Centrum voor Wiskunde en Informatica

REPORT*RAPPORT*

Universal coalgebra: a theory of systems

J.J.M.M. Rutten

Computer Science/Department of Software Technology

CS-R9652 1996

Report CS-R9652
ISSN 0169-118X

CWI
P.O. Box 94079
1090 GB Amsterdam
The Netherlands

P.O. Box 94079, 1090 GB Amsterdam (NL)
Kruislaan 413, 1098 SJ Amsterdam (NL)
Telephone +31 20 592 9333
Telefax +31 20 592 4199

Universal Coalgebra: a Theory of Systems

J.J.M.M. Rutten
CWI
P.O. Box 94079, 1090 GB Amsterdam, The Netherlands
email: `janr@cwi.nl`, http: `www.cwi.nl/~janr`

Abstract

In the semantics of programming, finite data types such as finite lists, have traditionally been modelled by initial algebras. Later final *coalgebras* were used in order to deal with *infinite* data types. Coalgebras, which are the dual of algebras, turned out to be suited, moreover, as models for certain types of automata and more generally, for (transition and dynamical) *systems*.

An important property of initial algebras is that they satisfy the familiar principle of induction. Such a principle was missing for coalgebras until the work of Aczel (1988) on a theory of non-wellfounded sets, in which he introduced a proof principle nowadays called *coinduction*. It was formulated in terms of *bisimulation*, a notion originally stemming from the world of concurrent programming languages (Milner, 1980; Park, 1981). Using the notion of *coalgebra homomorphism*, the definition of bisimulation on coalgebras can be shown to be formally dual to that of congruence on algebras (Aczel and Mendler, 1989).

Thus the three basic notions of universal algebra: algebra, homomorphism of algebras, and congruence, turn out to correspond to: coalgebra, homomorphism of coalgebras, and bisimulation, respectively. In this paper, the latter are taken as the basic ingredients of a theory called *universal coalgebra*. Some standard results from universal algebra are reformulated (using the afore mentioned correspondence) and proved for a large class of coalgebras, leading to a series of results on, e.g., the lattices of subcoalgebras and bisimulations, simple coalgebras and coinduction, and a covariety theorem for coalgebras similar to Birkhoff's variety theorem.

AMS Subject Classification (1991): 68Q10, 68Q55
CR Subject Classification (1991): D.3.1, F.1.2, F.3.2
Keywords & Phrases: Coalgebra, algebra, dynamical system, transition system, bisimulation, universal coalgebra, universal algebra, congruence, homomorphism, induction, coinduction.

Contents

1 Introduction

In the semantics of programming, data types are usually presented as algebras (cf. [LS81]). For instance, the collection of finite words A^* over some alphabet A is an algebra

$$\langle A^*,\ \alpha : (1 + (A \times A^*)) \to A^* \rangle,$$

where α maps $*$ (the sole element of the singleton set $1 = \{*\}$) to the empty word and a pair $\langle a, w \rangle$ to $a \cdot w$. This example is typical in that A^* is an *initial* algebra. Initial algebras are generalizations of *least* fixed points, and satisfy familiar inductive proof and definition principles.

For *infinite* data structures, the dual notion of *coalgebra* has been used as an alternative to the algebraic approach [AM82]. For instance, the set A^∞ of finite and infinite words over A can be described by the pair

$$\langle A^\infty,\ \gamma : A^\infty \to (1 + (A \times A^\infty)) \rangle,$$

where γ maps the empty word to $*$ and a non-empty word to the pair consisting of its head (the first letter) and tail (the remainder). It is a *co*algebra because γ is a function *from* the carrier set A^∞ to an expression involving A^∞, that is, $1 + (A \times A^\infty)$, as opposed to the algebra above, where α was a function *into* the carrier set A^*. Again the example is typical because A^∞ is a *final* coalgebra, which generalizes the notion of *greatest* fixed point.

Coalgebras had previously been found to be suitable for the description of the dynamics of systems such as deterministic automata (cf. [AM80, MA86]). Traditionally these are represented as tuples

$$\langle Q,\ A,\ \delta : Q \times A \to Q,\ \beta : Q \to B \rangle,$$

consisting of a set of states Q, an input alphabet A, a next state function δ, and an output function β (in addition an initial state is often specified as well). Alternatively, such an automaton can be represented as a coalgebra of the form

$$\langle Q,\ \alpha : Q \to (Q^A \times B) \rangle,$$

where Q^A is the set of all functions from A to Q, and α can be defined in an obvious manner from δ and β (and vice versa). Coalgebras are nowadays also used to describe the behaviour of classes in object-oriented languages, see [Rei95] and [Jac96b, Jac96c].

Similarly, Peter Aczel uses a coalgebraic description of (nondeterministic transition) systems in constructing a model for a theory of non-wellfounded sets [Acz88]. Maybe more importantly, he also introduces a proof principle for final coalgebras called strong extensionality. It is formulated in terms of the notion of *bisimulation relation*, originally stemming from the field of concurrency semantics [Mil80, Par81]. Using the notion of *coalgebra homomorphism*, the definition of bisimulation can be shown [AM89, RT94] to be formally dual to that of *congruence* on algebras. Moreover one can formulate definition and proof principles for final coalgebras (generalizing Aczel's principle of strong extensionality), which are the coalgebraic counterpart of the inductive principles for initial algebras, and which therefore are called *coinductive* [RT93].

These observations, then, have led to the development in the present paper of a general theory of coalgebras called *universal coalgebra*, along the lines of universal algebra. Universal algebra (cf. [Coh81, MT92]) deals with the features common to the many well-known examples of algebras such as groups, rings, etc. The central concepts are Σ-algebra, homomorphism of Σ-algebras, and congruence relation. The corresponding notions [RT94] on the coalgebra side are: coalgebra, homomorphism of coalgebras, and bisimulation equivalence. These notions constitute the basic ingredients of our theory of universal coalgebra. (More generally, the notion of *substitutive relation* corresponds to that of bisimulation relation; hence congruences, which are substitutive *equivalence* relations, correspond to bisimulation equivalences.) Adding to this the above-mentioned observation that various dynamical systems (automata, transition systems, and many others as we shall see) can be represented as coalgebras, makes that we speak of universal coalgebra as *a theory of* *systems*. We shall go even as far as, at least for the context of the present paper, to consider coalgebras and systems as synonyms.

The correspondence between the basic elements of the theories of algebra and coalgebra are summarized in the following table:

Universal algebra:	Universal coalgebra:
(Σ-)algebra	coalgebra = system
algebra homomorphism	system homomorphism
substitutive relation (congruence relation)	bisimulation relation (bisimulation equivalence)

As mentioned above, universal algebra plays a guiding role in the development of universal algebra as a theory of coalgebras (= systems). Much of this involves replacing the central notions from universal algebra by the corresponding coalgebraic notions, and see whether the resulting statements can actually be proved. Often, facts on Σ-algebras turn out to be valid (in their translated version) for systems as well. Examples are basic observations on quotients and subsystems, and the so-called three isomorphism theorems. In other cases, more can be said in the world of coalgebras about the dual of an algebraically important notion than about that notion itself. For instance, initial algebras play a role of central importance. Initial coalgebras are usually trivial but *final* coalgebras are most relevant. A related example: initial algebras are *minimal*: they have no proper subalgebras. This property is equivalent to the familiar *induction* proof principle. Dually, final coalgebras are *simple*: they have no proper quotients, which can be interpreted as a so-called *coinductive* proof principle.

In a previous paper [Rut95], the above programme has been carried out for one particular family of systems: unlabelled nondeterministic transition systems (also called frames). As it turns out, all observations on such systems apply to many other kinds of systems as well, such as deterministic and nondeterministic automata, binary systems, and hyper systems. Also the afore mentioned infinite data structures, which can be interpreted as dynamical systems as well, are examples to which the theory applies.

All these different examples can be conveniently described in one single framework, using (some basic) category theory. Each of these classes of systems turns out to be the collection of coalgebras of a particular *functor*, and different functors are needed for different types of systems. In that respect, the world of universal algebra is simpler because of the existence of a general, non-categorical way of describing *all* Σ-algebras at the same time, namely as sets with operations, the type of which is specified by the signature Σ. (A categorical treatment is also feasible in the algebraic case, though; see [Man76].)

The generality of the coalgebraic theory presented here thus lies in the fact that all results are formulated for coalgebras of a collection of well-behaved functors on the category of sets and functions, and thereby apply to a great number of different systems. This number can be seen to be larger still by varying the category involved. Taking, for instance, the category of complete metric spaces rather than simply sets allows us to deal with (discrete time) dynamical systems (Section 16).

Some familiarity with the basic elements of category theory, therefore, will be of use when reading this paper. The notions of category and functor will be assumed to be known. Section 18 has been included to provide some background information. It contains some basic definitions, facts (and notation) both on sets and functors on the category of sets, and is to be consulted when needed. The family of (nondeterministic labelled) *transition systems* [Kel76, Plo81] will be used as a running example throughout the first sections of the paper. Moreover, the reader might want to refer to [Rut95], where many of the present observations are proved in a less abstract way for this family.

Deep insights about groups are not obtained by studying universal algebra. Nor will universal coalgebra lead to difficult theorems about (specific types of) systems. Like universal algebra, its possible merit consists of the fact that it '... tidies up a mass of rather trivial detail, allowing us to concentrate our powers on the hard core of the problem.' ([Coh81]). There are maybe two aspects that we might want to add to this. Firstly, induction principles are well-known and much used. The coinductive definition and proof principles for coalgebras are less well-known by far, and often even not very clearly formulated. Universal coalgebra offers a simple context for a good understanding of coinduction. Secondly, many families of systems look rather different from the outside, and so do the corresponding notions of bisimulation. A systematic study of coalgebras brings to light many sometimes unexpected similarities.

This paper both gives an overview of some of the existing insights in the theory of coalgebras, and, in addition, presents some new material. Section 17 contains a brief description per section of which results have been taken from the literature, as well as a discussion of related work. In summary, the present theory was preceded by [Rut95], which at its turn builds on previous joint work with Turi [RT93, RT94], from which a number of results on final systems is taken. Many observations that are folklore in the context of particular examples (such as transition systems) are generalized to arbitrary systems. The section on the existence of final systems is based on results from Barr [Bar93]. The work of Jacobs on a coalgebraic semantics for object-oriented programming [Jac96b] and coalgebraic specification [Jac95] has greatly influenced the sections on cofree systems.

A synopsis of the contents is given by the second column of the following table, which extends the one above. Its first column shows the corresponding algebraic notions. (See Section 13 for a discussion on the formal relationship between the algebraic and the coalgebraic notions.)

Universal algebra:	Universal coalgebra:
(Σ-)algebra	coalgebra = system
algebra homomorphism	system homomorphism
substitutive relation	bisimulation relation
congruence	bisimulation equivalence
subalgebra	subsystem
minimal algebra (no proper subalgebras) $\Longleftrightarrow$ induction proof principle	minimal system (no proper subsystems)
simple algebra (no proper quotients)	simple system (no proper quotients) $\Longleftrightarrow$ coinduction proof principle
initial algebra (is minimal, plus: induction definition principle)	initial system (often trivial)
final algebra (often trivial)	final system (is simple, plus: coinduction definition principle)
free algebra (used in algebraic specification)	free system (often trivial)
cofree algebra (often trivial)	cofree system (used in coalgebraic specification)
variety (closed under subalgebras, quotients, and products) $\Longleftrightarrow$ definable by a quotient of a free algebra	variety (closed under subsystems, quotients, and products)
covariety (closed under subalgebras, quotients, and coproducts)	covariety (closed under subsystems, quotients, and coproducts) $\Longleftrightarrow$ definable by a subsystem of a cofree system

Note that this table is not to suggest that the theory of systems is dual to that of algebras. (If so the paper would end here.) It is true that certain facts about algebras can be dualized and then apply to systems. For instance, the fact that the quotient of a system with respect to a bisimulation equivalence is again a system is dual to the fact that the quotient of an algebra with respect to a congruence yields again an algebra. However, many notions that are defined in both worlds in the same way, have entirely different properties. Examples are initial algebras and initial systems, of which the former are of prime interest. Dually, final algebras are usually trivial whereas final systems have interesting properties. Similarly, free algebras and *cofree* systems are important: If A_V is an algebra that is free on a given set V, then we think of the elements of V as variables, with which (equations and therewith) classes of algebras can be defined. The intuition behind a system S_C that is cofree on a set C is rather different. The elements of C are best thought of as (labels or) colours. The system S_C is in that view a universally (state) coloured system with the universal property that for any system S (of the same type as S_C) and any 'colouring' $f : S \to C$, there exists a unique colour preserving homomorphism $\tilde{f} : S \to S_C$. Also S_C gives rise to the definition of classes of systems, called covarieties, dualizing the notion of variety of algebras. Covarieties will be characterized in a similar way (Section 15) to Birkhoff's variety theorem. The paper is concluded with a coalgebraic treatment of one-dimensional discrete time dynamical systems. In particular, it is shown how the technique of *symbolic dynamics* in essence is based on the existence of a cofree coalgebra in the category of complete metric spaces.

2 Coalgebras, homomorphisms, and bisimulations

The basic notions of coalgebra, homomorphism, and bisimulation relation are introduced. A running example for this section will be the family of labelled transition systems. Many more examples will follow in Section 3.

Let $F : \mathit{Set} \to \mathit{Set}$ be a functor. An *F-coalgebra* or *F-system* is a pair (S, α_S) consisting of a set S and a function $\alpha_S : S \to F(S)$. The set S is called the *carrier* of the system, also to be called the set of *states*; the function α_S is called the *F-transition structure* (or *dynamics*) of the system. When no explicit reference to the functor (i.e., the type of the system) is needed, we shall simply speak of *system* and *transition structure*. Moreover, when no explicit reference to the transition structure is needed, we shall often use S instead of (S, α_S).

Example 2.1 Consider *labelled transition systems* $(S, \to_S, A)$, consisting of a set S of states, a transition relation $\to_S \subseteq S \times A \times S$, and a set A of labels [Kel76, HP79, Plo81]. As always, $s \xrightarrow{a}_S s'$ is used to denote $\langle s, a, s' \rangle \in \to_S$. Define

$$B(X) = \mathcal{P}(A \times X) = \{V \subseteq A \times X\},$$

for any set X. (We shall see below that B is a functor from *Set* to *Set*.) A labelled transition system $(S, \to_S, A)$ can be represented as a B-system (S, α_S) by defining

$$\alpha_S : S \to B(S), \quad s \mapsto \{\langle a, s' \rangle \mid s \xrightarrow{a}_S s'\}.$$

And conversely, any B-system (S, α_S) corresponds to a transition system $(S, A, \to_S)$ by defining

$$s \xrightarrow{a}_S s' \iff \langle a, s' \rangle \in \alpha_S(s).$$

In other words, the class of all labelled transition systems coincides with the class of all B-systems. □

Let (S, α_S) and (T, α_T) be two F-systems, where F is again an arbitrary functor. A function $f : S \to T$ is a *homomorphism of F-systems*, or *F-homomorphism*, if $F(f) \circ \alpha_S = \alpha_T \circ f$:

$$\begin{tikzcd} S \arrow[r, "f"] \arrow[d, "\alpha_S"'] & T \arrow[d, "\alpha_T"] \\ F(S) \arrow[r, "F(f)"'] & F(T). \end{tikzcd}$$

Intuitively, homomorphisms are functions that preserve and reflect F-transition structures (see the example below). We sometimes write $f : (S, \alpha_S) \to (T, \alpha_T)$ to express that f is a homomorphism. The identity function on an F-system (S, α_S) is always a homomorphism, and the composition of two homomorphisms is again a homomorphism. Thus the collection of all F-systems together with F-system homomorphisms is a category, which we denote by Set_F.

Example 2.1, continued: Let $(S, A, \to_S)$ and $(T, A, \to_T)$ be two labelled transition systems with the same set A of labels, and let (S, α_S) and (T, α_T) be the corresponding representations as B-systems. Per definition, a B-homomorphism $f : (S, \alpha_S) \to (T, \alpha_T)$ is a function $f : S \to T$ such that $B(f) \circ \alpha_S = \alpha_T \circ f$, where the function $B(f)$, also denoted by $\mathcal{P}(A \times f)$, is defined by

$$B(f)(V) = \mathcal{P}(A \times f)(V) = \{\langle a, f(s)\rangle \mid \langle a, s\rangle \in V\}.$$

Note that B is defined both on sets and on functions. Moreover, B can be shown to preserve identities: $B(1_S) = 1_{B(S)}$, and compositions: $B(f \circ g) = B(f) \circ B(g)$. In other words, B is indeed a functor. One can easily prove that the equality $B(f) \circ \alpha_S = \alpha_T \circ f$ is equivalent to the following two conditions:

1. for all s' in S, if $s \xrightarrow{a}_S s'$ then $f(s) \xrightarrow{a}_T f(s')$;
2. for all t in T, if $f(s) \xrightarrow{a}_T t$ then there is s' in S with $s \xrightarrow{a}_T s'$ and $f(s') = t$.

Thus a homomorphism is a function that is transition preserving and reflecting. □

An F-homomorphism $f : S \to T$ with an inverse $f^{-1} : T \to S$ which is also a homomorphism is called an *isomorphism* between S and T. As usual, $S \cong T$ means that there exists an isomorphism between S and T. An injective homomorphism is called *monomorphism*. Dually, a surjective homomorphism is called *epimorphism*. Given systems S and T, we say that S can be *embedded* into T if there is a monomorphism from S to T. If there exists an epimorphism from S to T, T is called a *homomorphic image* of S. In that case, T is also called a *quotient* of S.

Remark 2.2 The above definitions are sufficient for our purposes but, more generally, monomorphisms could be defined as homomorphism that are *mono* in the category Set_F: that is, homomorphisms $f : S \to T$ such that for all homomorphisms $k : U \to S$ and $l : U \to S$: if $f \circ k = f \circ l$ then $k = l$. Clearly injective homomorphisms are mono. One can show that for a large class of functors, the converse of this statement holds as well. A dual remark applies to epimorphisms. Further details are given in Proposition 4.7. □

The following properties will be useful.

Proposition 2.3 *Every bijective homomorphism is necessarily an isomorphism.*

Proof: If $f : (S, \alpha_S) \to (T, \alpha_T)$ is an F-homomorphism and $g : T \to S$ is an inverse of f then

$$\begin{aligned} &\alpha_S \circ g \\ &\quad = \; F(g) \circ F(f) \circ \alpha_S \circ g \\ &\quad = \; F(g) \circ \alpha_T \circ f \circ g \\ &\quad = \; F(g) \circ \alpha_T, \end{aligned}$$

thus g is a homomorphism. □

Lemma 2.4 *Let S, T, and U be systems, and $f : S \to T$, $g : S \to U$, and $h : U \to T$ any functions.*

1. *If $f = h \circ g$, g is surjective, and f and g are homomorphisms, then h is a homomorphism.*
2. *If $f = h \circ g$, h is injective, and f and h are homomorphisms, then g is a homomorphism.*

Proof: We prove 1., the proof of 2. is similar. Consider $u \in U$ and let $s \in S$ be such that $g(s) = u$. Then

$$\begin{aligned}
&F(h) \circ \alpha_U(u) \\
&\quad = \; F(h) \circ \alpha_U \circ g(s) \\
&\quad = \; F(h) \circ F(g) \circ \alpha_S(s) \\
&\quad = \; F(f) \circ \alpha_S(s) \\
&\quad = \; \alpha_T \circ f(s) \\
&\quad = \; \alpha_T \circ h \circ g(s) \\
&\quad = \; \alpha_T \circ h(u).
\end{aligned}$$

□

We now come to the third fundamental notion of universal coalgebra. A *bisimulation* between two systems is intuitively a transition structure respecting relation between sets of states. Formally, it is defined, for an arbitrary set functor $F : Set \to Set$, as follows [AM89]: Let (S, α_S) and (T, α_T) be F-systems. A subset $R \subseteq S \times T$ of the Cartesian product of S and T is called an *F-bisimulation* between S and T if there exists an F-transition structure $\alpha_R : R \to F(R)$ such that the projections from R to S and T are F-homomorphisms:

$$\begin{tikzcd}
S \arrow[d, "\alpha_S"'] & R \arrow[l, "\pi_1"'] \arrow[r, "\pi_2"] \arrow[d, "\alpha_R"] & T \arrow[d, "\alpha_T"] \\
F(S) & F(R) \arrow[l, "F(\pi_1)"] \arrow[r, "F(\pi_2)"'] & F(T).
\end{tikzcd}$$

We shall also say, making explicit reference to the transition structures, that (R, α_R) is a bisimulation between (S, α_S) and (T, α_T). If $(T, \alpha_T) = (S, \alpha_S)$ then (R, α_R) is called a bisimulation *on* (S, α_S). A *bisimulation equivalence* is a bisimulation that is also an equivalence relation. Two states s and t are called *bisimilar* if there exists a bisimulation R with $\langle s, t \rangle \in R$. (See Section 17 for some references to alternative categorical approaches to bisimulation.)

Example 2.1, continued: Consider again two (labelled transition systems represented as) B-systems (S, α_S) and (T, α_T). We show that a B-bisimulation between S and T is a relation $R \subseteq S \times T$ satisfying, for all $\langle s, t \rangle \in R$,

1. for all s' in S, if $s \xrightarrow{a}_S s'$ then there is t' in T with $t \xrightarrow{a}_T t'$ and $\langle s', t' \rangle \in R$;

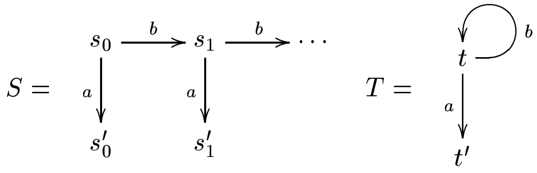
2. for all t' in T, if $t \xrightarrow{a}_T t'$ then there is s' in S with $s \xrightarrow{a}_S s'$ and $\langle s', t' \rangle \in R$,

which is the familiar definition of bisimulation from concurrency theory [Mil80, Par81]. For let R be a B-bisimulation with transition structure $\alpha_R : R \to B(R)$. As before, α_R induces a relation $\to_R \subseteq R \times A \times R$. Let $\langle s, t \rangle \in R$, and suppose $s \xrightarrow{a}_S s'$. Because $s = \pi_1 \langle s, t \rangle$ this implies $\pi_1 \langle s, t \rangle \xrightarrow{a}_S s'$, and because π_1 is a homomorphism, it follows that there is $\langle s'', t' \rangle \in R$ with $\langle s, t \rangle \xrightarrow{a}_R \langle s'', t' \rangle$ and $\pi_1 \langle s'', t' \rangle = s'$. Thus $\langle s', t' \rangle \in R$. Because π_2 is a homomorphism it follows that $t \xrightarrow{a}_T t'$, which concludes the proof of clause 1. Clause 2 is proved similarly. Conversely, suppose R satisfies clauses 1 and 2. Define $\alpha_R : R \to \mathcal{P}(R)$, for $\langle s, t \rangle \in R$, by

$$\alpha_R \langle s, t \rangle = \{ \langle s', t' \rangle \in R \mid s \xrightarrow{a}_S s' \text{ and } t \xrightarrow{a}_T t' \}.$$

It is immediate from clauses 1 and 2. that the projections are homomorphisms from (R, α_R) to (S, α_S) and (T, α_T). (Note that in general α_R is not the only transition structure on R having this property.)

