A Coalgebraic Perspective on Minimization and Determinization*

Jiří Adámek[1], Filippo Bonchi[2], Mathias Hülsbusch[3], Barbara König[3], Stefan Milius[1], Alexandra Silva[4,**]

[1] Technische Universität Braunschweig [2] CNRS - ENS Lyon
[2] Universität Duisburg-Essen [4] Radboud University Nijmegen

Abstract. Coalgebra offers a unified theory of state based systems, including infinite streams, labelled transition systems and deterministic automata. In this paper, we use the coalgebraic view on systems to derive, in a uniform way, abstract procedures for checking behavioural equivalence in coalgebras, which perform (a combination of) minimization and determinization. First, we show that for coalgebras in categories equipped with *factorization structures*, there exists an abstract procedure for equivalence checking. Then, we consider coalgebras in categories without suitable factorization structures: under certain conditions, it is possible to apply the above procedure after transforming coalgebras with *reflections*. This transformation can be thought of as some kind of determinization. We will apply our theory to the following examples: conditional transition systems and (non-deterministic) automata.

1 Introduction

Finite automata are one of the most basic structures in computer science. One particularly interesting problem is that of minimization: given a (non-)deterministic finite automaton is there an equivalent one which has a minimal number of states?

Given a regular language L, minimal deterministic automata (DA) can be thought of as the canonical acceptors of the given language L. A minimal automaton is universal, in the sense that given any automaton which recognizes the same language (and where all states are reachable) there is a unique mapping into the minimal one. Similar notions exist for other kinds of transition systems such as Mealy machines or labelled transition systems. However, in many interesting cases, such as for non-deterministic automata (NDA) or for weighted automata, what it means to be a minimal system is not yet clear. Typically, for NDA one first determinizes the automaton and then minimizes it, since for DA minimization algorithms are well-known ([16]).

It is the main aim of this paper to find a general notion of canonicity for a large class of transition systems, in a uniform manner. This encompasses two things: (i) casting

* The work of Mathias Hülsbusch and Barbara König was partially supported by the DFG project Bevaviour-GT. The work of Alexandra Silva was partially supported by Fundação para a Ciência e a Tecnologia, Portugal, under grant number SFRH/BPD/71956/2010.

** Also affiliated to Centrum Wiskunde & Informatica (Amsterdam, The Netherlands) and HASLab / INESC TEC, Universidade do Minho (Braga, Portugal).

the automata and the intended equivalence in a general framework; and (ii) using the general framework to devise algorithms to minimize (and determinize) the automata, yielding a canonical representative. To study all the types of automata mentioned above (and more) in a uniform setting, we use *coalgebras*.

For a functor $F\colon \mathbf{C} \to \mathbf{C}$, on a category $\mathbf{C}$, an F-coalgebra is a pair (X, α), where X is an object of $\mathbf{C}$ representing the "state space" of the system and $\alpha\colon X \to FX$ is a morphism of $\mathbf{C}$ defining the "transitions" of the states. For instance, given an input alphabet A, DAs are coalgebras for the functor $2 \times (-)^A\colon \mathbf{Set} \to \mathbf{Set}$ and NDAs are coalgebras for the functor $A \times (-) + 1\colon \mathbf{Rel} \to \mathbf{Rel}$, where $\mathbf{Set}$ is the category of sets and functions and $\mathbf{Rel}$ the category of sets and relations.

The strength of the coalgebraic approach lies in the fact that many important notions, such as behavioural equivalence, are uniquely determined by the type of the system. Under mild conditions, functors F have a final coalgebra (unique up to isomorphism) into which every F-coalgebra can be mapped via a unique homomorphism. The final coalgebra can be viewed as the universe of all possible behaviours: the unique homomorphism into the final coalgebra maps every state of a coalgebra to its behaviour. This provides a general notion of behavioural equivalence: two states are equivalent iff they are mapped to the same element of the final coalgebra. In the case of DAs, the final coalgebra is $\mathcal{P}(A^*)$ (the set of all languages over input alphabet A) and the unique homomorphism is a *function* mapping each state to the language that it accepts. In the case of NDAs, as shown in [13], the final coalgebra is A^* (the set of all finite words over A) and the unique homomorphism is a *relation* linking each state with all the words that it accepts. In both cases, the induced behavioural equivalence is language equivalence. The base category chosen to model the system plays an important role in the obtained equivalence. For instance, NDAs can alternatively be modelled as coalgebras for the functor $2 \times \mathcal{P}(-)^A\colon \mathbf{Set} \to \mathbf{Set}$, where $\mathcal{P}$ is the powerset functor, but then the induced behavioural equivalence is bisimilarity (which is finer than language equivalence).

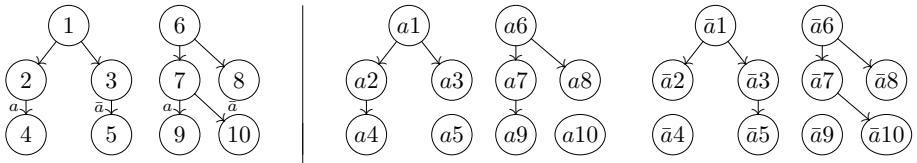
For a functor F on $\mathbf{Set}$, the *image* of an F-coalgebra under the unique morphism is its *minimal representative* (with respect to the induced behavioural equivalence) that, in the finite case, can be computed via ordinary partition refinement algorithms. For functors on categories not equipped with proper image factorization structures (such as $\mathbf{Rel}$, for instance) the situation is less clear-cut. This observation instantiates to the well-known fact that for every DA there exists an equivalent minimal automaton, while for NDAs the uniqueness of minimal automata is not guaranteed.

