TECHNISCHE UNIVERSITÄT DRESDEN

Fakultät Informatik

TUD–FI05–02–Januar 2005

Frank Felfe

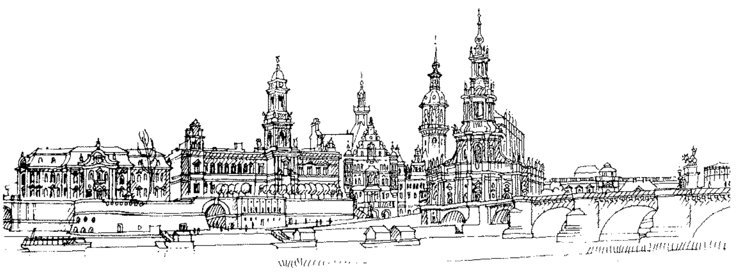
Institut für Theoretische Informatik

An Approach to Computable Coalgebras based on Recursive Functions

Technische Berichte

Technical Reports

ISSN 1430-211X



Technische Universität Dresden
Fakultät Informatik
D-01062 Dresden
Germany

URL: http://www.inf.tu-dresden.de/

An Approach to Computable Coalgebras based on Recursive Functions *

Frank Felfe
TU-Dresden
Institute of Theoretical Computer Science
D-01062 Dresden, Germany
felfe@tcs.inf.tu-dresden.de

January 21, 2005

This paper proposes a notion of computable coalgebras based on numbered sets and recursive functions similar to the notion of computable algebra. A model that is final for the category of computable coalgebras is constructed. An investigation of the computability of the final computable coalgebra motivates the use of partial structure maps. So a notion of computable coalgebra with partial structure map is developed. It is shown that the final model in the partial case has better computability properties than the final model in the total case.

*The author acknowledges the suport by grants from the DFG within the PhD Programme GK 334 ”Specification of discrete processes and systems of processes by operational models and logic”.

1 Introduction

Coalgebras are being used for modeling computation systems such as labeled transition systems or classes in object oriented programming languages [18]. Usually coalgebras are based on the category SET which contains far more sets and functions than can be computed in terms of Turing machines. Especially when using final semantics one gets models that are not implementable on a machine.

Consider the example of the functor $F(X) = \{0,1\} \times X$ which can be used to model binary streams. The final model (Z, ζ) for the class of F-coalgebras contains all functions $f : \omega \to \{0,1\}$. The structure map ζ maps some f to $(f(0), \lambda x \in \omega . f(x+1))$. While the structure map may be regarded computable, not all $f : \omega \to \{0,1\}$ can represented in a machine. Even in the case of unbounded memory capacity, the state space of an implementation of the final model can not contain non-Turing computable functions.

In algebra the problem of implementability has led to the development of the notion of computable or effective algebra [23]. The main idea is to consider enumerations from sets of natural numbers to the carriers of algebras such that the algebra functions can be tracked by recursive functions on the domain of the enumerations. This way the algebra functions can be simulated by recursive functions.

In this paper we develop a notion of computable coalgebra similar to the notion of computable algebra. A coalgebra that is final for the class of computable coalgebras is constructed and shown that it lacks important computability properties. This motivates the development of a notions of partial computable coalgebra for which the final coalgebra is shown to have nicer properties than the final model for total computable coalgebras.

1.1 Related Work

Our theory of computable coalgebras will be based on the theory of numbered sets or indexed sets. Indexed sets have been studied by Mal'cev [11] and Erschov [5] [6] [7]. The main idea of indexed sets is that every element of a set can be addressed by a natural number. The great disadvantage of this approach is that these sets must be countable.

To overcome this pitfall Weihrauch [24] developed a theory for so called type 2 sets where every object is defined as the limit of a sequence of finite objects. A way to define computability for real valued functions has been developed by Grzegorczyk [10]. He defines real numbers to be computable if they are limits of computable Cauchy sequences of rationals. These two approaches have been compared by Speen et al [22] and found to be equivalent.

The notion of effective algebra has been studied in depth by Tucker et al [23]. It is based on indexed sets and recursive functions. They show, that every computable or semicomputable algebra can be specified using initial algebra semantics and a finite or enumerable term rewrite system respectively. The case of final algebras was studied by Bergstra et al [23].

The notion of partial coalgebra has been used by Goldin et al [9] to model nontermination of transition systems. The case of infinite silent transition steps which would lead to nontermination is represented by nondefinedness of the transition for the respective state. The use of partial coalgebras in a computability context has been proposed by Reichel et al [8]. It was motivated by the fact that in a computable environment not all partial coalgebras can be modeled by total coalgebras for some functor $F(X) + 1$, where the case of undefinedness of the structure map is captured by the right injection.

A more general approach to partial coalgebra would be to consider total coalgebras for some functor $F(X)+X$ where the right injection can not be observed. This abstraction from $F(X)+X$ to $F(X)$ could lead to a partial mapping in case repeated application of the structure map does never result in the left injection. Such functors and abstractions were considered in [20] and [17] without defining partial coalgebras.

Computability in coalgebras has so far only been considered by Pattinson [14]. This work focuses on computability on final coalgebras by using an approximation-based approach and shows how computable functions can be defined in final coalgebras. The question if some model is implementable on a machine is not discussed. There exists a notion of recursive coalgebra that was introduced by Osius [13]. The idea of that notion is to extend the coalgebra by some kind of induction scheme to allow inductive definitions. The relation between our approach and recursive coalgebras is shortly discussed at the end of this paper.

2 Preliminaries

It is assumed that the reader is familiar with the theory of coalgebras and the theory of categories. For an introduction to coalgebras see [21] .

We will rephrase two definitions of final objects for a category that are needed in this paper. First we give the standard definition for an object of a category that is final in that category.

Definition 2.1 *An object Z of a category $\mathbb{C}$ is called final object of that category if for every object A of $\mathbb{C}$ there is a unique arrow $! : A \to Z$.*

It turns out that the requirement for Z to be an object of the category is sometimes too strict such that no final object exists. There might, however, exist objects with the finality properties outside of the category. The following two definitions taken from [12] give a notion of final object in a super-category.

Definition 2.2 *Let $\mathbb{C}$ be a category, $\mathbb{D}$ be a subcategory of $\mathbb{C}$ and A be an object of $\mathbb{C}$. A collection of arrows $\mathcal{S} : (f_B : B \to A)_{B \in ob(\mathbb{D})}$ is called a sink and denoted by $\mathcal{S} : \mathbb{D} \Rightarrow A$.*

If $\mathcal{S} : (f_B : B \to A)_{B \in ob(\mathbb{D})}$ is a sink and $h : A \to C$ is an arrow in $\mathbb{C}$ then $\mathcal{S}' : (h \circ f_B : B \to C)_{B \in ob(\mathbb{D})}$ is a sink denoted by $h \circ \mathcal{S}$.

A Sink $\mathcal{S}$ is epi if for all morphisms $h : A \to C$ and $g : A \to C$, $g \circ \mathcal{S} = h \circ \mathcal{S}$ implies $g = h$.

Now we can define the notion of final object that is located in a super-category. In order to keep apart the two notions of final object we will distinguish them as ”final object of a category” and ”object that is final for a category”.

Definition 2.3 *Let $\mathbb{C}$ be a category, $\mathbb{D}$ be a subcategory of $\mathbb{C}$ and A be an object of $\mathbb{C}$. We say that A is final object for $\mathbb{D}$ in $\mathbb{C}$ if the following is satisfied:*

- *there exists a unique sink $\mathcal{S} : \mathbb{D} \Rightarrow A$;*

- *the sink $\mathcal{S}$ is epi*

From the definitions of the sink and the final object follows immediately that an object of a category that is final for that category is the final object of that category.

Proposition 2.4 *Let Z be an object of category $\mathbb{C}$ that is final for the category $\mathbb{C}$ then Z is the final object of $\mathbb{C}$.*

2.1 Basic recursion theory

While this paper is concerned with coalgebraic issues and no recursion theoretic results are presented, a basic notion and results for recursive functions will be needed to understand the following paper. For a detailed discussion of recursion theory see [4] and [3].

The set of natural numbers is denoted by ω. Computable functions are defined as partial recursive functions. The class of partial recursive functions R^n of some arity n is enumerable and there exists a $n+1$-ary partial recursive function $\Phi^n : \omega \times \omega^n \to \omega$ such that for every $\phi \in R^n$ there exists some $e \in \omega$ such that $\Phi^n(e, x) = \phi(x)$, $x \in \omega^n$. The number e is called the (recursive) index of ϕ. Since in this paper only unary recursive functions are used we will omit the arity and write Φ instead of Φ^1. A partial recursive function f is called total on some set $\Omega \subseteq \omega$ if $f(n)$ is defined for all $n \in \Omega$.

A relation $R \subseteq \omega^n$ is called decidable or recursive if there exists a total recursive function $\chi : \omega^n \to \{0, 1\}$ such that $\chi(x) = 1$ if $x \in R$ and $\chi(x) = 0$ otherwise. Such χ is called characteristic function. A number n is a characteristic index of relation R if n is index of the characteristic function χ of R.

A relation $R \subseteq \omega^n$ is called semidecidable or recursively enumerable if there exists a partial recursive function $f : \omega^n \to \omega$ such that $x \in R$ iff $f(x)$ is defined. A number n is a characteristic index of R if n is index of such f. A relation $R \subseteq \omega^n$ is called co-semidecidable or co-recursively enumerable if the relation $\overline{R}$ where $x \in \overline{R}$ iff $x \notin R$ is semidecidable.

While every partial recursive function can be named by an index it is not decidable if two partial recursive functions $\Phi^n(x)$ and $\Phi^n(y)$ for $x \neq y, x, z \in \omega$ are equal. Likewise it is in general not decidable if some partial recursive function is defined for some $n \in \omega$.

