For instance, set $\{1, 2, 3\}$ is captured by relation $\Phi_{\{1,2,3\}} = \{(1,1), (2,2), (3,3)\}$. We also need to define the binary relation composition operator

$$b(R \cdot S)c \equiv \langle \exists\, a \;::\; b\, R\, a \,\wedge\, a\, S\, c \rangle \tag{5}$$

(read $R \cdot S$ as "R after S") and to assert a rule which will prove convenient,

$$(f\, b)R(g\, a) \equiv b(f^\circ \cdot R \cdot g)a \tag{6}$$

where f and g are functions and $\_^\circ$ denotes relational converse:

$$a(R^\circ)b \equiv b\, R\, a \tag{7}$$

In this context, we reason:

$$\langle \forall\, x \;::\; x \in P \;\Rightarrow\; c(x) \in\, Pred(\mathsf{F})(P) \rangle$$

$\equiv \qquad \{ \; \forall\text{-one point rule} \; \}$

$$\langle \forall\, y, x \;:\; y = x :\; x \in P \;\Rightarrow\; c(y) = c(x) \,\wedge\, c(x) \in\, Pred(\mathsf{F})(P) \rangle$$

$\equiv \qquad \{ \; \forall\text{-trading} \; \}$

$$\langle \forall\, y, x \;::\; y = x \,\wedge\, x \in P \;\Rightarrow\; c(y) = c(x) \,\wedge\, c(x) \in\, Pred(\mathsf{F})(P) \rangle$$

$\equiv \qquad \{ \; (4) \text{ twice} \; \}$

$$\langle \forall\, y, x \;::\; y\, \Phi_P\, x \;\Rightarrow\; c(y)\, \Phi_{Pred(\mathsf{F})(P)}\, c(x) \rangle$$

$\equiv \qquad \{ \; \text{rule (6)} \; \}$

$$\langle \forall\, y, x \;::\; y\, \Phi_P\, x \;\Rightarrow\; y(c^\circ \cdot \Phi_{Pred(\mathsf{F})(P)} \cdot c)x \rangle$$

$\equiv \qquad \{ \; \text{rule (3)} \; \}$

$$\Phi_P\, \subseteq c^\circ \cdot \Phi_{Pred(\mathsf{F})(P)} \cdot c \tag{8}$$

---

[3] By $y\, R\, x$ we mean the fact that pair (y, x) belongs to R. (Similarly for $y\, S\, x$.)

Predicate $Pred(\mathsf{F})(P)$ is defined in [10] (Def. 4.1.1) by induction on the structure of polynomial F. Regarding F as a relator [4] and representing P by its coreflexive Φ_P, $Pred(\mathsf{F})(P)$ coincides with relation $\mathsf{F}\,\Phi_P$. Thus we resume to (8) and calculate further:

$$\Phi_P \ \subseteq c^\circ \cdot \mathsf{F}\,\Phi_P \cdot c$$

$$\equiv \qquad \{\ \text{see (10) below}\ \}$$

$$c \cdot \Phi_P \ \subseteq \mathsf{F}\,\Phi_P \cdot c \tag{9}$$

where the last step is justified by the first of the following laws of the relational calculus,

$$f \cdot R \subseteq S \ \equiv \ R \subseteq f^\circ \cdot S \tag{10}$$

$$R \cdot f^\circ \subseteq S \ \equiv \ R \subseteq S \cdot f \tag{11}$$

known as the *shunting rules* [7] [5]. Note that coalgebra c complies with this rule because it is a function.

Altogether, we arrive at (9), a quite compact version of (2). The sections which follow will give evidence of the advantages of such a transformation.

3   Invariants are bisimulations

We move on to the second cornerstone of coalgebra theory — bisimilarity. This is based on the concept of *bisimulation* which is given by Jacobs [10] as follows:

Definition 2. *A bisimulation for coalgebras $c : X \rightarrow \mathsf{F}(X)$ and $d : Y \rightarrow \mathsf{F}(Y)$ is a relation $R \subseteq X \times Y$ which is* closed under c and d:

$$(x, y) \in R \Rightarrow (c(x), d(y)) \in Rel(\mathsf{F})(R). \tag{12}$$

for all $x \in X$ and $y \in Y$.

(a)   (b)

This time $Rel(\mathsf{F})(R)$ stands for the relational *lifting* of R via functor F which, in our relational setting, is captured by notation $\mathsf{F}\,R$.

An exercise at all similar to the one carried out in the previous section will show (12) PF-transformed into

$$c \cdot R \ \subseteq \mathsf{F}\,R \cdot d \tag{13}$$

(cf. diagram (a) above left) where X and Y are the carriers of coalgebras c and d, respectively. By comparing (13) against (9) we conclude that invariants are *special* cases of bisimulations: exactly those which are coreflexive relations, cf. diagram (b).

---

[4] The concept of a *relator* [5] extends that of a *functor* to relations: $\mathsf{F}\,A$ describes a parametric type while $\mathsf{F}\,R$ is a relation from $\mathsf{F}\,A$ to $\mathsf{F}\,B$ provided R is a relation from A to B. Relators are monotone and commute with composition, converse and the identity.

[5] Functions are denoted by lowercase characters (eg. f, g, ϕ) and function application will be abbreviated by juxtaposition, eg. $f\,a$ instead of $f(a)$.

We will see briefly that this conclusion brings about its benefits, as much of the theory of coalgebraic invariants stems directly from that of bisimulations [6]. We will address this one first.