A concrete example of a bisimulation relation between two transition systems is the following. Consider two systems S and T:



Then $\{\langle s_i, s_j\rangle \mid i,j \geq 0\} \cup \{\langle s_i', s_j'\rangle \mid i,j \geq 0\}$ is a bisimulation on S. And $\{\langle s_i, t\rangle \mid i \geq 0\} \cup \{\langle s_i', t'\rangle \mid i \geq 0\}$ is a bisimulation between S and T. Note that the function $f : S \to T$ defined by $f(s_i) = t$ and $f(s_i') = t'$ is a homomorphism, and that there exists no homomorphism from T to S. □

The last observation of the example above (that f is a homomorphism) is an immediate consequence of the following fundamental relationship between homomorphisms and bisimulations.

Theorem 2.5 *Let (S, α_S) and (T, α_T) be two systems. A function $f : S \to T$ is a homomorphism if and only if its graph $G(f)$ is a bisimulation between (S, α_S) and (T, α_T).*

Proof: Let $\alpha : G(f) \to F(G(f))$ be such that $(G(f), \alpha)$ is a bisimulation between (S, α_S) and (T, α_T). Let π_1 and π_2 be the projections from $G(f)$ to S and T. Because π_1 is bijective it has an inverse $\pi_1^{-1} : (S, \alpha_S) \to (G(f), \alpha)$ which is also a homomorphism. Because $f = \pi_2 \circ \pi_1^{-1}$, also f is a homomorphism.

Conversely, suppose f is a homomorphism. We can take $F(\pi_1)^{-1} \circ \alpha_S \circ \pi_1$ as a transition structure on $G(f)$. This clearly turns π_1 into a homomorphism. The same holds for π_2:

$$\begin{aligned}
&F(\pi_2) \circ (F(\pi_1)^{-1} \circ \alpha_S \circ \pi_1) \\
&\quad = \quad F(\pi_2 \circ {\pi_1}^{-1}) \circ \alpha_S \circ \pi_1 \\
&\quad = \quad F(f) \circ \alpha_S \circ \pi_1 \\
&\quad = \quad \alpha_T \circ f \circ \pi_1 \\
&\quad = \quad \alpha_T \circ \pi_2.
\end{aligned}$$

(Because $F(\pi_1)$ is mono, there is only one transition structure on $G(f)$.) □

Therefore homomorphisms are sometimes called *functional bisimulations*.

3 Systems, systems, systems, ...

The coalgebras, homomorphisms, and bisimulations of a number of functors that can be considered as the basic building blocks for most systems are described. (All functors that are used are described in Section 18.)

Deterministic systems

Deterministic systems exist in many different forms. The simplest ones are coalgebras of the identity functor $I(S) = S$:

$$\begin{tikzcd} S \arrow[d, "\alpha_S"'] \\ S, \end{tikzcd} \qquad s \longrightarrow_S s' \iff \alpha_S(s) = s'.$$

The notation $s \longrightarrow_S s'$ for $\alpha_S(s) = s'$ is used as a shorthand, which puts emphasis on the fact that α_S actually gives the *dynamics* of the system (S, α_S), and should be read as: in state s the system

S can make a transition step to state s'. The arrow notation will turn out to be particularly useful for the characterization of homomorphisms and bisimulations. Formally, the above equivalence is simply stating that any function is also a (functional) relation. Conversely, it is often convenient to define the dynamics of a system by specifying its transitions (in particular when dealing with nondeterministic systems, see below). For instance, specifying for the set of natural numbers transitions

$$0 \longrightarrow 1 \longrightarrow 2 \longrightarrow \cdots,$$

defines the deterministic system ($\mathcal{N}$, succ), where succ is the successor function.

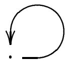
A homomorphism between two deterministic systems (S, α_S) and (T, α_T) is a function $f : S \to T$ satisfying for all s in S,

$$s \longrightarrow s' \;\Rightarrow\; f(s) \longrightarrow f(s').$$

(Note that we have dropped the subscripts from $\longrightarrow_S$ and $\longrightarrow_T$, a convention we shall often apply.) Thus homomorphisms are *transition preserving* functions. A bisimulation between deterministic systems S and T is any relation $R \subseteq S \times T$ such that, for all $s \in S$ and $t \in T$,

$$\langle s, t \rangle \in R \text{ and } s \longrightarrow s' \text{ and } t \longrightarrow t' \;\Rightarrow\; \langle s', t' \rangle \in R.$$

Thus bisimulations between deterministic systems are transition respecting relations. For instance, there is an obvious bisimulation relation between the above system ($\mathcal{N}$, succ), and the system



Not only are there many deterministic systems (take any set and any function from the set to itself), many of them have a more interesting dynamics than one would expect at first sight, in spite of the fact that the functor at stake is trivial. For instance, let A be any set (alphabet) and let $A^{\mathcal{Z}}$ be the set of all so-called bi-infinite sequences (words) over A. It can be given the following dynamics:

$$\begin{tikzcd} A^{\mathcal{Z}} \arrow[d, "\text{shift}"'] \\ A^{\mathcal{Z}}, \end{tikzcd} \qquad \text{shift}(\phi) = \lambda m.\, \phi(m+1).$$

This example is of central importance in the theory of *symbolic dynamics* (cf. [BP96]). There the set of bi-infinite words is supplied with a metric, by which the shift example becomes even more interesting. See Section 16 for some observations about such 'metric systems'.

Termination

Any set S carries a coalgebra structure of the constant functor $F(S) = 1$:

$$\begin{tikzcd} S \arrow[d, "\alpha_S"'] \\ 1, \end{tikzcd} \qquad s{\downarrow} \iff \alpha_S(s) = *,$$

where $1 = \{*\}$. Thus S can be viewed as a system with trivial dynamics, in which no state can take a step and every state s terminates, as expressed by the arrow notation $s{\downarrow}$. Any function between such systems trivially is a homomorphism and any relation a bisimulation. Thus the category Set_1 of all systems of the constant functor is just (isomorphic to) the category of sets.

Deterministic systems with termination are coalgebras of the functor $F(S) = 1 + S$:

$$\begin{tikzcd} S \arrow[d, "\alpha_S"'] \\ 1+S, \end{tikzcd} \qquad s \longrightarrow s' \iff \alpha_S(s) = s', \quad s{\downarrow} \iff \alpha_S(s) = *.$$

Such a system can in a state s either make a transition to a state s' or terminate. Homomorphisms (and bisimulations) are as before, with the additional property that terminating states are mapped to (related to) terminating states. Note that homomorphisms not only preserve but also reflect transitions: if $f : S \to T$ is a homomorphism and $f(s) \longrightarrow t$, for $s \in S$ and $t \in T$, then there exists $s' \in S$ with $s \longrightarrow s'$ and $f(s') = t$.

An example of a deterministic system with termination is the system of the *extended natural numbers* [AM82] $\bar{\mathcal{N}} = \{0, 1, 2, \ldots\} \cup \{\infty\}$, with dynamics

$$\begin{tikzcd} \infty \arrow[loop left] & \cdots \arrow[r] & 2 \arrow[r] & 1 \arrow[r] & 0{\downarrow}, \end{tikzcd}$$

which, equivalently, can be defined as

$$\begin{tikzcd} \bar{\mathcal{N}} \arrow[d, "\mathrm{pred}"'] \\ 1+\bar{\mathcal{N}}, \end{tikzcd} \qquad \mathrm{pred}(n) = \begin{cases} * & \text{if } n = 0 \\ n-1 & \text{if } 0 < n \neq \infty \\ \infty & \text{if } n = \infty. \end{cases}$$

In this system, a natural number n can take precisely n transition steps and then terminates, and the additional number ∞ only takes a step to itself and hence never terminates.

Input

Systems in which state transitions may depend on input are coalgebras of the functor $F(S) = S^A$ (recall that $S^A = \{f : A \to S\}$):

$$\begin{tikzcd} S \arrow[d, "\alpha_S"'] \\ S^A, \end{tikzcd} \qquad s \xrightarrow{a} s' \iff \alpha_S(s)(a) = s',$$

where A is any set (to be thought of as an input alphabet) and the arrow can be read as: in state s and given input a, the system can make a transition to state s'. Typical examples of deterministic systems with input are *deterministic automata*, which traditionally are represented as pairs $(Q, \delta : (Q \times A) \to Q)$, consisting of a set Q of states and a state transition function δ that for every state q and input symbol a in A determines the next state $\delta\langle q, a\rangle$. (Often an initial state and a set of final states is specified as well, but they can be dealt with separately.) As observed in the introduction, in [AM82, MA86], such automata are precisely the deterministic systems with input mentioned above, because of the following bijection:

$$\{f : Q \times A \to Q\} \cong \ \{f : Q \to Q^A\}.$$

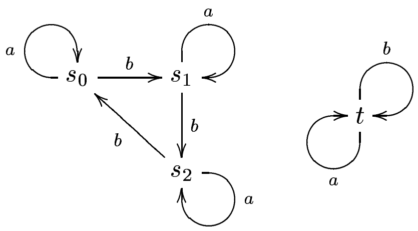
A homomorphism between (S, α_S) and (T, α_T) is any function $f : S \to T$ satisfying for all s in S, a in A,

$$s \xrightarrow{a} s' \ \Rightarrow f(s) \xrightarrow{a} f(s').$$

A bisimulation between systems S and T is now a relation $R \subseteq S \times T$ such that, for all a in A,

$$\langle s, t\rangle \in R \text{ and } s \xrightarrow{a} s' \text{ and } t \xrightarrow{a} t' \ \Rightarrow \langle s', t'\rangle \in R.$$

For instance, all states in the following two systems are bisimilar:



Output

Transitions may also yield an output. Thus we consider coalgebras of the functor $F(S) = A \times S$:

$$\begin{tikzcd} S \arrow[d, "\alpha_S"'] \\ A \times S, \end{tikzcd} \qquad s \xrightarrow{a} s' \iff \alpha_S(s) = \langle a, s' \rangle,$$

where A is an arbitrary set and the arrow can be read as: in state s, one can 'observe' the output a, and the system can make a transition to the state s'. An intuition that often applies is to consider the output a as the 'observable effect' of the state transition. (Note that the same arrow notation $\xrightarrow{a}$ is used both for transitions with input and with output. In general, the right interpretation follows from the context.) Such systems are also called *deterministic labelled transition systems* [Plo81]. Homomorphisms and bisimulations can be characterized by an obvious variation on the descriptions above. A concrete example is the set A^ω of infinite sequences over A, with

$$\begin{tikzcd} A^\omega \arrow[d, "\langle h,t \rangle"'] \\ A \times A^\omega, \end{tikzcd} \qquad \langle a_0, a_1, \ldots \rangle \xrightarrow{a_0} \langle a_1, a_2, \ldots \rangle.$$

The pair $\langle h, t \rangle$ assigns to an infinite sequence its *head* (the first element) and *tail* (the remainder). Adding the possibility of termination yields, for instance, the following two variations, where the functors involved are $F(S) = 1 + (A \times S)$ and $F(S) = A + (A \times S)$:

$$\begin{tikzcd} S \arrow[d] \\ 1 + (A \times S), \end{tikzcd} \qquad \begin{tikzcd} S \arrow[d] \\ A + (A \times S). \end{tikzcd}$$

An example of the first type is the set A^∞ of finite and infinite streams, with

$$\begin{tikzcd} A^\infty \arrow[d, "\pi"'] \\ 1 + (A \times A^\infty), \end{tikzcd} \qquad \epsilon\downarrow; \ \langle a_0, a_1, \ldots \rangle \xrightarrow{a_0} \langle a_1, a_2, \ldots \rangle.$$

Similarly, the set A^∞_+ of non-empty finite and infinite streams over A is an example of the last type, $S \to A + (A \times S)$.

Binary systems

Binary systems are coalgebras of the functor $F(S) = S \times S$. Now a transition yields two new states:

$$\begin{tikzcd} S \arrow[d, "\alpha_S"'] \\ S \times S, \end{tikzcd} \qquad s \longrightarrow \langle s_1, s_2 \rangle \iff \alpha_S(s) = \langle s_1, s_2 \rangle.$$

A homomorphism between binary systems S and T is any function $f : S \to T$ satisfying for all s in S,

$$s \longrightarrow \langle s_1, s_2 \rangle \;\Rightarrow f(s) \longrightarrow \langle f(s_1), f(s_2) \rangle.$$

Similarly for bisimulations. A concrete example of a binary system is the set $\mathcal{Z}$ of integers with transitions

$$\begin{tikzcd} \cdots & -2 \arrow[l] \arrow[r, shift left] & -1 \arrow[l, shift left] \arrow[r, shift left] & 0 \arrow[l, shift left] \arrow[r, shift left] & 1 \arrow[l, shift left] \arrow[r] & \cdots. \end{tikzcd}$$

Note that the fact that there are two outgoing transitions from each state should in this context not be interpreted as a form of nondeterminism (see below): the system is perfectly deterministic in that for each state one transition is possible, leading to a pair of new states. The system can equivalently be defined by

$$\begin{tikzcd} \mathcal{Z} \arrow[d, "\langle \text{pred, succ} \rangle"'] \\ \mathcal{Z} \times \mathcal{Z}, \end{tikzcd} \qquad m \longrightarrow \langle m-1, m+1 \rangle.$$

Variations of binary systems can be obtained by adding labels (output) and the possibility of termination:

$$\begin{tikzcd} S \arrow[d] \\ S \times A \times S, \end{tikzcd} \qquad \begin{tikzcd} S \arrow[d] \\ (A \times S) \times (A \times S), \end{tikzcd} \qquad \begin{tikzcd} S \arrow[d] \\ 1 + ((A \times S) \times (A \times S)). \end{tikzcd}$$

Examples of such systems are, respectively: the set of infinite node-labelled binary trees, where each node is assigned its label in A, together with the nodes of the two subtrees; the set of infinite *arc*-labelled binary trees, where a node is mapped to the two nodes of its subtrees, each together with the label of the corresponding arc; and the set of all arc-labelled binary trees with finite and infinite branches.

Nondeterministic systems

From one state, several transitions may be possible:

$$\begin{tikzcd} S \arrow[d, "\alpha_S"'] \\ \mathcal{P}(S), \end{tikzcd} \qquad s \longrightarrow s' \iff s' \in \alpha_S(s).$$

A variation of this type of systems is obtained by adding labels, thus considering coalgebras of the functor $F(S) = \mathcal{P}(A \times S)$:

$$\begin{tikzcd} S \arrow[d, "\alpha_S"'] \\ \mathcal{P}(A \times S), \end{tikzcd} \qquad s \xrightarrow{a} s' \iff \langle a, s' \rangle \in \alpha_S(s).$$

These are the nondeterministic labelled transition systems of Example 2.1, where homomorphisms and bisimulations have been characterized as transition preserving and reflecting functions and relations. Often one wishes to consider systems with *bounded* nondeterminism, in which from an arbitrary state only a finite number of transitions is possible. Such systems can be modelled using the finite powerset functor:

$$\begin{tikzcd} S \arrow[d, "\alpha_S"'] \\ \mathcal{P}_f(A \times S), \end{tikzcd}$$

and are called *finitely branching*. Yet another class of systems are the coalgebras of the functor $F(S) = \mathcal{P}_f(S)^A$:

$$\begin{tikzcd} S \arrow[d, "\alpha_S"'] \\ \mathcal{P}_f(S)^A, \end{tikzcd}$$

which are called *image finite*: for every s in S and a in A, the number of reachable states $\{s' \mid s \xrightarrow{a} s'\}$ is finite.

Hyper systems

The contravariant powerset functor can be used to model *hyper* systems, in which a state can make nondeterministically a step to a *set* of states:

$$\begin{tikzcd} S \arrow[d, "\alpha_S"'] \\ \bar{\mathcal{P}}(\bar{\mathcal{P}}(S)), \end{tikzcd} \qquad s \longrightarrow V \iff V \in \alpha_S(s).$$

(Recall that $\bar{\mathcal{P}}(S) = 2^S$ and thus $\bar{\mathcal{P}}(\bar{\mathcal{P}}(S)) = (2^S)^S$.) Here the arrow $s \longrightarrow V$ should be read as: from state s the system can reach the set V of states (but not necessarily each individual element of V). Using the definition of the contravariant powerset functor, one can show that a homomorphism between hyper systems S and T is any function $f : S \to T$ satisfying, for all s in S and $W \subseteq T$,

$$s \to f^{-1}(W) \iff f(s) \to W.$$

Bisimulations are generally not so easy to characterize. For the special case of a bisimulation *equivalence* $R \subseteq S \times S$ on a hypersystem S, the following holds[1]: for all s and s' in S,

$$\langle s, s' \rangle \in R \Rightarrow \quad (\text{for every } R\text{-equivalence class } V \subseteq S,\ s \to V \iff s' \to V).$$

The reader is invited to try and model hyper systems using the *co*variant powerset functor, to find that the notions of homomorphism and bisimulation are rather different in that case. This example illustrates the importance of functors, which operate both on sets and on functions, in a theory of coalgebras.

[1] This type of bisimulation seems to be underlying many of the recently proposed *probabilistic* bisimulations [LS91, vGSS95]. It was found in joint work with Erik de Vink.

More examples

Some further examples are given, using functors that combine some of the basic constructions mentioned above.

Transducers (cf. [Mil75]): essentially are (automata and hence) systems with input and output, possibly with termination, such as

$$\begin{tikzcd}
S \arrow[d] & S \arrow[d] & S \arrow[d] \\
(B \times S)^A, & B \times S^A, & C + (A \times S^B).
\end{tikzcd}$$

More traditionally (but isomorphically), transducers of the first type are represented as sets S together with a transition function of type

$$(S \times B)^{S \times A}.$$

Note that in the second type of system, the output does not depend on the input.

Graphs: A directed (1-)graph (V, E) consists of a set V of points (vertices) and an edge relation $E \subseteq V \times V$, representing the arcs of the graph. Graphs are in one-to-one correspondence with nondeterministic systems because of the bijection

$$\{f : V \to \mathcal{P}(V)\} \cong \mathcal{P}(V \times V).$$

Note that the standard notion of graph homomorphism is a function preserving the arc relation [SS93], without necessarily reflecting it. In contrast, a homomorphism of (graphs as) nondeterministic systems both preserves and reflects the arcs, as a consequence of the categorical definition of homomorphism of F-coalgebras. Nevertheless, the traditional way of representing graphs and arc-preserving homomorphisms between them can be modelled in the present framework by considering the following, so to speak many-sorted coalgebraic definition[2]: Consider the functor

$$F : (\mathit{Set} \times \mathit{Set}) \to (\mathit{Set} \times \mathit{Set}), \quad \langle X, Y \rangle \mapsto \langle 1, X \times X \rangle.$$

A graph (V, E) can be represented as a coalgebra of F by defining:

$$\begin{tikzcd}
(V, E) \arrow[d, "{(1, \langle s,t \rangle)}"] \\
(1, V \times V),
\end{tikzcd}$$

where $s : E \to V$ and $t : E \to V$ are the projections from E to V, which we call *source* and *target*. An F-homomorphism

$$\begin{tikzcd}
(V, E) \arrow[r, "{(f,g)}"] \arrow[d, "{(1, \langle s,t \rangle)}"'] & (V', E') \arrow[d, "{(1, \langle s',t \rangle)}"] \\
(1, V \times V) \arrow[r, "{(1, f \times f)}"'] & (1, V' \times V'),
\end{tikzcd}$$

is a pair of functions $f : V \to V'$ and $g : E \to E'$ such that

$$f(s(e)) = s'(g(e)), \quad f(t(e)) = t'(g(e)),$$

which is the usual definition of graph homomorphism.

Frames and models: A *frame* in the world of modal logic (cf. [Gol87]) is a directed graph, and thus (as we have seen above) can be represented as a nondeterministic system. A *model* (V, E, f) is

[2] This definition was suggested by Andrea Corradini.

a frame (V, E) together with a function $f : \Phi \rightarrow \mathcal{P}(V)$, where Φ is a collection of atomic formulas in some given modal logic. Intuitively, f specifies for each formula in which states v in V it holds. Because of the isomorphism

$$\{f : \Phi \rightarrow \mathcal{P}(V)\} \cong \{f : V \rightarrow \mathcal{P}(\Phi)\},$$

it is easily verified that models correspond to systems of type:

$$\begin{tikzcd} V \arrow[d] \\ \mathcal{P}(\Phi) \times \mathcal{P}(V). \end{tikzcd}$$

As it turns out, homomorphisms and bisimulations for these systems correspond precisely to the so-called *p-morphisms* and *zig-zag relations* of modal logic.

Resumptions: are systems of type

$$\begin{tikzcd} S \arrow[d] \\ (\mathcal{P}(B \times S))^A. \end{tikzcd}$$

In other words, resumptions are nondeterministic systems with input and output. They play a central role in the semantics of (nondeterministic and parallel) programming languages (cf. [HP79, BV96]).

4 Limits and colimits of systems

We want to prove statements like: the union of a collection of bisimulations is again a bisimulation; the quotient of a system with respect to a bisimulation equivalence is again a system; and: the kernel of a homomorphism is a bisimulation equivalence. These facts are well-known for certain systems such as nondeterministic labelled transition systems. As it turns out, they do not depend on particular properties of such examples, and actually apply to (almost) all systems we have seen sofar. Therefore, this section presents a number of basic categorical constructions that will enable us, in the subsequent sections, to prove all these statements for all systems at the same time.