It is a known fact in recursion theory, that the set of indices of all recursive functions that are total on some domain Ω is not recursively enumerable. Since we will use this result we give a proof of a special case for recursive functions from ω to $\{0, 1\}$. This shall demonstrate the diagonalisation method that is often used to show that some set is not recursive.

Theorem 2.5 *A set that contains at least one index for every total recursive function from ω to $\{0, 1\}$ can not be recursively enumerable.*

Proof Assume there exists a recursively enumerable set I containing at least one index for every total recursive functions. Then there exists a recursive function $f : \omega \to \omega$ those range is I. Consider the function $g : \omega \to \{0, 1\}$ with

$$g(n) = 1 - \Phi(f(n), n).$$

Obviously g is total recursive and different to every total recursive function with index in I. Hence I can not be an index set for all total recursive functions. $\square$

Alternatively such results can be derived from Rice's Theorem, a central result in recursion theory which states, that the only sets of partial recursive functions that have a recursive index set are the empty set and the set of all partial recursive functions. An index set for some set of recursive functions contains all indices of all functions in the set.

There exists a bijective encoding function $\langle \_, \_ \rangle : \omega^2 \to \omega$ that is total recursive such that the projections $\pi_{1,2} : \omega \to \omega$ with $\pi_1(\langle n, m \rangle) = n$ and $\pi_1(\langle n, m \rangle) = n$ are recursive functions too.

2.2 Categories of Numberings

In order to use recursive functions in coalgebras we need a notion for encoding elements of the coalgebra as natural numbers. Like in the theory of computable algebras we use numbered sets.

Definition 2.6 *A numbering of a set A (numbered Set) is a pair (Ω_A, ν_A) consisting of a set Ω_A of natural numbers and a (total) surjection $\nu_A : \Omega_A \to A$.*

A morphism between two numberings $\nu_A : \Omega_A \to A$ and $\nu_B : \Omega_B \to B$ is a function $f : A \to B$, which can be tracked by a function $f_\Omega : \Omega_A \to \Omega_B$, i.e. $\nu_B \circ f_\Omega = f \circ \nu_A$ or equivalently the following diagram commutes in SET.

$$\begin{array}{ccc} A & \xrightarrow{f} & B \\ {\scriptstyle \nu_A}\uparrow & & \uparrow{\scriptstyle \nu_B} \\ \Omega_\alpha & \xrightarrow[f_\Omega]{} & \Omega_\beta \end{array}$$

Numberings and morphisms define the category NSET *of numberings.*

Please note that the tracking function is not part of the morphism. There could be several functions that track the same morphism. Of course, for each tracking function the diagram must commute. A tracking function, on the other hand, can only track one morphism.

Some $n \in \Omega_A$ can be regarded to encode the $a \in A = \nu_A(n)$. Due to the surjectivity of ν_A this allows us to address every $a \in A$ by some natural number. This property is used to introduce a computability notion into the category of numberings. The kernel relation $\equiv_{\nu_A} \subseteq \Omega_A \times \Omega_A$ on the numbering $\nu_A : \Omega_A \to A$ is for $x, y \in \Omega_A$ defined by $x \equiv_{\nu_A} y$ iff $\nu_A(x) = \nu_A(y)$.

Definition 2.7 *A morphism $f : \nu_A \to \nu_B$ is called effective if it can be tracked by a recursive function f_Ω that is total on the domain of ν_A. If e is recursive index of that f_Ω then we also say that e tracks f.*

A numbering $\nu_A : \Omega_A \to A$ is called computable, semicomputable, co-semicomputable if the domain Ω_A and the relation $\equiv_{\nu_A}$ are both recursive, recursively enumerable, co-recursively enumerable respectively.

The recursivity of Ω_A allows us to decide wether $n \in \omega$ encodes some $a \in A$. The recursivity of the kernel relation allows to computationally distinguish numbers that encode different elements of the set. Since numberings will be used as carriers for coalgebras one would like to use computable numbering which give maximum control over the enumerated set.

The category formed by numbered sets and effective morphisms will be called eNSet. It is of special interest and it will be shown that the category of coalgebras on this category has a final object.

3 Computable Functors

For building coalgebras on numberings we need a notion of effectivity for functors. We consider endofunctors on the category NSet. The notion of effective functor was introduced in [14] and is here slightly modified to represent the different degrees of computability.

Definition 3.1 *A functor $F : \text{NSet} \to \text{NSet}$ is called effective if there is a recursive function ϕ_F such that whenever $f : (\Omega_A, \nu_A) \to (\Omega_{A'}, \nu_{A'}) \in \text{NSet}$ and $Ff = g : (\Omega_B, \nu_B) \to (\Omega_{B'}, \nu_{B'})$ and f_Ω tracks f then $\phi_F(f_\Omega)$ tracks g.*

The effective functor fulfils the least requirement, namely that recursive tracking functions are recursively mapped to recursive tracking functions. Please note, that an effective functor on NSet is also a functor on the subcategory eNSet. The more strict computable functor also requires recursive domains of numberings to be mapped to recursive domains and recursive kernels to be mapped to recursive kernels.

Definition 3.2 *An effective functor F is called computable if whenever $F(\Omega_A, \nu_A) = (\Omega_B, \nu_B)$ then*

1. *if Ω_A is a recursive (recursively enumerable, co-recursively enumerable) set so is Ω_B;*
2. *if $\equiv_{\nu_A}$ is recursive (recursively enumerable, co-recursively enumerable) so is $\equiv_{\nu_B}$ and there is a recursive function ψ_F such that if e is characteristic index of $\equiv_{\nu_A}$, then $\psi_F(e)$ is characteristic index of $\equiv_{\nu_B}$.*

We will use the following notation: For some $\nu_A : \Omega_A \to A$ and $F : \text{NSet} \to \text{NSet}$ we write $F(\nu_A) : \Omega_{F(A)} \to F(A)$. Please note, that F is not defined on sets, thus $F(A)$ is not a formally correct term. Here we use it to denote the enumerated set of $F(\nu_A)$ to make diagrams more readable. One has to be careful, because If ν_A^1 and ν_A^2 are two enumerations of the same set A then it is not guaranteed that $F(\nu_A^1)$ and $F(\nu_A^2)$ enumerate the same set.

Likewise we use $F(f_\Omega)$ to denote a tracking function for $F(f)$. Again one has to be careful, because the tracking function is not part of the morphism and there might be more than one tracking function for $F(f)$. The following diagram illustrates the case for the morphism $F(f)$ where $f : \nu_A \to \nu_B$:

$$\begin{array}{ccc} F(A) & \xrightarrow{F(f)} & F(B) \\ {\scriptstyle F(\nu_A)}\uparrow & & \uparrow{\scriptstyle F(\nu_B)} \\ \Omega_{F(A)} & \xrightarrow[F(f_\Omega)]{} & \Omega_{F(B)} \end{array}$$

The following lemma states that computable and effective functors can be constructed from computable and effective functors.

Lemma 3.3 *The composition of two effective functors yields an effective functor. The composition of two computable functors yields a computable functor.*

Proof Assume effective $F, G : \text{NSET} \to \text{NSET}$, with corresponding ϕ_F and ϕ_G. Let $f : \nu_A \to \nu_B$ be tracked by f_Ω. Then $F(f)$ is tracked by $\phi_F(f_\Omega)$ and $G(F(f))$ is tracked by $\phi_G(\phi_F(f_\Omega))$ making $G \circ F$ an effective functor.

Now assume F and G computable and $\nu_A : \Omega_A \to A$ be a computable numbering. Since F is computable $F(\nu_A)$ is computable and since G is computable $G(F(\nu_A))$ is computable too. $\square$

The notion effective and computable functor is extended to endofunctors on SET in the following way:

Definition 3.4 *A functor $F : \text{SET} \to \text{SET}$ is called effective/computable, if there is an effective/computable $G : \text{NSET} \to \text{NSET}$ with $U \circ G = F \circ U$, where $U : \text{NSET} \to \text{SET}$ is the canonical forgetful functor. In this case we call G an effective extension of F.*

Some examples of effective functors:

Example 3.5 *Let $\nu_C : \Omega_C \to C$ be the (constant) numbering of some set C. The constant functor $F_C : \text{NSET} \to \text{NSET}$ maps each $\nu_A \in \text{NSET}$ to ν_C and each $f : \nu_A \to \nu_B$ to the identity mapping $id : \nu_C \to \nu_C$. Since the identity mapping can be tracked by the identity function, which is computable, F_C is effective. If additionally ν_C is computable so is F_C.* $\square$

In the next examples div is the symbol for integer division and mod is remainder of integer division.