It is our aim to, on the one hand, offer a procedure to perform ordinary partition refinement for categories with suitable factorization structures (such as $\mathbf{Set}$, wherein DAs are modelled), yielding the *minimization* of a coalgebra. On the other hand, we want to offer an alternative procedure for categories without proper factorization structures: we describe a general setting for *determinizations* and show how to obtain a single algorithm that does determinization and minimization simultaneously. It is worth to note that the latter approach holds for functors for which a final coalgebra does not exist.

Our work was motivated by several examples, considering coalgebras in various underlying categories. In this paper, we take one example in $\mathbf{Set}$ and two examples in $\mathcal{K}\ell(T)$, the Kleisli category for a monad T. More precisely, we consider DAs in $\mathbf{Set}$, NDAs in $\mathbf{Rel}$, which is $\mathcal{K}\ell(\mathcal{P})$ where $\mathcal{P}$ is the powerset monad, and conditional

transition systems in $\mathcal{K}\ell(T)$ where T is the input monad. For DAs, we recover the usual Hopcroft minimization algorithm [16]. Instantiation to NDAs gives us (a part of) Brzozowski's algorithm [7]: the obtained automata coincide with *átomata*, that are a new kind of "canonical" NDAs recently introduced in [8].

Conditional Transition Systems (CTS). To better illustrate our work, we employ transition systems labelled with *conditions* that have similarly been studied in [14,10]. Consider the transition system (1) below where transitions are decorated with conditions $a, \bar{a}$, where intuitively $\bar{a}$ stands for "not a". Labelled transitions are either present or absent, depending on whether a or $\bar{a}$ hold. Unlabelled transitions are always present (they can be thought of as two transitions labelled a and $\bar{a}$).



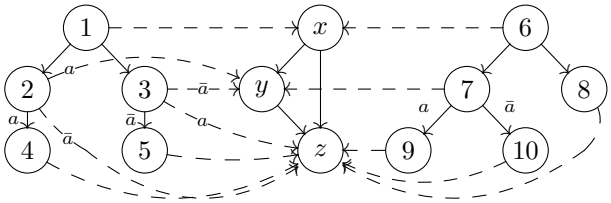
(1) (2)

The environment can make one choice, which can not be changed later: it decides whether to take either a or $\bar{a}$. Regardless of the specific choice of the environment, the two states 1 and 6 in (1) above will be bisimilar. If a holds then the systems above would be instantiated to the left half of transition system (2) above. Instead if a does not hold then we obtain the right half. In both cases, the instances of the states 1 and 6 are bisimilar.

This shows that one possible way to solve the question whether two states are *always* bisimilar consists in enumerating all conditions and to create suitably many instantiations of the transition system. Then the resulting transition system can be minimized with respect to bisimilarity. This is analogous to the steps of determinization and minimization for NDAs. Indeed, the base category of coalgebras of CTSs, as $\mathbf{Rel}$ for NDAs, has no suitable factorization structures. In order to minimize (both NDAs and CTSs), coalgebras should be transformed via *reflections* that, in the case of NDAs means determinizing, while for CTS, means instantiating CTSs for all the conditions.

In this work, we will study both constructions in a general setting and also show how they can be combined into a single algorithm. For CTSs this mean that we will provide an algorithm that checks if two states are bisimilar under all the possible conditions, without performing all the possible instantiations.

Now, what would be a canonical representative of the systems above? In other words, is there a system into which CTS (1) can be mapped? In the example above, it is relatively easy to see that that system would be the transition system consisting of states x, y, z in (3) below. One would map both 1 and 6 to x, 7 to y, $4, 5, 9$ and 10 to z. What about 2 and 3? We want to map 2 to y whenever a holds and to z whenever $\bar{a}$ holds, dually for 3. In order to do that we need to work in a category where we can represent such *conditional* maps. As we will show in the sequel, by modelling CTS as coalgebras in a Kleisli category this will be possible. The full mapping is represented below.



(3)

An extended version of this paper [2] contains all the proofs, further details and examples. In particular, we instantiate our theory to the case of linear weighted automata [6].

2 Background material on coalgebras

We assume some prior knowledge of category theory (categories, functors, monads, limits and adjunctions). Definitions can be found in [3]. However, to establish some notation, we recall some basic definitions. We denote by Ord the class of all ordinals. Let $\mathbf{Set}$ be the category of sets and functions. Sets (and other objects) are denoted by capital letters $X, Y, \ldots$ and functions (and other morphisms) by lower case $f, g, \ldots, \alpha, \beta, \ldots$ We write $\emptyset$ for the empty set, 1 for the singleton set, typically written as $1 = \{\bullet\}$, and 2 for the two elements set $2 = \{0, 1\}$. The collection of all subsets of a set X is denoted by $\mathcal{P}(X)$ and the collection of functions from a set X to a set Y is denoted by Y^X. We write $g \circ f$ for function composition, when defined. The product of two sets X, Y is written as $X \times Y$, while the coproduct, or disjoint union, as $X + Y$. These operations, defined on sets, can analogously be defined on functions, yielding (bi-)functors. A category $\mathbf{C}$ is called *concrete* if a faithful functor $U\colon \mathbf{C} \to \mathbf{Set}$ is given.

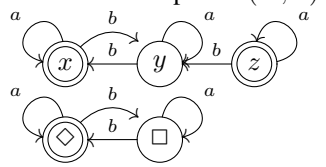
Definition 2.1 (Coalgebra). *Given an endofunctor $F\colon \mathbf{C} \to \mathbf{C}$ an* (F-)coalgebra *is a pair (X, α), where X is an object of $\mathbf{C}$ and $\alpha\colon X \to FX$ a morphism in $\mathbf{C}$. A* (coalgebra) homomorphism *$f\colon (X, \alpha) \to (Y, \beta)$ between two coalgebras $\alpha\colon X \to FX$ and $\beta\colon Y \to FY$ is a $\mathbf{C}$-morphism $f\colon X \to Y$ such that $Ff \circ \alpha = \beta \circ f$.*

An F-coalgebra (Ω, ω) is *final* if for any F-coalgebra (X, α) there exists a unique homomorphism $beh_X\colon (X, \alpha) \to (\Omega, \omega)$. If $\mathbf{C}$ is concrete we can define behavioural equivalence. Given an F-coalgebra (X, α) and $x, y \in UX$, we say that x and y are *behaviourally equivalent*, written $x \approx y$, if and only if there exist an F-coalgebra (Z, γ) and a homomorphism $f\colon (X, \alpha) \to (Z, \gamma)$ such that $Uf(x) = Uf(y)$. If a final F-coalgebra exists, we have a simpler characterization of behavioural equivalence: $x \approx y$ iff $Ubeh_X(x) = Ubeh_X(y)$.