There are three basic constructions in the category Set_F of F-systems that are needed: the formation of coproducts (sums), coequalizers, and pullbacks (cf. Section 18). In this section, they are discussed in some detail for arbitrary F-systems. The family of labelled transition systems is used again as a running example.

(We shall see that in Set_F coproducts and coequalizers exist, for arbitrary functors F. If the functor F preserves pullbacks, then pullbacks exist in Set_F. For completeness, a general description of limits and colimits of systems is presented at the end of this section. In summary, Set_F has all colimits, for any F. And Set_F has all limits that are preserved by F.)

Coproducts

Coproducts (as well as coequalizers and, more generally any type of colimit) in Set_F are as easy as they are in the category *Set*. The *coproduct* (or sum) of two F-systems (S, α_S) and (T, α_T) can be constructed as follows. Let $i_S : S \rightarrow (S+T)$ and $i_T : T \rightarrow (S+T)$ be the injections of S and T into their disjoint union. It is easy to see that there is a unique function $\gamma : (S+T) \rightarrow F(S+T)$ such that both i_S and i_T are homomorphisms:

$$\begin{tikzcd} S \arrow[r, "i_S"] \arrow[d, "\alpha_S"'] & S+T \arrow[d, dashed, "\gamma"] & T \arrow[l, "i_T"'] \arrow[d, "\alpha_T"] \\ F(S) \arrow[r, "F(i_S)"'] & F(S+T) & F(T). \arrow[l, "F(i_T)"] \end{tikzcd}$$

The function γ acts on S as $F(i_S) \circ \alpha_S$ and on T as $F(i_T) \circ \alpha_T$. The system $(S+T,\, \gamma)$ has the following universal property: for any system (U, α_U) and homomorphisms $k : (S, \alpha_S) \to (U, \alpha_U)$ and $l : (T, \alpha_T) \to (U, \alpha_U)$ there exists a unique homomorphism $h : (S+T,\, \gamma) \to (U, \alpha_U)$ making the following diagram commute:

$$\begin{tikzcd}
& U & \\
S \arrow[ur, "k"] \arrow[r, "i_S"'] & S+T \arrow[u, dashed, "h"] & T. \arrow[ul, "l"'] \arrow[l, "i_T"]
\end{tikzcd}$$

That is $(S+T,\, \gamma)$ is the coproduct of (S, α_S) and (T, α_T). Similarly, the coproduct of an indexed family $\{S_i\}_{i \in I}$ of systems can be constructed.

Example 4.1 Recall from Example 2.1 that labelled transition systems (lts) are B-systems where $B(X) = \mathcal{P}(A \times X)$. The coproduct of two lts's (S, α_S) and (T, α_T) consists of the disjoint union $S+T$ of the sets of states together with a B-transition structure $\gamma : S+T \to B(S+T)$, defined for s in S and t in T by

$$\gamma(s) = \alpha_S(s), \quad \gamma(t) = \alpha_T(t).$$

Because $A \times S \subseteq A \times (S+T)$ and $A \times T \subseteq A \times (S+T)$ (identifying for convenience $S+T$ and $S \cup T$), this defines indeed a function from $S+T$ into $B(S+T)$. □

Coequalizers

Next we show how in Set_F a coequalizer of two homomorphisms can be constructed. Consider two homomorphisms $f : (S, \alpha_S) \to (T, \alpha_T)$ and $g : (S, \alpha_S) \to (T, \alpha_T)$. We have to find a system (U, α_U) and a homomorphism $h : (T, \alpha_T) \to (U, \alpha_U)$ such that

1. $h \circ f = h \circ g$;
2. for every homomorphism $h' : (T, \alpha_T) \to (U', \alpha_{U'})$ such that $h' \circ f = h' \circ g$, there exists a unique homomorphism $l : (U, \alpha_U) \to (U', \alpha_{U'})$ with the property that $l \circ h = h'$.

Since (per definition) f and g are functions $f : S \to T$ and $g : S \to T$ in Set, there exists a coequalizer $h : T \to U$ in Set (see Section 18). Consider $F(h) \circ \alpha_T : T \to F(U)$. Because

$$\begin{aligned}
&F(h) \circ \alpha_T \circ f \\
&\quad = \quad F(h) \circ F(f) \circ \alpha_S \\
&\quad = \quad F(h \circ f) \circ \alpha_S \\
&\quad = \quad F(h \circ g) \circ \alpha_S \\
&\quad = \quad F(h) \circ F(g) \circ \alpha_S \\
&\quad = \quad F(h) \circ \alpha_T \circ g,
\end{aligned}$$

and $h : T \to U$ is a coequalizer, there exists a unique function $\alpha_U : U \to F(U)$ making the following diagram commute:

$$\begin{tikzcd}
S \arrow[r, shift left, "f"] \arrow[r, shift right, "g"'] \arrow[d, "\alpha_S"] & T \arrow[r, "h"] \arrow[d, "\alpha_T"'] & U \arrow[d, dashed, "\alpha_U"] \\
F(S) \arrow[r, shift left, "F(f)"] \arrow[r, shift right, "F(g)"'] & F(T) \arrow[r, "F(h)"] & F(U)
\end{tikzcd}$$

Thus (U, α_U) is an F-system and h is a homomorphism. One easily checks that the universal property (2) is satisfied.

Example 4.1, continued: Let (S, α_S) and (T, α_T) be again two lts's and consider homomorphisms $f,\, g : (S, \alpha_S) \to (T, \alpha_T)$. Let R be the smallest equivalence relation on T that contains the set

$$\{\langle f(s), g(s)\rangle \mid s \in S\},$$

and let $q : T \to T/R$ be the function that maps t in T to its R-equivalence class $[t]_R$. Then T/R can be supplied with a B-transition structure $\alpha_R : T/R \to B(T/R)$ by specifying transitions

$$[t]_R \xrightarrow{a} [t']_R \iff \exists t'' \in [t']_R,\, t \xrightarrow{a}_T t''.$$

It is moreover the only possible choice for α_R making $q : T \to T/R$ into a homomorphism. A special instance of this example is obtained by taking a bisimulation equivalence on a B-system, say

$$\pi_1,\, \pi_2 : (R, \alpha_R) \to (T, \alpha_T).$$

Then the coequalizer of π_1 and π_2 is the quotient T/R, showing that the quotient of an lts with respect to a bisimulation equivalence yields again an lts. This observation will be generalized in Proposition 5.8. □

The results above are summarized for future reference in the following.

Theorem 4.2 *Let $F : Set \to Set$ be any functor. In the category Set_F of F-systems, all coproducts and all coequalizers exist, and are constructed as in Set.* □

(Weak) pullbacks

The existence of *pullbacks* in Set_F depends on the functor F. More specifically, if $F : Set \to Set$ preserves pullbacks then pullbacks exist in Set_F: Let $f : (S, \alpha_S) \to (T, \alpha_T)$ and $g : (U, \alpha_U) \to (T, \alpha_T)$ be homomorphisms. Let

$$\begin{tikzcd} P \arrow[r, "\pi_1"] \arrow[d, "\pi_2"'] & S \arrow[d, "f"] \\ U \arrow[r, "g"'] & T \end{tikzcd}$$

be the pullback of f and g in *Set*, with $P = \{\langle s, u\rangle \mid f(s) = g(u)\}$. Because F preserves pullbacks,

$$\begin{tikzcd} F(P) \arrow[r, "F(\pi_1)"] \arrow[d, "F(\pi_2)"'] & F(S) \arrow[d, "F(f)"] \\ F(U) \arrow[r, "F(g)"'] & F(T) \end{tikzcd}$$

is a pullback of $F(f)$ and $F(g)$ in *Set*. Consider $\alpha_S \circ \pi_1 : P \to F(S)$ and $\alpha_U \circ \pi_2 : P \to F(U)$. Because

$$\begin{aligned} & F(f) \circ \alpha_S \circ \pi_1 \\ = \; & \alpha_T \circ f \circ \pi_1 \\ = \; & \alpha_T \circ g \circ \pi_2 \\ = \; & F(g) \circ \alpha_U \circ \pi_2, \end{aligned}$$

there exists, by the fact that $F(P)$ is a pullback, a unique function $\alpha_P : P \to F(P)$ such that $F(\pi_1) \circ \alpha_P = \alpha_S \circ \pi_1$ and $F(\pi_2) \circ \alpha_P = \alpha_U \circ \pi_2$. Thus (P, α_P) is an F-system, and π_1 and π_2 are homomorphisms. It is easily verified that (P, α_P) is a pullback of f and g in Set_F.

Note that as a consequence, the pullback (P, α_P) is a bisimulation on S and U: $P \subseteq S \times U$ and the projections π_1 and π_2 are homomorphisms.

As it turns out, the pullback of two homomorphisms is a bisimulation even in the case that F only preserves *weak* pullbacks (cf. Section 18).

Theorem 4.3 *Let $F : Set \rightarrow Set$ be a functor that preserves weak pullbacks, and let $f : (S, \alpha_S) \rightarrow (T, \alpha_T)$ and $g : (U, \alpha_U) \rightarrow (T, \alpha_T)$ be homomorphisms of F-systems. Then the pullback $(P,\ \pi_1,\ \pi_2)$ of f and g in Set is a bisimulation on S and T.*

Proof: The proof is essentially the same as the proof of the existence of pullbacks in Set_F in case F preserves pullbacks. The only difference is that $F(P)$ is now, by assumption, a *weak* pullback. As a consequence, there exists again a (no longer necessarily unique) transition structure $\alpha_P : P \rightarrow F(P)$ on P such that π_1 and π_2 are homomorphisms. □

Example 4.1, continued: Let $f : (S, \alpha_S) \rightarrow (T, \alpha_T)$ and $g : (U, \alpha_U) \rightarrow (T, \alpha_T)$ be homomorphisms of lts's. Because lts's are B-systems and the functor B preserves weak pullbacks (cf. Section 18), the above argumentation applies. The following gives a more direct construction. As above, let $P = \{\langle s, u \rangle \mid f(s) = g(u)\}$. It can be supplied with a B-transition structure by specifying transitions

$$\langle s, u \rangle \xrightarrow{a} \langle s', u' \rangle \iff f(s') = g(u') \text{ and } s \xrightarrow{a}_S s' \text{ and } u \xrightarrow{a}_U u'.$$

It is straightforward to prove that the projections from P to S and U are homomorphisms. Thus P is a bisimulation. A special case is obtained by taking only one homomorphism $f : (S, \alpha_S) \rightarrow (T, \alpha_T)$ and considering the pullback of f and f. The resulting set is $P = \{\langle s, s' \rangle \mid f(s) = f(s')\}$, which is the *kernel* of f. It follows that it is a bisimulation (equivalence). Again, this will be proved in greater generality in Proposition 5.7. □

Because this result will be called upon time and again, and because all functors we have seen in the examples sofar do preserve weak pullbacks (but for the contravariant powerset functor, cf. Section 18), we shall assume in the sequel that when talking about an arbitrary functor F, it preserves weak pullbacks:

Convention 4.4 *In the rest of this paper, set functors $F : Set \rightarrow Set$ are assumed to preserve weak pullbacks. If (the proof of) a lemma, proposition, or theorem actually makes use of this assumption, then it is marked with an asterisk.* □

Limits and colimits, generally

This section is concluded with the observation that the above constructions of coproducts, coequalizers, and pullbacks can be generalized by means of the so-called forgetful functor $U : Set_F \rightarrow Set$, which sends systems to their carrier: $U(S, \alpha_S) = S$, and homomorphisms $f : (S, \alpha_S) \rightarrow (T, \alpha_T)$ to the function $f : S \rightarrow T$. (see, e.g., [Bar93]).

Theorem 4.5 *The functor $U : Set_F \rightarrow Set$ creates colimits. This means that any type of colimit in Set_F exists, and is obtained by first constructing the colimit in Set and next supplying it (in a unique way) with an F-transition structure.* □

Similarly, there is the following general statement about limits in Set_F.

Theorem 4.6 *If $F : Set \rightarrow Set$ preserves a (certain type of) limit, then the functor $U : Set_F \rightarrow Set$ creates that (type of) limit. This means that any type of limit in Set that is preserved by F also exists in Set_F, and is obtained by first constructing the limit in Set and next supplying it (in a unique way) with an F-transition structure.* □

Epi's and mono's in Set_F

Using the results of this section, we are now in a position to supply the details announced in Remark 2.2 about epi's and mono's in the category Set_F of F-systems.

Proposition 4.7 *Let $F : Set \to Set$ be a functor and $f : (S, \alpha_S) \to (T, \alpha_T)$ an F-homomorphism.*

1. *The homomorphism f is an epimorphism (i.e., surjective) if and only if f is epi in the category Set_F.*
2. *If the homomorphism f is a monomorphism (i.e., injective) then it is mono in the category Set_F. If the functor F preserves weak pullbacks then the converse is also true: if f is mono then it is injective.*

Proof: We use the following categorical characterization of epi's [Bor94][Proposition 2.5.6]. Let C be an arbitrary category. An arrow $a : A \to B$ in C is epi if and only if the following diagram is a pushout in C:

$$\begin{tikzcd}
A \arrow[r, "a"] \arrow[d, "a"'] & B \arrow[d, "1_B"] \\
B \arrow[r, "1_B"'] & B.
\end{tikzcd}$$

By Theorem 4.5, the forgetful functor $U : Set_F \to Set$ creates colimits and hence pushouts. Moreover it is easily verified that U preserves any colimit that it creates. So in particular U preserves pushouts. Thus we obtain the following equivalence:

$$\begin{tikzcd}
(S, \alpha_S) \arrow[r, "f"] \arrow[d, "f"'] & (T, \alpha_T) \arrow[d, "1_T"] \\
(T, \alpha_T) \arrow[r, "1_T"'] & (T, \alpha_T)
\end{tikzcd} \text{ is a pushout in } Set_F \iff \begin{tikzcd}
S \arrow[r, "f"] \arrow[d, "f"'] & T \arrow[d, "1_T"] \\
T \arrow[r, "1_T"'] & T
\end{tikzcd} \text{ is a pushout in } Set.$$

As a consequence, the homomorphism f is epi in Set_F if and only if the function f is epi, and hence surjective, in Set.

For mono's, the dual of the characterization for epi's can be used. Let C be an arbitrary category. An arrow $a : A \to B$ in C is mono if and only if the following diagram is a pullback in C:

$$\begin{tikzcd}
A \arrow[r, "1_A"] \arrow[d, "1_A"'] & A \arrow[d, "a"] \\
A \arrow[r, "a"'] & B.
\end{tikzcd}$$

Assume that F preserves weak pullbacks. Because weak pullbacks of mono's are ordinary pullbacks (Proposition 18.2), F preserves pullbacks of mono's. By Theorem 4.6, the forgetful functor $U : Set_F \to Set$ creates such pullbacks. Again one can easily prove that U also preserves them. The argument can now be finished as in the case of epi's. □

5 Basic facts on bisimulations

This section deals with arbitrary F-systems. All results are straightforward for the special case of labelled transition systems. In fact, some of them have already been proved for lts's in Example 4.1.

Let S, T and U be three F-systems with transition structures α_S, α_T and α_U, respectively.

Proposition 5.1 *The diagonal Δ_S of a system S is a bisimulation.*

Proof: Follows from Theorem 2.5 and the observation that Δ_S equals the graph of the identity $1_S : S \to S$. □

The inverse of a bisimulation is a bisimulation.

Theorem 5.2 *Let (R, α_R) be a bisimulation between systems S and T. The inverse R^{-1} of R is a bisimulation between T and S.*

Proof: Let $i : R \to R^{-1}$ be the isomorphism sending $\langle s,t\rangle \in R$ to $\langle t,s\rangle \in R^{-1}$. Then $(R^{-1}, F(i) \circ \alpha_R \circ i^{-1})$ is a bisimulation between T and S. □

Next we show that the composition and union of bisimulations is again a bisimulation. The following lemma will be helpful.

Lemma 5.3 *The image $\langle f,g\rangle(T) = \{\langle f(t), g(t)\rangle \mid t \in T\}$ of two homomorphisms $f : T \to S$ and $g : T \to U$ is a bisimulation on S and U.*

Proof: Consider the following diagram:

$$\begin{tikzcd}
 & \langle f,g\rangle(T) \arrow[dl, "\pi_1"'] \arrow[dr, "\pi_2"] \arrow[d, shift left, "i"] & \\
S & T \arrow[l, "f"] \arrow[r, "g"'] \arrow[u, shift left, "j"] & U,
\end{tikzcd}$$

where the function j is defined by $j(t) = \langle f(t), g(t)\rangle$, the function i is any right inverse for j (which exists by the axiom of choice because j is surjective): $j \circ i = 1$, and π_1 and π_2 are projections. Note that everything in this diagram commutes. The set $\langle f,g\rangle(T)$ can be given a transition structure $\gamma : \langle f,g\rangle(T) \to F(\langle f,g\rangle(T))$ by defining

$$\gamma = F(j) \circ \alpha_T \circ i.$$

It follows that $(\langle f,g\rangle(T), \gamma)$ is a bisimulation on S and U because

$$\begin{aligned}
&F(\pi_1) \circ \gamma \\
&\quad = \quad F(\pi_1) \circ F(j) \circ \alpha_T \circ i \\
&\quad = \quad F(\pi_1 \circ j) \circ \alpha_T \circ i \\
&\quad = \quad F(f) \circ \alpha_T \circ i \\
&\quad = \quad \alpha_S \circ f \circ i \\
&\quad = \quad \alpha_S \circ \pi_1,
\end{aligned}$$

and similarly for π_2. □

Theorem* 5.4 [3] *The composition $R \circ Q$ of two bisimulations $R \subseteq S \times T$ and $Q \subseteq T \times U$ is a bisimulation between S and U.*

Proof: Recall from Section 18 that $R \circ Q$ is equal to the image $\langle r_1 \circ x_1, q_2 \circ x_2\rangle(X)$ of the pullback:

$$\begin{tikzcd}
 & & X \arrow[dl, "x_1"'] \arrow[dr, "x_2"] & & \\
 & R \arrow[dl, "r_1"'] \arrow[dr, "r_2"] & & Q \arrow[dl, "q_1"'] \arrow[dr, "q_2"] & \\
S & & T & & U.
\end{tikzcd}$$

[3] Recall from Convention 4.4 that the asterisk indicates the assumption that the functor F preserves weak pullbacks.

(Here x_i, r_i, and q_i are projections.) Because F preserves weak pullbacks, the pullback X can be supplied with a transition structure, by Theorem 4.3. By Lemma 5.3, $R \circ Q$ is a bisimulation on S and U. □

Similarly, the union of bisimulations is again a bisimulation.

Theorem 5.5 *The union $\bigcup_k R_k$ of a family $\{R_k\}_k$ of bisimulations between systems S and T is again a bisimulation.*

Proof: Recall from Section 18 that $\bigcup_k R_k$ is the image of

$$S \xleftarrow{k} \textstyle\sum_i R_i \xrightarrow{l} T,$$

where k and l are the componentwise projections. By Theorem 4.2, the coproduct of a family of systems is again a system. It follows from Lemma 5.3 that the union is a bisimulation. □

Corollary 5.6 *The set of all bisimulations between systems S and T is a complete lattice, with least upperbounds and greatest lowerbounds given by*

$$\bigvee_k R_k = (\bigcup_k R_k),$$

$$\bigwedge_k R_k = \bigcup \{R \mid R \textit{ is a bisimulation between } S \textit{ and } T \textit{ with } R \subseteq \bigwedge_k R_k\}.$$

In particular, the greatest bisimulation between S and T exists, and is denoted by $\sim_{\langle S,T \rangle}$. It is the union of all bisimulations:

$$\sim_{\langle S,T \rangle} = \bigcup \{R \mid R \textit{ is a bisimulation between } S \textit{ and } T \ \}.$$

We shall write

$$s \sim_{\langle S,T \rangle} t \iff \langle s,t \rangle \in \sim_{\langle S,T \rangle} \iff \textit{there exists a bisimulation } R \textit{ with } \langle s,t \rangle \in R.$$

The greatest bisimulation on one single system S, denoted by $\sim_S$, is a bisimulation equivalence. □

We shall simply write $\sim$ for the greatest bisimulation relation when the systems are clear from the context. Moreover, we write $\sim_F$ when explicit reference to the type of systems is needed.

Bisimulation equivalences and homomorphisms are related by the following two propositions.

Proposition* 5.7 *The kernel $K(f)$ of a homomorphism $f : S \to T$ is a bisimulation equivalence.*

Proof: Note that $K(f) = G(f) \circ G(f)^{-1}$. $K(f)$ is transitive. The result follows from Theorem 5.4. An alternative proof consists of the observation that $K(f)$ is a pullback of f with itself and then applying Theorem 4.3. □

Conversely, any bisimulation equivalence on a system is the kernel of a homomorphism:

Proposition 5.8 *Let R be a bisimulation equivalence on a system S. Let $\epsilon_R : S \to S/R$ be the quotient map of R. Then there is a unique transition structure $\alpha_{S/R} : S/R \to F(S/R)$ on S/R such that $\epsilon_R : S \to S/R$ is a homomorphism:*

$$\begin{tikzcd}
S \arrow[r, "\epsilon_R"] \arrow[d, "\alpha_S"'] & S/R \arrow[d, dashed, "\alpha_{S/R}"] \\
F(S) \arrow[r, "F(\epsilon_R)"'] & F(S/R).
\end{tikzcd}$$

Proof: Immediate from the observation that ϵ_R is a coequalizer of the projections from R to S and Theorem 4.2. Alternatively and more concretely, $\alpha_{S/R}$ can be defined on an R-equivalence class by $F(\epsilon_R) \circ \alpha_S(s)$, where s is any element of the equivalence class. □

The following facts will be useful.

Proposition* 5.9 *Let $f : S \to T$ be a homomorphism.*

1. *If $R \subseteq S \times S$ is a bisimulation on S, then $f(R)$ is a bisimulation on T.*
2. *If $Q \subseteq T \times T$ is a bisimulation on T, then $f^{-1}(Q)$ is a bisimulation on S.*

Proof: Immediate from Theorem 5.4, and the observation that $f(P) = G(f)^{-1} \circ P \circ G(f)$ and $f^{-1}(Q) = G(f) \circ Q \circ G(f)^{-1}$. □

6 Subsystems

Let (S, α_S) be a system and let V be a subset of S with inclusion mapping $i : V \to S$. If there exists a transition structure α_V on V such that $i : (V, \alpha_V) \to (S, \alpha_S)$ is a homomorphism, then V is called a *subsystem* (or subcoalgebra) of S. There is at most one such transition structure.