4 Calculating bisimulations

Let us first show how the classical definition of bisimulation used in process algebra (due to Milner and Park [17]) can be retrieved from (13) simply by instantiating F to the powerset relator $\mathcal{P}X = \{S \mid S \subseteq X\}$. We need the universal property of the *power-transpose* isomorphism Λ

$$f = \Lambda R \equiv R = \in \cdot f \tag{14}$$

which converts binary relations to set-valued functions [7, 16], where $A \xleftarrow{\in} \mathcal{P}A$ is the membership relation. In [7] the powerset relator is defined by

$$\mathcal{P}R = (\in \backslash (R \cdot \in)) \cap (\in \backslash (R^\circ \cdot \in))^\circ \tag{15}$$

where $\cap$ denotes relation intersection and $R \backslash S$ denotes relational division,

$$a(R \backslash S)c \equiv \langle \forall b \,:\, b\,R\,a \,:\, b\,S\,c \rangle$$

a relational operator whose semantics is captured by universal property

$$R \cdot X \subseteq S \equiv X \subseteq R \backslash S \tag{16}$$

The main ingredient of the calculation below is (14), which ensures that every powerset coalgebra uniquely determines a binary relation on its carrier (Λ is a bijection). In this context, let R be a bisimulation between two powerset coalgebras ΛS and ΛU. We reason:

$$(\Lambda S) \cdot R \subseteq (\mathcal{P}R) \cdot (\Lambda U)$$

$\equiv$ { unfolding $\mathcal{P}R$ (15) }

$$(\Lambda S) \cdot R \subseteq (\in \backslash (R \cdot \in)) \cap (\in \backslash (R^\circ \cdot \in))^\circ \cdot (\Lambda U)$$

$\equiv$ { distribution (since ΛU is a function) thanks to (11) }

$$(\Lambda S) \cdot R \subseteq (\in \backslash (R \cdot \in)) \cdot (\Lambda U) \,\wedge\, (\Lambda S) \cdot R \subseteq (\in \backslash (R^\circ \cdot \in))^\circ \cdot (\Lambda U)$$

$\equiv$ { property $R \backslash (S \cdot f) = (R \backslash S) \cdot f$; converses }

$$(\Lambda S) \cdot R \subseteq \in \backslash (R \cdot \in \cdot \Lambda U) \,\wedge\, R^\circ \cdot (\Lambda S)^\circ \subseteq (\Lambda U)^\circ \cdot (\in \backslash (R^\circ \cdot \in))$$

$\equiv$ { shunting rules (10,11) and property above }

$$(\Lambda S) \cdot R \subseteq \in \backslash (R \cdot \in \cdot \Lambda U) \,\wedge\, (\Lambda U) \cdot R^\circ \subseteq \in \backslash (R^\circ \cdot \in \cdot \Lambda S)$$

---

[6] It is interesting to note that Lemma 4.2.2 in [10] proves that relation $\{(x,x) \mid x \in P\}$ is a bisimulation yielded by invariant P, but no further advantage is taken from this fact.

$$\equiv \qquad \{ \ (16) \text{ twice } \}$$

$$\in \cdot (\Lambda S) \cdot R \subseteq R \cdot \in \cdot \Lambda U \ \wedge \ \in \cdot (\Lambda U) \cdot R^\circ \subseteq R^\circ \cdot \in \cdot \Lambda S$$

$$\equiv \qquad \{ \ \text{cancellation} \in \cdot (\Lambda R) = R \text{ four times } \}$$

$$S \cdot R \subseteq R \cdot U \ \wedge \ U \cdot R^\circ \subseteq R^\circ \cdot S$$

The two conjuncts state that R and its converse are *simulations* between state transition relations S and U, which corresponds to the Park-Milner definition [7]: *a bisimulation is a* simulation *between two LTS such that its converse is also a simulation, where a simulation between two LTS S and U is a relation R such that, if $(p, q) \in R$, then for all p' such that $(p', p) \in S$, then there is a q' such that $(p', q') \in R$ and $(q', q) \in U$* — see diagram on the right.
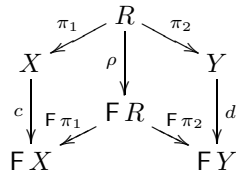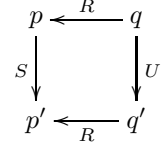
Next we want want to check (13) against another (also coalgebraic) definition of bisimulation due to Aczel & Mendler [1]: *given two coalgebras $c : X \to F(X)$ and $d : Y \to F(Y)$ an F-bisimulation is a relation $R \subseteq X \times Y$ which can be extended to a coalgebra ρ such that projections π_1 and π_2 lift to F-coalgebra morphisms.* (See diagram aside.)

Jacobs [10] spends some time in proving the equivalence between the two definitions. Our proof will be much shorter and calculational thanks to a small trick: we identify (13) as an instance of Reynolds "arrow combinator" $R \leftarrow S$ which, given R and S, relates two functions f and g as follows:

$$f(R \leftarrow S)g \ \equiv \ f \cdot S \subseteq R \cdot g \tag{17}$$

The fact that we can write

$$c(\mathsf{F}\, R \leftarrow R)d \tag{18}$$

instead of $c \cdot R \subseteq \mathsf{F}\, R \cdot d$ (13) to mean that R is a bisimulation between F coalgebras c and d is of great notational, conceptual and calculational advantage. As far as notation is concerned, (18) is very appropriate for telling that c and d produce $\mathsf{F}\, R$-related outputs $c\, y$ and $d\, x$ provided their inputs are R-related ($y\, R\, x$). Conceptually, $\mathsf{F}\, R \leftarrow R$ may be regarded as a relation involving all coalgebras which are R-bisimilar. But it is the calculational power implicit in (18) which really justifies the recasting of (13) in terms of Reynolds' arrow combinator. This has been studied in detail in [2] (if not earlier), a paper which derives elegant and manageable PF-properties such, for instance

$$id \leftarrow id = id \tag{19}$$

$$(R \leftarrow S)^\circ = R^\circ \leftarrow S^\circ \tag{20}$$

$$R \leftarrow S \subseteq V \leftarrow U \ \Leftarrow R \subseteq V \ \wedge \ U \subseteq S \tag{21}$$

$$k(f \leftarrow g)h \equiv k \cdot g = f \cdot h \tag{22}$$

---

[7] The pointwise definition of simulation is better preceived once $S \cdot R \subseteq R \cdot U$ is re-written into $R \subseteq S \setminus (R \cdot U)$, recall (16) — similarly for the other conjunct. Matteo Vaccari [21] performs a calculation similar to the above starting directly from this pointwise definition.