Example 2.2. (DA) A deterministic automaton over the alphabet A is a pair (X, α), where X is a set of states and $\alpha\colon X \to 2 \times X^A$ is a function that to each state x associates a pair $\alpha(x) = \langle o_x, t_x \rangle$, where o_x, the output value, determines if a state x is final ($o_x = 1$) or not ($o_x = 0$); and t_x, the transition function, returns for each $a \in A$ the next state. DAs are coalgebras for the functor $FX = 2 \times X^A$ on $\mathbf{Set}$. The final coalgebra for this functor is $(\mathcal{P}(A^*), \omega)$ where $\mathcal{P}(A^*)$ is the set of languages over A and, for a language L, $\omega(L) = \langle \varepsilon_L, L_a \rangle$, where ε_L determines whether or not the

empty word is in the language ($\varepsilon_L = 1$ or $\varepsilon_L = 0$, resp.) and, for each input letter a, L_a is the *derivative* of L: $L_a = \{w \in A^* \mid aw \in L\}$. From any DA (X, α), there is a unique homomorphism beh_X into $\mathcal{P}(A^*)$ which assigns to each state its behaviour (that is, the language that the state recognizes). Two states are behaviourally equivalent iff they accept the same language.

Take $A = \{a, b\}$ and consider the DAs on the right. We call the topmost (X, α) where $X = \{x, y, z\}$ and $\alpha\colon X \to 2 \times X^A$ maps x to the pair $\langle 1, \{a \mapsto x,\ b \mapsto y\}\rangle$, y to $\langle 0, \{a \mapsto y,\ b \mapsto x\}\rangle$ and z to $\langle 1, \{a \mapsto z,\ b \mapsto y\}\rangle$. The bottom one is (Z, γ) where $Z = \{\diamond, \square\}$ and $\gamma\colon Z \to 2 \times Z^A$ maps $\diamond$ to $\langle 1, \{a \mapsto \diamond,\ b \mapsto \square\}\rangle$ and $\square$ to $\langle 0, \{a \mapsto \square,\ b \mapsto \diamond\}\rangle$.



As an example of a coalgebra homomorphism, take the function $e\colon X \to Z$ mapping x, z to $\diamond$ and y to $\square$.

Non-deterministic automata (NDA) can be described as coalgebras for the functor $2 \times \mathcal{P}(-)^A$ (on $\mathbf{Set}$): to each input in A, we assign a set of possible successors states. Unfortunately, the resulting behavioural equivalence is not language equivalence (as for DAs), but bisimilarity (i.e., it only identifies states having the same branching structure). In [21,13], it is shown that in order to retrieve language equivalence for NDAs, one should consider coalgebras in a Kleisli category. In what follows, we introduce Kleisli categories, in which we model NDAs and CTSs as coalgebras. While objects in a Kleisli category are sets, morphisms are generalized functions that incorporate side effects, such as non-determinism, specified by a monad (see [3,13,18]).

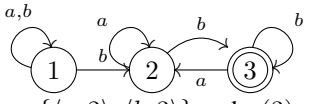
Definition 2.3 (Kleisli Category). *Let $(T\colon \mathbf{Set} \to \mathbf{Set}, \eta, \mu)$ (or simply T) be a monad on $\mathbf{Set}$. Its* Kleisli category $\mathcal{K}\ell(T)$ *has sets as objects and a morphism $X \to Y$ in $\mathcal{K}\ell(T)$ is a function $X \to TY$. The identity id_X is η_X and the composition $g \circ f$ of $f\colon X \to Y$, $g\colon Y \to Z$ (i.e., functions $f\colon X \to TY$, $g\colon Y \to TZ$) is $\mu_Z \circ Tg \circ f$.*

In the following we will employ overloading and use the same letter to both denote a morphism in $\mathcal{K}\ell(T)$ and the corresponding function in $\mathbf{Set}$. Furthermore, note that $\mathbf{Set}$ can be seen as a (non-full) subcategory of $\mathcal{K}\ell(T)$, where each function $f\colon X \to Y$ is identified with $\eta_Y \circ f$. Every Kleisli category $\mathcal{K}\ell(T)$ is a concrete category where $UX = TX$ and $Uf = \mu_Y \circ Tf$ for an object X and a morphism $f\colon X \to Y$.

To define coalgebras over Kleisli categories we need the notion of lifting of a functor, which we define here directly, but could otherwise be specified via a distributive law (for details see [13,19]): a functor $\overline{F}\colon \mathcal{K}\ell(T) \to \mathcal{K}\ell(T)$ is called a *lifting of* $F\colon \mathbf{Set} \to \mathbf{Set}$ whenever it coincides with F on $\mathbf{Set}$, seen as a subcategory of $\mathcal{K}\ell(T)$.

Since F and $\overline{F}$ coincide on objects, $\overline{F}$-coalgebras in $\mathcal{K}\ell(T)$ are of the form $X \to TFX$, where intuitively the functor F describes the explicit branching, i.e. choices which are visible to the observer, and the monad T the implicit branching, i.e. side-effects, which are there but cannot be observed directly. In this way, the implicit branching is part of the underlying category and is also present in the morphism from any coalgebra into the final coalgebra. As in functional programming languages such as Haskell, the idea is to "hide" computational effects underneath a monad and to separate them from the (functional) behaviour as much as possible.