Example 3.6 *An effective extension of the functor $F(X) = 2 \times X = \{0,1\} \times X$ in NSET, where X represents the next state and $\{0,1\}$ some output, can be defined as follows: Let $\nu_X : \Omega_X \to X$ be the numbering of X. The numbering $\nu_{2\times X} : \Omega_{2\times X} \to 2 \times X$ is defined by*

$$n \mapsto (n \textit{ mod } 2, \nu_X(n \textit{ div } 2))$$

with domain

$$n \in \Omega_{2\times X} \Leftrightarrow (n \textit{ div } 2) \in \Omega_X.$$

This means, if n enumerates $x \in X$ then $2 \times n$ enumerates $(0, x)$ and $2 \times n + 1$ enumerates $(1, x)$. If Ω_X is recursive so is $\Omega_{2\times X}$. The characteristic function $\chi_X : \Omega_X \times \Omega_X \to \{0,1\}$ of $\equiv_X$ is mapped to $\chi_{2\times X} : \Omega_{2\times X} \times \Omega_{2\times X} \to \{0,1\}$, defined by

$$\chi_{2\times X}(n,m) = \begin{cases} 1 & \textit{if } (n \textit{ mod } 2 = m \textit{ mod } 2) \wedge \chi_X(\nu_X(n \textit{ div } 2), \nu_X(m \textit{ div } 2)) = 1 \\ 0 & \textit{otherwise.} \end{cases}$$

. In other words, (n,m) is in $\chi_{2\times X}$ exactly if n and m have the same output and their next states are in χ_X. A morphism $f : \nu_X \to \nu_Y$ is mapped to $F(f) : F(\nu_X) \to F(\nu_Y)$ with $F(f)(0,a) = (0, f(a))$ and $F(f)(1,a) = (1, f(a))$. If f_Ω is tracking function of f, then $F(f_\Omega)$ is a tracking function for $F(f)$ with

$$F(f_\Omega)(n) = (n \textit{ mod } 2) + 2 * f_\Omega(n \textit{ div } 2)).$$

The functor property is checked by

$$\begin{aligned}
F(g_\Omega) \circ F(f_\Omega)(n) &= F(g_\Omega)((n \textit{ mod } 2) + 2 * f_\Omega(n \textit{ div } 2))) \\
&= (((n \textit{ mod } 2) + 2 * f_\Omega(n \textit{ div } 2)) \textit{ mod } 2) \\
&\quad +2 \times g_\Omega(((n \textit{ mod } 2) + 2 * f_\Omega(n \textit{ div } 2)) \textit{ div } 2) \\
&= ((n \textit{ mod } 2) \textit{ mod } 2) + 2 \times g_\Omega((2 * f_\Omega(n \textit{ div } 2)) \textit{ div } 2) \\
&= (n \textit{ mod } 2) + 2 \times g_\Omega(f_\Omega(n \textit{ div } 2)) \\
&= F(g_\Omega \circ f_\Omega)(n).
\end{aligned}$$

The mappings of the characteristic function and the tracking function are clearly computable. Hence F is a computable functor. □

The exponent and powerset functors are clearly not computable because the cardinalities of function spaces and powersets are in general larger than the cardinalities of their arguments that are countable. For cartesian functors we can state the following:

Lemma 3.7 *Suppose $F : \text{SET} \to \text{SET}$ is build from computable constant functors and the identity functor by means of $+$ and $\times$. Then F is computable.*

Proof Proof: Numberings for $+$ and $\times$ can be constructed with the tupling of natural numbers making them computable. By Lemma 3.3 all functors build from computable constant functors and the identity functor by means of $+$ and $\times$ are computable. □

An effective extension to the exponent functor can not be defined because for counable sets A, B the function space $A \Rightarrow B$ is not a countable set. Also, the set $A \Rightarrow B$ contains functions that are not computable with respect to possible numberings of A and B such that the eval morphism of such an exponent object would not be effective. We are, however, interested in effective morphisms.

In order to make the eval function of a possible effective exponent functor effective, one would have to restrict to effective morphisms, i.e. a category that only contains effective morphisms which is eNSET. An exponent object $\nu_X^{\nu_C}$ in this category could be thought of containing all effective morphisms from ν_C to ν_X or all total recursive functions from Ω_C to Ω_X that track a morphism. Next, we define an endofunctor on eNSET that, if given some numbering ν_C, maps any numbering ν_X to a numbering of the effective morphisms from ν_C to ν_X.

Example 3.8 *We use an enumeration function $\Phi : \omega \times \omega \to \omega$ for partial recursive functions to construct an exponent object. Let $\nu_C : \Omega_C \to C$ and $\nu_X : \Omega_X \to X$ be numberings. The function $\hat{\nu}_{C \Rightarrow X} : \Omega_{C \Rightarrow X} \to C \Rightarrow X$ is defined by*

$$\Omega_{C \Rightarrow X} = \{n \mid \Phi(n) \textit{ tracks a morphism between } \nu_C \textit{ and } \nu_X\},$$

$$n \mapsto \lambda n_C \in \Omega_C.\nu_X(\Phi(n, n_c)), \textit{ where } \nu_C(n_c) = c.$$

$\hat{\nu}_{C \Rightarrow X}$ is not a numbering because it is not surjective. We define $\nu_{C \Rightarrow X} : \Omega_{C \Rightarrow X} \to range(\hat{\nu}_{C \Rightarrow X})$, $\nu_{C \Rightarrow X}(n) = \hat{\nu}_{C \Rightarrow X}(n)$ which is a numbering. We will show, that the mapping $EX_{\nu_C} :$ eNSET $\to$

ENSET, $\nu_X \mapsto \nu_{C\Rightarrow X}$ *is a functor. Let* $f : \nu_X \to \nu_Y$ *be tracked by* f_Ω *then* $EX_{\nu_C}(f)$ *is tracked by*

$$\lambda n \in \omega.\Phi^{-1}(\lambda x \in \omega.f_\Omega(\Phi(n,x))).$$

In other words $EX_{\nu_C}(f)$ *maps some function* $h : C \to X$ *to* $f(h)$*. To show the functor property let* $f : \nu_X \to \nu_Y, g : \nu_Y \to \nu_Z$ *be tracked by* f_Ω *and* g_Ω *respectively. Then*

$$\begin{aligned} EX_{\nu_C}(g_\Omega) \circ EX_{\nu_C}(f_\Omega) &= \lambda n.\Phi^{-1}(g_\Omega(\lambda x.\Phi(n,x))) \circ \lambda n.\Phi^{-1}(f_\Omega(\lambda x.\Phi(n,x))) \\ &= \Phi^{-1}(g_\Omega(\lambda x.\Phi(\lambda n.\Phi^{-1}(f_\Omega(\lambda x.\Phi(n,x))),x))) \\ &= \lambda n.\Phi^{-1}(g_\Omega(\lambda x.\Phi(\Phi^{-1}(f_\Omega(\lambda x.\Phi(n,x))),x))) \\ &= \lambda n.\Phi^{-1}(g_\Omega(f_\Omega(\lambda x.\Phi(n,x)))) \\ &= EX_{\nu_C}(g_\Omega \circ f_\Omega) \end{aligned}$$

The functor EX_{ν_C} is effective because the mapping of the tracking function is a recursive function. It is not computable because neither is $\Omega_{EX_{\nu_C}}$ recursive nor is $\equiv_{\nu_{C\Rightarrow X}}$ decidable. This follows directly from Rice's theorem. □

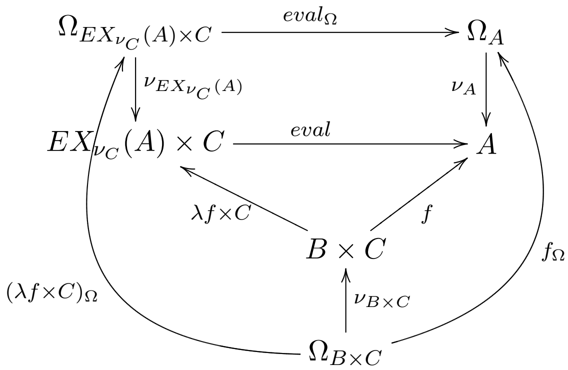
Now we want to show, that the functor EX_{ν_C} is a constant exponent functor on ENSET that maps some ν_X to the exponent object ${\nu_X}^{\nu_C}$. First we need a product functor. For the numberings $\nu_A : \Omega_A \to A$ and $\nu_B : \Omega_B \to B$ let

$$\nu_{A\times B} : \Omega_{A\times B} \to A \times B, \langle n,m\rangle \mapsto (\nu_A(n), \nu_B(m)).$$

$\nu_{A\times B}$ together with the projections of $A \times B$ is a product of ν_A and ν_B in NSET because $A \times B$ is product of A and B in SET. The projections can be tracked by the projections π_1 and π_2 of the tupling function. We define the product functor $\_ \times \_ :$ NSET $\times$ NSET $\to$ NSET by $(\nu_A, \nu_B) \mapsto \nu_{A\times B}$ and the mapping of arrows of $\times$ in SET. Since $\langle \_, \_\rangle, \pi_1$ and π_2 are computable, $\times$ is also a product functor in ENSET.

Theorem 3.9 *The functor* EX_{ν_C} *is a constant exponent functor in the category* ENSET*. The numbering* $\nu_{EX_{\nu_C}(\nu_X)}$ *is an exponent object.*

Proof We have to show that there exists a morphism $eval : \nu_{EX_{\nu_C}(A)\times C} \to \nu_A$ with the property that for each $f : \nu_{B\times C} \to \nu_A$ there exists exactly one $\lambda f : \nu_B \to \nu_{EX_{\nu_C}(A)}$ such that $f = eval \circ \lambda f \times A$.



Let $a \in A, b \in B, c, x \in C$ and $n, m \in \omega$, we define:

$$eval : (\lambda x.g(x), c) \mapsto g(c) \text{ tracked by } eval_\Omega : n \mapsto \Phi(\pi_1(n), \pi_2(n))$$

$$\lambda f : b \mapsto \lambda c.f(b, c) \text{ tracked by } (\lambda f)_\Omega : n \mapsto \Phi^{-1}(\lambda m.f_\Omega(\langle \pi_1(n), m \rangle))$$

then the diagram commutes. □

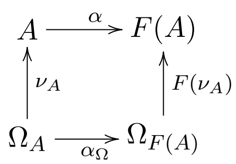
4 Computable Coalgebras

Having some categories and a notion of functor we can define coalgebras in the usual way as morphism from some object ν_A to $F(\nu_A)$. Unlike in computable algebra the numbering is part of the carrier. This allows us to reuse standard coalgebra notation.

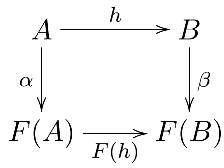
Definition 4.1 *A numbered coalgebra for some functor F : NSET → NSET is a tuple $(\nu_A : \Omega_A \to A, \alpha : \nu_A \to F(\nu_A))$.*

Let (ν_A, α) and (ν_B, β) be numbered coalgebras for some functor F. A morphism $h : \nu_A \to \nu_B$ is a numbered coalgebra homomorphism if $\beta \circ h = F(h) \circ \alpha$.