From property (21) we learn that the combinator is monotonic on the left hand side — and thus facts

$$S \leftarrow R \subseteq (S \cup V) \leftarrow R \qquad (23)$$

$$\top \leftarrow S = \top \qquad (24)$$

hold [8] — and anti-monotonic on the right hand side — and thus

$$R \leftarrow \bot = \top \qquad (25)$$

and the two distributive properties which follow:

$$S \leftarrow (R_1 \cup R_2) = (S \leftarrow R_1) \cap (S \leftarrow R_2) \qquad (26)$$

$$(S_1 \cap S_2) \leftarrow R = (S_1 \leftarrow R) \cap (S_2 \leftarrow R) \qquad (27)$$

Let us see how the properties above explain those of bisimulation by themselves. Property (20) ensures that the converse of a bisimulation is also a bisimulation. This turns out to be an equivalence:

$$
\begin{array}{ll}
R \text{ is a bisimulation} & \\
\equiv \quad \{ \ (18) \ \} & \qquad d((\mathsf{F}\,R)^\circ \leftarrow R^\circ)c \\
c(\mathsf{F}\,R \leftarrow R)d & \equiv \quad \{ \ \text{relator } \mathsf{F} \ \} \\
\equiv \quad \{ \ \text{converse} \ \} & \qquad d(\mathsf{F}(R^\circ) \leftarrow R^\circ)c \\
d(\mathsf{F}\,R \leftarrow R)^\circ c & \equiv \quad \{ \ (18) \ \} \\
\equiv \quad \{ \ (20) \ \} & \qquad R^\circ \text{ is a bisimulation}
\end{array}
$$

Next, we recall the definition of a coalgebra morphism:

Definition 3. *Let* $(X, p : X \longrightarrow \mathsf{F}\,X)$ *and* $(Y, q : Y \longrightarrow \mathsf{F}\,Y)$ *be* coalgebras *for functor* F. *A morphism* connecting p and q *is a function* h *between their carriers such that* $q \cdot h = \mathsf{F}\,h \cdot p$.

Clearly, property (22) tells immediately that coalgebra morphisms are bisimulations.

The easy calculation of $\mathsf{F}\,id \leftarrow id = id$ (19) ensures id is a bisimulation between a given coalgebra and itself. On the other side of the spectrum, (25) tells us that $\bot$ is a bisimulation for *any* pair of coalgebras c and d. (Just introduce points in $\mathsf{F}\,\bot \leftarrow \bot$ simplify.)

Let us now see how the fact that bisimulations are closed under union,

$$c(\mathsf{F}\,R_1 \leftarrow R_1)d \ \wedge \ c(\mathsf{F}\,R_2 \leftarrow R_2)d \Rightarrow c(\mathsf{F}(R_1 \cup R_2) \leftarrow (R_1 \cup R_2))d \qquad (28)$$

stems from properties (21,23) and (26). First we PF-transform (28) to

$$(\mathsf{F}\,R_1 \leftarrow R_1) \cap (\mathsf{F}\,R_2 \leftarrow R_2) \subseteq \mathsf{F}(R_1 \cup R_2) \leftarrow (R_1 \cup R_2)$$

---

[8] Cf. $f \cdot S \cdot g^\circ \subseteq \top \equiv \text{TRUE}$ concerning (24).

and reason:

$$(\mathsf{F}\,R_1 \leftarrow R_1) \cap (\mathsf{F}\,R_2 \leftarrow R_2)$$

$$\subseteq \qquad \{ \ (23) \ (\text{twice}) \ ; \ \text{monotonicity of} \cap \ \}$$

$$((\mathsf{F}\,R_1 \cup \mathsf{F}\,R_2) \leftarrow R_1) \cap ((\mathsf{F}\,R_1 \cup \mathsf{F}\,R_2) \leftarrow R_2)$$

$$= \qquad \{ \ (26) \ \}$$

$$(\mathsf{F}\,R_1 \cup \mathsf{F}\,R_2) \leftarrow (R_1 \cup R_2)$$

$$\subseteq \qquad \{ \ \mathsf{F} \ \text{is monotonic; (21)} \ \}$$

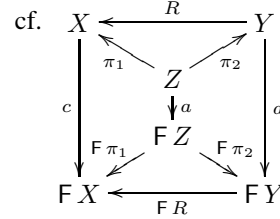$$\mathsf{F}(R_1 \cup R_2) \leftarrow (R_1 \cup R_2)$$

Finally, we address the equivalence between Jacobs's and Aczel-Mendler's definitions of bisimulation. To the set of known rules about (17) we add the following law [9]

$$(r \cdot s^\circ) \leftarrow (f \cdot g^\circ) = (r \leftarrow f) \cdot (s \leftarrow g)^\circ \ \Leftarrow \text{pair } r, s \text{ is a tabulation} \qquad (29)$$

where a pair of functions $A \xleftarrow{\ r\ } C \xrightarrow{\ s\ } B$ form a tabulation iff split function $\langle r, s \rangle$ is injective, that is, iff $r^\circ \cdot r \cap s^\circ \cdot s = id$ holds [10].