Example 2.4. (NDA) Consider the powerset monad $TX = \mathcal{P}(X)$. The Kleisli category $\mathcal{K}\ell(\mathcal{P})$ coincides with the category $\mathbf{Rel}$ of sets and relations. As an example of a lifting, take $FX = A \times X + 1$ in $\mathbf{Set}$ (with $1 = \{\bullet\}$). The functor F lifts to $\overline{F}$ in $\mathbf{Rel}$ as follows: for any $f\colon X \to Y$ in $\mathbf{Rel}$ (that is $f\colon X \to \mathcal{P}(Y)$ in $\mathbf{Set}$), $\overline{F}f\colon A \times X + 1 \to A \times Y + 1$ is defined as $\overline{F}f(\bullet) = \{\bullet\}$ and $\overline{F}f(\langle a, x\rangle) = \{\langle a, y\rangle \mid y \in f(x)\}$. Non-deterministic automata over the input alphabet A can be regarded as coalgebras in $\mathbf{Rel}$ for the functor $\overline{F}$. A coalgebra $\alpha\colon X \to \overline{F}X$ is a function $\alpha\colon X \to \mathcal{P}(A \times X + 1)$, which assigns to each state $x \in X$ a set which contains $\bullet$ if x is final and $\langle a, y\rangle$ for all transitions $x \xrightarrow{a} y$. For instance, the automaton on the right is the coalgebra (X, α), where $X = \{1, 2, 3\}$ and $\alpha\colon X \to \mathcal{P}(\{a, b\} \times X + \{\bullet\})$ is defined as follows: $\alpha(1) = \{\langle a, 1\rangle, \langle b, 1\rangle, \langle b, 2\rangle\}$, $\alpha(2) = \{\langle a, 2\rangle, \langle b, 3\rangle\}$ and $\alpha(3) = \{\bullet, \langle a, 2\rangle, \langle b, 3\rangle\}$. In [13], it is shown that the final $\overline{F}$-coalgebra (in $\mathbf{Rel}$) is the set A^* of words. For an NDA (X, α), the unique coalgebra homomorphism beh_X into A^* is the relation that links every state in X with all the words in A^* that it accepts.



Example 2.5. (CTS) We shortly discuss how to specify the example from the introduction in a Kleisli category. All the details can be found in [2] .

We use the input monad $TX = X^A$, where A is a set of conditions or inputs (for the example of the introduction $A = \{a, \bar{a}\}$). Given a function $f\colon X \to Y$, $Tf\colon TX \to TY$ is $f^A\colon X^A \to Y^A$ defined for all $g \in X^A$ and $a \in A$ as $f^A(g)(a) = f(g(a))$.

Note that a morphism $f\colon X \to Y$ in the Kleisli category over the input monad is a function $f\colon X \to Y^A$. For instance, the dashed arrows in the introduction describe a morphism in $\mathcal{K}\ell(T)$: state 2 is mapped to y if condition a holds and to z if $\bar{a}$ holds.

We will use the countable powerset functor $FX = \mathcal{P}_c(X)$ as endofunctor, which is lifted to $\mathcal{K}\ell(T)$ as follows: a morphism $f\colon X \to Y$ in $\mathcal{K}\ell(T)$, which is a function of the form $f\colon X \to Y^A$, is mapped to $\overline{F}f\colon \mathcal{P}_c(X) \to \mathcal{P}_c(Y)$ with $\overline{F}f(X')(a) = \{f(x)(a) \mid x \in X'\}$ for $X' \subseteq X$, $a \in A$. Hence, CTS (1) from the introduction is modelled by a morphism $\alpha\colon X \to \mathcal{P}_c(X)$ in $\mathcal{K}\ell(T)$ (i.e., a function $\alpha\colon X \to \mathcal{P}_c(X)^A$), where $X = \{1, \ldots, 10\}$ and $A = \{a, \bar{a}\}$. For instance $\alpha(1)(a) = \alpha(1)(\bar{a}) = \{2, 3\}$, $\alpha(2)(a) = \{4\}$, $\alpha(2)(\bar{a}) = \emptyset$. The entire coalgebra α is represented by the matrix on the right.

α	1	2	3	4	5	6	7	8	9	10
a	$\{2,3\}$	$\{4\}$	$\emptyset$	$\emptyset$	$\emptyset$	$\{7,8\}$	$\{9\}$	$\emptyset$	$\emptyset$	$\emptyset$
$\bar{a}$	$\{2,3\}$	$\emptyset$	$\{5\}$	$\emptyset$	$\emptyset$	$\{7,8\}$	$\{10\}$	$\emptyset$	$\emptyset$	$\emptyset$

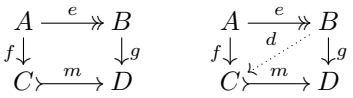
Note that the above $\alpha\colon X \to \mathcal{P}_c(X)^A$ can be seen as a coalgebra for the functor $FX = \mathcal{P}_c(X)^A$ in $\mathbf{Set}$, which yields ordinary A-labelled transition systems. However, the resulting behavioural equivalence (that is, ordinary bisimilarity) would be inadequate for our intuition, since it would distinguish the states 1 and 6. In [2], we prove that behavioural equivalence of $\overline{F}$-coalgebras coincides with the expected one.

3 Minimization via $(\mathcal{E}, \mathcal{M})$-Factorizations

We now introduce the notion of minimization of a coalgebra and its iterative construction that generalizes the minimization of transition systems via partition refinement. This notion is parametrized by two classes $\mathcal{E}$ and $\mathcal{M}$ of morphisms that form a factorization structure for the considered category $\mathbf{C}$.