The letters $\mathcal{A}, \mathcal{B}$ etc will be used to denote coalgebras. The following diagrams in SET commute for numbered coalgebras and numbered coalgebra homomorphisms:



numbered coalgebra



numbered coalgebra homomorphism

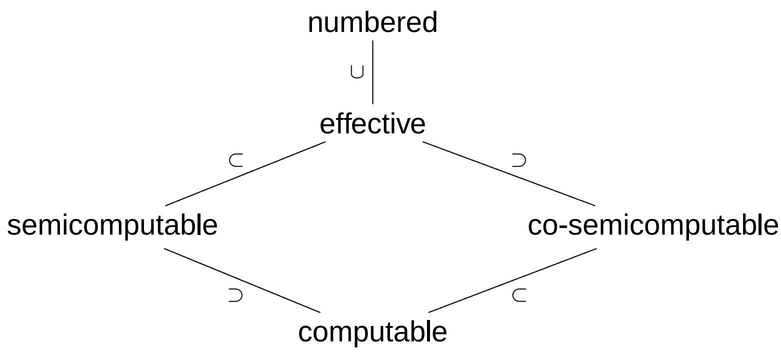
Please note that again the tracking function is not part of the coalgebra. The structure map consists only of a function from the set that is numbered by the carrier ν_A to the set that is numbered by $F(\nu_A)$. There could be several possible tracking function for one structure map.

The numbered coalgebra represents the most general case of coalgebras in this paper. Computability notion is introduced as restrictions on numbered coalgebras. The reason is that in general one does not know if some numbered coalgebra is computable. Some coalgebraic construction of a coalgebra from computable coalgebras may yield only a numbered coalgebra.

Notation 4.2 *Numbered coalgebras inherit the attributes effective, computable, semicomputable and co-semicomputable from their carriers and morphisms. For example an effective coalgebra has an effective morphism as structure map. A computable coalgebra contains a computable numbering as carrier and an effective morphism as structure map.*

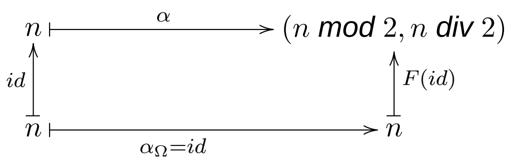
The properties of the single coalgebra types are listed in figure 1.

Numbered, effective, computable, semicomputable, and co-semicomputable coalgebras and their homomorphisms form categories. The following picture shows the subcategory relation of these categories. The category of numbered coalgebras contains all other categories and the category of computable coalgebras is contained in all other categories.



In this paper we will consider the coalgebras to be numbered unless specified differently. We will use the notation NSET_F for the category of numbered F-coalgebras and $c\mathrm{NSET}_F$ for the category of computable F-coalgebras.

Example 4.3 *We give an example of a computable coalgebra for the computable binary stream functor from example 3.6. Let the carrier of the coalgebra be $id : \omega \to \omega$ the identity enumeration of natural numbers. The structure map $\alpha : id \to F(id), n \mapsto (n \textit{ mod } 2, n \textit{ div } 2)$ can be tracked by the function $\alpha_\Omega = id$ because $id = \alpha_\Omega$ and $\alpha = F(id)$.*



The structure map can be regarded to output a binary encoding of the natural number it was started with. This coalgebra is computable because, the domain of the carrier is ω which is recursive, the kernel of the carrier is $\Delta_\omega = \{\langle n, n\rangle \mid n \in \omega\}$ which is decidable and the structure map can be tracked by the recursive function id. □

The notion of bisimulation is given in the usual way:

	domain	kernel	structure map
numbered	/	/	/
effective	/	/	recursive
semicomputable	recursively enumerable	semidecidable	recursive
co-semicomputable	co-recursively enumerable	co-semidecidable	recursive
computable	recursive	decidable	recursive

Figure 1: Computability properties of the coalgebra types

Definition 4.4 *Let $\mathcal{A} = (\nu_A : \Omega_A \to A, \alpha)$ and $\mathcal{B} = (\nu_B : \Omega_B \to B, \beta)$ be numbered F-coalgebras. A numbering $\nu_R : \Omega_R \to R$ for $R \subseteq A \times B$ is a bisimulation between the coalgebras $\mathcal{A}$ and $\mathcal{B}$ if there exists a transition structure $\alpha_R : \nu_R \to F(\nu_R)$ such that the projections from R to A and B are morphisms from (ν_R, α_R) to $\mathcal{A}$ and $\mathcal{B}$.*

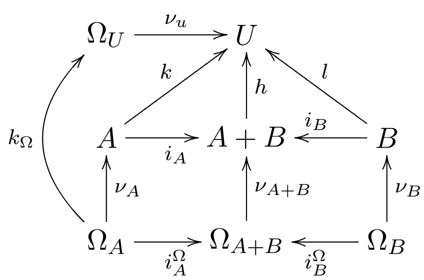
$a \in A$ and $b \in B$ are bisimilar if there exists a bisimulation $\nu_R : \Omega_R \to R$ between $\mathcal{A}$ and $\mathcal{B}$ such that $(a, b) \in R$.

A bisimulation that is an equivalence relation is called bisimulation equivalence.

The reminder of this section will show some basic results for numbered coalgebras, homomorphisms and bisimulations. Since the primary goal of the paper is to construct a numbered coalgebra that is final for the class of computable coalgebras we will restrict to those results that are needed.

Theorem 4.5 *Let F : NSET $\to$ NSET be a functor. In the category NSET$_F$ of numbered coalgebras all coproducts exist.*

Proof Consider the following diagram in SET:



$A + B$ is the coproduct in SET. We have to show, that there exists a numbering for $A + B$ and all the tracking functions.

The numbering ν_{A+B} can be given as:

$$
\begin{aligned}
2n \in \Omega_{A+B} &\Leftrightarrow n \in \Omega_A \\
2n + 1 \in \Omega_{A+B} &\Leftrightarrow n \in \Omega_A \\
\nu_{A+B}(2n) &= i_A(\nu_A(n)) \\
\nu_{A+B}(2n + 1) &= i_B(\nu_B(n)).
\end{aligned}
$$

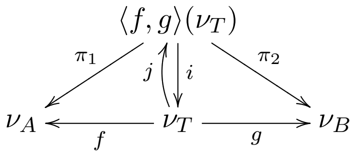
With $i_A^\Omega(n) = 2n$ and $i_B^\Omega(n) = 2n + 1$ can the injections be shown to commute. The tracking function h_Ω is given by $h_\Omega(2n) = k_\Omega(n), h_\Omega(2n + 1) = l_\Omega$ such that $h \circ \nu_{A+B} = \nu_U \circ h_\Omega$. $\square$

For computable coalgebras the construction of the coproduct would yield a computable coalgebra what can be seen from the definition of the numbering ν_{A+B} and the tracking functions.

The next result is an adaption of a standard result from [21]. Here F is an arbitrary functor on the category NSET. The proof can be adapted to numbered coalgebras because it mainly contains of compositions of morphisms in NSET$_F$.

Lemma 4.6 *The image $\langle f, g\rangle(\nu_T)$ of two numbered coalgebra homomorphisms $f : \nu_T \to \nu_A$ and $g : \nu_T \to \nu_B$ is a bisimulation between (ν_A, α) and (ν_B, β).*

Proof Consider the following diagram in NSET:



Morphism j is defined by $j(t) = \langle f(t), g(t)\rangle$ for which a tracking function can be given by f_Ω, g_Ω and the tupling of numbers. Function i is any right inverse for j, $j \circ i = 1$, π_1 and π_2 are projections. The transition structure $\gamma : \langle f, g\rangle(\nu_T) \to F(\langle f, g\rangle(\nu_T))$ is defined by $\gamma = F(j) \circ \alpha_T \circ i$.

$(\langle f, g\rangle(\nu_T), \gamma)$ is bisimulation between (ν_A, α) and (ν_B, β) because

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

The same holds for π_2. □

Using the previous result we can proof the existence of greatest bisimulations between numbered coalgebras.

Theorem 4.7 *The union $\bigcup_k \nu_R^k$ of a finite family $\{\nu_R^k\}_k$ of bisimulations between numbered coalgebras $\mathcal{A}$ and $\mathcal{B}$ is again a bisimulation.*

Proof $\bigcup_k \nu_R^k = \langle \pi_1, \pi_2\rangle(\sum_k \nu_R^k)$. Since the coproduct of two numbered coalgebras is again a numbered coalgebra it follows from lemma 4.6 that the union is a bisimulation. □

Corolary 4.8 *The set of all bisimulations between numbered coalgebras $\mathcal{A}$ and $\mathcal{B}$ is a complete lattice with least upper bound given by $\bigvee_k \nu_R^k = \bigcup_k \nu_R^k$.*

The greatest bisimulation between $\mathcal{A}$ and $\mathcal{B}$ exists and is denoted by $\sim_{AB} = \bigcup\{\nu_R \mid \nu_R$ is a bisimulation between $\mathcal{A}$ and $\mathcal{B}$.

The existence of greatest bisimulations between numbered coalgebras implies the existence of a greatest bisimulation between computable coalgebras. It does not imply computability properties for the greatest bisimulation such as a computable numbering and effective projections.