Proposition 6.1 *Let (S, α_S) be a system and let $i : V \to S$ be a subset of S. If $k, l : V \to F(V)$ are such that i is a homomorphism both from (V, k) to (S, α_S) and from (V, l) to (S, α_S), then $k = l$.*

Proof: If V is non-empty, the equality follows from $F(i) \circ k = \alpha_S \circ i = F(i) \circ l$ and the fact that $F(i)$ is mono, by Proposition 18.1. The case that $V = \emptyset$ is trivial. □

For instance, a subsystem of a labelled transition system (Example 2.1) is a set of states that is closed under (outgoing) transitions; subsystems of graphs are (full) subgraphs; and subsystems of trees are subtrees.

The empty set and S are always subsystems of (S, α_S). A system is called *minimal* if it does not have any proper subsystem (i.e., different from $\emptyset$ and S).

Subsystems can be characterized in terms of bisimulations as follows.

Proposition* 6.2 *Let S be a system. A subset $V \subseteq S$ is a subsystem if and only if the diagonal Δ_V of V is a bisimulation on S.*

Proof: Let $i : V \to S$ be the inclusion homomorphism of a subsystem V in S. Because Δ_V is equal to $G(i)$ (the graph of i), it is a bisimulation by Theorem 2.5. For the converse, suppose that $\Delta_V = G(i)$ is a bisimulation on (S, α_S). Because $\pi_1 : G(i) \to V$ is an isomorphism, the transition structure on $G(i)$ induces a transition structure on V. □

Theorem* 6.3 *Let S and T be two systems and $f : S \to T$ a homomorphism.*

1. *If $V \subseteq S$ is a subsystem of S, then $f(V)$ is a subsystem of T.*
2. *If $W \subseteq T$ is a subsystem of T, then $f^{-1}(W)$ is a subsystem of S.*

Proof: The theorem follows, by Proposition 5.9 and Proposition 6.2, from the observation that

$$\Delta_{f(V)} = f(\Delta_V), \quad \text{and} \quad \Delta_{f^{-1}(W)} = f^{-1}(\Delta_W).$$

□

Unions and intersections of subsystems are again subsystems.

Theorem* 6.4 *The collection of all subsystems of a system S is a complete lattice, in which least upperbounds and greatest lowerbounds are given by union and intersection.*

Proof: Let $\{V_k\}_k$ be a collection of subsystems of a system S.

$\bigcup_k V_k$**:** For every k, the set Δ_{V_k} is a bisimulation by Proposition 6.2. Because

$$\Delta_{\bigcup_k V_k} = \bigcup_k \Delta_{V_k},$$

it follows from Theorem 5.5 that it is a bisimulation. Thus $\bigcup_k V_k$ is a subsystem, again by Proposition 6.2.

$\bigcap_k V_k$**:** By Proposition 18.3, F preserves intersections. More specifically, F transforms the pullback diagram of the intersection of the sets $\{V_k\}_k$ into a pullback diagram of the sets $F(\{V_k\}_k)$ (see the proof of Proposition 18.3). It follows from Theorem 4.6 that there exists a (unique) transition structure on $\bigcap_k V_k$ such that the inclusion mapping from $\bigcap_k V_k$ to S is a homomorphism. □

Theorem 6.4 allows us to give the following definitions. Let (S, α_S) be a system and X a subset of S. The subsystem of (S, α_S) *generated* by X, denoted by $\langle X \rangle$, is defined as

$$\langle X \rangle = \bigcap \{ V \mid V \text{ is a subsystem of } S \text{ and } X \subseteq V \}.$$

So $\langle X \rangle$ is the *smallest* subsystem of S containing X. If $S = \langle X \rangle$ for some subset X of S then S is said to be *generated* by X. The subsystem generated by a singleton set $\{s\}$ is denoted by $\langle s \rangle$.

Dually, one can also look at the greatest subsystem $[X]$ of S that is *contained* in X: using again Theorem 6.4, it is defined by

$$[X] = \bigcup \{ V \mid V \text{ is a subsystem of } S \text{ and } V \subseteq X \}.$$

There is the following characterization, which will be of use in the sequel.

Proposition 6.5 *Let X be a subset of a system S and $i : [X] \to S$ the inclusion homomorphism. Any homomorphism $f : T \to S$ such that $f(T) \subseteq X$, factorizes through $[X]$. That is, there exists a unique homomorphism $f' : T \to [X]$ such that*

$$\begin{tikzcd} T \arrow[r, "f"] \arrow[dr, "f'"'] & S \\ & {[X].} \arrow[u, dashed, two heads, "i"'] \end{tikzcd}$$

Proof: By Theorem 6.3, $f(T)$ is a subsystem of S and since $f(T) \subseteq X$, by assumption, it follows that $f(T) \subseteq [X]$. Defining $f'(t) = f(t)$ gives us a function with $i \circ f' = f$. It is a homomorphism by Lemma 2.4. It is unique because i is mono. □

Examples 6.6 Some examples of subsystems.

1. Let $(S,\ \alpha_S)$ be a labelled transition system (Example 2.1). The subsystem $\langle s \rangle$ generated by an element s in S consists of all states

$$\bigcup_{n \geq 0} \{ s' \mid \exists s_0, \ldots, s_n,\ s = s_0 \longrightarrow \cdots \longrightarrow s_n = s' \}.$$

2. Recall from Section 3 that a directed graph is a system of type

$$\begin{tikzcd} S \arrow[d, "\alpha"'] \\ \mathcal{P}(S), \end{tikzcd} \qquad s \longrightarrow s' \iff s' \in \alpha(s).$$

One can define the largest subsystem $C(S)$ of S in which all states have a self cycle, by

$$C(S) = [\{s \in S \mid s \longrightarrow s\}].$$

Generally $C(S)$ is a strict subset of $\{s \in S \mid s \longrightarrow s\}$. For instance, if $S = \{s, s'\}$ with transitions $s \longrightarrow s$ and $s \longrightarrow s'$, then the subsystem $C(S)$ is empty. □

This section is concluded with a note on the size of subsystems generated by one element. For future reference, we give the following definition.

Definition 6.7 A functor F *bounded* if there exists a set V such that for every F-system (S, α_S) and every s in S, there exists an injective function from the carrier of the subsystem $\langle s \rangle$ into the set V (cf. [KM96]). □

In other words, the size of any subsystem generated by one element is bounded by the size of V. As we shall see in Section 10, this condition is sufficient to guarantee the existence of a final F-coalgebra.

Examples 6.8 Two examples of functors that are bounded, and one of a functor that is not.

1. $\mathcal{P}_f(S) = \{V \mid V \subseteq S \text{ and } V \text{ is finite }\}$: Let (S, α_S) be a $\mathcal{P}_f$-system and s in S. For any n, there are only finitely many reachable states. Therefore $\langle s \rangle$ has at most countably many elements, and can be embedded in $\mathcal{N}$. More generally, any type of powerset functor $\mathcal{P}_\kappa$, which assigns to a set the set of all subsets with cardinality less than or equal to a given cardinal κ, is bounded.

2. $F(S) = A \times (B \rightarrow S)$: Let (S, α_S) be an F-system and s in S. If κ is the size (cardinality) of B then the number of elements of $\langle s \rangle$ is bounded by $\sum\{\kappa^i \mid i \geq 0\}$. Any set with at least that number of elements is a bound for F.

3. $\mathcal{P}$: The unrestricted powerset is not bounded. □

7 Three isomorphism theorems

This section contains three theorems, which are well-known in universal algebra, on the existence of isomorphisms between systems.

The first isomorphism theorem states that any homomorphism factorizes through a pair consisting of an epimorphism and a monomorphism.

Theorem* 7.1 *(First isomorphism theorem)*
Let $f : S \rightarrow T$ be a homomorphism. Then there is the following factorization of f:

$$\begin{tikzcd}
S \arrow[rr, "f"] \arrow[dr, "f'"'] \arrow[ddr, bend right, "\epsilon_{K(f)}"'] & & T \\
& f(S) \arrow[ur, "i"'] & \\
& S/K(f), \arrow[u, "\cong"'] \arrow[uur, bend right, "\mu"'] &
\end{tikzcd}$$

where i is the inclusion monomorphism of $f(S)$ in T, μ is a monomorphism, f' is an epimorphism (with $f'(s) = f(s)$ for all s), and $\epsilon_{K(f)}$ is the quotient map of the kernel $K(f)$ of f.

Proof: By Theorem 6.3, $f(S)$ is a subsystem of T. It follows from Lemma 2.4 that f' is a homomorphism, and because it is surjective, it is an epimorphism. By Proposition 5.7, $K(f)$ is a bisimulation equivalence on S, and by (the proof of) Proposition 5.8, $S/K(f)$ is the coequalizer of the projection homomorphism of $K(f)$. The homomorphisms from $S/K(f)$ to $f(S)$ and T are given by the coequalizer property. Since the former is bijective, it is an isomorphism by Proposition 2.3. The latter is a monomorphism because i is. □

Theorem* 7.2 *Let $f : S \to T$ be a homomorphism and R a bisimulation equivalence on S which is contained in the kernel of f. Then there is a unique homomorphism $\bar{f} : S/R \to T$ such that $f = \bar{f} \circ \epsilon_R$:*

$$\begin{tikzcd}
S \arrow[r, "\epsilon_R"] \arrow[dr, "f"'] & S/R \arrow[d, dashed, "\bar{f}"] \\
& T
\end{tikzcd}$$

Proof: There is precisely one function $\bar{f}$ for which $\bar{f} \circ \epsilon_R = f$. It follows from Lemma 2.4 that it is a homomorphism. Alternatively, the existence of the homomorphism $\bar{f}$ is given by fact that S/R is a coequalizer of the projection homomorphisms from R to S. □

The second isomorphism theorem states that there is a ono-to-one correspondence between subsystems of a quotient of a system S and quotients of subsystems of S.

Theorem* 7.3 *(Second isomorphism theorem)*
Let S be a system, T a subsystem of S, and R a bisimulation equivalence on S. Let T^R be defined by $T^R = \{s \in S \mid \exists t \in T,\ (s,t) \in R\}$. The following facts hold:

1. *T^R is a subsystem of S.*
2. *$Q = R \cap (T \times T)$ is a bisimulation equivalence on T.*
3. *$T/Q \cong T^R/R$.*

Proof: Since $T^R = \pi_1({\pi_2}^{-1}(T))$, it is a subsystem of S by Theorem 6.3. Because $R \cap (T \times T) = \pi_1^{-1}(T) \cap \pi_2^{-1}(T)$, it is a subsystem of R, by the same theorem, and hence it is a bisimulation equivalence on T. Consider the quotient homomorphism $\epsilon_R : S \to S/R$, and let $\epsilon : T \to S/R$ be its restriction to T. Because $\epsilon(T) = \epsilon_R(T^R) = T^R/R$, and $K(\epsilon) = Q$, it follows from Theorem 7.1 that $T/Q \cong T^R/R$. □

Let S be a system, T a subsystem of S, and R a bisimulation equivalence on S. If $R \cap (T \times T) = \Delta_T$ then R is said to *separate* T (because, equivalently: for all $t, t' \in T$, if $t \neq t'$ then $(t,t') \notin R$). In this case, the above theorem yields that $T \cong T^R/R$.

Theorem* 7.4 *(Third isomorphism theorem)*
Let S be a system, and let R and Q be bisimulation equivalences on S such that $R \subseteq Q$. There is a unique homomorphism $\theta : S/R \to S/Q$ such that $\theta \circ \epsilon_R = \epsilon_Q$:

$$\begin{tikzcd}
S \arrow[r, "\epsilon_R"] \arrow[dr, "\epsilon_Q"'] & S/R \arrow[d, dashed, "\theta"] \\
& S/Q.
\end{tikzcd}$$

Let R/Q denote the kernel of θ: it is a bisimulation equivalence on S/R and induces an isomorphism $\theta' : (S/R)/(R/Q) \to S/Q$ such that $\theta = \theta' \circ \epsilon_{R/Q}$:

$$\begin{tikzcd}
S/R \arrow[r, "\epsilon_{R/Q}"] \arrow[d, "\theta"'] & (S/R)/(R/Q) \arrow[dl, dashed, "\theta'"] \\
S/Q. &
\end{tikzcd}$$

Proof: The existence of θ follows from Theorem 7.2. Because ϵ_Q is surjective also θ is surjective. The existence of the isomorphism θ' is now given by Theorem 7.1. □

8 Simple systems and coinduction

An algebra is called simple if it does not have proper quotients. We apply the same definition to systems: a system S is *simple* if it has no proper quotients (homomorphic images): i.e., every epimorphism $f : S \rightarrow T$ is an isomorphism. Theorem 8.1 below gives a number of equivalent characterizations of simplicity, the most important of which is the so-called *coinduction proof principle*: for every bisimulation R on S, $R \subseteq \Delta_S$ (recall that $\Delta_S = \{\langle s, s\rangle \mid s \in S\}$). Equivalently, for all s and s' in S,

$$s \sim_S s' \ \Rightarrow s = s'.$$

The principle states that in order to prove the equality of two states in S, it is sufficient to show that they are bisimilar. We shall see examples of the use of this surprisingly strong proof principle in Section 12. In Section 13, it will be related to the more familiar principle of induction in a way that will justify the chosen terminology.

Theorem* 8.1 *Let S be a system. The following are equivalent:*

1. *S is simple.*
2. *S satisfies the coinduction proof principle: for every bisimulation R on S, $R \subseteq \Delta_S$.*
3. *Δ_S is the only bisimulation equivalence on S.*
4. *For any system T, and functions $f : T \rightarrow S$ and $g : T \rightarrow S$: if f and g are homomorphisms then $f = g$.*
5. *The quotient homomorphism $\epsilon : S \rightarrow S/\sim$, where $\sim$ denotes the greatest bisimulation on S, is an isomorphism.*

Proof: $1 \Rightarrow 3$: Let R be a bisimulation equivalence on S and consider the quotient homomorphism $\epsilon_R : S \rightarrow S/R$. If S is simple then ϵ_R is an isomorphism. Thus $R = \Delta_S$.
$3 \Rightarrow 1$: Let $f : S \rightarrow T$ be an epimorphism. Since the kernel of f is a bisimulation equivalence, it follows from 3 that it is equal to Δ_S. By Theorem 7.1, $S/\Delta_S \cong f(S)$, hence $S \cong T$. Thus S is simple.
$2 \Rightarrow 4$: Let T be a system, and let $f : T \rightarrow S$ and $g : T \rightarrow S$ be homomorphisms. Define

$$Q = \{\langle f(t), g(t)\rangle \mid t \in T\}.$$

Since $Q = G(f)^{-1} \circ G(g)$ (recall that $G(f)$ is the graph of f), it is a bisimulation by Theorem 5.4. It follows from 2 that $Q \subseteq \Delta_S$. Thus $f = g$.
$4 \Rightarrow 2$: Let R be a bisimulation on S. By definition, its projections $\pi_1 : R \rightarrow S$ and $\pi_2 : R \rightarrow S$ are homomorphisms. It follows from 4 that $\pi_1 = \pi_2$, hence $R \subseteq \Delta_S$.
$3 \Leftrightarrow 2$: Immediate from the observation that the greatest bisimulation on S is an equivalence.
$1 \Rightarrow 5$: Immediate.
$5 \Rightarrow 3$: Suppose that $\epsilon : S \rightarrow S/\sim$ is an isomorphism. Let R be a bisimulation equivalence on S. Because $R \subseteq\sim$ and $\sim$ is the kernel of ϵ, there exists by Theorem 7.2 a (unique) homomorphism $\bar{f} : S/R \rightarrow S/\sim$ such that $\bar{f} \circ \epsilon_R = \epsilon$. Since ϵ is an isomorphism this implies that ϵ_R is injective. Thus $R = \Delta_S$. □

Every system can be made simple by taking the quotient with respect to its greatest bisimulation. This is a consequence of the following.

Proposition* 8.2 *For every system S and bisimulation equivalence R on S, the quotient S/R is simple if and only if $R = \sim$.*

Proof: $\Leftarrow$: Let Q be a bisimulation on $S/\sim$. We show that $Q \subseteq \Delta_{S/\sim}$. Then it follows from Theorem 8.1 that $S/\sim$ is simple. Consider $\epsilon : S \to S/\sim$. By Proposition 5.9 $\epsilon^{-1}(Q)$ is a bisimulation on S and hence is included in $\sim$. This implies $Q \subseteq \Delta_{S/\sim}$.
$\Rightarrow$: Let Q be a bisimulation on S. We show that $Q \subseteq R$. By definition the projections $\pi_1 : Q \to S$ and $\pi_2 : Q \to S$ are homomorphisms. Consider the compositions $\epsilon \circ \pi_1 : Q \to S/R$ and $\epsilon \circ \pi_2 : Q \to S/R$. By assumption, S/R is simple. It follows from Theorem 8.1 that $\epsilon \circ \pi_1 = \epsilon \circ \pi_2$, whence $Q \subseteq R$. Therefore $R = \sim$. □

9 Final systems

An F-system (P, π) is *final*[4] if for any F-system (S, α_S) there exists precisely one homomorphism $f_S : (S, \alpha_S) \to (P, \pi)$. (In other words, (P, π) is a final object in the category Set_F. As a consequence, any two final systems are isomorphic.) Final systems are of special interest because they have a number of pleasant properties.

First of all, the transition structure on a final system is an isomorphism (Lambek, cf. [SP82]).

Theorem 9.1 *Let (P, π) be a final F-system. Then $\pi : P \to F(P)$ is an isomorphism.*

Proof: Because $(F(P), F(\pi))$ is an F-system, there exists by the finality of P a unique homomorphism $f : (F(P), F(\pi)) \to (P, \pi)$. Again by finality, the composition of the homomorphisms π and f:

$$\begin{tikzcd}
P \arrow[r, "\pi"] \arrow[d, "\pi"'] & F(P) \arrow[r, "f"] \arrow[d, "F(\pi)"] & P \arrow[d, "\pi"] \\
F(P) \arrow[r, "F(\pi)"'] & F(F(P)) \arrow[r, "F(f)"'] & F(P).
\end{tikzcd}$$

is equal to 1_P, since 1_P is also a homomorphism. It follows from the fact that f is a homomorphism that the reverse composition equals the identity on $F(P)$. □

Final systems allow coinductive proofs.

Theorem 9.2 *[RT93] Final systems (are simple and hence) satisfy the coinduction proof principle: for any bisimulation R on P, $R \subseteq \Delta_P$.*

Proof: Immediate from Theorem 8.1(4). □

A final system can be considered as a universal domain of canonical representatives for bisimulation equivalences classes in the following way.

Theorem* 9.3 *[RT93] Let S be an F-system, P a final F-system, and $f_S : S \to P$ the unique homomorphism from S to P. For all s and s' in S,*

$$s \sim_S s' \iff f_S(s) = f_S(s').$$

Thus $f_S(s)$ represents the $\sim_S$-equivalence class of s.

Proof: $\Rightarrow$: Let R be a bisimulation on S with $\langle s, s' \rangle \in R$. By Proposition 5.9(1), $f_S(R)$ is a bisimulation on P. By definition, $\langle f_S(s), f_S(s') \rangle \in f_S(R)$. By Theorem 9.2, P satisfies the coinduction proof principle: $f_S(R) \subseteq \Delta_P$, whence $f_S(s) = f_S(s')$.

$\Leftarrow$: Because Δ_P is a bisimulation on P, $f_S^{-1}(\Delta)$ is a bisimulation on S, by Proposition 5.9(2). If $f_S(s) = f_S(s')$ then $\langle s, s' \rangle \in f_S^{-1}(\Delta)$, thus $s \sim_S s'$. □

[4] We prefer *final* to *terminal*, which we associate with malady.

The element $f_S(s)$ in the final system can be viewed as the 'observable behaviour' of s. (For that reason, f_S is called *final semantics* in [RT93].) The following simple example may serve to illustrate this.

Example 9.4 Consider the functor $F(S) = A \times S$ of deterministic transition systems with output. For this functor, $(A^\omega, \langle \mathrm{h,t} \rangle)$ (Section 3) is final: Consider a system S with dynamics $\langle v, n \rangle : S \to (A \times S)$. The function from $f_S : S \to A^\omega$, which assigns to a state s in S with transitions

$$s \xrightarrow{a_0} s_1 \xrightarrow{a_1} \cdots,$$

the infinite word

$$\langle a_0, a_1, \ldots \rangle \quad (= \langle v(s), v(n(s)), v(n(n(s))), \ldots \rangle),$$

is the only homomorphism between S and A^ω. If the output symbols a_i are interpreted as the observations corresponding to the individual transition steps, then $f_S(s)$ can be viewed as the observable behaviour of the entire transition sequence (computation) starting in s. □

Finally, the *existence* of a unique homomorphism from a given system into a final system P can be used as a way of giving definitions. Therefore, P is said to satisfy the *coinduction definition principle*. We shall see examples of this in Section 11.

10 Existence of final systems

A final F-system need not always exist. For instance, if F is the powerset functor $\mathcal{P}$ (of non-deterministic systems) and P were a final $\mathcal{P}$-system, then Theorem 9.1 implies $P \cong \mathcal{P}(P)$, a contradiction because the cardinality of $\mathcal{P}(P)$ is strictly greater than that of P. For many functors, though, final systems do exist. We shall briefly describe two ways of constructing final systems and give some concrete examples.

For many functors $F : \mathit{Set} \to \mathit{Set}$, the following construction yields a final system. Let $! : F(1) \to 1$ be the unique function from $F(1)$ to the one element set 1. The *inverse limit* of the following sequence:

$$\begin{tikzcd} 1 & F(1) \arrow[l, "!"'] & F^2(1) \arrow[l, "F(!)"'] & \cdots, \arrow[l, "F^2(!)"'] \end{tikzcd}$$

where $F^{n+1} = F \circ F^n$, is defined as

$$P = \{ \langle x_0, x_1, x_2, \ldots \rangle \mid \forall n \geq 0,\ x_n \in F^n(1) \text{ and } F^n(!)(x_{n+1}) = x_n \}.$$

The set P is a categorical limit of the sequence. If $F(P)$ is again a limit of the same sequence, then F is called *(ω^{op}-)continuous*. In that case, there exists a unique (mediating) bijection from $F(P)$ to P, the inverse of which, say $\pi : P \to F(P)$, turns P into an F-system (P, π), which can be easily shown to be a final F-system (cf. [SP82]).