Definition 3.1 (Factorization Structures). *Let* $\mathbf{C}$ *be a category and let* $\mathcal{E}$, $\mathcal{M}$ *be classes of morphisms in* $\mathbf{C}$. *The pair* $(\mathcal{E}, \mathcal{M})$ *is called a* factorization structure *for* $\mathbf{C}$ *whenever*

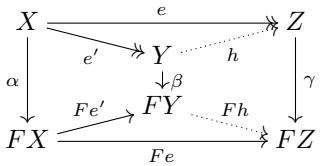
- $\mathcal{E}$ *and* $\mathcal{M}$ *are closed under composition with isos.*
- $\mathbf{C}$ *has* $(\mathcal{E}, \mathcal{M})$-factorizations *of morphisms, i.e., each morphism* f *of* $\mathbf{C}$ *has a factorization* $f = m \circ e$ *with* $e \in \mathcal{E}$ *and* $m \in \mathcal{M}$.
- $\mathbf{C}$ *has the unique* $(\mathcal{E}, \mathcal{M})$-diagonalization property*: for each commutative square as shown on the left-hand side with* $e \in \mathcal{E}$ *and* $m \in \mathcal{M}$ *there exists a unique* diagonal*, i.e., a morphism* d *such that the diagram on the right-hand side commutes (i.e.,* $d \circ e = f$ *and* $m \circ d = g$*). If all morphisms in* $\mathcal{E}$ *are epis we call* $(\mathcal{E}, \mathcal{M})$ *a* right factorization structure*.*



In any category with an $(\mathcal{E}, \mathcal{M})$-factorization structure, the classes $\mathcal{E}, \mathcal{M}$ are closed under composition and factorizations of morphisms are unique up to iso (see [3]). For $\mathbf{Set}$ we always consider below the factorization structure $(\mathcal{E}, \mathcal{M})$ with $\mathcal{E}$ = epimorphims (surjections) and $\mathcal{M}$ = monomorphisms (injections); for the category $\mathbf{Set}^{\mathbf{op}}$ we take the corresponding structure $(\mathcal{M}, \mathcal{E})$, i.e., where the epic part consists of functions that in $\mathbf{Set}$ are monomorphisms, analogously with $\mathcal{E}$. Morphisms from $\mathcal{E}$ are drawn using double-headed arrows $A \twoheadrightarrow B$, whereas morphisms from $\mathcal{M}$ are depicted using arrows of the form $A \rightarrowtail B$. Whenever the endofunctor F preserves $\mathcal{M}$-morphisms, which we assume in the following, the factorization structure can be straightforwardly lifted to coalgebra homomorphisms (see [17]).

Assumption 3.2 *We assume that* $\mathbf{C}$ *is a complete category with a right* $(\mathcal{E}, \mathcal{M})$*-factorization structure and* $\mathbf{C}$ *is* $\mathcal{E}$*-cowellpowered, i. e., every object* X *only has a set of* $\mathcal{E}$*-quotients (i.e.,* $\mathcal{E}$*-morphisms with domain* X *up to isomorphism of the codomains). We also assume that* $F\colon \mathbf{C} \to \mathbf{C}$ *is a functor preserving* $\mathcal{M}$*, i. e., if* $m \in \mathcal{M}$ *then* $Fm \in \mathcal{M}$.

Definition 3.3 (Minimization). *The* minimization *of a coalgebra* $\alpha\colon X \to FX$ *is the greatest* $\mathcal{E}$*-quotient coalgebra. More precisely, the minimization is a coalgebra* (Z, γ) *with a homomorphism* $e\colon (X, \alpha) \twoheadrightarrow (Z, \gamma)$ *with* $e \in \mathcal{E}$ *such that for any other coalgebra homomorphism* $e'\colon (X, \alpha) \twoheadrightarrow (Y, \beta)$ *with* $e' \in \mathcal{E}$ *there exists a (necessarily) unique coalgebra homomorphism* $h\colon (Y, \beta) \to (Z, \gamma)$ *such that* $e = h \circ e'$.

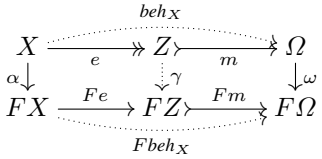


Remark 3.4. (1) Since $\mathbf{C}$ is $\mathcal{E}$-cowellpowered and $\mathcal{E}$ consists of epimorphisms, the $\mathcal{E}$-quotient coalgebras of a coalgebra (X, α) form a pre-ordered set: a quotient coalgebra $e'\colon (X, \alpha) \twoheadrightarrow (Y', \beta')$ is larger than $e\colon (X, \alpha) \twoheadrightarrow (Y, \beta)$ iff there exists a coalgebra homomorphism $h\colon (Y, \beta) \to (Y', \beta')$ with $e' = h \circ e$; notice that h is uniquely determined and $h \in \mathcal{E}$ by the properties of factorization systems. Thus, the minimization is simply the greatest element in the pre-order of $\mathcal{E}$-quotient coalgebras of (X, α).
(2) While in $\mathbf{Set}$ the minimization is also determined by the strict minimality of the number of states, this is not necessarily true for other categories (see Example 4.10).

(3) We often speak about (Z, γ) (without explicitly referring to the morphism e) or even just the object Z as the minimization of the given coalgebra.

Theorem 3.8 will show that under Assumption 3.2 the minimization always exists, even when there is no final coalgebra. When the final coalgebra exists, minimization is the quotient of the unique morphism.

Proposition 3.5 (Minimization and Final Coalgebra). *If the final coalgebra $\omega\colon \Omega \to F\Omega$ exists, then – for a given coalgebra $\alpha\colon X \to FX$ – the minimization $\gamma\colon Z \to FZ$ can be obtained by factoring the unique coalgebra homomorphism $beh_X\colon (X,\alpha) \to (\Omega,\omega)$ into an $\mathcal{E}$-morphism and an $\mathcal{M}$-morphism.*



Note that whenever the concretization functor $U\colon \mathbf{C} \to \mathbf{Set}$ maps $\mathcal{M}$-morphisms to injections, $x, y \in UX$ are behaviourally equivalent ($x \approx y$) iff $Ue(x) = Ue(y)$.