Definition 4.9 *Let $\nu_R : \Omega_R \to R$ be a bisimulation equivalence on the numbered coalgebra (ν_A, α). The quotient ν_A/R is defined as $\nu_{A/R} : \Omega_A \to A/R$ where A/R is the quotient of A by R and $\nu_{A/R}(n) = [\nu_A(n)]$. The quotient map $\varepsilon_R : \nu_A \to \nu_{A/R}$ is defined by $a \in A \mapsto [a]$.*

Please note that the numbering of R does not influence the quotient. It must be mentioned in the definition, because ν_R is a bisimulation. The quotient of a numbered coalgebra by a bisimulation equivalence defines a quotient coalgebra:

Proposition 4.10 *Let $\nu_R : \Omega_R \to R$ be a bisimulation equivalence on a numbered coalgebra (ν_A, α). Let $\varepsilon_R : \nu_A \to \nu_{A/R}$ be the quotient map of R. Then there is a unique transition structure $\alpha_{A/R} : \nu_{A/R} \to F(\nu_{A/R})$ such that ε_R is a numbered coalgebra morphism.*

Proof When forgetting the numberings, the unique transition structure $\alpha_{A/R} : A/R \to F(A/R)$ is known to exist. Remains to show that there exists a tracking function for $\alpha_{A/R}$. We use $\alpha^{\Omega}_{A/R} = F(id) \circ \alpha_\Omega \circ id^{-1}$.

$$
\begin{array}{ccc}
\Omega_{F(A)} & \xleftarrow{\alpha_\Omega} & \Omega_A \\
{\scriptstyle F(id)}\downarrow & & \downarrow{\scriptstyle id} \\
\Omega_{F(A/R)} & \xleftarrow{\alpha^{\Omega}_{A/R}} & \Omega_{A/R} = \Omega_A
\end{array}
$$

Then the coalgebra diagram for $(\nu_{A/R}, \alpha_{A/R})$ commutes because

$$
\begin{aligned}
\alpha_{A/R} \circ \nu_{A/R} &= \alpha_{A/R} \circ \varepsilon_R \circ \nu_A \circ id^{-1} \\
&= F(\varepsilon_R) \circ \alpha \circ \nu_A \circ id^{-1} \\
&= F(\varepsilon_R) \circ F(\nu_A) \circ \alpha_\Omega \circ id^{-1} \\
&= F(\nu_{A/R}) \circ F(id) \circ \alpha_\Omega \circ id^{-1} \\
&= F(\nu_{A/R}) \circ \alpha^{\Omega}_{A/R}
\end{aligned}
$$

□

At last we can make two statements about the computability of quotient coalgebras.

Lemma 4.11 *Let $\nu_R : \Omega_R \to R$ be a bisimulation equivalence on a numbered coalgebra (ν_A, α) for some effective functor F then*

1. *If α is tracked by a recursive function so is the quotient coalgebra $(\nu_{A/R}, \alpha_{A/R})$.*
2. *If ν_R and ν_A are computable (semicomputable, co-semicomputable) so is $\nu_{A/R}$.*

Proof Immediate from the definitions □

5 Final Model

In coalgebra theory the existence of a final coalgebra is a central issue because co-recursion and final semantics are important in coalgebraic specification. Thus, it is interesting to know if there exist final objects in the particular categories of coalgebras presented.

For the category of numbered coalgebras the answer is negative. For instance in the case of the binary stream functor the carrier of a final coalgebra would contain the set of all binary streams which has a cardinality larger than the set of natural numbers, making a numbering impossible.

This section will focus on final models for categories of effective coalgebras. We will show, that there exists a final coalgebra for the category of effective coalgebras. The category of computable or semi-computable coalgebras does not possess a final object, however, there exist effective coalgebras, that are final for the class of computable or semi-computable coalgebras.

Proposition 5.1 *The categories of computable coalgebras and semi-computable coalgebras do in general not have a final object.*

Proof We show an example of a functor which does not admit a final coalgebra. Let F be the binary stream functor from example 3.6. First, we show that every element of a semi-computable F-coalgebra defines a computable binary stream. Let $(\nu_A : \Omega_A \to A, \alpha)$ be a semi-computable F-coalgebra, $\nu_A(n) = a, n \in \Omega_A$ and $g : \omega \times \omega \to \{0, 1\}$ be the function that computes the output of n after m steps defined by

$$g(n, 0) = \alpha(n) \text{ mod } 2$$
$$g(n, m + 1) = g(\alpha(n) \text{ div } 2, m)$$

. Then the binary stream defined by this n can be given by $f : \omega \to \{0, 1\}, f(m) \mapsto g(n, m)$. Since α is total recursive on Ω_A f is total recursive, such that every element of a semi-computable (and also computable) F-coalgebra defines a computable binary stream.

Next we show, that every computable binary stream represents the observable behavior of an element of a computable F- coalgebra. Let $f : \omega \to \{0, 1\}$ be a total recursive function specifying a computable binary stream. Consider an F-coalgebra with carrier $id : \omega \to \omega$ and structure map $\alpha_\Omega : n \in \omega \mapsto f(n) + 2 * (n + 1)$. The element 0 defines the stream $f' : \omega \to: \{0, 1\}$:

$$\begin{aligned} f'(m) &= (f(m) + 2 * (m + 1)) \text{ mod } 2 \\ &= f(m) \end{aligned}$$

and hence $f = f'$.

If there existed a final computable F-coalgebra then all computable binary streams would occur in it. So we could use the recursive domain of the carrier and the computable tracking function of the structure map to enumerate all computable binary streams, i.e. all total recursive functions from ω to $\{0, 1\}$. This contradicts the fact that the set of total recursive functions is not recursively enumerable by theorem 2.5. Since a computable coalgebra is also semi-computable, a final semi-computable F-coalgebra would contain all computable binary streams too, such that the same argument applies as above and no final semi-computable coalgebra exists for F.

Alternatively one could consider decidability of the kernel relation of some final computable coalgebra. From Rice's theorem follows then, that the equality of two recursive functions is not decidable. □

Even though there does in general not exist a computable final coalgebra for effective functors on NSET there might exist one in some extreme cases, usually of very simple functors. The final coalgebras for such functors are very simple and do not have any practical use. Here are 2 such examples.

Example 5.2 *Consider the computable functor Id : NSET $\to$ NSET that maps objects and arrows to themselves. The coalgebra with the carrier $\nu_* : \{0\} \to \{*\}, \nu_*(0) = *$ and the identity function as structure map is final for the class of computable Id-coalgebras. The final morphism maps every element of a coalgebra to $*$ and is tracked by $f_\Omega : \omega \to \{0\}, n \mapsto 0$. (ν_*, id) is computable.* □

Example 5.3 *Consider the constant functor F_C from example 3.5 that maps each numbering to the constant numbering $\nu_C : \Omega_C \to C$, and the effective coalgebra (ν_C, id), where id is the identity function mapping $c \in C$ to itself. There exists a unique morphism from every numbered F- coalgebra (ν_A, α) to (ν_C, id), namely α, because the homomorphism property holds: $F_C(!_A) \circ \alpha = F_C(\alpha) \circ \alpha = id \circ \alpha$.*

Hence, (ν_C, id) is final for the class of (semi-)computable F_C-coalgebras iff ν_C is computable. If ν_C is only semicomputable then it is only final for the class off semi-computable F_C-coalgebras. □

As we have seen above there does in general not exist a final computable coalgebra. This problem stems from the fact that sets of indices of total recursive functions are not recursive. Nevertheless the sets of indices of total recursive functions do exist but are not recursive.

This suggests that there might exist effective coalgebras that are outside the category of computable coalgebras for some effective functor, which otherwise have the property of final coalgebras, i.e. there exists a unique morphism from every computable coalgebra to it. We will show that such a coalgebra exists for every effective functor by constructing such a numbered final coalgebra and show that it is effective.

A known approach for constructing the final F-coalgebra is as quotient of the disjoint union of all F-coalgebras with respect to its greatest bisimulation (see [21]). Such a final coalgebra contains all possible behavior from all F-coalgebras.

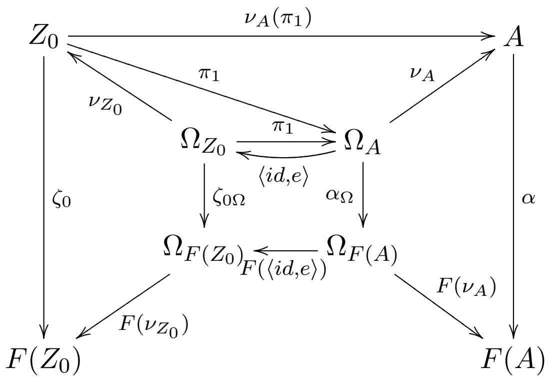
This approach is modified in the following way: We assume an effective functor $F : \text{NSET} \to \text{NSET}$ with recursive ϕ_F as mapping for the tracking functions. The behavior of some element a of some computable coalgebra (ν_A, α) is characterized by the tracking function α_Ω of α and any $n \in \Omega_A$ with $\nu_A(n) = a$. So we use the set of all pairs (n, ϕ) of natural numbers and recursive functions such that ϕ is tracking function of some computable coalgebra those carrier numbering domain contains n:

$$Z_0 = \{(n, f) \mid n \in \omega, f \text{ recursive }, \exists(\nu_A : \Omega_A \to A, \alpha).(n \in \Omega_A \wedge f \text{ tracks } \alpha)\}$$

The numbering for Z_0 is given using the enumeration function for partial recursive functions and tupling of natural numbers

$$\nu_{Z_0} : \Omega_{Z_0} \to Z_0, \langle n, e\rangle \mapsto (n, \Phi(e))$$

where $\langle n, e\rangle \in \Omega_{Z_0}$ iff $(n, \Phi(e)) \in Z_0$. When constructing the structure map for Z_0 we have to consider, that (n, ϕ) is intended to represent the behavior of some element a encoded by n and tracking function ϕ. The final morphism for coalgebra (ν_A, α) will map a to (n, ϕ). This fact is depicted in the following diagram in SET showing computable coalgebra (ν_A, α) and the numbered coalgebra (ν_{Z_0}, ζ_0):



The tracking function α_Ω has the index e such that $\Phi(\pi_2(\langle n, e\rangle)) = \alpha_\Omega$. $F(\langle id, e\rangle)$ is given by $\phi_F(\langle id, e\rangle)$. The tracking function

$$\zeta_{0\Omega} = F(\langle id, e)\rangle) \circ \Phi(\pi_2) \circ \pi_1$$

with $F(\langle id, e\rangle) = F(\langle \pi_1, \pi_2\rangle\rangle)$ defines the structure map

$$\zeta_0 : Z_0 \to F(Z_0) = F(\nu_{Z_0}) \circ \zeta_{0\Omega} \circ \nu_{Z_0}^{-1}$$

making (ν_{Z_0}, ζ_0) a numbered coalgebra. $\nu_{Z_0}^{-1}$ is an arbitrary inverse of ν_{Z_0}. ζ_Ω is independent of the actual choice of the inverse, because the right projections of two different tuples that enumerate the same element of Z_0 denote the same recursive functions such that the application of the definition of $\zeta_{0\Omega}$ yields the same mapping.