Let the class of *polynomial* functors consist of all functors that we can build from the following basic functors: the constant functor A (where A is any set), the identity functor I, sum $+$, product $\times$, and the function space functor $F(X) = X^A$, where A again an arbitrary set. (Note that this definition if somewhat non-standard in that the function space functor is usually not included.)

Theorem 10.1 *All polynomial functors are continuous and hence have a final system.*

Below we give a few concrete examples.

Examples 10.2 An explicit description of some final systems is given, on the basis of which a direct proof—not using the continuity of the functor—of their finality can be easily given as well. (The sets A, B, and C below are arbitrary.)

1. $I(S) = S$: The one element set 1 is a final system for the identity functor.

2. $F(S) = A \times S$: For this functor, the system (A^ω, $\langle$h,t$\rangle$) is final (cf. Example 9.4).

3. $F(S) = 1 + (A \times S)$: the system (A^∞, π) (Section 3) is final.

4. A special case: if $A = 1$ then the previous final system is (isomorphic to) ($\bar{\mathcal{N}}$, pred), the set of extended natural numbers (Section 3).

5. $F(S) = A \times S^B$: the system (A^{B^*}, π) is final [MA86], where

$$\pi : A^{B^*} \to A \times (A^{B^*})^B$$

is defined, for ϕ in A^{B^*}, by $\pi(\phi) = \langle \phi(\epsilon),\, \psi \rangle$, with for b in B and v in B^*,

$$\psi(b)(v) = \phi(\langle b \rangle \cdot v).$$

(Here ϵ is the empty sequence and $\cdot$ denotes concatenation of finite sequences.)

6. $F(S) = C + (A \times S^B)$: Note that this example subsumes all of the above examples. The following set can be given a transition structure similarly to the definition of π in the previous example, turning it into a final system [Jac96c]:

$$\{\phi : B^* \to (A + C) \mid \forall v \in B^*,\, \phi(v) \in C \Rightarrow\ (\forall w \in B^*,\, \phi(v \cdot w) = \phi(v))\}.$$

7. $F(S) = 1 + ((A \times S) \times (A \times S))$: The following system (T, τ) is final. It consists of the set T of all binary arc-labelled trees (possibly containing finite and infinite branches):

$$\begin{array}{rcl} T & = & \{\phi : \{0,1\}^* \to (1 + (A \times A)) \mid \\ & & \forall v \in \{0,1\}^*,\, \phi(v) \in 1 \Rightarrow\ (\forall w \in \{0,1\}^*,\, \phi(v \cdot w) = \phi(v))\}, \end{array}$$

and the function $\tau : T \to 1 + ((A \times T) \times (A \times T))$, defined for $\phi \in T$ by

$$\tau(\phi) = \begin{cases} * & \text{if } \phi(\epsilon) = * \\ (\langle a_1, \phi_1 \rangle,\, \langle a_2, \phi_2 \rangle) & \text{if } \phi(\epsilon) = \langle a_1, a_2 \rangle, \end{cases}$$

where ϕ_i is defined for $v \in \{0,1\}^*$ by $\phi_i(v) = \phi(\langle a_i \rangle \cdot v)$.

□

The class of polynomial functors contains most but not all of the functors we have encountered sofar. Notably the powerset functor $\mathcal{P}$ is not polynomial. Now we have already seen at the beginning of this section that no final system exists for this functor. However, we shall see that for the *finite* powerset $\mathcal{P}_f$ a final system exists. It cannot be obtained by the inverse limit construction described above, since $\mathcal{P}_f$ is not continuous. Fortunately there exist other more general ways of finding final systems, one of which is discussed next. It is a variation on the following naive approach, which *almost* works. First we form the coproduct (disjoint union) of all F-systems:

$$(U, \beta) = \bigsqcup \{(S_i, \alpha_i) \mid (S_i, \alpha_i) \text{ is an } F\text{-system }\}.$$

Next the quotient of U is taken with respect to the greatest bisimulation on U:

$$(P, \pi) = (U/\sim_U,\, \beta_{\sim_U}).$$

We claim that (P, π) is final: let (S, α) be any F-system. There exists a homomorphism from S to P by composing the embedding homomorphism of S into the coproduct U with the quotient homomorphism $\epsilon : U \to P$. Because P is simple by Theorem 8.2, this homomorphism is unique by Theorem 8.1.

This argument has, of course, a flaw: the coproduct of *all* systems does not exist (its carrier would generally be a proper class). For many functors, however, it is not necessary to take the coproduct of all systems, but it is sufficient to consider only a 'generating' set of designated systems. More precisely, suppose that (F is such that) there exists a set of F-systems

$$\mathcal{G} = \{(G_i, \alpha_i) \mid i \in I\}$$

(for some index set I), with the property that

$$\forall (S, \alpha_S)\, \forall s \in S\, \exists (G_i, \alpha_i) \in \mathcal{G},\ (G_i, \alpha_i) \cong \langle s \rangle.$$

(Recall that $\langle s \rangle$ is the subsystem of S generated by the singleton set $\{s\}$.) Such a set $\mathcal{G}$ is called a *set of generators*[5] because every F-system (S, α_S) can be obtained as a quotient of a coproduct of elements of $\mathcal{G}$ as follows: Choose for any $s \in S$ a system G_s in $\mathcal{G}$. Then there exists a surjective homomorphism

$$q : \bigsqcup \{G_s \mid s \in S\} \to\ S,$$

which is determined by the homomorphisms, for all $s \in S$, $G_s \xrightarrow{\cong} \langle s \rangle \to S$ (the latter homomorphism is the embedding of the subsystem $\langle s \rangle$ in S).

Now the construction proceeds as before: let

$$(U, \beta) = \bigsqcup \{(G_i, \alpha_i) \mid (G_i, \alpha_i) \in \mathcal{G}\},$$

and let again

$$(P, \pi) = (U/\sim_U,\, \beta_{\sim_U}).$$

We claim that (P, π) is final: let (S, α) be any F-system. Because (P, π) is simple (as before), it is sufficient to prove the existence of a homomorphism from any system S to P. Consider the following diagram:

$$\begin{tikzcd}
\bigsqcup\{G_s \mid s \in S\} \arrow[rr, "q"] \arrow[dr, "l"'] & & S \arrow[dd, dashed, "f_S"] \\
& (U, \beta) \arrow[dr, "\epsilon"'] & \\
& & (P, \pi).
\end{tikzcd}$$

The existence of the homomorphism q was established above, and $\epsilon : (U, \beta) \to (P, \pi)$ is the quotient homomorphism. The homomorphism l is determined by the embeddings, for all $s \in S$, of G_s in U. The existence of the homomorphism f_S follows from Theorem 7.2, whose conditions can be seen to be fulfilled as follows: by the First Isomorphism Theorem (7.1) $S \cong \bigsqcup\{G_s \mid s \in S\}/K(q)$ (recall that K stands for kernel); $K(q)$ is a bisimulation, by Proposition 5.7, and hence $l(K(q))$ is a bisimulation on U, by Proposition 5.9; consequently, $l(K(q)) \subseteq \sim_U = K(\epsilon)$, which implies $K(q) \subseteq K(\epsilon \circ l)$. This concludes the proof of the finality of (P, π). We have proven the following.

Theorem* 10.3 *Any functor F for which a set of generators exists, has a final F-system.* □

For all bounded functors (Definition 6.7), a set of generators exists.

Theorem* 10.4 *For every bounded functor F, a set of generators, and hence a final F-system, exists.*

[5] See [Bor94][Proposition 4.5.2] for two equivalent characterizations of this notion.

Proof: Let V be a set such that for any system (S, α_S) and any subsystem $\langle s \rangle = (T, \beta)$ of (S, α_S), T can be embedded in V. The following is a set of generators for F:

$$\{(U, \gamma) \mid U \subseteq V \text{ and } \gamma : U \to F(U)\}.$$

For let $\langle s \rangle = (T, \beta)$ be a subsystem of a system (S, α_S), and let $i : T \to V$ be injective. Let $b : T \to i(T)$ be the corresponding bijection. Then $\langle s \rangle$ is isomorphic with $(i(T),\, F(b) \circ \beta \circ b^{-1})$. Applying Theorem 10.3 yields the existence of a final system. □

Examples 10.5 The above results apply to many functors.

1. The functor $F(S) = A \times S^B$ is bounded (Examples 6.8) and hence has a final system (which we already knew from Theorem 10.1). In fact, it is not to difficult to prove that all polynomial functors are bounded.
2. A prototypical example of a functor that is not polynomial, $\mathcal{P}_f$, is bounded by $\mathcal{N}$ (Examples 6.8). Hence a final $\mathcal{P}_f$-system exists.
3. Similarly, the functor $F(S) = (\mathcal{P}_f(S))^A$ of image finite labelled transition systems (Section 3) is bounded, and thus has a final system (P, π). Using a word from the world of programming semantics [Mil75], we call the elements of P (image finite) *processes*. □

In the same way, one can find sets of generators for all possible combinations of the basic functors mentioned above:

Theorem 10.6 *For all functors that we can build from the following basic functors: the polynomial ones (A, I, $+$, $\times$, $(-)^A$), and $\mathcal{P}_f$, a set of generators exists. Consequently, all these functors have a final coalgebra.* □

The proof of the existence of a final system for bounded functors is more general but at the same time less constructive than the first method, for continuous functors, where explicit descriptions of final systems (as in Examples 10.2) can be easily given. In general, we have no such concrete characterizations for the final systems of functors involving $\mathcal{P}_f$, such as the set of processes in Examples 10.5. (See however [RT94] for a description of a final system for $\mathcal{P}_f$ as a subset of an inverse limit.)

11 Examples of coinductive definitions

We mention the general principle of coinductive definitions and give a few examples.

Let S be a set and (P, π) a final F-system. Given a transition structure $\alpha : S \to F(S)$ there exists by the finality of P a unique homomorphism $f_\alpha : S \to P$. Thus, specifying a transition structure α on S uniquely defines a function $f_\alpha : S \to P$ which is consistent with that specification in that it is a homomorphism:

$$\begin{tikzcd} S \arrow[r, dashed, "\exists!\, f_\alpha"] \arrow[d, "\forall\, \alpha"'] & P \arrow[d, "\pi"] \\ F(S) \arrow[r, "F(f_\alpha)"'] & F(P). \end{tikzcd}$$

We say that the function f_α is defined by coinduction from (the specification) α. As we shall see shortly, α gives 'the first step' of the action of f_α. Therefore f_α is sometimes called the *coinductive extension* of α.

Examples 11.1 Typically the coinduction definition principle is used to define functions from (products of) a final system to itself. Here are a few examples.

1. *'Zipping' two infinite streams*: Recall from Examples 10.2 that the system $(A^\omega, \langle h, t\rangle)$ of infinite streams over A is final for the functor $F(S) = A \times S$. In order to define a function f_{zip} that merges two streams into one by alternatingly taking an element from the first and the second, we define a transition structure $zip : (A^\omega \times A^\omega) \rightarrow A \times (A^\omega \times A^\omega)$ by

   $$zip\langle v, w\rangle = (a, \langle w, v'\rangle), \quad \text{where } \langle h, t\rangle(v) = \langle a, v'\rangle.$$

   Then by coinduction there exists a unique homomorphism $f_{zip} : A^\omega \times A^\omega \rightarrow A^\omega$. Because it is a homomorphism, it satisfies

   $$\langle h, t\rangle(\, f_{zip}\langle a \cdot v', w\rangle) = (a, \, f_{zip}\langle w, v'\rangle\,)$$

   Note that this equation expresses the fact that f_{zip} is a consistent extension of *zip*: it repeats infinitely often the first step of *zip*, namely taking the first element from the left stream and swapping the remainder of the left stream with the right stream.

2. *'Zipping' two infinite streams*: We repeat the same example with a small variation of presentation. Rather than defining the function *zip* directly, we specify the corresponding transitions in $A^\omega \times A^\omega$ by means of the following conditional rule

   $$\frac{v \xrightarrow{a} v'}{\langle v, w\rangle \overset{a}{\Longrightarrow} \langle w, v'\rangle}.$$

   We use the symbol $\longrightarrow$ for the transitions in A^ω (determined by the function $\langle h, t\rangle$), and the symbol $\Longrightarrow$ for the transitions in $A^\omega \times A^\omega$ that we are defining. The rule should be read as: if the transition of the premise (upper part) is possible, then so is the transition of the conclusion (lower part). Then $\Longrightarrow$ is formally defined as the smallest relation on $A^\omega \times A^\omega$ that satisfies this rule. From $\Longrightarrow$, we derive an alternative definition of the function *zip*, by putting

   $$zip\langle v, w\rangle = (a, \langle w, v'\rangle) \iff \langle v, w\rangle \overset{a}{\Longrightarrow} \langle w, v'\rangle.$$

   The function f_{zip} can now be conveniently characterized by the rule

   $$\frac{v \xrightarrow{a} v'}{f_{zip}\langle v, w\rangle \xrightarrow{a} f_{zip}\langle w, v'\rangle},$$

   which is identical in shape to the rule that has been used in the definition of *zip*.

3. *Defining concrete infinite streams*: Let in the previous example a and b be elements of A. The infinite streams $(ab)^\omega$ and $(ba)^\omega$ can be formally introduced by putting a transition structure on the set $2 = \{0, 1\}$ as follows:

   $$0 \overset{a}{\Longrightarrow} 1 \text{ and } 1 \overset{b}{\Longrightarrow} 0.$$

   By coinduction there there exists a unique homomorphism $f : 2 \rightarrow A^\infty$ satisfying:

   $$f(0) \xrightarrow{a} f(1) \text{ and } f(1) \xrightarrow{b} f(0).$$

   Now put $(ab)^\omega = f(0)$ and $(ba)^\omega = f(1)$. Similarly one defines a^ω and b^ω.

4. *Concatenation of streams*: The system (A^∞, π) of finite and infinite streams over A is final for the functor $F(S) = 1 + (A \times S)$ (Examples 10.2). The concatenation of two streams can be defined by specifying the following transitions on $A^\infty \times A^\infty$ (using a notation similar to that of the previous example):

   $$\frac{v \xrightarrow{a} v'}{\langle v, w\rangle \overset{a}{\Longrightarrow} \langle v', w\rangle} \quad \frac{v\downarrow \ \text{ and } w \xrightarrow{a} w'}{\langle v, w\rangle \overset{a}{\Longrightarrow} \langle v, w'\rangle} \quad \frac{v\downarrow \ \text{ and } w\downarrow}{\langle v, w\rangle\Downarrow}.$$

   (The transitions in the premises correspond to the transition structure π.) As before, this defines a transition structure *conc* : $A^\infty \times A^\infty \to 1 + (A^\infty \times A^\infty)$, by

   $$conc\langle v, w\rangle = \begin{cases} * & \text{if } \langle v, w\rangle\Downarrow \\ (a, \langle v', w'\rangle) & \text{if } \langle v, w\rangle \overset{a}{\Longrightarrow} \langle v', w'\rangle. \end{cases}$$

   By coinduction, there exists a unique function $f_{conc} : A^\infty \times A^\infty \to A^\infty$. For notational convenience, we shall write $v \cdot w = f_{conc}\langle v, w\rangle$. Again the resulting function f_{conc} satisfies the same (in shape, that is) rules that have been used in the definition of *conc* above:

   $$\frac{v \xrightarrow{a} v'}{v \cdot w \xrightarrow{a} v' \cdot w} \quad \frac{v\downarrow \ \text{ and } w \xrightarrow{a} w'}{v \cdot w \xrightarrow{a} v \cdot w'} \quad \frac{v\downarrow \ \text{ and } w\downarrow}{v \cdot w\downarrow}.$$

   Note that $v \cdot \epsilon = \epsilon \cdot v = v$ does not come out of this characterization immediately: we shall prove it in Section 12 using the coinduction proof principle.

5. *Concatenation of other structures*: Without mentioning the details, let us observe that concatenation of other structures like trees or processes (Examples 10.5) can be defined in essentially the same way as the previous example.

6. *Addition of natural numbers*: A special case of concatenation of streams is obtained by taking $A = 1$. Now the functor looks like $F(S) = 1 + S$ (since $1 + (A \times S) = 1 + (1 \times S) \cong 1 + S$.) Recall from Examples 10.2 that it has $(\bar{\mathcal{N}}$, pred), the extended natural numbers, as a final system. We write $\oplus$ for the function f_{conc} in this case, which satisfies as before

   $$\frac{n \longrightarrow n'}{n \oplus m \longrightarrow n' \oplus m} \quad \frac{n\downarrow \ \text{ and } m \longrightarrow m'}{n \oplus m \longrightarrow n \oplus m'} \quad \frac{n\downarrow \ \text{ and } m\downarrow}{n \oplus m\downarrow}.$$

   We shall prove in Section 12 that $\oplus$ indeed is (a coinductively defined version of) addition.

7. *Merging two processes*: The system of nondeterministic processes (P, π) is final for the functor $F(S) = A \to \mathcal{P}_f(S)$ (Examples 10.5). In the same style as above, we define a merge (or interleaving) operation on $P \times P$ by specifying the following transitions:

   $$\frac{p \xrightarrow{a} p'}{\langle p, q\rangle \overset{a}{\Longrightarrow} \langle p', q\rangle} \quad \frac{q \xrightarrow{a} q'}{\langle p, q\rangle \overset{a}{\Longrightarrow} \langle p, q'\rangle} \quad \frac{p\downarrow \ \text{ and } q\downarrow}{\langle p, q\rangle\Downarrow}.$$

   As before, this determines a transition structure $merge : P \to (A \to \mathcal{P}_f(P))$. (Note that one has to check that the transition relation $\Longrightarrow$ is image finite, which is immediate from the fact that $\longrightarrow$ is.) By coinduction, there exists a function $f_{merge} : P \times P \to P$, which is characterized by

   $$\frac{p \xrightarrow{a} p'}{f_{merge}\langle p, q\rangle \xrightarrow{a} f_{merge}\langle p', q\rangle} \quad \frac{q \xrightarrow{a} q'}{f_{merge}\langle p, q\rangle \xrightarrow{a} f_{merge}\langle p, q'\rangle} \quad \frac{p\downarrow \ \text{ and } q\downarrow}{f_{merge}\langle p, q\rangle\downarrow}.$$

   The merge of two processes is sometimes called *parallel composition*.

□

A common feature of all the examples above is that the definition of a function $f : S \to P$ by coinduction amounts to the definition of a transition structure $\alpha : S \to F(S)$. A good understanding of coinduction, therefore, should be based on a thorough knowledge of *transition system specifications*, of which we have seen examples above. The classification of *schemes* or *formats* of such transition system specifications (as in, e.g., [GV92] and [vG96]) could be called the study of *corecursion*, in the same way as recursion theory studies schemes for inductive definitions. See also [MD97] for some further thoughts on corecursion in the context of non-wellfounded set theory.

12 Examples of proofs by coinduction

Recall from Section 8 the coinduction proof principle, for a system S:

> for every bisimulation R on S, $R \subseteq \Delta_S$.

As we have seen, the principle is valid for all simple systems and hence for all final systems. It is quite a bit more powerful than one might suspect at first sight.

Examples 12.1 Typically the coinduction proof principle is used to prove properties of coinductively defined functions, such as the ones defined in Examples 11.1.

1. *'Zipping' infinite streams*: We prove that $zip\langle a^\omega, b^\omega \rangle = (ab)^\omega$. The relation $R \subseteq A^\omega \times A^\omega$, consisting of the following two pairs

   $$R = \{\langle zip\langle a^\omega,\, b^\omega\rangle,\, (ab)^\omega\rangle,\ \langle zip\langle b^\omega,\, a^\omega\rangle,\, (ba)^\omega\rangle\}$$

   is a bisimulation: We have to prove (cf. Section 3) for all a in A and $\langle v, w\rangle$ in R:

   (a) $v \xrightarrow{a} v'$ and $w \xrightarrow{a} w' \ \Rightarrow \langle v', w'\rangle \in R$.

   Consider the first pair of R. The only transition step of its left component is

   $$zip\langle a^\omega,\, b^\omega\rangle \xrightarrow{a} zip\langle b^\omega,\, a^\omega\rangle,$$

   whereas its right component can take the step

   $$(ab)^\omega \xrightarrow{a} (ba)^\omega.$$

   The pair of resulting states, $\langle zip\langle b^\omega,\, a^\omega\rangle,\, (ba)^\omega\rangle$, is again an element of R. Thus we have proved that the first pair in R has the bisimulation property. Similarly for the second pair. Now A^ω is final and hence satisfies the coinduction proof principle, which tells us that $R \subseteq \Delta_{A^\omega}$, proving the equality we were after.

2. *Concatenating the empty stream*: For any finite or infinite stream $v \in A^\infty$, left concatenation with the empty stream ϵ is the identity, because $R = \{\langle \epsilon \cdot v,\, v\rangle \mid v \in A^\infty\}$ is easily shown to be a bisimulation on the final system A^∞; that is (cf. Section 3), for all $\langle v, w\rangle$ in R, (a) above holds as well as

   (b) $v{\downarrow} \iff w{\downarrow}$.

   Similarly, right concatenation with ϵ is the identity.

3. *Concatenation of streams is associative*: This follows by coinduction from the fact that

   $$R = \{\langle (u \cdot v) \cdot w,\ u \cdot (v \cdot w)\rangle \mid u,\, v,\, w\, \in A^\infty\}$$

   is a bisimulation relation on A^∞. Rather than showing this, it turns out to be convenient to prove that $S = R \cup \Delta_{A^\infty}$ is a bisimulation. Consider a pair in S. If it is in Δ_{A^∞} then there is nothing to prove since by Proposition 5.1, the diagonal is a bisimulation. Otherwise, the pair is of the form $\langle (u \cdot v) \cdot w,\ u \cdot (v \cdot w) \rangle$ in R. If $u = \epsilon$ then it follows from the previous example that

$$(\epsilon \cdot v) \cdot w = \ v \cdot w = \ \epsilon \cdot (v \cdot w),$$

which implies that our pair is in Δ_{A^∞} after all, bringing us back to the case we have already considered. If u is not the empty sequence, it can take an a step to u', for some a and u'. In that case, there are transitions

$$(u \cdot v) \cdot w \xrightarrow{a} (u' \cdot v) \cdot w, \quad \text{and } u \cdot (v \cdot w) \xrightarrow{a} u' \cdot (v \cdot w),$$

which concludes the proof since the resulting pair is in R again. The reader is invited to prove directly that R (without taking the union with Δ_{A^∞}) is a bisimulation. This is quite possible but involves a few more case distinctions (as to whether v and w are empty or not).