Example 3.6 (DA, Minimal Automata). Recall that DAs are coalgebras for the functor $FX = 2 \times X^A$ on $\mathbf{Set}$ (Example 2.2). In this case, minimization corresponds to the well known minimization of deterministic automata. For instance, the minimization of the top automaton (X, α) in Example 2.2 yields the automaton (Z, γ) (on the bottom).

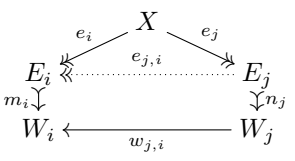
We now describe a construction that – given a coalgebra (X, α) – obtains the minimization γ without going via the final coalgebra. This closely resembles the partition refinement algorithm for minimizing deterministic automata or for computing bisimilarity. Whenever the construction below becomes stationary, we obtain the minimization. In many examples the constructed sequence might even become stationary after finitely many steps. The construction is reminiscent of the construction (in the dual setting) of the initial algebra by Adámek [1], for the coalgebraic version see Worrel [24] and Adámek and Koubek [4]. As in those papers, our construction works for ordinals beyond ω. Hereafter 1 denotes the final object of $\mathbf{C}$.

Construction 3.7 *Recall the final chain $W\colon \mathsf{Ord} \to \mathbf{C}$ given by*

$$W_0 = 1, \qquad W_{i+1} = FW_i, \qquad W_j = \lim_{i<j} W_j \quad (j \text{ a limit ordinal.})$$

This is the unique chain, up to natural isomorphism, whose connecting morphisms $w_{i,j}$ fulfil (a) $w_{i+1,j+1} = Fw_{i,j}$ and (b) for limit ordinals they form a limit cone.

As we do not assume that F has a final coalgebra, the chain W need not converge. Every coalgebra $\alpha\colon X \to FX$ defines a unique canonical cone $(\alpha_i\colon X \to W_i)_{i\in\mathsf{Ord}}$ *on W with the property that $\alpha_{i+1} = F\alpha_i \circ \alpha\colon X \to FW_i = W_{i+1}$. Let $e_i\colon X \twoheadrightarrow E_i$, $m_i\colon E_i \rightarrowtail W_i$ be an $(\mathcal{E},\mathcal{M})$-factorization of α_i. Then, we obtain an ordinal indexed chain (E_i) of quotients of X with the connecting morphisms $e_{j,i}$ obtained by diagonalization for $i < j$, as depicted on the right.*



Theorem 3.8. *For every F-coalgebra (X, α), its minimization is E_i, for some $i \in \mathsf{Ord}$.*

More precisely, there exists an ordinal i such that E_i carries a coalgebra structure $\varepsilon\colon E_i \to FE_i$ such that $e_i\colon (X, \alpha) \to (E_i, \varepsilon)$ is the minimization; for details see the proof of Theorem 3.8 in the extended version of this paper [2].

By the above theorem, minimizations always exist even when there is no final coalgebra. Worrell [24] shows that for a finitary functor $F\colon \mathbf{Set} \to \mathbf{Set}$, the final chain W_i converges at the final coalgebra in $\omega + \omega$ iterations. The chain E_i, instead, converges at the minimization in ω iterations.

Theorem 3.9. *Let $F\colon \mathbf{Set} \to \mathbf{Set}$ be a finitary functor. Then for every F-coalgebra (X, α), its minimization is E_ω.*

In our examples, we will use the following construction which is closer to the standard minimzation algorithm and to any reasonable implementation of Construction 3.7.

Theorem 3.10. *The chain $(E_i)_{i\in\mathsf{Ord}}$ of Construction 3.7 can also be defined as follows:*
(a) Factor the unique morphism $d_0\colon X \to 1$ into $e_0\colon X \twoheadrightarrow E_0$ and $n_0\colon E_0 \rightarrowtail 1$.
(b) Given $e_i\colon X \twoheadrightarrow E_i$, factor $d_{i+1} = Fe_i \circ \alpha$ into $e_{i+1}\colon X \twoheadrightarrow E_{i+1}$ and $n_{i+1}\colon E_{i+1} \rightarrowtail FE_i$.
(c) For a limit ordinal j, form a limit of the preceding chain $(E_i)_{i<j}$, obtaining $\hat{E}_j$ and $\hat{e}_j\colon X \to \hat{E}_j$ as mediating morphism. Factor $\hat{e}_j$ into $e_j\colon X \twoheadrightarrow E_j$ and $n_j\colon E_j \rightarrowtail \hat{E}_j$.

By instantiating the above construction to the case of DAs, we obtain the standard minimization algorithm by Hopcroft [16].

4 Determinization via Reflections

For several categories there are no suitable factorization structures. This can for instance be observed in $\mathbf{Rel}$, wherein we model non-deterministic automata as coalgebras. It is known that minimization of non-deterministic automata is not unique. The usual procedure is to first construct the corresponding deterministic automaton (via the powerset construction), which is then minimized in a second step. In this section, we will give a general framework for determinization-like constructions in the form of reflections, which can also be applied to other settings, such as conditional transition systems. For non-deterministic automata we will obtain an automaton which is "backward-deterministic", i.e., for every state and each letter there is exactly one predecessor. Then we will show how reflections can be combined with the minimization.

Definition 4.1 (Reflective Subcategory). *Let $\mathbf{S}$ be a subcategory of $\mathbf{C}$. Let X be an object of $\mathbf{C}$. An* $\mathbf{S}$-reflection *for X is a morphism $\eta_X\colon X \to X'$, where X' is an $\mathbf{S}$-object, such that for every other morphism $f\colon X \to Y$ with Y in $\mathbf{S}$ there exists a unique $\mathbf{S}$-morphism $f'\colon X' \to Y$ such that $f = f' \circ \eta_X$. $\mathbf{S}$ is called a* reflective subcategory *of $\mathbf{C}$ whenever each $\mathbf{C}$-object has an $\mathbf{S}$-reflection.*

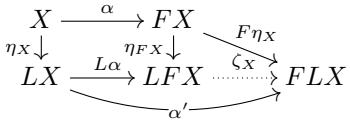
This definition is equivalent to saying that the functor embedding $\mathbf{S}$ into $\mathbf{C}$ has a left adjoint $L\colon \mathbf{C} \to \mathbf{S}$ called *reflector*. The morphisms η_X form the unit of this adjunction.