Below we show that (29) is what matters in proving the equivalence between Jacobs' definition of bisimulation (once PF-transformed) and that of Aczel & Mendler:

$$c(\mathsf{F}\,R \leftarrow R)d$$

$$\equiv \qquad \{ \ \text{tabulate } R = \pi_1 \cdot \pi_2^\circ \ \}$$

$$c(\mathsf{F}(\pi_1 \cdot \pi_2^\circ) \leftarrow (\pi_1 \cdot \pi_2^\circ))d$$

$$\equiv \qquad \{ \ \text{relator commutes with composition and converse} \ \}$$

$$c(((\mathsf{F}\,\pi_1) \cdot (\mathsf{F}\,\pi_2)^\circ) \leftarrow (\pi_1 \cdot \pi_2^\circ))d$$

$$\equiv \qquad \{ \ \text{new rule (29)} \ \}$$

$$c((\mathsf{F}\,\pi_1 \leftarrow \pi_1) \cdot ((\mathsf{F}\,\pi_2)^\circ \leftarrow \pi_2^\circ))d$$

$$\equiv \qquad \{ \ \text{converse rule (20)} \ \}$$

$$c((\mathsf{F}\,\pi_1 \leftarrow \pi_1) \cdot (\mathsf{F}\,\pi_2 \leftarrow \pi_2)^\circ)d$$

$$\equiv \qquad \{ \ (5) \ \}$$

$$\langle \exists\, a \ :: \ c(\mathsf{F}\,\pi_1 \leftarrow \pi_1)a \ \wedge \ d(\mathsf{F}\,\pi_2 \leftarrow \pi_2)a \rangle$$

cf.

Clearly, the meaning of the last line above is exactly Aczel-Mendler's definition (cf. diagram): it states that *there exists a coalgebra a whose carrier is the "graph" of bisimulation R and which is such that projections π_1 and π_2 lift to the corresponding coalgebra morphisms.*

---

[9] The proof of (29) can be found in [13].

[10] It is a standard result that every R can be factored in a tabulation $R = r \cdot s^\circ$ [7]. An obvious and easy to check tabulation is $r, s := \pi_1, \pi_2$ [13], which boils down to pairwise equality of pairs: $(b, a) = (d, c)$ equivalent to $b = d \ \wedge \ a = c$.

Note how simple the proof is. The elegance of the calculation lies in the synergy with Reynolds' arrow combinator. To the best of our knowledge, such a synergy is new in the literature [11].

5   Calculating invariants

Let us write $\mathsf{F}\,\varPhi_P \xleftarrow{\;c\;} \varPhi_P$ to denote the fact that P is an invariant (9), which we abbreviate to $\mathsf{F}\,\varPhi \xleftarrow{\;c\;} \varPhi$ since predicates and coreflexives are in one to one correspondence. (We will use uppercase Greek letters to denote such coreflexives and will refer to them as "invariants" with no further explanation).

This notation suggests a category Pred of "predicates as objects" as a suitable universe for describing coalgebraic systems subject to invariants. Pred's objects are predicates, represented by coreflexives. An arrow $\varPsi \xleftarrow{\;f\;} \varPhi$ in Pred means a function which ensures property $\varPsi$ on its output whenever property $\varPhi$ holds on its input. Arrows in Pred can therefore be seen as *proof-obligations* for the underlying functions [12]. Formally:

$$\varPsi \xleftarrow{\;f\;} \varPhi \;\;\equiv\;\; f(\varPsi \leftarrow \varPhi)f \;\;\equiv\;\; f \cdot \varPhi \subseteq \varPsi \cdot f \tag{30}$$

Clearly, any relator (in Rel) restricts to a functor in Pred. In particular, the functorial image of an arrow $\varPsi \xleftarrow{\;f\;} \varPhi$ is well-typed, cf.

$$\mathsf{F}\,\varPsi \xleftarrow{\;\mathsf{F}f\;} \mathsf{F}\,\varPhi$$

$$\equiv \qquad \{ \;(30)\; \}$$

$$\mathsf{F}f \cdot \mathsf{F}\,\varPhi \subseteq \mathsf{F}\,\varPsi \cdot \mathsf{F}f$$

$$\equiv \qquad \{ \;\text{functors}\; \}$$

$$\mathsf{F}\,(f \cdot \varPhi) \subseteq \mathsf{F}\,(\varPsi \cdot f)$$

$$\Leftarrow \qquad \{ \;\mathsf{F}\text{ is monotone; }(30)\; \}$$

$$\varPsi \xleftarrow{\;f\;} \varPhi$$

Such a "predicates as types" view carries over universal constructs. As Pred's hom sets are included in Set, in order to verify whether a particular universal property in

---

[11] For a longer bi-implication proof of this equivalence see Backhouse and Hoogendijk's work on final *dialgebras* [6]. A proof of the same result is implicit in Corollary 3.1 of [19] which invokes a result by Carboni *et al* [8] on extending functors to relators.