In order to get rid of duplicate behavior in (ν_{Z_0}, ζ_0) we build the quotient with respect to the greatest bisimulation $\sim_{Z_0 Z_0}$

$$(\nu_Z, \zeta) = (\nu_{Z_0}, \zeta_0)|_{\sim_{Z_0 Z_0}}.$$

Proposition 5.4 *The coalgebra $\mathcal{Z} = (\nu_Z, \zeta)$ is final for the class of computable coalgebras for the functor F.*

Proof First, we have to show that $\mathcal{Z}$ forms a unique sink by constructing a morphism $!_\mathcal{A}$ from any computable coalgebra $\mathcal{A} = (\nu_A, \alpha)$ to it. The morphism $!_\mathcal{A} : \mathcal{A} \to \mathcal{Z}, a \mapsto (n, \alpha)$ where $\nu_A(n) = a$ is tracked by $\langle id, e\rangle$, the tupling of the identity function and any index e of α_Ω. $!_\mathcal{A}$ is a numbered coalgebra homomorphism because

$$\begin{aligned}
\zeta \circ !_\mathcal{A} &= F(\nu_Z) \circ \zeta_\Omega \circ \nu_Z^{-1} \circ !_\mathcal{A} \\
&= F(\nu_Z) \circ F(\langle id, e\rangle) \circ \alpha_\Omega \circ \pi_1 \circ \nu_Z^{-1} \circ !_\mathcal{A} \\
&= F(!_\mathcal{A}) \circ F(\nu_A) \circ \alpha_\Omega \circ \pi_1 \circ \nu_Z^{-1} \circ !_\mathcal{A} \\
&= F(!_\mathcal{A}) \circ \alpha \circ \nu_A \circ \pi_1 \circ \nu_Z^{-1} \circ !_\mathcal{A} \\
&= F(!_\mathcal{A}) \circ \alpha.
\end{aligned}$$

Hence $\mathcal{S} = (!_\mathcal{A} : \mathcal{A} \to \mathcal{Z})_{\mathcal{A} \in ob(c\text{NSET}_F)}$ is a sink. It is unique because $\mathcal{Z}$ is the quotient by the greatest bisimulation, hence simple and there is at most one arrow from every coalgebra in NSET_F to it.

Epiness: Consider an element a of some F-coalgebra $\mathcal{A}$, an F-coalgebra $\mathcal{B}$ and $f, g : \mathcal{Z} \to \mathcal{B}$ with $f \circ \mathcal{S} = g \circ \mathcal{S}$. Since $!_\mathcal{A}$ is unique there is exactly one $z \in Z$ that is an homomorphic image of a. If $f(!_\mathcal{A}(a)) = g(!_\mathcal{A}(a))$ then $f(z) = g(z)$. Since this holds for all elements of all F-coalgebras, it holds that for all $z \in Z$ $f(z) = g(z)$. Hence $f = g$. □

Example 5.5 *We consider the final coalgebra for the class of computable coalgebras for the binary stream functor from example 3.6. Some $\langle n, e\rangle \in \Omega_Z$ behaves as follows:*

$$\begin{array}{ccc}
(n, \Phi(e)) & \overset{\zeta}{\longmapsto} & (\Phi(e)(n) \textit{ mod } 2, (\Phi(e)(n) \textit{ div } 2, \Phi(e))) \\
\uparrow \nu_Z & & \uparrow F(\nu_Z) \\
\langle n, e\rangle & \underset{\zeta_\Omega}{\longmapsto} & (\Phi(e)(n) \textit{ mod } 2) + 2 * \langle \Phi(e)(n) \textit{ div } 2, e\rangle
\end{array}$$

One can think of this final coalgebra to be identical to the coalgebra containing all computable binary streams, i.e. all total recursive functions $\phi : \omega \to \{0, 1\}$ that are enumerated by Φ. The structure map would map some $\phi : \omega \to \{0, 1\}$ to $(\phi(0), \lambda x.\phi(x + 1))$. □

Next we consider the computability properties of the constructed final coalgebra $\mathcal{Z}$.

Theorem 5.6 *Let F : NSET $\to$ NSET be an effective functor with $range(F(\nu)) \neq \emptyset$ for any numbering ν. The numbered coalgebra $\mathcal{Z}$ that is final for the class of computable F-coalgebras is effective but not computable.*

Proof $\mathcal{Z}$ is effective, because it is tracked by a recursive function which follows from the definition of $\zeta_{0\Omega}$ and lemma 4.11.

$\mathcal{Z}$ can not be computable because Ω_Z can not be recursive. This is because $\pi_2(\Omega_Z)$ would be an index set, since by definition of Z_0 either all indices of a recursive function are in it or none. By Rice's theorem such a set is either empty or ω. ω would include the index for the nowhere defined function which can not track a computable coalgebra because we don't allow partial structure maps. The case of empty Ω_Z is not possible because the range of $F(\nu)$ is nonempty such that there is at least the tracking function that maps all of $dom(\nu)$ to an existing element of $dom(F(\nu))$. □

The case of the constant functor to the empty set is deliberately excluded because theorem 5.6 would not hold for it. In order to have a structure map to the empty set the carrier of the coalgebra must be empty too, including the domain of the numbering. Hence ν_Z would be a numbering of the empty set which is trivially computable.

A coalgebra that is final for the class of semi-computable coalgebras for some functor can be constructed analogously. The computable coalgebra property is only used in the definition of Z_0. Thus, changing the definition to semi-computable coalgebras results in an effective coalgebra that is final for the class of semi-computable coalgebras. If we used effective coalgebras, then the result would be an effective coalgebra, that is final for the class of effective coalgebras.

Corolary 5.7 *The category of effective coalgebras has a final object.*

Proof The application of the definition of (ν_Z, ζ) to effective coalgebras yields by proposition 2.4 a final effective coalgebra. □

As we have seen, a reason for the fact that a coalgebra that is final for the class of computable coalgebras is not computable is the non-recursivity of the domain of its carrier. There are, however, cases of simple functors in which the domain of the carrier is recursively enumerable or its kernel is decidable. In general the domain and the kernel of the final effective coalgebra are non-decidable. To illustrate this, let's look at two examples of coalgebras that are final for the class of coalgebras for some functor.

Example 5.8 *Consider the constant Functor F_C from example 3.5. Then*

$$\Omega_Z = \{\langle n, e\rangle \mid \Phi(e, n) \downarrow \wedge\, \Phi(e, n) \in \Omega_C\}$$

which is recursively enumerable exactly if Ω_C is recursively enumerable because then $\Phi(e, n) \in \Omega_C$ is semidecidable.

The kernel $\equiv_{\nu_Z}$ can be decided by checking if $\Phi(e_1, n_1) = \Phi(e_2, n_2)$, which is possible because $n_i \in$ dom(Φ_{e_i}). Hence, if ν_C is semicomputable so is the final coalgebra for the class of computable coalgebras for the constant effective functor to ν_C. □

Example 5.9 *We look at the properties of the final effective coalgebra for the class of computable coalgebras for the binary stream functor from example 3.6.*

The domain of the carrier is not recursively enumerable because then the set of total recursive function from ω to $\{0, 1\}$ would be recursively enumerable as we have seen in the proof of proposition 5.1.

The kernel of the carrier is co-semicomputable because given two elements of the domain of the carrier we can compute the respective binary streams they represent. If their binary streams are different, then they are different at a finite position and we find out computationally after finite time. □

For coalgebraic specification this means, that when using final semantics the specified model does exist but does not have the properties of a computable coalgebra. Neither can one computationally determine whether 2 states behave equally nor can the state space be decided. However, the structure map is implementable in the sense that it is a recursive function. A consequence is that the greatest bisimulation is not decidable which has consequences when using co-induction combined with computability properties. In non-trivial cases one can at best expect the greatest bisimulation to be co-semidecidable.

The undecidability of the carrier is a disadvantage, because you are not able to give a concrete specification of it, e.g. as an algebra. This problem stems from the fact that you cannot recursively enumerate the set of total recursive functions. Recursively enumerable would be the set of partial recursive functions. Thus, the consideration of computable coalgebras motivates the consideration of numbered coalgebras with partial tracking functions and therefore with partial structure map.

6 Partial Computable Coalgebras

In this section we will develop a theory of computable coalgebras with partial structure map. The goal is to construct a final partial coalgebra and show that it has better computability properties than the final total coalgebra.

6.1 Partial Numbered Coalgebras

The notion of numbering remains the same as for computable coalgebras, i.e. we use (total) surjective maps from sets of natural numbers to some set A. A numbering is computable if its domain is recursive and if additionally the kernel is decidable. Morphisms are defined as follows:

Definition 6.1 *A partial morphism $f : (\nu_A : \Omega_A \to A) \rightharpoonup (\nu_B : \Omega_B \to B)$ is a partial function $f : A \rightharpoonup B$, which can be tracked by a partial function $f_\Omega : \Omega_A \rightharpoonup \Omega_B$, i.e. $f \circ \nu_A$ is defined iff $\nu_B \circ f_\Omega$ is defined and if both are defined then $f \circ \nu_A = \nu_B \circ f_\Omega$.*

A partial morphism is called effective if it can be tracked by a partial recursive function.

Numbered Sets and partial morphisms form the category PNSET. Functors in PNSET reflect the same idea as Functors in NSET but have to be defined for partial morphisms.