In our examples in $\mathcal{K}\ell(T)$, the unit η of the reflection will *not* coincide with the natural transformation η of the monad T. It is well-known that for a monad $T\colon \mathbf{Set} \to \mathbf{Set}$ the category $\mathbf{Set}$ is coreflective in $\mathcal{K}\ell(T)$, whereas here we need a reflective subcategory.

Example 4.2. (NDA) The category $\mathbf{Set}^{\mathbf{op}}$ is a reflective subcategory of $\mathbf{Rel}$. The reflector L is the contravariant powerset functor, i.e., for a relation $R\colon X \to Y$ we have $L(R)\colon \mathcal{P}(X) \to \mathcal{P}(Y)$ in $\mathbf{Set}^{\mathbf{op}}$ where $L(R)$ maps $Y' \subseteq Y$ to $R^{-1}(Y')$. The reflection $\eta_X : X \to \mathcal{P}(X)$ relates an element $x \in X$ with $X' \subseteq X$ if and only if $x \in X'$.

(CTS) For $\mathcal{K}\ell(T)$ where T is the input monad, we have the following situation: since every function $f\colon X \to Y^A$ corresponds to a function $f'\colon A \times X \to Y$ by currying, the category $\mathcal{K}\ell(T)$ is isomorphic to the co-Kleisli category over the comonad $VX = A \times X$ on $\mathbf{Set}$. Hence, $\mathbf{Set}$ is both reflective and coreflective in $\mathcal{K}\ell(T)$. The reflection is the Kleisli morphism $\eta_X\colon X \to A \times X$ with $\eta_X(x)(a) = \langle a, x\rangle$. The reflector L coincides with V on objects and takes the product of the state set X with the label set A. More concretely, for a morphism $f\colon X \to Y$ in $\mathcal{K}\ell(T)$ we obtain a morphism $Lf\colon A \times X \to A \times Y$ in $\mathbf{Set}$ with $Lf(\langle a, x\rangle) = \langle a, f(x)(a)\rangle$.

Definition 4.3 (Reflection of Coalgebras). *Let $\mathbf{S}$ be a reflective subcategory of a category $\mathbf{C}$ and let $L\colon \mathbf{C} \to \mathbf{S}$ be the reflector. Assume that $\mathbf{S}$ is preserved by the endofunctor F. Then, given a coalgebra $\alpha\colon X \to FX$ in $\mathbf{C}$ we reflect it into $\mathbf{S}$, obtaining a coalgebra $\alpha'\colon LX \to FLX$ by the following construction:*



Note that the existence of a unique morphism ζ_X is guaranteed by Definition 4.3, since F preserves $\mathbf{S}$ and hence FLX is an object of $\mathbf{S}$.

That the above construction indeed gives a reflection of coalgebras for F is a special instance of a known result (see for instance Hermida and Jacobs [15], Corollary 2.15).

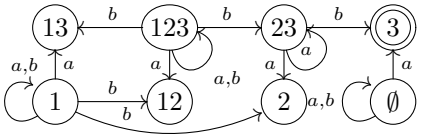
Whenever $\mathbf{C}$ is a concrete category (with concretization functor U) and $x, y \in UX$ it holds that $x \approx y \iff U\eta_X(x) \approx U\eta_Y(y)$. Two states in UX are behaviourally equivalent if and only if this holds for their images in the reflected coalgebra.

Proposition 4.4. *Let $\mathbf{S}$ be a reflective subcategory of $\mathbf{C}$, which is preserved by the endofunctor F. The category of F-coalgebras in $\mathbf{S}$ is a reflective subcategory of the category of F-coalgebras in $\mathbf{C}$.*

A limit in a reflective subcategory $\mathbf{S}$ is also a limit in $\mathbf{C}$. Hence, if the final coalgebra exists in the subcategory $\mathbf{S}$, it is also the final coalgebra in $\mathbf{C}$. In particular, whenever $\mathbf{S}$ is complete, the chain (W_i) (Construction 3.7) in $\mathbf{S}$ will coincide with the chain in $\mathbf{C}$.

Example 4.5. (NDA) We will first study the effect of a reflection on a non-deterministic automaton, for which we use the reflective subcategory $\mathbf{Set}^{\mathbf{op}}$ of $\mathbf{Rel}$ (see Example 4.2). The effect of the reflection on coalgebras is a powerset automaton which is however "backwards-deterministic": more specifically, given a coalgebra $\alpha\colon X \to A \times X + 1$ in $\mathbf{Rel}$, the reflected coalgebra $\alpha'\colon \mathcal{P}(X) \to A \times \mathcal{P}(X) + 1$ is a relation

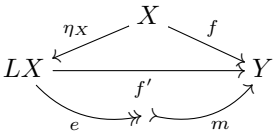
which lives in $\mathbf{Set}^{\mathbf{op}}$ and, when seen as a function, maps $\langle a, X' \rangle$ with $X' \subseteq X$ to $\{x \in X \mid \exists x' \in X' \colon \langle a, x' \rangle \in \alpha(x)\}$ (the set of a-predecessors of X') and $\bullet$ to $\{x \in X \mid \bullet \in \alpha(x)\}$ (the set of final states, the unique final state of the new automaton). For instance, the reflection of the NDA (X, α) in Example 2.4 is the above backwards-deterministic automaton. Note that the above automaton has a single final state (consisting of the set of final states of the original automaton) and every state has a unique predecessor for each alphabet letter. Hence, it can be seen as a function $\alpha' \colon A \times Y + 1 \to Y$ (i.e., an algebra for the functor $FY = A \times Y + 1$). Note that $\mathbf{Set}$ is *not* a reflective subcategory of $\mathbf{Rel}$ – it is instead coreflective – and hence both categories have different final coalgebras. However for the reflective subcategory $\mathbf{Set}^{\mathbf{op}}$, we have exactly the same final coalgebra as for $\mathbf{Rel}$, which, as shown in [13], is the *initial algebra* in $\mathbf{Set}$.