[12] See [15], where this view of proof obligations is actually extended to arbitrary binary relations. This is suitable for specification languages such as eg. VDM, where *the inclusion of a subtyping mechanism which allows truth-valued functions forces the type checking here to rely on proofs* [11].

the latter lifts to a universal in Pred it is enough to check whether the corresponding diagram still exists and the universal arrow in Set is still an arrow in Pred. The fact that composition satisfies constraints,

$$\Psi \xleftarrow{\ g \cdot f\ } \Phi \ \Leftarrow\ \Psi \xleftarrow{\ g\ } \Upsilon \ \wedge\ \Upsilon \xleftarrow{\ f\ } \Phi \tag{31}$$

stems directly from (30), as does the obvious rule concerning identity:

$$\Psi \xleftarrow{\ id\ } \Phi \ \equiv\ \Phi \subseteq \Psi \tag{32}$$

From (31) we infer also that exponential g^Φ (which should be such that $g^\Phi\ f = g \cdot f$) is well-typed. For a slightly more elaborate example consider, for instance, functional *products* in the new setting:

$$\Psi \xleftarrow{\ \pi_1\ } \Psi \times \Upsilon \xrightarrow{\ \pi_2\ } \Upsilon \tag{33}$$

Clearly, the proof-obligations associated to the two projections

$$\pi_1 \cdot (\Psi \times \Upsilon) \subseteq \Psi \cdot \pi_1 \quad , \quad \pi_2 \cdot (\Psi \times \Upsilon) \subseteq \Upsilon \cdot \pi_2$$

are instances of Reynolds abstraction theorem [18, 22, 2]:

$$\mathsf{G}\,A \xleftarrow{\ f\ } \mathsf{F}\,A \ \text{is polymorphic} \equiv \langle \forall\,R \ :: \ f(\mathsf{G}\,R \leftarrow \mathsf{F}\,R)f \rangle \tag{34}$$

So there is nothing to prove. To show that $\langle f, g \rangle$ is indeed an arrow in Pred we need to recall the universal property of relational splits [7]

$$X \subseteq \langle R, S \rangle \ \equiv\ \pi_1 \cdot X \subseteq R \ \wedge\ \pi_2 \cdot X \subseteq S \tag{35}$$

and that $\times$-absorption holds. We reason:

$$\Psi \times \Upsilon \xleftarrow{\ \langle f, g \rangle\ } \Phi$$

$\equiv \qquad \{\ \text{definition (30)}\ \}$

$$\langle f, g \rangle \cdot \Phi \subseteq (\Psi \times \Upsilon) \cdot \langle f, g \rangle$$

$\equiv \qquad \{\ \text{absorption law for relational product}\ \}$

$$\langle f, g \rangle \cdot \Phi \subseteq \langle \Psi \cdot f, \Upsilon \cdot g \rangle$$

$\equiv \qquad \{\ \text{universal law for relational product (35)}\ \}$

$$\pi_1 \cdot \langle f, g \rangle \cdot \Phi \subseteq \Psi \cdot f \ \wedge\ \pi_2 \cdot \langle f, g \rangle \cdot \Phi \subseteq \Upsilon \cdot g$$

$\equiv \qquad \{\ \text{cancellation law for functional product}\ \}$

$$f \cdot \Phi \subseteq \Psi \cdot f \ \wedge\ g \cdot \Phi \subseteq \Upsilon \cdot g$$

$\equiv \qquad \{\ \text{definition (30)}\ \}$

$$\Phi \xleftarrow{\ f\ } \Phi \ \wedge\ \Upsilon \xleftarrow{\ g\ } \Phi$$

As expected, a coalgebra $\mathsf{F}\,\Phi \xleftarrow{\;c\;} \Phi$ in Pred maintains property Φ invariant (in the same way an algebra $\Upsilon \xleftarrow{\;a\;} \mathsf{F}\,\Upsilon$ declares Υ as a *compatible* predicate).

Final coalgebras (and initial algebras) exist and coincide with the ones in Set. Let us check, in this respect, the diagram of *unfold* (aside), where $\nu\mathsf{F}$ denotes the final coalgebra [13] and $[\![c]\!]$ is the coinductive extension, or *unfold*, of coalgebra c. We reason:

$$
\begin{array}{c}
\mathsf{F}(\nu\mathsf{F}) \xleftarrow{\;out\;} \nu\mathsf{F} \\
\mathsf{F}[\![c]\!] \uparrow \qquad\qquad \uparrow [\![c]\!] \\
\mathsf{F}\,\Phi \xleftarrow{\;c\;} \Phi
\end{array}
$$

$$\nu\mathsf{F} \xleftarrow{\;[\![c]\!]\;} \Phi$$

$\equiv$ $\quad$ { definition (30) }

$$[\![c]\!] \cdot \Phi \subseteq [\![c]\!]$$

$\Leftarrow$ $\quad$ { relational coinduction fusion: $[\![T]\!] \cdot S \subseteq [\![R]\!] \;\Leftarrow\; T \cdot S \subseteq \mathsf{F}\,S \cdot R$ }

$$c \cdot \Phi \subseteq \mathsf{F}\,\Phi \cdot c$$

$\equiv$ $\quad$ { definition (30) }

$$\mathsf{F}\,\Phi \xleftarrow{\;c\;} \Phi$$

We close this section by showing how the "invariants as bisimulations" approach helps in developing of a number of simple, yet powerful rules to reason about "invariant-typed" coalgebras. Our calculations below address three such rules.