4. *Concatenation of trees and processes is associative*: by similar proofs.

5. *Addition of natural numbers*: In Examples 11.1, addition ($\oplus$) on the (extended) natural numbers $\bar{\mathcal{N}}$ has been defined in terms of concatenation. Here we show that $\oplus$ has the usual properties in terms of the successor function. Let $s : \bar{\mathcal{N}} \to \bar{\mathcal{N}}$ be the inverse of $pred : (1 + \bar{\mathcal{N}}) \to \bar{\mathcal{N}}$, restricted to $\bar{\mathcal{N}}$. Thus it is defined as usual, with $s(\infty) = \infty$. Because $pred(s(n)) = n$ there is a transition $s(n) \longrightarrow n$, for any n in $\bar{\mathcal{N}}$. The following holds, for any n and m in $\bar{\mathcal{N}}$:

   (a) $0 \oplus m = m$

   (b) $s(n) \oplus m = \ s(n \oplus m)$.

   The first statement follows from example 2. above. The second follows by coinduction on $\bar{\mathcal{N}}$ from the fact that $R = \{\langle s(n) \oplus m,\ s(n \oplus m)\rangle \mid n, m \in \Delta_{\bar{\mathcal{N}}}\} \cup \ \Delta_{\bar{\mathcal{N}}}$ is a bisimulation, which is immediate since both $s(n) \oplus m$ and $s(n \oplus m)$ can take a step to $n \oplus m$, and $\langle n \oplus m,\ n \oplus m\rangle$ is in $\Delta_{\bar{\mathcal{N}}}$. Note that it follows from the previous example that addition is associative.

6. *Addition of natural numbers is commutative*: Not much of a surprise, really. But just for the fun of it, we present a proof by coinduction (which the reader may want to compare with the more familiar proof using mathematical induction). We prove, for all n and m,

   (a) $n \oplus s(m) = \ s(n) \oplus m$: This follows by coinduction from the fact that

   $$R = \{\langle n \oplus s(m),\ s(n) \oplus m\rangle \mid n, m \in \bar{\mathcal{N}}\} \cup \Delta_{\bar{\mathcal{N}}}$$

   is a bisimulation: Consider a pair $\langle n \oplus s(m),\ s(n) \oplus m\rangle$. If $n = 0$ then both components make a transition to $0 \oplus m$ and we are done, since $\langle 0 \oplus m,\ 0 \oplus m\rangle \in \Delta_{\bar{\mathcal{N}}}$. Otherwise, we have transitions

   $$n \oplus s(m) \longrightarrow pred(n) \oplus s(m), \quad \text{and } s(n) \oplus m \longrightarrow n \oplus m.$$

   Now note that $\langle pred(n) \oplus s(m),\ n \oplus m\rangle = \ \langle pred(n) \oplus s(m),\ s(pred(n)) \oplus m\rangle$, which is in R.

   (b) $n \oplus m = \ m \oplus n$: Using statement (a) as a lemma, we prove that the relation

   $$Q = \{\langle n \oplus m,\ m \oplus n\rangle \mid n, m \in \bar{\mathcal{N}}\}$$

   is a bisimulation. Consider a pair $\langle n \oplus m,\ m \oplus n\rangle$ and suppose that both are different from 0 (the other three cases are trivial), say, $n = \ s(n')$ and $m = \ s(m')$. Then there are transitions

   $$n \oplus m \longrightarrow n' \oplus m, \quad \text{and } m \oplus n \longrightarrow m' \oplus n.$$

   Now observe that

   $$n' \oplus m = \ n' \oplus s(m') = \ [\text{the lemma (a)}] \ \ s(n') \oplus m' = \ n \oplus m',$$

   which implies that $\langle n' \oplus m,\ m' \oplus n\rangle$ is in Q.

(Clearly, concatenation of streams over a set A with *more* than one element is generally not commutative.)

7. *The merge of processes*: is commutative, since $R = \{\langle f_{merge}\langle p,q\rangle,\ f_{merge}\langle q,p\rangle\rangle \mid p,q \in P\}$ is a bisimulation.

13 Induction and coinduction

Why did we call the coinductive proof principle of Section 8 by that name? How does it relate to induction? In short, coinduction is dual to induction in the following sense. Recall that a system S satisfies the coinduction proof principle if and only if it satisfies one of the following two (by Theorem 8.1) equivalent conditions:

1. S is simple, that is, it has *no proper quotients*.
2. For every bisimulation relation R on S, $R \subseteq \Delta_S$.

There is also the following dual proof principle for *algebras*. We say that an algebra A satisfies the *induction proof principle* whenever one of the following two conditions, which turn out to be equivalent, holds:

3. A is minimal, that is, it has *no proper subalgebras*.
4. For every congruence relation R on A, $\Delta_A \subseteq R$.

To make this more precise, we shall give the categorical definitions of algebra, homomorphism of algebras, and congruence relation, which are the algebraic counterparts of the coalgebraic notions of coalgebra, homomorphism of coalgebras, and bisimulation, respectively (cf. Section 1). Then the equivalence of 3 and 4 is proved. Next these notions and the induction principle are illustrated by the example of the natural numbers, which will make clear that the above, somewhat abstractly formulated induction principle, is just the familiar principle of mathematical induction.

Let $F : Set \to Set$ be a functor. An *F-algebra* is a pair (A, α_A) consisting of a set A and a function $\alpha_A : F(A) \to A$. Let (A, α_A) and (B, α_B) be two F-algebras. A function $f : A \to B$ is a *homomorphism of F-algebras* if $f \circ \alpha_A = \alpha_B \circ F(f)$:

$$\begin{tikzcd}
F(A) \arrow[r, "F(f)"] \arrow[d, "\alpha_A"'] & F(B) \arrow[d, "\alpha_B"] \\
A \arrow[r, "f"'] & B.
\end{tikzcd}$$

Intuitively, homomorphisms are functions that preserve the F-algebra structure. An *F-congruence* relation between two F-algebras (A, α_A) and (B, α_B) is a subset $R \subseteq A \times B$ for which there exists an F-algebra structure $\alpha_R : F(R) \to R$ such that the projections from R to A and B are homomorphisms of F-algebras:

$$\begin{tikzcd}
F(A) \arrow[d, "\alpha_A"'] & F(R) \arrow[l, "F(\pi_1)"'] \arrow[r, "F(\pi_2)"] \arrow[d, "\alpha_R"] & F(B) \arrow[d, "\alpha_B"] \\
A & R \arrow[l, "\pi_1"] \arrow[r, "\pi_2"'] & B.
\end{tikzcd}$$

(This definition of congruence is not to be confused with a congruence *equivalence* relation, which is an equivalence relation on one and the same algebra that is moreover respected by the operators. In fact, the above definition is more general.)

Example 13.1 Any Σ-algebra [Coh81] is an F-algebra for a suitable choice of F (see, for instance, [RT94]). Here we look at one particular type of algebras, namely triples $(A,\, 0_A \in A,\, s_A : A \to A)$ consisting of a (carrier) set A, a constant 0_A, and a unary (successor) function s_A. A concrete example are the natural numbers $(\mathcal{N}, 0, s)$. Such algebras can be represented as algebras of the functor $F(X) = 1 + X$, by defining

$$\alpha_A : (1 + A) \to A, \quad * \mapsto 0_A, \quad a \mapsto s_A(a).$$

If we have two such algebras $(A,\, 0_A \in A,\, s_A : A \to A)$ and $(B,\, 0_B \in B,\, s_B : B \to B)$ and represent them as F-algebras (A, α_A) and (B, α_B), then one readily verifies that a function $f : A \to B$ is a $(1 + -)$-homomorphism from (A, α_A) to (B, α_B) if and only if it satisfies the usual definition of homomorphism:

$$f(0_A) = 0_B, \;\; f(s_A(a)) = s_B(f(a)).$$

Similarly, it is easy to prove that a $(1 + -)$-congruence relation $R \subseteq A \times B$ between (A, α_A) and (B, α_B) is *substitutive*:

$$\langle 0_A, 0_B \rangle \in R, \;\; \langle a, b \rangle \in R \Rightarrow \langle s_A(a), s_B(b) \rangle \in R.$$

□

As already mentioned above, an F-algebra A satisfies the *induction proof principle* if it satisfies clauses 3 and 4, which are next shown to be equivalent: If R is a congruence on A then $\pi_1(R) \cap \pi_2(R)$ is a subalgebra of A. Assuming 3, this subalgebra is equal to A, or equivalently, $\Delta_A \subseteq R$. This proves 4. Conversely, if $A' \subseteq A$ is a subalgebra and $i : A' \to A$ is the inclusion homomorphism then the kernel of i is easily shown to be a congruence on A', and hence on A. Assuming 4, $\Delta_A \subseteq R$, which implies $A \subseteq A'$.

We have seen that all final systems are simple and hence satisfy the coinduction proof principle. Dually, an *initial* algebra (for which there exists precisely one homomorphism into any given algebra) is minimal and hence satisfies the induction principle.

Example 13.1, continued: The algebra $(\mathcal{N}, 0, s)$ of the natural numbers is initial and hence minimal. Now minimality amounts to the well-known principle of mathematical induction: for all $P \subseteq \mathcal{N}$,

$$\text{if:} \quad 0 \in P \quad \text{and: for all } n \in \mathcal{N} \quad (n \in P \Rightarrow s(n) \in P) \quad \text{then:} \quad P = \mathcal{N},$$

since the if-part of the implication is just the assertion that P is a subalgebra of $\mathcal{N}$. □

Note that for proofs by induction, formulation 3 is mostly used, whereas proofs by coinduction are best given, as we have seen in Section 12, using 2 (which is the dual of 4 rather than 3).

Although we have only compared induction and coinduction as *proof* principles, the corresponding *definition* principles are similarly related. The main observation there is that definitions by induction use the universal property of initiality, as opposed to definitions by coinduction, which are based on finality.

14 Comparing systems

Any deterministic system is a special kind of nondeterministic system. Similarly, any binary tree can be turned into a deterministic system by 'cutting away' all left branches.

Such statements can be formalized using the the following (categorical) notion. Let $F : \mathit{Set} \to \mathit{Set}$ and $G : \mathit{Set} \to \mathit{Set}$ be two functors. A *natural transformation* ν from F to G, denoted by

$\nu : F \rightarrow G$, is a family $\{v_S : F(S) \to G(S) \mid S \in \mathit{Set}\}$ of functions satisfying the following naturality property: for any function $f : S \to T$, the following diagram commutes:

$$\begin{tikzcd}
F(S) \arrow[r, "F(f)"] \arrow[d, "\nu_S"'] & F(T) \arrow[d, "\nu_T"] \\
G(S) \arrow[r, "G(f)"'] & G(T).
\end{tikzcd}$$

Any F-system (S, α_S) can now be viewed as a G-system by composing α_S with ν_S. Moreover, if $f : (S, \alpha_S) \to (T, \alpha_T)$ is an F-homomorphism then it is also a G-homomorphism of the resulting G-systems; and, similarly, any F-bisimulation between F-systems is also a G-bisimulation of the resulting systems:

$$\begin{tikzcd}
S \arrow[r, "f"] \arrow[d, "\alpha_S"'] & T \arrow[d, "\alpha_T"] \\
F(S) \arrow[r, "F(f)"] \arrow[d, "\nu_S"'] & F(T) \arrow[d, "\nu_T"] \\
G(S) \arrow[r, "G(f)"'] & G(T)
\end{tikzcd}
\qquad
\begin{tikzcd}
S \arrow[d, "\alpha_S"'] & R \arrow[l, "\pi_1"'] \arrow[r, "\pi_2"] \arrow[d, "\alpha_R"] & T \arrow[d, "\alpha_T"] \\
F(S) \arrow[d, "\nu_S"'] & F(R) \arrow[l, "F(\pi_1)"] \arrow[r, "F(\pi_2)"'] \arrow[d, "\nu_R"] & F(T) \arrow[d, "\nu_T"] \\
G(S) & G(R) \arrow[l, "G(\pi_1)"] \arrow[r, "G(\pi_2)"'] & G(T).
\end{tikzcd}$$

The above is summarized in the following. (Recall—Corollary 5.6—that $\sim_F$ denotes the greatest F-bisimulation between two systems.)

Theorem 14.1 *A natural transformation $\nu : F \rightarrow G$ between set functors induces a functor, denoted by $\nu \circ (-) : \mathit{Set}_F \to \mathit{Set}_G$ which maps (S, α_S) to $(S, \nu_S \circ \alpha)$ and an F-homomorphism $f : (S, \alpha_S) \to (T, \alpha_T)$ to the G-homomorphism $f : (S, \nu_S \circ \alpha_S) \to (T, \nu_T \circ \alpha_T)$. Moreover, this functor preserves bisimulations: for any s and t, $s \sim_F t \Rightarrow s \sim_G t$.* □

Examples 14.2 A few examples of the use of natural transformations.

1. *Relabeling*: Any function $l : A \to B$ induces a natural transformation $\lambda : A \times (-) \rightarrow B \times (-)$, defined for a set S by $\lambda_S\langle a, s\rangle = \langle l(a), s\rangle$. Let $\alpha_S : S \to (A \times S)$ and $\alpha_T : T \to (B \times T)$ be deterministic transition systems with labels in A and B, respectively. Then a $(B \times (-))$-homomorphism $f : (S, \lambda_S \circ \alpha_S) \to (T, \alpha_T)$ is characterized by

$$s \xrightarrow{a} s' \Rightarrow \; f(s) \xrightarrow{l(a)} f(s').$$

2. *Restriction*: Let $\mu_S : \mathcal{P}_f((A \cup B) \times S) \rightarrow \mathcal{P}_f(B \times S)$ be defined, for any set S and $V \subseteq S$ by $\mu_S(V) = V \cap (B \times S)$. Then composing a nondeterministic transition system $\alpha_S : S \to \mathcal{P}_f((A \cup B) \times S)$ with μ_S amounts to restricting its behavior to B-steps only.

Given, conversely, a G-system (C, γ), there exist, under some conditions on F, a corresponding F-system (S_C, α) that when viewed as a G-system $(S_C, \nu_{S_C} \circ \alpha)$, 'resembles' (C, γ) most. More precisely:

Theorem* 14.3 *Let F and G be functors and $\nu : F \rightarrow G$ a natural transformation. Suppose that for any set V, the functor $V \times F$ has a final system (where V is the constant functor that sends any set to V). Then there exists for any G-system (C, γ) an F-system (S_C, α) and a G-homomorphism $\epsilon : (S_C, \nu_{S_C} \circ \alpha) \to (C, \gamma)$ satisfying the following universal property: for any F-system (U, α_U)*

and any G-homomorphism $f : (U, \nu_U \circ \alpha_U) \rightarrow (C, \gamma)$ there exists a unique F-homomorphism $\tilde{f} : (U, \alpha_U) \rightarrow (S_C, \alpha)$ such that $\epsilon \circ \tilde{f} = f$:

$$\begin{tikzcd}
U \arrow[r, dashed, "\exists\,\tilde{f}"'] \arrow[rr, bend left, "\forall f"] \arrow[d, "\alpha_U"'] & S_C \arrow[r, "\epsilon"'] \arrow[d, "\alpha"] & C \arrow[dd, "\gamma"] \\
F(U) \arrow[r, dashed, "F(\tilde{f})"'] \arrow[d, "\nu_U"'] & F(S_C) \arrow[d, "\nu_{S_C}"] & \\
G(U) \arrow[r, dashed, "G(\tilde{f})"] \arrow[rr, bend right, "G(f)"'] & G(S_C) \arrow[r, "G(\epsilon)"] & G(C).
\end{tikzcd}$$

The F-system (S_C, α) (and ϵ) is called *cofree on* the G-system (C, γ). Note that the functor $V \times F$ is bounded whenever F is, in which case a $(V \times F)$-final system exists by Theorem 10.4.

Proof: By assumption, $S \times F$ has a final system (T, τ). Let $\tau = \langle \pi_1, \pi_2 \rangle$, where $\pi_1 : T \rightarrow S$ and $\pi_2 : T \rightarrow F(T)$. By Theorem 14.1, $(T, \nu_T \circ \pi_2)$ is a G-system. Let $B = \{t \in T \mid t \sim_G \pi_1(t)\}$. Define $(S_C, \alpha) = [B]$, the largest F-subsystem of (T, π_2) that is contained in the subset B:

$$\begin{tikzcd}
S_C \arrow[r, "i"] \arrow[d, "\alpha"'] & T \arrow[d, "\pi_2"] \\
F(S_C) \arrow[r, "F(i)"'] & F(T),
\end{tikzcd}$$

where i is the inclusion F-homomorphism. It is by Theorem 14.1 also a G-homomorphism $i : (S_C, \nu_{S_C} \circ \alpha) \rightarrow (T, \nu_T \circ \pi_2)$. By Theorem 2.5, its graph is a G-bisimulation, hence $c \sim_G i(c)$ for any c in S_C. By definition of B also $i(c) \sim_G \pi_1(i(c))$, and because composition of bisimulation relations is again a bisimulation (Theorem 5.4), it follows that $c \sim_G \pi_1(i(c))$. Therefore the graph of $\pi_1 \circ i$ is a G-bisimulation, and so $\pi_1 \circ i : (S_C, \nu_{S_C} \circ \alpha) \rightarrow (C, \gamma)$ is a G-homomorphism, by Theorem 2.5. That is, the outer square below commutes:

$$\begin{tikzcd}
S_C \arrow[r, "i"] \arrow[d, "\alpha"'] & T \arrow[r, "\pi_1"] \arrow[d, "\pi_2"'] & C \arrow[dd, "\gamma"] \\
F(S_C) \arrow[r, "F(i)"'] \arrow[d, "\nu_{S_C}"'] & F(T) \arrow[d, "\nu_T"'] & \\
G(S_C) \arrow[r, "G(i)"'] & G(T) \arrow[r, "G(\pi_1)"'] & G(C).
\end{tikzcd}$$

(Note that the right rectangle generally does *not* commute.) Define $\epsilon = \pi_1 \circ i$. We claim that (S_C, α) and ϵ satisfy the universal property of the theorem: Consider any F-system (U, α_U) and G-homomorphism $f : (U, \nu_U \circ \alpha_U) \rightarrow (C, \gamma)$. By finality of T, there exists a unique $S \times F$-homomorphism $h : U \rightarrow T$:

$$\begin{tikzcd}
U \arrow[r, "h"] \arrow[d, "{\langle f, \alpha_U \rangle}"'] & T \arrow[d, "{\langle \pi_1, \pi_2 \rangle}"] \\
S \times F(U) \arrow[r, "(S \times F)(h)"'] & S \times F(T).
\end{tikzcd}$$

Commutativity of this diagram implies that $\pi_1 \circ h = f$ and $h : (U, \alpha_U) \rightarrow (T, \pi_2)$ is an F-homomorphism. By Theorem 2.5, its graph is an F-bisimulation and hence, by Theorem 14.1, a G-bisimulation between $(U, \nu_U \circ \alpha_U)$ and $(T, \nu_T \circ \pi_2)$. Thus $u \sim_G h(u)$, for any $u \in U$. Because f is a G-homomorphism, also $u \sim_G f(u)$, again by Theorem 2.5. Because inverse and composition of bisimulations yield bisimulations again (Theorems 5.2 and 5.4), it follows that $h(u) \sim_G f(u) = \pi_1(h(u))$. Thus $h(U) \subseteq B$, which implies, by Proposition 6.5, that h factorizes through $S_C = [B]$: there exists a unique F-homomorphism $\tilde{f} : (U, \alpha_U) \to (S_C, \alpha)$ such that

$$\begin{tikzcd}
U \arrow[r, "h"] \arrow[dr, "\tilde{f}"'] & T \\
 & S_C. \arrow[u, dashed, "i"']
\end{tikzcd}$$

By Theorem 14.1, it is also a G-homomorphism from $(U, \nu_U \circ \alpha_U)$ to $(S_C, \nu_{S_C} \circ \alpha)$. Since

$$\epsilon \circ \tilde{f} = \pi_1 \circ i \circ \tilde{f} = \pi_1 \circ h = f,$$

$\tilde{f}$ is the F-homomorphism we have been looking for. Its uniqueness follows from that of h and the factorization. □

(By a standard argument in category theory, it follows that the assignment of (S_C, α) to (C, γ) actually extends to a functor from Set_G to Set_F, which is right adjoint to $\nu \circ (-)$.)

Examples 14.4 We give a few examples of cofree systems.

1. A simple instantiation of Theorem 14.3 is obtained by taking $G = 1$, the functor that is constant 1. Then there is only one natural transformation from a functor F to 1, and the functor it induces from Set_F to Set sends each F-system to its carrier. (Recall from Section 3 that $Set_1 \cong Set$.) If F is bounded then it follows from the construction above that, for a set C, the final $(C \times F)$-system S_C is cofree on C (cf. [Jac95]). We like to think of the elements of C as 'colours'. In that view, S_C can be regarded as a universally C-coloured F-system: $\epsilon : S_C \to C$ gives the colours of the states in S_C; and for any F-system U and any 'colouring' $f : U \to C$ there exists a unique F-homomorphism $\tilde{f} : U \to S_C$ which is colour consistent: $\epsilon \circ \tilde{f} = f$:

$$\begin{tikzcd}
U \arrow[r, dashed, "\exists\, \tilde{f}"'] \arrow[rr, bend left, "\forall\, f"] & S_C \arrow[r, "\epsilon"'] & C.
\end{tikzcd}$$

2. For a concrete example of the preceding situation, let $F = I$, the identity functor. Then for any set C, the system $\langle h, t \rangle : C^\omega \to C \times C^\omega$ is a final $(C \times I)$-system, and so $t : C^\omega \to C^\omega$, with colouring $h : C^\omega \to C$, is cofree on the set C.

15 Cofreeness and covarieties of systems

We show how any subsystem of a cofree system determines a well-behaved class of systems, called a covariety, and briefly illustrate how this can been seen as a way of system specification.