Definition 6.2 *A functor F : PNSET $\to$ PNSET is effective if there exists a recursive function ϕ_F such that whenever $f : \nu_A \rightharpoonup \nu_{A'} \in$ PNSET and $Ff = g : \nu_B \rightharpoonup \nu_{B'}$ and f_Ω tracks f then $\phi_F(f_\Omega)$ tracks g.*

Definition 6.3 *An effective functor F in PNSET is called computable if whenever $F(\nu_A : \Omega_A \to A) = (\nu_B : \Omega_B \to B)$ then*

- *if Ω_A is a recursive (recursively enumerable, co-recursively enumerable) set so is Ω_B;*
- *if $\equiv_{\nu_A}$ is recursive (recursively enumerable, co-recursively enumerable) so is $\equiv_{\nu_B}$ and there is a recursive function ψ_F such that if e is characteristic index of $\equiv_{\nu_A}$, then $\psi_F(e)$ is characteristic index of $\equiv_{\nu_B}$.*

A numbered partial coalgebra is again a morphism from some numbered set ν_A to $F(\nu_A)$.

Definition 6.4 *A numbered partial coalgebra for some effective functor F : PNSET $\to$ PNSET is a tuple $(\nu_A : \Omega_A \to A, \alpha : \nu_a \rightharpoonup F(\nu_A))$.*

Let (ν_A, α) and (ν_B, β) be numbered partial coalgebras for some functor F. A morphism $h : \nu_A \to \nu_B$ is a numbered partial coalgebra homomorphism if h is a total function, $\beta \circ h$ is defined iff $F(h) \circ \alpha$ is defined and if both are defined then $\beta \circ h = F(h) \circ \alpha$.

We use total morphisms because the intention of homomorphism is to be a mapping of the structure. A partial homomorphism could not fulfil this intention, because it would only map some part of the structure of a coalgebra. Please note that homomorphisms map elements of the carrier that are not in the domain of the structure map to elements that are not in the domain of the image of this structure map. Thus, non-definedness is treated as structural property that is preserved by homomorphisms.

Partial numbered coalgebras with an effective structure map and computable, semicomputable or co-semicomputable carrier we will call partial effective coalgebras, partial computable coalgebras, partial semicomputable coalgebras and partial co-semicomputable coalgebras respectively. The category formed by partial numbered F-coalgebras and partial morphisms is denoted by $\mathrm{PNS}\textsc{et}_F$, the category formed by partial computable F-coalgebras and partial effective morphisms is denoted by $c\mathrm{PNS}\textsc{et}_F$.

Bisimulation is defined as usual:

Definition 6.5 *Let $\mathcal{A} = (\nu_A : \Omega_A \to A, \alpha)$ and $\mathcal{B} = (\nu_B : \Omega_B \to B, \beta)$ be partial numbered F-coalgebras. A numbering $\nu_R : \Omega_R \to R$ for $R \subseteq A \times B$ is a bisimulation between the coalgebras $\mathcal{A}$ and $\mathcal{B}$ if there exists a transition structure $\alpha_R : R \to F(R)$ such that the projections from R to A and B are partial coalgebra morphisms.*

$a \in A$ and $b \in B$ are bisimilar if there exists a bisimulation ν_R for $\mathcal{A}$ and $\mathcal{B}$ such that $\langle a, b\rangle \in R$.

The next results are needed to define a coalgebra that is final for the class of partial computable coalgebras. They are equivalents of the presented results for total numbered coalgebras. The proofs are omitted, because they are mostly equal to the ones presented above. However, one has to take care of the case that a structure map is not defined for an element of a carrier.

Theorem 6.6 *Let $F : \mathrm{PNS}\textsc{et} \to \mathrm{PNS}\textsc{et}$ be a functor. In the category $\mathrm{PNS}\textsc{et}_F$ of partial numbered coalgebras all coproducts exist.*

Lemma 6.7 *The image $\langle f, g\rangle(\nu_T)$ of two partial numbered coalgebra homomorphisms $f : \nu_T \to \nu_A$ and $g : \nu_T \to \nu_B$ is a bisimulation between partial numbered coalgebras (ν_A, α) and (ν_B, β).*

Theorem 6.8 *The union $\bigcup_k \nu_R^k$ of a family $\{\nu_R^k\}_k$ of bisimulations between partial numbered coalgebras $\mathcal{A}$ and $\mathcal{B}$ is again a bisimulation.*

Corolary 6.9 *The set of all bisimulations between partial numbered coalgebras $\mathcal{A}$ and $\mathcal{B}$ is a complete lattice with least upper bound given by $\bigvee_k \nu_R^k = \bigcup_k \nu_R^k$. The greatest bisimulation between $\mathcal{A}$ and $\mathcal{B}$ exists and is denoted by $\sim_{AB} = \bigcup\{\nu_R \mid \nu_R$ is a bisimulation between $\mathcal{A}$ and $\mathcal{B}$.*

Definition 6.10 *Let $\nu_R : \Omega_R \to R$ be a bisimulation equivalence on the partial numbered coalgebra (ν_A, α). The quotient ν_A/R is defined as $\nu_{A/R} : \Omega_A \to A/R$ where A/R is the quotient of A by R and $\nu_{A/R}(n) = [\nu_A(n)]$.*

Proposition 6.11 *Let ν_R be a bisimulation equivalence on a partial numbered coalgebra (ν_A, α). Let $\varepsilon_R : \nu_A \to \nu_{A/R}$ be the quotient map of R. Then there is a unique transition structure $\alpha_{A/R} : \nu_{A/R} \to F(\nu_{A/R})$ such that ε_R is a partial numbered coalgebra morphism.*

Proof We cannot simply use the respective result for coalgebras in SET. Instead one could consider a total numbered coalgebra (ν'_A, α') for the functor $F + 1$ which behaves like (ν_A, α) but maps to 1 if the structure map is not defined. Now the proof from lemma 4.10 can be used for (ν'_A, α). The resulting numbered quotient coalgebra $(\nu_{A'/R}, \alpha_{A'/R})$ is turned into a partial numbered quotient coalgebra by forgetting the mappings to 1. This is possible, because the transformation represents an isomorphism between partial numbered F-coalgebras and total numbered $F + 1$ coalgebras. □

Lemma 6.12 *Let $\nu_R : \Omega_R \to R$ be a bisimulation equivalence on a partial numbered coalgebra (ν_A, α) for some effective functor F then*

1. *If α is tracked by a recursive function so is the quotient coalgebra $(\nu_{A/R}, \alpha_{A/R})$.*
2. *If ν_R and ν_A are computable (semicomputable, co-semicomputable) so is $\nu_{A/R}$.*

6.2 Final Partial Computable Coalgebras

For constructing a final partial numbered coalgebra we use the same method as for total numbered coalgebras. The carrier of the final coalgebra contains all pairs of natural numbers and partial recursive functions

$$Y_0 = \{(n, f) \mid n \in \omega, f \text{ partial recursive } \}$$

with the numbering using the enumeration function Φ

$$\nu_{Y_0} : \Omega_{Y_0} \to Y_0, \langle n, e\rangle \mapsto (n, \Phi(e))$$

where $\Omega_{Y_0} = \omega$, then

$$\gamma_{0\Omega} = F(\langle id, \pi_2\rangle) \circ \Phi(\pi_2) \circ \pi_1$$

defines the structure map $\gamma_0 : \nu_{Y_0} \to F(\nu_{Y_0}) = F(\nu_{Y_0}) \circ \gamma_{0\Omega} \circ \nu_{Y_0}^{-1}$ making (ν_{Y_0}, γ_0) a partial numbered coalgebra with computable tracking function.

At last we have to build the quotient of (ν_{Y_0}, γ_0) with respect to its greatest bisimulation $\sim_{Y_0 Y_0}$:

$$(\nu_Y, \gamma) = (\nu_{Y_0}, \gamma_0)/ \sim_{Y_0 Y_0}$$

Lemma 6.13 *Let F : PNSET $\to$ PNSET be an effective functor with $range(F(\nu)) \neq \emptyset$. The partial numbered coalgebra $\mathcal{Y} = (\nu_Y, \gamma)$*

1. *has an effective structure map;*
2. *has a carrier with recursive domain;*
3. *has a carrier with kernel that is not decidable;*
4. *is final for the class of parital computable coalgebras for the effective functor F.*

Proof (1) The recursivity of $\gamma_{0,\Omega}$ follows from its definition. From lemma 6.12 follows that γ_Ω is recursive, hence (ν_Y, γ) is effective.

(2) Ω_Y is by definition 6.10 equal to Ω_{0Y} which is defined as ω.

(3) Follows from the undecidability of the domain of recursive functions.

(4) First we show that $\mathcal{Y}$ forms a unique sink. The morphism $!_\mathcal{A} : \mathcal{A} \to \mathcal{Y}, a \mapsto (n, \alpha)$ with $\nu_A(n) = a$ is tracked by $\langle id, e\rangle$ the tupling of the identity function and an index e of α_Ω. $!_\mathcal{A}$ is a partial computable coalgebra homomorphism because

$$\begin{aligned}
\gamma \circ !_\mathcal{A} &= F(\nu_Y) \circ \gamma_\Omega \circ \nu_Y^{-1} \circ !_\mathcal{A} \\
&= F(\nu_Y) \circ F(\langle id, e\rangle) \circ \alpha_\Omega \circ \pi_1 \circ \nu_Y^{-1} \circ !_\mathcal{A} \\
&= F(!_\mathcal{A}) \circ F(\nu_A) \circ \alpha_\Omega \circ \pi_1 \circ \nu_Y^{-1} \circ !_\mathcal{A} \\
&= F(!_\mathcal{A}) \circ \alpha \circ \nu_A \circ \pi_1 \circ \nu_Y^{-1} \circ !_\mathcal{A} \\
&= F(!_\mathcal{A}) \circ \alpha
\end{aligned}$$

Hence $\mathcal{S} = (!_\mathcal{A} : \mathcal{A} \to \mathcal{Y})_{\mathcal{A} \in ob(c\mathrm{PNSET}_F)}$ is a sink. It is unique because $\mathcal{Y}$ is the quotient by the greatest bisimulation, hence simple and there is at most one arrow from every computable coalgebra in $c\mathrm{PNSET}_F$ to it.