Separation rule:

$$\mathsf{F}\,(\Phi \cdot \Psi) \xleftarrow{\;c\;} \Phi \cdot \Psi \;\;\Leftarrow\;\; \mathsf{F}\,\Phi \xleftarrow{\;c\;} \Phi \;\;\wedge\;\; \mathsf{F}\,\Psi \xleftarrow{\;c\;} \Psi \qquad (36)$$

This rule enables the decomposition of the proof obligation of a compound invariant into two separate proof obligations, one per conjunct. Its calculation is as follows:

$$\mathsf{F}\,\Phi \xleftarrow{\;c\;} \Phi \;\;\wedge\;\; \mathsf{F}\,\Psi \xleftarrow{\;c\;} \Psi$$

$\equiv$ $\quad$ { (30) twice }

$$c \cdot \Phi \subseteq \mathsf{F}\,\Phi \cdot c \;\wedge\; c \cdot \Psi \subseteq \mathsf{F}\,\Psi \cdot c$$

$\Rightarrow$ $\quad$ { monotonicity of composition (twice) }

$$c \cdot \Phi \cdot \Psi \subseteq \mathsf{F}\,\Phi \cdot c \cdot \Psi \;\wedge\; \mathsf{F}\,\Phi \cdot c \cdot \Psi \subseteq \mathsf{F}\,\Phi \cdot \mathsf{F}\,\Psi \cdot c$$

$\Rightarrow$ $\quad$ { transitivity }

$$c \cdot \Phi \cdot \Psi \subseteq \mathsf{F}\,\Phi \cdot (\mathsf{F}\,\Psi \cdot c)$$

$\equiv$ $\quad$ { relator F and (30) }

$$\mathsf{F}\,(\Phi \cdot \Psi) \xleftarrow{\;c\;} \Phi \cdot \Psi$$

---

[13] Since diagrams in Pred bear coreflexives on nodes, $\nu\mathsf{F}$ abbreviates $id_{\nu\mathsf{F}}$, meaning that no invariant constrains the carrier of the final coalgebra. Similarly for $\mathsf{F}(\nu\mathsf{F})$.

Interleaving rule:

$$\mathsf{F}\,(\Phi \times \Psi) \xleftarrow{\;c\,|||\,d\;} \Phi \times \Psi \;\;\Leftarrow\;\; \mathsf{F}\,\Phi \xleftarrow{\;c\;} \Phi \;\wedge\; \mathsf{F}\,\Psi \xleftarrow{\;c\;} \Psi \tag{37}$$

where $|||$ is an interleaving operator defined by $c \,|||\, d \stackrel{\mathrm{def}}{=} \delta \cdot (c \times d)$ whenever F has a distributive law $\delta : \mathsf{F}\,\Phi \times \mathsf{F}\,\Psi \longrightarrow \mathsf{F}\,(\Phi \times \Psi)$ corresponding to the Kleisli composition of F's left and right strength (see [12] for details). The calculation of (37) follows:

$$\mathsf{F}\,\Phi \xleftarrow{\;c\;} \Phi \;\;\wedge\;\; \mathsf{F}\,\Psi \xleftarrow{\;c\;} \Psi$$

$$\equiv \quad \{\ (30)\ \text{twice}\ \}$$

$$c \cdot \Phi \subseteq \mathsf{F}\,\Phi \cdot c \;\wedge\; c \cdot \Psi \subseteq \mathsf{F}\,\Psi \cdot c$$

$$\Rightarrow \quad \{\ \text{monotonicity of product and composition}\ \}$$

$$\delta \cdot (c \cdot \Phi \times d \cdot \Psi) \subseteq \delta \cdot (\mathsf{F}\,\Phi \cdot c \times \mathsf{F}\,\Psi \cdot d)$$

$$\Rightarrow \quad \{\ \times \text{ relator}\ \}$$

$$\delta \cdot (c \times d) \cdot (\Phi \times \Psi) \subseteq \delta \cdot (\mathsf{F}\,\Phi \times \mathsf{F}\,\Psi) \cdot (c \times d)$$

$$\Rightarrow \quad \{\ \delta\text{'s free theorem (34)}\ \}$$

$$\delta \cdot (c \times d) \cdot (\Phi \times \Psi) \subseteq \mathsf{F}\,(\Phi \times \Psi) \cdot \delta \cdot (c \times d)$$

$$\equiv \quad \{\ \text{definition of } c \,|||\, d \text{ and (30)}\ \}$$

$$\mathsf{F}\,(\Phi \times \Psi) \xleftarrow{\;c\,|||\,d\;} \Phi \times \Psi$$

Pipeline. For F a monad,

$$\mathsf{F}\,\Phi \xleftarrow{\;c\bullet d\;} \Phi \;\;\Leftarrow\;\; \mathsf{F}\,\Phi \xleftarrow{\;c\;} \Phi \;\wedge\; \mathsf{F}\,\Phi \xleftarrow{\;d\;} \Phi \tag{38}$$

where $c \bullet d$ corresponds to the Kleisli composition of c and d. We calculate:

$$(c \bullet d) \cdot \Phi$$

$$= \quad \{\ \text{definition of Kleisli composition}\ \}$$

$$\mu \cdot \mathsf{F}\,c \cdot d \cdot \Phi$$

$$\subseteq \quad \{\ \mathsf{F}\,\Phi \xleftarrow{\;d\;} \Phi \ \text{ and monotonicity}\ \}$$

$$\mu \cdot \mathsf{F}\,c \cdot \mathsf{F}\,\Phi \cdot d$$

$$= \quad \{\ \mathsf{F}\ \text{relator}\ \}$$

$$\mu \cdot \mathsf{F}\,(c \cdot \Phi) \cdot d$$

$$\subseteq \quad \{\ \mathsf{F}\,\Phi \xleftarrow{\;c\;} \Phi \ \text{ and monotonicity}\ \}$$

$$\mu \cdot \mathsf{F}\,(\mathsf{F}\,\Phi \cdot c) \cdot d$$

$$= \qquad \{\ \mathsf{F} \text{ relator and } \mu\text{'s free theorem (34)}\ \}$$

$$\mathsf{F}\,\Phi \cdot \mu \cdot \mathsf{F}\,c \cdot d$$

$$= \qquad \{\ \text{definition of Kleisli composition}\ \}$$

$$\mathsf{F}\,\Phi \cdot (c \bullet d)$$

6   Calculating assertions

As mentioned in the introduction to this paper, the third main ingredient of coalgebraic reasoning identified in [10] is a language of modal *assertions* in which specifications of the behaviour of systems can be expressed. Clearly, invariants bring about a *"next time"* modal operator,