Let in this section $F : Set \to Set$ be a bounded functor, and C a set, of colours. Let S_C, with colouring $\epsilon : S_C \to C$, be an F-system that is cofree on C. Recall from Examples 14.4 that we can take for S_C any final $(C \times F)$-system, which exists because $C \times F$ is bounded. Consider a subsystem $i : S \to S_C$. Let the class $\mathcal{K}(S)$ consist of all F-systems (U, α_U) with the property that for any colouring function $f : U \to C$, the (by cofreeness uniquely determined) F-homomorphism

$\tilde{f}$ factorizes though S:

$$\begin{tikzcd}
 & C \\
U \arrow[ur, "\forall f"] \arrow[r, "\tilde{f}"] \arrow[dr, dashed] & S_C \arrow[u, "\epsilon"'] \\
 & S. \arrow[u, "i"']
\end{tikzcd}$$

(Note that f and ϵ are functions and the other arrows are F-homomorphisms.) Such classes are well behaved in the following sense.

Theorem* 15.1 *The class $\mathcal{K}(S)$ of F-systems defined above is closed under the formation of*

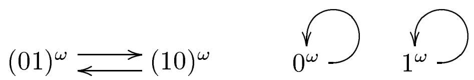
1. *subsystems;*
2. *homomorphic images;*
3. *and sums.* □

Such a class is called a *covariety*.

Proof:

1. Let U be a system in $\mathcal{K}(S)$ and $j : U' \to U$ a subsystem. Any colouring $f' : U' \to C$ can be extended to a colouring $f : U \to C$ such that $f \circ j = f'$. Because $\epsilon \circ \tilde{f} \circ j = f \circ j = f'$, the unique extension of f' to an F-homomorphism from U' to S_C is $\tilde{f'} = \tilde{f} \circ j$. Because U is in $\mathcal{K}(S)$, $\tilde{f}$ factorizes through S, and hence so does $\tilde{f'}$. Thus U' is in $\mathcal{K}(S)$.
2. Let U be a system in $\mathcal{K}(S)$ and $q : U \to U'$ a surjective homomorphism. Any colouring $f' : U' \to C$ induces a colouring $f = f' \circ q$ on U. Because $\epsilon \circ \tilde{f'} \circ q = f' \circ q = f$, it follows from the cofreeness of S_C that $\tilde{f} = \tilde{f'} \circ q$. Because U is in $\mathcal{K}(S)$ there exists $g : U \to S$ such that $i \circ g = \tilde{f}$. The kernel $K(q)$ is included in $K(\tilde{f})$, since $\tilde{f} = \tilde{f'} \circ q$. The fact that q is a surjective homomorphism, implies the existence (by Theorems 7.1 and 7.2) of a homomorphism $g' : U' \to S$ such that $g' \circ q = g$. Since $\tilde{f'} \circ q = \tilde{f} = i \circ g = i \circ g' \circ q$, it follows from the surjectivity of g that $\tilde{f'} = i \circ g'$. Thus U' is in $\mathcal{K}(S)$.
3. A family of colourings $\{f_i\}$ on a family $\{U_i\}$ of systems in $\mathcal{K}(S)$ determines a colouring $\sum f_i : \sum_i U_i \to C$. Because each of the induced homomorphisms $\tilde{f}_i$ factorizes through S, their sum $\sum_i \tilde{f}_i : \sum_i U_i \to S_C$ is readily seen to factorize through S as well. □

Example 15.2 An example of such a class definition is obtained by taking $F = I$, the identity functor, and $C = 2 = \{0, 1\}$. The system $t : 2^\omega \to 2^\omega$, with colouring $h : 2^\omega \to 2$, is cofree on the set 2. Consider the following subsystem S of 2^ω:



The class $\mathcal{K}(S)$ contains all systems (U, α_U) which consist of one and two cycles only: $\alpha_U \circ \alpha_U(u) = u$, for all u in U. □

Returning to the general case again, the following theorem is a kind of converse of the previous one. It states that any covariety is determined by a subsystem of a cofree system.

Theorem* 15.3 *For any covariety $\mathcal{K}$ there exists a set of colours C and a subsystem S of the cofree F-system S_C, such that $\mathcal{K} = \mathcal{K}(S)$.*

Proof: Let $\mathcal{K}$ be a covariety. By assumption F is bounded, say by a set C. Let S_C and $\epsilon : S_C \to C$ be as before. Define a subsystem $i : S \to S_C$ by

$$S = \bigcup \{\tilde{f}(U) \mid U \in \mathcal{K} \text{ and } f : U \to C\}.$$

(Recall that $\tilde{f}(U)$ is a subsystem of S_C by Theorem 6.3, and that the union of subsystems is again a subsystem by Theorem 6.4.) Clearly, $\mathcal{K} \subseteq \mathcal{K}(S)$. For the converse, first note that $S \in \mathcal{K}$: this follows from the fact that S is the image of a homomorphism

$$q : \sum_{s \in S} U_s \to S,$$

where for each s in S an F-system $U_s \in \mathcal{K}$ and a colouring $f_s : U_s \to C$ have been choosen such that $s \in \tilde{f}_s(U_s)$; and where q is determined by the homomorphisms $\tilde{f}_s$. Now let T be any F-system in $\mathcal{K}(S)$, and $t \in T$. The size of the subsystem $\langle t \rangle$ of T is bounded by that of C, because F is bounded by C. Thus there exists a colouring $f : T \to C$ that is injective on $\langle t \rangle$. Because $T \in \mathcal{K}$, the induced homomorphism $\tilde{f}$ factorizes through S via some homomorphism g:

$$\begin{tikzcd}
 & C \\
T \arrow[ur, "f"] \arrow[r, "\tilde{f}"] \arrow[dr, dashed, "g"'] & S_C \arrow[u, "\epsilon"'] \\
 & S. \arrow[u, "i"']
\end{tikzcd}$$

Because $f = \epsilon \circ i \circ g$ and f is injective on $\langle t \rangle$, also g is injective on $\langle t \rangle$. Thus $\langle t \rangle \cong f(\langle t \rangle)$. Since the latter is a subsystem of S, which we have shown to be in $\mathcal{K}$, also $\langle t \rangle$ is in $\mathcal{K}$. Because T is the image of the homomorphism

$$\sum_{t \in T} \langle t \rangle \to T$$

which is determined by the inclusions of the subsystems $\langle t \rangle$ in T, it follows that $T \in \mathcal{K}$. □

The above characterization of classes of systems is inspired by Birkhoff's variety theorem for algebras (cf. [MT92][Theorem 5.2.16]), which states that a class of algebras is closed under the formation of subalgebras, quotients, and *products*, if and only if it is equationally definable. There is also another theorem by Birkhoff, which asserts the soundness and completeness of a logical calculus for equations of varieties. It is unclear what a counterpart of the latter should be for systems.

For some specific types of systems, theorems similar to Theorem 15.3 exist. In particular, it would be interesting to connect our covariety theorem to the characterizations, in [Gol93], of certain classes of frames.

16 Dynamical systems and symbolic dynamics

The generality of the coalgebraic view on systems is further illustrated by a brief account of so-called *one dimensional discrete time dynamical systems* (X, f), consisting of a complete metric space X (with distance function d_X) and a continuous function $f : X \to X$. Such systems are coalgebras of the identity functor on the category *Met* of complete metric spaces and continuous functions between them. Thus we are changing the scene for the first time by looking at a category

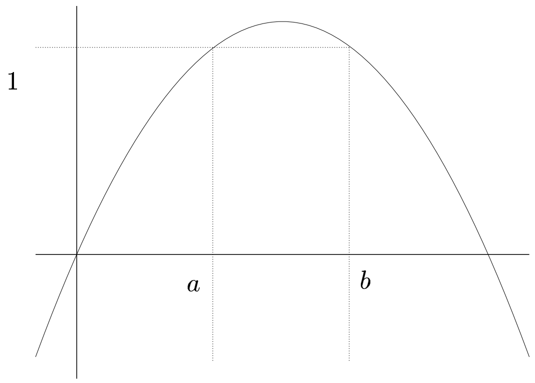


Figure 1: The graph of f_μ, for $\mu > 4$.

different from *Set*. One of the main themes in the theory of dynamical systems is the systematic study of *orbits*: if $x \in X$ then its orbit is the set

$$\{x,\, f^1(x),\, f^2(x),\, f^3(x),\, \ldots\},$$

where $f^{n+1}(x) = f(f^n(x))$. (In our terminology, the orbit of x is just the subsystem $\langle x \rangle$ of (X, f) generated by the singleton x.) Questions to be addressed are, for instance, whether a point x is periodic ($x = f^n(x)$, for some $n \geq 0$); whether there are many such periodic points and how they are distributed over X (e.g., do they form a dense subset?); and whether orbits $\langle x \rangle$ and $\langle y \rangle$ are similar if we know that x and y are close, that is, $d_X(x, y)$ is small. Here we shall briefly discuss one important technique that is used in the world of dynamical systems to answer some of such questions, called *symbolic dynamics* (cf. [BP96]), by giving a coalgebraic account of one particular example, taken from [Dev86]. As it turns out, the notion of cofreeness plays a crucial role.

Let $\mathcal{R}$ denote the set of real numbers. The concrete example we shall consider is the *quadratic family* of systems $(\mathcal{R}, f_\mu)$, which are parameterized by a real number μ, and for which f_μ is defined by

$$f_\mu : \mathcal{R} \to \mathcal{R}, \;\; f_\mu(x) = \mu x(1 - x).$$

More specifically, we shall assume μ to be fixed with $\mu > 4$. (The reason for this choice is that the maximum $\mu/4$ in this case is strictly bigger than 1.) We shall write f for f_μ. Let a and b in $\mathcal{R}$ be the points with $f(a) = 1 = f(b)$ and $a < b$.

A quick look at Figure 1 tells us that the dynamics of f on the intervals $(-\infty, 0)$ and $(1, +\infty)$ is easily understood: all orbits tend to $-\infty$. The same applies to the interval (a, b), since it is mapped by f to $(1, +\infty)$, bringing us back to the previous case. Possibly more interesting dynamic behavior may be expected from elements in the intervals $[0, a]$ and $[b, 1]$. Now note that f maps each of these intervals bijectively to $[0, 1]$. Consequently, $[0, a] \cap f^{-1}(a, b)$ and $[b, 1] \cap f^{-1}(a, b)$ have uninteresting dynamics as well: those points are mapped by f^2 to $(1, +\infty)$, where all orbits go to $-\infty$. This leaves us with $([0, a] \cup [b, 1]) \cap f^{-1}([0, a] \cup [b, 1])$, which consists of four closed intervals. Continuing in this way, we find a set

$$J = \bigcap_{i=0}^{\infty} (f^i)^{-1}([0, a] \cup [b, 1]),$$

which can alternatively be characterized as the largest subsystem of $(\mathcal{R}, f)$ that is contained in $[0, 1]$. Its dynamics can in a surprisingly simple way be explained using symbolic dynamics, which we explain next using our own coalgebraic idiom.

Let 2 be the set $\{0,1\}$ with the discrete metric ($d_2(0,1)=1$). As before we shall consider the elements of 2 as colours. Consider the functor

$$2\times - : \ Met \rightarrow Met, \ \ X \mapsto 2\times X,$$

where the Cartesian product is supplied with distance function

$$d(\langle i,x\rangle,\ \langle j,y\rangle) = d_2(i,j) + 1/2\cdot d_X(x,y).$$

The set of infinite sequences $(2^\omega,\ \langle h,t\rangle)$ (where $h:2^\omega \rightarrow 2$ and $t:2^\omega \rightarrow 2^\omega$ are the head and tail functions), supplied with distance function

$$d_{2^\omega}(v,w) = \sum_{i=0}^{\infty} \frac{d_2(v_i,w_i)}{2^i},$$

is a final $(2\times -)$-system: a proof is omitted but can be given using the techniques for the solution of metric domain equations of [AR89] and [RT93]. Consequently, $(2^\omega, t)$ is a dynamical system that is cofree on the metric space 2. Now define a colouring $c: J \rightarrow 2$ of J by

$$c(x) = \begin{cases} 0 & \text{if } x \in [0,a], \\ 1 & \text{if } x \in [b,1]. \end{cases}$$

By the universal property of the cofree system $(2^\omega,t)$ there exists a unique homomorphism $\tilde{c}: (J,f) \rightarrow (2^\omega,t)$ with $h\circ\tilde{c} = c$. This homomorphism $\tilde{c}$ can readily be shown to be an isomorphism.

Thus $(\mathcal{R},f)$ falls apart into two subsystems: $(\mathcal{R} - J, f)$, where all orbits tend to $-\infty$, and (J,f), whose dynamics is the same as that of $(2^\omega,t)$. The gain of this symbolic interpretation of (J,f) is that the dynamics of $(2^\omega,t)$ is well understood: it is the prototypical example of a *chaotic* system.

17 Notes and related work

The use of final coalgebras in the semantics of systems (including automata and infinite data types such as trees) dates back at least to [AM82]. Also Peter Aczel modelled (transition) systems as coalgebras, in constructing a model for a theory of non-wellfounded sets [Acz88]. In a subsequent paper on final coalgebras [AM89], a categorical definition of bisimulation was given. (Later we found that a variation also occurs in [Ken88].) This categorical definition and the characterization of (final coalgebras and) coinduction in terms thereof, has been the starting point of the present paper. It generalizes and extends [Rut95], where part of the theory of universal coalgebra is developed for the special case of labelled transition systems. That paper was preceded by joint work with Daniele Turi [RT93, RT94, TR97] on final coalgebra semantics for concurrent programming languages.

The aim of the present paper has been both to give an overview of some of the existing insights in coalgebra as well as to present some new material. Below we briefly describe per section which results have been taken from the literature.

Our references for universal algebra have been [Coh81] and [MT92]; for category theory, [ML71, Bor94]. The definition in Section 2 of F-bisimulation is from [AM89]. Theorem 2.5 generalizes [RT94][Proposition 2.8]. Most observations in Section 4 are standard in category theory (cf. [Bar93]). Some of the results in Sections 5 and 6 are generalizations of similar observations in [RT94] and [Rut95], on the category of labelled transition systems. The notion of bounded functor is taken from [KM96], and is ultimately due to [Bar93]. Sections 7 and 8 generalize similar results from [Rut95]. The results on final systems in Section 9 are from [RT93]. The results presented in Section 10 are from [Bar93, Bar94], which build on [AM89]. Their presentation has been influenced by [Tur96] and [KM96]. The example of the extended natural numbers in Sections 11 and 12 was developed jointly with Bart Jacobs and Bill Rounds. The comparison of induction and coinduction in Section 13 extends the characterization in [RT94], which was given in terms of congruences and bisimulations (see also [HJ96]). The recent work by Bart Jacobs on coalgebraic specification [Jac95] and his use of cofreeness in a coalgebraic semantics for object-oriented programming [Jac96b] have been a source of inspiration for the writing of Section 14. The variety theorems of Section 15 answer a question raised in [Rut95]. Section 16 gives a a coalgebraic account of the dynamics of the quadratic family of dynamical systems, which occurs in [Dev86]. This section does not present any new results but for the observation that one of the essential ingredients in symbolic dynamics is the notion of cofree system.

In addition to the references mentioned above, work on (final) coalgebraic semantics includes [Tur96], which gives a systematic comparison of final coalgebra and initial algebra semantics for concurrent languages; [Rei95], on object-oriented programming; [HL95], on a model for the lambda calculus; [Len96], on a higher-order concurrent language; [Jac96a], on behaviour refinement in object-oriented programming.

The following papers are using non-wellfounded sets as the starting point for semantics: [Rut92], [Acz94], and [FHL94], on processes and non-wellfounded sets; [BM96], a recent textbook on non-wellfounded sets and circularity; and [MD97], where corecursion is further studied in that context.

Other categorical approaches to bisimulation include [Abr91], on a domain for bisimulation; [WN95], on categories of transition systems; [Pit94], [Fio96], and [Pit96], on mixed induction-coinduction principles on domains in terms of relational properties; [HJ96], on functors on categories of relations; [JNW96], on a characterization of bisimulation in terms of open maps and presheaves. In [BV96], a metric domain for bisimulation can be found.

Acknowledgements

The numerous conversations with Bart Jacobs over the past two years during his stay at the CWI have greatly increased my understanding of category theory in general and of coalgebras in particular. In addition to the above mentioned sections, also the idea of coalgebra as a general theory of (dynamical) systems has been influenced by our intensive interaction. Many thanks to Andrea Corradini, Jan van Eijk, Jan Heering, Furio Honsell, Wilfried Meyer Viol, Larry Moss, Prakash Panangaden, Maarten de Rijke, Bill Rounds, Erik de Vink, for discussions; to Andrea Corradini, Bart Jacobs, and Elena Marchiori, for detailed comments on a draft version of this paper; to Kees van Kemenade, for making the picture in Section 16. As always, I have learned much from the comments of the Amsterdam Concurrency Group, headed by Jaco de Bakker.

18 Appendix

This section is intended to give an overview of some basic facts on sets and categories, and also to mention one or two facts that may be less familiar. (The latter are indicated as propositions.) It is to be consulted when needed.

On sets

Composition of functions $f : S \to T$ and $g : T \to U$ is written as $g \circ f : S \to U$. We write 0 for the empty set, and $1 = \{*\}$ for the one element set. The identity function on a set S is denoted by $1_S : S \to S$. The sets of natural numbers and integers are denoted by

$$\mathcal{N} = \{0, 1, 2, \ldots\}, \quad \mathcal{Z} = \{0, 1, -1, 2, -2, \ldots\}.$$

The set of functions between sets S and T is denoted by

$$S^T = \{f : S \to T\}.$$

Let A be any set. The following notation will be used for sets of *streams* (or sequences, or lists) over A:

1. A^*: the set of all finite streams of elements of A; ϵ denotes the empty stream.

2. A^+: the set of all non-empty finite streams.

3. A^ω: the set of all infinite streams.

4. $A^\infty = A^* \cup A^\omega$: the set of all finite and infinite streams.

5. $A^\infty_+ = A^+ \cup A^\omega$: the set of all non-empty finite and infinite streams.

Let S be any set and R an equivalence relation on S. Let the quotient set S/R be defined by $S/R = \{[s]_R \mid s \in S\}$, with $[s]_R = \{s' \mid (s, s') \in R\}$. Let $\epsilon_R : S \to S/R$ be the surjective mapping sending each element s to its equivalence class $[s]_R$. It is called the *quotient map* of R.

The *diagonal* (or equality) Δ_S of a set S is given by

$$\Delta_S = \{\langle s, s\rangle \in S \times S \mid s \in S\}.$$

Let $f : S \to T$ be any mapping. The *kernel* $K(f)$ and the *graph* $G(f)$ of f are defined as follows:

$$K(f) = \{\langle s, s'\rangle \mid f(s) = f(s')\},$$

$$G(f) = \{\langle s, f(s)\rangle \mid s \in S\}.$$

For subsets $V \subseteq S$ and $W \subseteq T$, let

$$f(V) = \{f(s) \mid s \in V\},$$

$$f^{-1}(W) = \{s \mid f(s) \in W\}.$$

The set $f(S)$ is called the *image* of f. More generally, for functions $f : S \to T$ and $g : S \to U$, the image of f and g is defined by

$$\langle f, g\rangle(S) = \{\langle f(s), g(s)\rangle \mid s \in S\}.$$

Also the following notation will be used: for $f : S \to T$, $R \subseteq S \times S$ and $Q \subseteq T \times T$,

$$f(R) = \{\langle f(s), f(s')\rangle \mid \langle s, s'\rangle \in R\},$$

$$f^{-1}(Q) = \{\langle s, s'\rangle \mid \langle f(s), f(s')\rangle \in Q\}.$$

Let S, T and U be sets, $R \subseteq S \times T$ a relation between S and T, and $Q \subseteq T \times U$ a relation between T and U. The *inverse* R^{-1} of R is defined by

$$R^{-1} = \{\langle t, s\rangle \mid \langle s, t\rangle \in R\},$$

and the *composition* $R \circ Q$ of R and Q is defined by

$$R \circ Q = \{\langle s, u\rangle \mid \exists t \in T,\ \langle s, t\rangle \in R \text{ and } \langle t, u\rangle \in Q\}.$$

Note the difference in order between function composition and relation composition.

On categories

Some familiarity with the following notions will be helpful (but is not strictly necessary for understanding the rest of the paper): category; functor; epi; mono; limit and colimit (in particular, pullback, coequalizer, initial object, final object); opposite category; product of categories.

On the category of sets

The category of sets and functions between them is denoted by *Set*. It is complete and cocomplete, i.e., all limits and colimits exist. A function is mono if and only if it is injective, and it is epi if and only if it is surjective.

Proposition 18.1 *Let $F : Set \to Set$ be an arbitrary functor. If $f : S \to T$ is mono and S is non-empty, then $F(f) : F(S) \to F(T)$ is mono as well.*

Proof: Let $s_0 \in S$ and define $g : T \to S$ by

$$g(t) = \begin{cases} s & \text{if there is (a unique) } s \in S \text{ with } t = f(s), \\ s_0 & \text{otherwise.} \end{cases}$$

Clearly, $g \circ f = 1_S$ and hence by functoriality of F, $F(g) \circ F(f) = F(1_S) = 1_{F(S)}$. Thus F(f) is injective, that is, mono. □

Below the functors that are used in this paper are described. First the basic functors are listed, which next are used to define a number of composed functors:

1. The identity functor: $I : Set \to Set$ sends sets and functions to themselves.

2. The constant functor A, where A is any set, maps any set to the set A, and any function to the identity function 1_A on A.

3. Coproduct (or sum):

   $$+ : Set \times Set \to Set$$

   It maps two sets to their disjoint union; a pair of functions $f : S \to S'$ and $g : T \to T'$ is mapped to $f + g : (S + T) \to (S' + T')$, sending s in S to $f(s)$ and $t \in T$ to $g(t)$. The coproduct of an indexed family of sets $\{S_i\}_i$ is denoted by

   $$\sum_i S_i.$$

4. Product:

   $$\times : Set \times Set \to Set$$

   It maps a pair of sets S and T to their Cartesian product $S \times T$; a pair of functions $f : S \to S'$ and $g : T \to T'$ is mapped to $f \times g : (S \times T) \to (S' \times T')$, sending $\langle s, t \rangle$ to $\langle f(s), g(t) \rangle$.