Epiness: Partial coalgebra homomorphisms are total, hence we can apply the proof that was used in the total case. Consider an element a of some partial numbered F-coalgebra $\mathcal{A}$, an partial numbered F-coalgebra $\mathcal{B}$ and $f, g : \mathcal{Y} \to \mathcal{B}$ with $f \circ \mathcal{S} = g \circ \mathcal{S}$. Since $!_\mathcal{A}$ is unique there is exactly one $y \in Y$ that is an homomorphic image of a. If $f(!_\mathcal{A}(a)) = g(!_\mathcal{A}(a))$ then $f(y) = g(y)$. Since this holds for all elements of all F-coalgebras, it holds that for all $y \in Y$ $f(y) = g(y)$. Hence $f = g$. □

The functors that maps all numberings to the empty numbering are again excluded because the kernel of the carrier would be decidable since all elements had the same behavior and would be bisimilar. There are cases in which the kernel is semi-decidable, i.e. for the constant functor to a computable numbering. However, in most nontrivial cases the kernel is completely undecidable, which follows from the undecidability of the domain of recursive functions.

In conclusion, the final coalgebra for the class of partial computable coalgebras and the final coalgebra for the class of partial computable coalgebras have different properties. While both are effective, the partial one has a recursive domain and the total one has not. The kernel of the carrier in the total case can have better properties than that in the partial case. However, for nontrivial functors it is usually at most co-semidecidable.

7 Computable vs. Recursive Coalgebra

In [13] Osius introduced the notion of recursive coalgebra. The question arises if our notion of effective coalgebra is related to the notion of recursive coalgebra. In order to answer this we need to extend the notion of effectivity to coalgebras in SET. The obvious way is to call a coalgebra effective iff its carrier can be extended with a numbering which results in an effective coalgebra.

Definition 7.1 *Let F : SET $\rightarrow$ SET be a functor. A coalgebra $(A, \alpha : A \rightarrow F(A))$ is called effective/computable if there exists a numbering $\nu_A : \Omega_A \rightarrow A$ and an effective/computable extension G for F such that (ν_A, α) is an effective/computable G-coalgebra in* NSET.

The computable coalgebras in SET are exactly the images of computable coalgebras in NSET under the canonical forgetful functor. The term recursive coalgebra is defined as follows:

Definition 7.2 *Let $F : \mathbb{C} \rightarrow \mathbb{C}$ be a functor. An F-coalgebra (C, ϕ) is recursive iff for every F-algebra (A, α) there exists a unique morphism $f : C \rightarrow A$ such that $f = \alpha \circ F(f) \circ \phi$.*

$$\begin{array}{ccc} FC & \xleftarrow{\ \phi\ } & C \\ {\scriptstyle F(f)}\downarrow & & \downarrow{\scriptstyle f} \\ FA & \xrightarrow[\ \alpha\ ]{} & A \end{array}$$

This definition represents some kind of induction scheme for coalgebra (C, ϕ) which would allow recursive function definitions. The following two examples of a computable coalgebra that is not recursive and a recursive coalgebra that is not computable show that there is no direct relation between the two notions.

Example 7.3 *We use the binary stream functor from example* **??** *which is an extension of the* SET*-functor $\{0, 1\} \times X$ and the following one element coalgebra $\mathcal{C} = (\{a\}, a \mapsto \langle 0, a\rangle)$. $\mathcal{C}$ is trivially computable, i.e. with numbering $0 \mapsto a$ and identity as tracking function.*

Consider the algebra $\mathcal{A} = (\{b, c\}, \langle x, y\rangle \mapsto y)$. The mappings $f_1 : a \mapsto b$ and $f_2 : a \mapsto c$ make the diagram above commute. Hence $\mathcal{C}$ is not recursive. □

Example 7.4 *The fact that recursive coalgebras are not restricted in the cardinality of their carriers but effective coalgebras are is used to show that there exist recursive coalgebras that are not effective. Assume some recursive coalgebra $\mathcal{C}$ for some functor which has an extension in* NSET *and is not empty. The coproduct of non-countably many copies of $\mathcal{C}$ would still be recursive but its carrier would be non-countable and hence not computable.* □

We can state the following proposition:

Proposition 7.5 *The notion of computable/semicomputable and effective computable are not equal to the notion of recursive coalgebra.*

The notion of recursive and effective coalgebra are based on fundamentally different ideas. Whereas that of effective coalgebra is intended to make coalgebras implementable on a machine, the notion of recursive coalgebra represents some induction scheme and does not restrict the category the coalgebras are build on. An open question to be investigated is if countable recursive coalgebras are effective.

8 Conclusion and Open Problems

We have defined a notion for computability in coalgebras based on enumerated sets and recursive functions for the total and partial case. Basic properties known from coalgebra theory have been shown for these computable coalgebras. Especially a final coalgebra with restricted computability properties has been constructed.

Restrictions in the functor class might lead to better properties for the coalgebras. For instance could computable data-types, discussed in [16], and co-datatypes [1] be used for functor definitions. In order to be useful in specifications, computable versions for exponent and powerset functors as well as bifunctors must be developed.

With regard to the computability properties of the final coalgebras it would be interesting to find criteria for the case that the kernel of the final models possess decidability properties. An approach for that could be the restriction of the functor class. Another issue is to find appropriate logics for computable coalgebras. Usually modal logics are considered for coalgebras as in [19] and [15]. Since we are using recursive functions, equational logics that are used for computable datatypes in [23] and [2], might also be an appropriate approach.

References

[1] T.Altenkirch *Representation of first order function types as terminal coalgebras*. In S. Abramski, editor, Typed Lambda Calculi and Applications, volume 2044 of Lect. Notes in Comp. Sci., pages 8-21, 2001.

[2] J. A. Bergstra, J. V. Tucker *Initial and final algebra semantics for data type specifications: two characterization theorems*. SIAM Journal on Computing, 12:366-387, 1983.

[3] G.S. Boolos, R.C. Jeffrey: *Computability and logic*. Cambridge University Press, 1974.

[4] R. L. Epstein, A. W. Carnielli: *Computable functions, logic, and the foundations of mathematics*. Wadsworth 2000.

[5] Ju.L.Eršov *Theorie der Nummerierungen I*. Zeitschr. f. math. Logik Grundl. d. Math. 19 (1973) 289-388.

[6] Ju.L.Eršov *Theorie der Nummerierungen II*. Zeitschr. f. math. Logik Grundl. d. Math. 21 (1975) 473-584.

[7] Ju.L.Eršov *Theorie der Nummerierungen III*. Zeitschr. f. math. Logik Grundl. d. Math. 23 (1977) 289-371.

[8] F. Felfe, H. Reichel *Why partial coalgebras ?*. In M. Wirsing, D. Pattinson, and R. Hennicker, editors, WADT 2002, Technical Report 0207, pages 4-7. Institut für Informatik, LMU München, 2002.

[9] D. Goldin, P. Wegner *Mathematical Models of Interactive Computing*. Technical Report CS 99-13, Brown University Providence, January 1999.

[10] A. Grzegorczyk *On the definitions of computable real continious functions*. Fund. Math. 44 (1957 61–74).

[11] A.I.Mal'cev *The metamathematics of algebraic systems*. Collected Papers: 1936–1967 (B.F.WellsIII,ed)(North-Holland, Amsterdam,1971).

[12] D. Mašulović *Towards coalgebraic behaviourism*. In L. Moss, editor, Proceedings of CMCS, number 65.1 in Electronic Notes in Theoretical Computer Science, 2002.

[13] G.Osius, *Categorical set Theory: A characterization of the category of sets*. J. of Pure and Appl. Algebra (1974) 79-119.

[14] D. Pattinson *Computable functions on final coalgebras*. In H. P. Gumm, editor, Proceedings of CMCS, volume 82.1 in ENTCS, 2003.

[15] D. Pattinson *Semantical Principles in the Modal Logic of Coalgebras*. In A. Ferieira and H. Reichel, editors, Proceedings of STACS, volume 2010 of LNCS. Springer, 2001.

[16] P. H. Rodenburg *Algebraic specifiability of data types with minimal computable parameters*. Theoretical Computer Science, 85:97-116, 1991.

[17] J. Rothe and D. Mašulović *Towards Weak Bisimulation for Coalgebras*. In A.Kurz, editor, Proceedings of CMCIM'2002, volume 68.1 of ENTCS.

[18] J. Rothe, H. Tews, B. Jacobs: *The Coalgebraic Class Specification Language CCSL*. JUCS, 7(2):175-193, 2001.

[19] M. Rößiger *Coalgebras, Clone Theory and Modal Logic*. PhD Thesis, Univ. of Dresden, Germany, 2000.

[20] J. Rutten *A note on co-induction and weak bisimilarity for while programs*. Theoretical Informatics and Application 33:393-400, 1999.

[21] J. Rutten: *Universal coalgebra: a theory of systems*. Theoretical Computer Science, 249(1):3–80, 1998.

[22] D.Speen, H.Schulz *On the equivalence of some approaches to computability on the real line*. Domains and Processes, Proc. 1st Intern. Symp. on Domain Theory, Shanghai, China, 1999 (Keimel, K. et al., eds.). Kluwer, Boston, 2001, 67–101.

[23] V.Stoltenberg-Hansen, J. V. Tucker *Effective Algebras*. In Handbook of Logic in computer science, pages 357-526, Oxford University Press, 1995.

[24] K. Weihrauch *Computability*. (Springer, Berlin, 1987).