$$c(\mathsf{F}\,\Phi \leftarrow \Phi)c \;\equiv\; c \cdot \Phi \subseteq \mathsf{F}\,\Phi \cdot c$$

$$\equiv \qquad \{\ \text{shunting (10)}\ \}$$

$$\Phi \subseteq \underbrace{c^{\circ} \cdot (\mathsf{F}\,\Phi) \cdot c}_{\bigcirc_c \Phi} \tag{39}$$

which holds for those states whose all immediate successors, if any, satisfy Φ. From this a PF-definition of the *"next time Φ holds"* modal operator emerges

$$\bigcirc_c \Phi \stackrel{\text{def}}{=} c^{\circ} \cdot (\mathsf{F}\,\Phi) \cdot c \tag{40}$$

which PF-transforms Def. 4.3.1 of [10]. So, assertion $\Phi \subseteq \bigcirc_c \Phi$ is an alternative statement of "Φ in an invariant" for coalgebra c.

This modal operator is easily shown to be the upper adjoint of Galois connection

$$\pi_c \Phi \subseteq \Psi \equiv \Phi \subseteq \bigcirc_c \Psi \tag{41}$$

whose lower adjoint is the *projection* operator $\pi_c \Phi \stackrel{\text{def}}{=} c \cdot \Phi \cdot c^{\circ}$ which is central to [14] in studying the PF-refactoring of data dependency theory (a part of database theory). From this, one immediately infers that $\bigcirc_c$ is monotonic and distributes over conjunction: $\bigcirc_c(\Phi \cdot \Psi) = (\bigcirc_c \Phi) \cdot (\bigcirc_c \Psi)$. Note that we express conjunction by composition because these two operators coincide on coreflexives:

$$\Phi \cap \Psi = \Phi \cdot \Psi \tag{42}$$

Such properties can then be used to reason about operator $\bigcirc_c$, as in, for example,

$$\Phi \text{ is an invariant}$$

$$\equiv \qquad \{\ (39)\ \}$$

$$\Phi \subseteq \bigcirc_c \Phi$$

$$\Rightarrow \qquad \{ \text{ monotonicity of } \bigcirc_c \text{ stemming from (41) } \}$$

$$\bigcirc_c \Phi \subseteq \bigcirc_c(\bigcirc_c \Phi)$$

$$\equiv \qquad \{ \text{ (39) } \}$$

$$\bigcirc_c \Phi \text{ is an invariant}$$

The whole construction of a modal logic relative to a coalgebra c, which is the basis of *assertion* reasoning in coalgebra theory, can be pursued along similar lines. Consider, for example, the definition of $\Box P$, the *henceforth P* operator of [10, Def. 4.2.8]:

$$(\Box P)x \;\; \overset{\text{def}}{=} \;\; \langle \exists\, Q \,:\, Q \text{ is invariant} :\, Q \subseteq P \,\wedge\, (Q\, x) \rangle$$

Converting predicates P and Q to coreflexives Φ and Ψ, respectively, and making explicit the *supremum* implict in the existential quantification one gets,

$$\Box\Phi = \langle \bigcup\, \Psi \,:\, \Psi \subseteq \bigcirc_c \Psi :\, \Psi \subseteq \Phi \rangle$$

$$= \qquad \{ \text{ trading [4] } \}$$

$$\langle \bigcup\, \Psi \,::\, \Psi \subseteq \bigcirc_c \Psi \,\wedge\, \Psi \subseteq \Phi \rangle$$

$$= \qquad \{ \cap\text{-universal} \}$$

$$\langle \bigcup\, \Psi \,::\, \Psi \subseteq \bigcirc_c \Psi \cap \Phi \rangle$$

$$= \qquad \{ \cap \text{ of coreflexives is composition (42) } \}$$

$$\langle \bigcup\, \Psi \,::\, \Psi \subseteq \Phi \cdot \bigcirc_c \Psi \rangle$$

which leads to a greatest (post)fixpoint definition:

$$\Box\Phi = \langle \nu\, \Psi \,::\, \Phi \cdot \bigcirc_c \Psi \rangle \tag{43}$$

We end this section by showing how the PF-transform (and in particular the replacement of intersection of coreflexives by composition (42)) together with the fixpoint calculus [3] speed up derivation of laws in such a logic. The law we have chosen to calculate is Lemma 4.2.9(ii) of [10]: $\Box\Phi \subseteq \Box\Box\Phi$. We drop subscript c of $\bigcirc_c$ (for economy of notation) and calculate:

$$\Box\Phi \subseteq \Box\Box\Phi$$

$$\equiv \qquad \{ \text{ (43) } \}$$

$$\Box\Phi \subseteq \langle \nu\, \Psi \,::\, (\Box\Phi) \cdot \bigcirc \Psi \rangle$$

$$\Leftarrow \qquad \{ \text{ greatest fixed point induction: } x \leq fx \,\Rightarrow\, x \leq \nu f \text{ [3] } \}$$

$$\Box\Phi \subseteq \Box\Phi \cdot \bigcirc(\Box\Phi)$$

$$\equiv \qquad \{ \, \Box\Phi \cdot \Phi = \Box\Phi \text{ thanks to (42), since } \Box\Phi \subseteq \Phi \, \}$$

$$\Box\Phi \;\subseteq\; \Box\Phi \cdot \Phi \cdot \bigcirc(\Box\Phi)$$

$\equiv\quad \{$ property (for Φ coreflexive) $\Phi \cdot R \subseteq S \;\equiv\; \Phi \cdot R \subseteq \Phi \cdot S \}$