5. Function space:

   $$\to : Set^{op} \times Set \to Set$$

   It maps a pair of sets S and T to the set $S \to T$ of all functions from S to T. A pair of functions $f : S' \to S$ and $g : T \to T'$ is mapped to $(f \to g) : (S \to T) \to (S' \to T')$, which sends $\phi \in S \to T$ to $g \circ \phi \circ f \in S' \to T'$. This functor will mostly be used with a fixed choice, a set A say, for the left argument. Then it is denoted as follows:

   $$(-)^A : Set \to Set.$$

6. Powerset:

   $$\mathcal{P} : \mathit{Set} \to \mathit{Set}$$

   It maps a set S to the set of all its subsets $\mathcal{P}(S) = \{V \mid V \subseteq S\}$. A function $f : S \to T$ is mapped to $\mathcal{P}(f) : \mathcal{P}(S) \to \mathcal{P}(T)$, which is defined, for any $V \subseteq S$, by $\mathcal{P}(f)(V) = f(V)$. We shall also encounter the *finite* powerset: $\mathcal{P}_f(S) = \{V \mid V \subseteq S \text{ and } V \text{ is finite }\}$.

7. Contravariant powerset:

   $$\bar{\mathcal{P}} : \mathit{Set}^{op} \to \mathit{Set}$$

   acts on sets as $\mathcal{P}$ does: $\bar{\mathcal{P}}(S) = \mathcal{P}(S)$. A function $f : S \to T$ is mapped to $\bar{\mathcal{P}}(f) : \bar{\mathcal{P}}(T) \to \bar{\mathcal{P}}(S)$, which is defined, for any $V \subseteq T$, by $\bar{\mathcal{P}}(f)(V) = f^{-1}(V)$. Because

   $$\{V \mid V \subseteq S\} \cong 2^S$$

   (by representing a subset by its characteristic function), the contravariant powerset functor could equivalently be described as $F(S) = 2^S$. (Note that the definition on functions would indeed be the same.) The contravariant powerset functor will in particular be considered in composition with itself:

   $$\bar{\mathcal{P}} \circ \bar{\mathcal{P}} : \mathit{Set} \to \mathit{Set}.$$

   One easily verifies that a function $f : S \to T$ is mapped by this composition to

   $$\bar{\mathcal{P}}(\bar{\mathcal{P}}(f)) : \bar{\mathcal{P}}(\bar{\mathcal{P}}(S)) \to \bar{\mathcal{P}}(\bar{\mathcal{P}}(T)), \quad \mathcal{V} \mapsto \{W \subseteq T \mid f^{-1}(W) \in \mathcal{V}\}.$$

Next a few examples are given of functors that are obtained by combining one or more of the basic functors mentioned above:

1. $F_1(S) = 1 + S$
2. $F_2(S) = A \times S$
3. $F_3(S) = 1 + (A \times S)$
4. $F_4(S) = S \times S$
5. $F_5(S) = \mathcal{P}(A \times S)$
6. $F_6(S) = (B \times S)^A$
7. $F_7(S) = 1 + ((A \times S) \times (A \times S))$

The definition of how these functors act on functions, follows from the definitions of the basic functors above. For instance, the functor F_6 sends a function $f : S \to T$ to the function $(B \times f)^A$, which maps a function ϕ in $(B \times S)^A$ to the function $\tilde{\phi}$ in $(B \times T)^A$, defined by $\tilde{\phi}(a) = \langle b, f(s)\rangle$, where $\phi(a) = \langle b, s\rangle$.

Next a few limit and colimit constructions in *Set* are described explicitly. A *pullback* of functions $f : S \to T$ and $g : U \to T$ is a triple $(P,\ k : P \to S,\ l : P \to U)$ with $f \circ k = g \circ l$ such that for any set X and functions $i : X \to S$ and $j : X \to U$ with $f \circ i = g \circ j$ there exists a unique (so-called mediating) function $h : X \to P$ with $k \circ h = i$ and $l \circ h = j$. In *Set*, a pullback of functions $f : S \to T$ and $g : U \to T$ always exists: the set

$$P = \{\langle s, u\rangle \in S \times U \mid f(s) = g(u)\},$$

with projections $\pi_1 : P \to S$ and $\pi_2 : P \to U$, is a pullback of f and g.

If (P, k, l) is a pullback of two functions f and g that are mono then k and l are mono.

We shall also need the following notion: a *weak* pullback is defined in the same way as a pullback, but without the requirement that the mediating function be unique. Weak and strong are the same if all functions involved are mono:

Proposition 18.2 *A weak pullback consisting of mono's is an (ordinary) pullback.* □

A *coequalizer* of two functions $f : S \to T$ and $g : S \to T$ is a pair $(U, c : T \to U)$ with $c \circ f = c \circ g$ such that for any function $h : T \to V$ with $h \circ f = h \circ g$ there exists a unique function $i : U \to V$ such that $i \circ c = h$. Also coequalizers always exist in *Set*: The quotient of T with respect to the smallest equivalence relation on T that contains the set

$$\{\langle f(s), g(s)\rangle \mid s \in S\}$$

is a coequalizer of f and g. For a set S and an equivalence relation R on S, the quotient map $\epsilon_R : S \to S/R$ can be readily seen to be the coequalizer of the projections from R to S:

$$\begin{tikzcd} R \arrow[r, shift left, "\pi_1"] \arrow[r, shift right, "\pi_2"'] & S \arrow[r, "\epsilon_R"] & S/R. \end{tikzcd}$$

The following diagrams show how in *Set*, the diagonal of a set S, and the kernel and the graph of a function $f : S \to T$ can be obtained as pullbacks:

$$\begin{tikzcd} \Delta_S \arrow[r, "\pi_1"] \arrow[d, "\pi_2"'] & S \arrow[d, "1_S"] \\ S \arrow[r, "1_S"'] & S, \end{tikzcd} \qquad \begin{tikzcd} K(f) \arrow[r, "\pi_1"] \arrow[d, "\pi_2"'] & S \arrow[d, "f"] \\ S \arrow[r, "f"'] & T, \end{tikzcd} \qquad \begin{tikzcd} G(f) \arrow[r, "\pi_1"] \arrow[d, "\pi_2"'] & S \arrow[d, "f"] \\ T \arrow[r, "1_T"'] & T. \end{tikzcd}$$

The composition of two relations can be described by means of pullback and image as follows. Consider two relations R and Q

$$\begin{tikzcd} & R \arrow[dl, "r_1"'] \arrow[dr, "r_2"] & & Q \arrow[dl, "q_1"'] \arrow[dr, "q_2"] & \\ S & & T & & U, \end{tikzcd}$$

with projections r_i and q_i. If we first take a pullback

$$\begin{tikzcd} & & X \arrow[dl, "x_1"'] \arrow[dr, "x_2"] & & \\ & R \arrow[dl, "r_1"'] \arrow[dr, "r_2"] & & Q \arrow[dl, "q_1"'] \arrow[dr, "q_2"] & \\ S & & T & & U, \end{tikzcd}$$

then it is easy to see that the composition of R and Q is the image of $r_1 \circ x_1$ and $q_2 \circ x_2$:

$$R \circ Q = \langle r_1 \circ x_1,\, q_2 \circ x_2\rangle(X).$$

The *union* of a collection of relations $\{R_i \subseteq S \times T\}_i$ can be obtained by means of coproduct and image: consider

$$\begin{tikzcd} S & \sum_i R_i \arrow[l, "k"'] \arrow[r, "l"] & T, \end{tikzcd}$$

where k and l are the componentwise projections. Then

$$\bigcup_i R_i = \langle k, l\rangle(\sum_i R_i).$$

The *intersection* of a collection $\{V_k\}_k$ of subsets of a set S can be constructed by means of a *generalized* pullback, which is so to speak a pullback of a whole family of arrows at the same time, as follows:

$$\begin{tikzcd} \bigcap_k V_k \arrow[r] \arrow[d] & V_k \arrow[d, "i_k"] \\ V_{k'} \arrow[r, "i_{k'}"'] & S, \end{tikzcd}$$

where $\{i_k : V_k \to S\}_k$ are the inclusion mappings. Note that all functions are mono.

Proposition 18.3 *Let $F : Set \to Set$ be a functor that preserves weak pullbacks, i.e., transforms weak pullbacks into weak pullbacks. Then F preserves intersections.*

Proof: Because F preserves weak pullbacks, the diagram above is transformed by F into a weak pullback diagram:

$$\begin{tikzcd}
F(\bigcap_k V_k) \arrow[r] \arrow[d] & F(V_k) \arrow[d, "F(i_k)"] \\
F(V_{k'}) \arrow[r, "F(i_{k'})"'] & F(S).
\end{tikzcd}$$

Because all functions in the original diagram are mono, and because F preserves mono's (Proposition 18.1), all functions in the second diagram are mono as well. By Proposition 18.2, the diagram is again a pullback in *Set*. Thus $F(\bigcap_k V_k)$ is (isomorphic to) $\bigcap_k F(V_k)$. □

As we shall see in Section 4 and Section 5, the requirement that functors preserve weak pullbacks is needed at various places in the theory. Therefore it is worthwhile to examine which functors have this property. First an easy proposition.

Proposition 18.4 *If a functor $F : Set \to Set$ preserves pullbacks then it also preserves* weak *pullbacks.* □

Many (combinations of the) functors mentioned above preserve pullbacks and hence weak pullbacks. To mention a few relevant examples: constant functors, identity, $A \times (-)$, $A + (-)$, $(-)^A$ (where A ia an arbitrary set). The proofs are easy. For instance, it is straightforward to prove that $A \times R$, where R is the pullback of two functions $f : S \to U$ and $g : T \to U$, is the pullback of the functions $A \times f : A \times S \to A \times U$ and $A \times g : A \times T \to A \times U$.

An exception is the (covariant) powerset functor: Consider $1 = \{0\}$ and $2 = \{0, 1\}$, and let $f : 2 \to 1$ be the unique constant function. Then

$$R = \{\langle 0,0\rangle,\ \langle 0,1\rangle,\ \langle 1,0\rangle,\ \langle 1,1\rangle\}$$

is a pullback of f with itself, but $\mathcal{P}(R)$ is not a pullback of $\mathcal{P}f$ with itself. It is, however, a weak pullback. More generally, it is not difficult to prove that $\mathcal{P}$ preserves weak pullbacks (cf. [Tur96]).

There is one functor in our list above that does not even preserve weak pullbacks. It is the contravariant powerset functor composed with itself ($\bar{\mathcal{P}} \circ \bar{\mathcal{P}}$). Take, for instance, $S = \{s_1, s_2, s_3\}$, $T = \{t_1, t_2, t_3\}$, $U = \{u_1, u_2\}$, $f : S \to U$ defined by $\{s_1 \mapsto u_1, s_2 \mapsto u_1, s_3 \mapsto u_2\}$ and $g : T \to U$ defined by $\{t_1 \mapsto u_1, t_2 \mapsto u_2, t_3 \mapsto u_2\}$. Then the image of the pullback of f and g is not a pullback and not even a weak pullback.

References

[Abr91] S. Abramsky. A domain equation for bisimulation. *Information and Computation*, 92(2):161–218, June 1991.

[Acz88] P. Aczel. *Non-well-founded sets.* Number 14 in CSLI Lecture Notes. Center for the Study of Languages and Information, Stanford, 1988.

[Acz94] P. Aczel. Final universes of processes. In S. Brookes, M. Main, A. Melton, M. Mislove, and D. Schmidt, editors, *Proceedings of the 9th International Conference on Mathematical Foundations of Programming Semantics*, volume 802 of *Lecture Notes in Computer Science*, pages 1–28. Springer-Verlag, 1994.

[AM80] M.A. Arbib and E.G. Manes. Machines in a category. *Journal of Pure and Applied Algebra*, 19:9–20, 1980.

[AM82] M.A. Arbib and E.G. Manes. Parametrized data types do not need highly constrained parameters. *Information and Control*, 52(2):139–158, 1982.

[AM89] P. Aczel and N. Mendler. A final coalgebra theorem. In D.H. Pitt, D.E. Ryeheard, P. Dybjer, A. M. Pitts, and A. Poigne, editors, *Proceedings category theory and computer science*, Lecture Notes in Computer Science, pages 357–365, 1989.

[AR89] P. America and J.J.M.M. Rutten. Solving reflexive domain equations in a category of complete metric spaces. *Journal of Computer and System Sciences*, 39(3):343–375, December 1989.

[Bar93] M. Barr. Terminal coalgebras in well-founded set theory. *Theoretical Computer Science*, 114(2):299–315, June 1993.

[Bar94] M. Barr. Additions and corrections to "terminal coalgebras in well-founded set theory". *Theoretical Computer Science*, 124(1):189–192, February 1994.

[BM96] J. Barwise and L.S. Moss. *Vicious Circles, On the Mathematics of Non-wellfounded Phenomena.* CSLI Lecture Notes. Center for the Study of Language and Information, Stanford, 1996.

[Bor94] F. Borceux. *Handbook of categorical algebra 1: basic category theory*, volume 50 of *Encyclopedia of mathematics and its applications.* Cambridge University Press, 1994.

[BP96] M.-P. Béal and D. Perrin. Symbolic dynamics and finite automata. Report IGM 96-18, Université de Marne-la-Vallée, 1996.

[BV96] J.W. de Bakker and E. de Vink. *Control Flow Semantics.* Foundations of Computing Series. The MIT Press, 1996.

[Coh81] P.M. Cohn. *Universal algebra*, volume 6 of *Mathematics and its applications.* D. Reidel Publishing Company, 1981.

[Dev86] R.L. Devaney. *An introduction to chaotic dynamical systems.* The Benjamin/Cummings Publishing Company, 1986.

[FHL94] M. Forti, F. Honsell, and M. Lenisa. Processes and hyperuniverses. In I. Privara, editor, *Proceedings of the 19th International Symposium on Mathematical Foundations of Computer Science*, volume 841 of *Lecture Notes in Computer Science*, pages 352–363. Springer-Verlag, 1994.

[Fio96] M. P. Fiore. A coinduction principle for recursive data types based on bisimulation. *Information and Computation*, 127(2):186–198, 1996.

[Gol87] R. Goldblatt. *Logics of time and computation.* Number 7 in CSLI Lecture Notes. Center for the Study of Language and Information, Stanford, 1987.

[Gol93] R. Goldblatt. *Mathematics of modality.* Number 43 in CSLI Lecture Notes. Center for the Study of Language and Information, Stanford, 1993.

[GV92] J.F. Groote and F. Vaandrager. Structured operational semantics and bisimulation as a congruence. *Information and Computation*, 100(2):202–260, October 1992.

[HJ96] C. Hermida and B. Jacobs. Structural induction and coinduction in a fibrational setting. Preprint, 1996.

[HL95] F. Honsell and M. Lenisa. Final semantics for untyped λ-calculus. In M. Dezani-Ciancaglini and G. Plotkin, editors, *Proceedings of the 2nd Conference on Typed Lambda Calculus and Applications*, volume 902 of *Lecture Notes in Computer Science*, pages 249–265. Springer-Verlag, 1995.

[HP79] M. Hennessy and G.D. Plotkin. Full abstraction for a simple parallel programming language. In J. Bečvář, editor, *Proceedings of 8th Symposium on Mathematical Foundations of Computer Science*, volume 74 of *Lecture Notes in Computer Science*, pages 108–120. Springer-Verlag, 1979.

[Jac95] B. Jacobs. Mongruences and cofree coalgebras. In V.S. Alagar and M. Nivat, editors, *Algebraic Methods and Software Technology*, number 936 in Lecture Notes in Computer Science, pages 245–260. Springer-Verlag, 1995.

[Jac96a] B. Jacobs. Behaviour-refinement of object-oriented specifications with coinductive correctness proofs. Report CSI-R9618, Computing Science Institute, University of Nijmegen, 1996. To appear in the proceedings of TAPSOFT'97.

[Jac96b] B. Jacobs. Inheritance and cofree constructions. In P. Cointe, editor, *European Conference on Object-Oriented Programming*, number 1098 in Lecture Notes in Computer Science, pages 210–231. Springer-Verlag, 1996.

[Jac96c] B. Jacobs. Objects and classes, co-algebraically. In B. Freitag, C.B. Jones, C. Lengauer, and H.-J. Schek, editors, *Object-Orientation with Parallelism and Persistence.* Kluwer Academic Publishers, 1996.

[JNW96] André Joyal, Mogens Nielsen, and Glynn Winskel. Bisimulation from open maps. *Information and Computation*, 127(2):164–185, 1996.

[Kel76] R.M. Keller. Formal verification of parallel programs. *Communications of the ACM*, 19(7):371–384, 1976.

[Ken88] R.E. Kent. The metric closure powerspace construction. In M. Main, A. Melton, M. Mislove, and D. Schmidt, editors, *Proceedings of the 3rd Workshop on Mathematical Foundations of Programming Language Semantics*, volume 298 of *Lecture Notes in Computer Science*, pages 173–199, New Orleans, 1988. Springer-Verlag.

[KM96] Y. Kawahara and M. Mori. A small final coalgebra theorem. 1996. To appear.

[Len96] M. Lenisa. Final semantics for a higher-order concurrent language. In H. Kirchner, editor, *Proceedings of CAAP'96*, volume 1059 of *Lecture Notes in Computer Science*, pages 102–118. Springer-Verlag, 1996.

[LS81] D.J. Lehmann and M.B. Smyth. Algebraic specification of data types: A synthetic approach. *Mathematical Systems Theory*, 14:97–139, 1981.

[LS91] K.G. Larsen and A. Skou. Bisimulation through probabilistic testing. *Information and Computation*, 94:1–28, 1991.

[MA86] E.G. Manes and M.A. Arbib. *Algebraic approaches to program semantics.* Texts and monographs in computer science. Springer-Verlag, 1986.

[Man76] E.G. Manes. *Algebraic theories*, volume 26 of *Graduate Texts in Mathematics.* Springer-Verlag, 1976.

[MD97] L.S. Moss and N. Danner. On the Foundations of Corecursion. *Bulletin of the IGPL*, 1997. Special issue on papers from WoLLIC 1995, to appear.

[Mil75] R. Milner. Processes: a mathematical model of computing agents. In H.E. Rose and J.C. Shepherdson, editors, *Logic Colloquium'73*, volume 80 of *Studies in Logic*, pages 157–173. North-Holland, 1975.

[Mil80] R. Milner. *A Calculus of Communicating Systems*, volume 92 of *Lecture Notes in Computer Science.* Springer-Verlag, Berlin, 1980.

[ML71] S. Mac Lane. *Categories for the Working Mathematician*, volume 5 of *Graduate Texts in Mathematics*. Springer-Verlag, New York, 1971.

[MT92] K. Meinke and J.V. Tucker. Universal algebra. In S. Abramsky, Dov.M. Gabbay, and T.S.E. Maibaum, editors, *Handbook of Logic in Computer Science*, volume 1, pages 189–411. Oxford University Press, 1992.

[Par81] D.M.R. Park. Concurrency and automata on infinite sequences. In P. Deussen, editor, *Proceedings 5th GI conference*, volume 104 of *Lecture Notes in Computer Science*, pages 15–32. Springer-Verlag, 1981.

[Pit94] A.M. Pitts. A co-induction principle for recursively defined domains. *Theoretical Computer Science*, 124(2):195–219, 1994.

[Pit96] A.M. Pitts. Relational properties of domains. *Information and Computation*, 127(2):66–90, 1996.

[Plo81] G.D. Plotkin. A structural approach to operational semantics. Report DAIMI FN-19, Aarhus University, Aarhus, September 1981.

[Rei95] H. Reichel. An approach to object semantics based on terminal coalgebras. *Mathematical Structures in Computer Science*, 5:129–152, 1995.

[RT93] J.J.M.M. Rutten and D. Turi. On the foundations of final semantics: non-standard sets, metric spaces, partial orders. In J.W. de Bakker, W.-P. de Roever, and G. Rozenberg, editors, *Proceedings of the REX Workshop on Semantics: Foundations and Applications*, volume 666 of *Lecture Notes in Computer Science*, pages 477–530, Beekbergen, June 1993. Springer-Verlag. FTP-available at `ftp.cwi.nl` as pub/CWIreports/AP/CS-R9241.ps.Z.

[RT94] J.J.M.M. Rutten and D. Turi. Initial algebra and final coalgebra semantics for concurrency. In J.W. de Bakker, W.-P. de Roever, and G. Rozenberg, editors, *Proceedings of the REX School/Symposium 'A decade of concurrency'*, volume 803 of *Lecture Notes in Computer Science*, pages 530–582. Springer-Verlag, 1994. FTP-available at `ftp.cwi.nl` as pub/CWIreports/AP/CS-R9409.ps.Z.

[Rut92] J.J.M.M. Rutten. Processes as terms: Non-well-founded models for bisimulation. *Mathematical Structures in Computer Science*, 2(3):257–275, 1992.

[Rut95] J.J.M.M. Rutten. A calculus of transition systems (towards universal coalgebra). In A. Ponse, M. de Rijke, and Y. Venema, editors, *Modal Logic and Process Algebra, a bisimulation perspective*, volume 53 of *CSLI Lecture Notes*, pages 231–256, Stanford, 1995. CSLI Publications. FTP-available at `ftp.cwi.nl` as pub/CWIreports/AP/CS-R9503.ps.Z.

[SP82] M.B. Smyth and G.D. Plotkin. The category-theoretic solution of recursive domain equations. *SIAM Journal of Computation*, 11(4):761–783, November 1982.

[SS93] G. Schmidt and T. Ströhlein. *Relations and graphs, discrete mathematics for computer scientists.* EATCS Monographs on Theoretical Computer Science. Springer-Verlag, New York, 1993.

[TR97] D. Turi and J.J.M.M. Rutten. On the foundations of final coalgebra semantics: non-well-founded sets, partial orders, metric spaces. *Mathematical Structures in Computer Science*, 1997. To appear.

[Tur96] D. Turi. *Functorial operational semantics and its denotational dual.* PhD thesis, Vrije Universiteit, Amsterdam, September 1996.

[vG96] R. van Glabbeek. The meaning of negative premises in transition system specifications II. Report STAN-CS-TN-95-16, Department of Computer Science, Stanford University, 1996. Extended abstract in: Automata, Languages and Programming, Proceedings 23th International Colloquium, ICALP '96, Paderborn, Germany, July 1996 (F. Meyer auf der Heide and B. Monien, eds.), LNCS 1099, Springer-Verlag, 1996, pp. 502-513.

[vGSS95] R.J. van Glabbeek, S.A. Smolka, and B. Steffen. Reactive, generative, and stratified models of probabilistic processes. *Information and Computation*, 121:59–80, 1995.

[WN95] G. Winskel and M. Nielsen. Models for concurrency. In S. Abramsky, Dov M. Gabbay, and T.S.E. Maibaum, editors, *Handbook of Logic in Computer Science*, volume 4, pages 1–148. Oxford Science Publications, 1995.