$$\Box\Phi \;\subseteq\; \Phi \cdot \bigcirc(\Box\Phi)$$

$\equiv\quad \{$ (43) and fixpoint calculus $(\nu_f \subseteq f\nu_f) \}$

$\quad true$

7   Epilogue

Invariants are constraints on the carrier of coalgebras which restrict their behavior in some desirable way but whose maintenance entails some kind of proof obligation discharge. An approach is put forward in this paper for reasoning about coalgebraic invariants which is both *compositional* and *calculational*: compositional because it is based on rules which break the complexity of such proof obligations across the structures involved; calculational because such rules are derived thanks to an algebra of invariants regarded as coreflexive bisimulations, which is what invariants are once encoded in the language of binary relations. Such calculational capabilities arise, in turn, from encoding bisimulations as instances of Reynolds *relation on functions*. In this process, functors which capture coalgebras' dynamics are generalized to relators and the objects of the underlying category are generalized to predicates.

The main contribution of the paper is the explicit adoption of such a constructive, calculational style in approaching the problem. Both [19, 6] already suggest a relational/relator-based approach to bisimulation, [6] actually generalizing from coalgebras to dialgebras. However, no relationship is established with the algebra of Reynolds *relation on functions* which, in close association with Reynolds abstraction theorem, naturally leads to a category (Pred) whose objects are predicates (invariants).

In a wider context, the explicit adoption of such a category has potential to support a constructive discipline of extended static checking (ESC) in a coalgebraic view of computer systems, but surely there is much work to be done before this becomes of practical use. On the theory side, the authors are aware of a connection between the "predicates as objects" approach and *Frege structures* [9] [14]. Quoting this reference:

> *A Frege structure is a lambda structure $\mathcal{F}$ on the set A together with a designated subset of A whose elements are called propositions (...) the propositional connectives are required to yield propositions as values only when they operate on propositions as arguments.*

Further investigating this connection is surely an interesting topic for future research.

References

1. Peter Aczel and Nax Mendler. A final coalgebra theorem. In *Category Theory and Computer Science*, pages 357–365, London, UK, 1989. Springer-Verlag.

---

[14] They thank Peter Dybjer for pointing out this connection.

2. K. Backhouse and R.C. Backhouse. Safety of abstract interpretations for free, via logical relations and Galois connections. *SCP*, 15(1–2):153–196, 2004.
3. R. Backhouse. Galois connections and fixed point calculus. In R. Crole, R. Backhouse, and J. Gibbons, editors, *Algebraic and Coalgebraic Methods in the Mathematics of Program Constuction*, pages 89–148. Springer Lect. Notes Comp. Sci. (2297), 2002.
4. R. Backhouse and D. Michaelis. Exercises in quantifier manipulation. In *MPC'06*, pages 70–81. Springer LNCS (4014), 2006.
5. R.C. Backhouse, P. de Bruin, P. Hoogendijk, G. Malcolm, T.S. Voermans, and J. van der Woude. Polynomial relators. In *AMAST'91*, pages 303–362. Springer, 1992.
6. R.C. Backhouse and P.F. Hoogendijk. Final dialgebras: From categories to allegories. *Informatique Theorique et Applications*, 33(4/5):401–426, 1999.
7. R. Bird and O. de Moor. Algebra of Programming. Series in Computer Science. Prentice-Hall International, 1997. C.A.R. Hoare, series editor.
8. A. Carboni, G. Kelly, and R. Wood. A 2-categorical approach to change of base and geometric morphisms I. Technical Report 90-1, Dept. of Pure Maths, Univ. Sydney, 1990.
9. William S. Hatcher. Review: Peter Aczel. Frege structures and the notions of proposition, truth and set. *The Journal of Symbolic Logic*, 51(1):244–246, 1986.
10. Bart Jacobs. Introduction to Coalgebra. Towards Mathematics of States and Observations. Draft Copy. Institute for Computing and Information Sciences, Radboud University Nijmegen, P.O. Box 9010, 6500 GL Nijmegen, The Netherlands.
11. C.B. Jones. *Systematic Software Development Using VDM*. Prentice-Hall Int., 1986.
12. A. Kock. Strong functors and monoidal monads. *Archiv für Mathematik*, 23:113–120, 1972.
13. J.N. Oliveira. *Invariants as coreflexive bisimulations — in a coalgebraic setting*, Dec. 2006. Presentation at the *IFIP WG 2.1 #62 Meeting*, Namur.
14. J.N. Oliveira. Pointfree foundations for (generic) lossless decomposition, 2007. (submitted).
15. J.N. Oliveira. Theory and applications of the PF-transform, Feb. 2008. Tutorial at Ler-NET'08, Piriápolis, Uruguay (slides available from the author's website).
16. J.N. Oliveira and C.J. Rodrigues. Transposing relations: from *Maybe* functions to hash tables. In *MPC'04*, volume 3125 of *LNCS*, pages 334–356. Springer, 2004.
17. D. Park. Concurrency and automata on infinite sequences. pages 561–572. Springer LNCS (104), 1981.
18. J.C. Reynolds. Types, abstraction and parametric polymorphism. *Information Processing 83*, pages 513–523, 1983.
19. Jan J.M.M. Rutten. Relators and metric bisimulations. *ENTCS*, 11:1–7, 1998.
20. J.J.M.M. Rutten. Coalgebraic foundations of linear systems (an exercise in stream calculus), 2007. Proceedings CALCO 2007, LNCS, Springer, to appear.
21. M. Vaccari. Calculational derivation of circuits, 1998. PhD thesis, Univ. S. Milano.
22. P.L. Wadler. Theorems for free! In *4th Int. Symp. on FPLCA*, London, Sep. 1989. ACM.