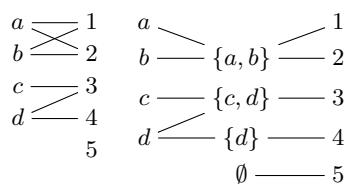
(CTS) Now we come back to the Kleisli category $\mathcal{K}\ell(T)$ over the input monad T (see Example 2.5) and coalgebras with endofunctor $\mathcal{P}_c$. As discussed in Example 4.2, $\mathbf{Set}$ is a reflective subcategory of $\mathcal{K}\ell(T)$. On coalgebras reflection has the following effect: given a coalgebra $\alpha \colon X \to \mathcal{P}_c(X)$ in $\mathcal{K}\ell(T)$ we obtain a reflected coalgebra $\alpha' \colon A \times X \to \mathcal{P}_c(A \times X)$ in $\mathbf{Set}$ with $\alpha'(\langle a, x \rangle) = \{\langle a, x' \rangle \mid x' \in \alpha(x)(a))\}$. That is, we generate the disjoint union of $|A|$ different transition systems, each of which describes the behaviour for some $a \in A$. For instance, the reflection of CTS (1) (formally introduced in Example 2.5, see also the introduction) is CTS (2) from the introduction.

We now consider other forms of factorizations that do not conform to Definition 3.1.

Definition 4.6 (Pseudo-Factorization). *Let $\mathbf{C}$ be a category and let $\mathbf{S}$ be a reflective subcategory with a factorization structure $(\mathcal{E}, \mathcal{M})$. Let $f \colon X \to Y$ be a morphism of $\mathbf{C}$ where Y is an object of $\mathbf{S}$. Take the unique morphism $f' \colon LX \to Y$ with $f' \circ \eta_X = f$ (which exists due to the reflection) and factor $f' = m \circ e$ with $m \in \mathcal{M}$, $e \in \mathcal{E}$. Then the decomposition $f = m \circ c$ with $c = e \circ \eta_X$ is called the* $(\mathcal{E}, \mathcal{M})$-pseudo-factorization *of f.*



Example 4.7. (NDA) Consider $\mathbf{Set}^{\mathbf{op}}$ as the reflective subcategory of $\mathbf{Rel}$ (Example 4.2). Given a relation $R \colon X \to Y$, let $\mathcal{Z} = \{R^{-1}(y) \mid y \in Y\} \subseteq \mathcal{P}(X)$ be the set of pre-images of elements of Y under R. Now define relations $R_c \colon X \to \mathcal{Z}$ with $R_c(x) = \{Z \in \mathcal{Z} \mid x \in Z\}$ and $R_m \colon \mathcal{Z} \to Y$ with $R_m(Z) = \{y \in Y \mid Z = R^{-1}(y)\}$. Note that $R_m \circ R_c = R$. As an example consider the relation R between sets $X = \{a, b, c, d\}$ and $Y = \{1, 2, 3, 4, 5\}$ visualized on the left (where $R(a) = R(b) = \{1, 2\}$, $R(c) = \{3\}$, $R(d) = \{3, 4\}$). Its pseudo-factorization into R_c and R_m is shown on the right. Here R_m maps elements of Y to their preimage under R in $\mathcal{P}(X)$.



(CTS) For $\mathbf{Set}$, the reflective subcategory of $\mathcal{K}\ell(T)$, where T is the input monad, we use the classical factorization structure with surjective and injective functions. Given a morphism $f \colon X \to Y$ in $\mathcal{K}\ell(T)$, seen as a function $f \colon X \to Y^A$, we define $Y' = \{y \in Y \mid \exists x \in X, a \in A \colon f(x)(a) = y\}$. Then $f_c \colon X \to Y'^A$ with $f_c(x)(a) = f(x)(a)$

and $f_m \colon Y' \to Y^A$ with $f_m(y)(a) = y$ for all $a \in A$, i.e., f_m is simply an injection without side-effects. Note that $f_m \circ f_c = f$ in $\mathcal{K}\ell(T)$.

Note that pseudo-factorizations enjoy the diagonalization property as in Definition 3.1 whenever g is a morphism of $\mathbf{S}$. However pseudo-factors are not necessarily closed under composition with the isos of $\mathbf{C}$.

Assumption 4.8 *We assume that* $\mathbf{S}$ *is a reflective subcategory of* $\mathbf{C}$*. We also assume that an endofunctor* F *of* $\mathbf{C}$ *is given preserving* $\mathbf{S}$*. And* $\mathbf{S}$ *and* F *fulfil Assumption 3.2.*

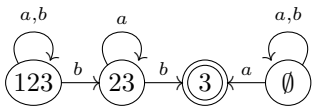
Theorem 4.9. *Given a coalgebra* $\alpha \colon X \to FX$ *in* $\mathbf{C}$*, the following four constructions obtain the same result (we also call this result the* minimization*):*

(i) Apply Construction 3.7 using the $(\mathcal{E}, \mathcal{M})$*-pseudo-factorizations of Definition 4.6.*
(ii) Reflect α *into the subcategory* $\mathbf{S}$ *according to Definition 4.3 and then apply Construction 3.7 using* $(\mathcal{E}, \mathcal{M})$*-factorizations.*
(iii) Apply the construction of Theorem 3.10 using $(\mathcal{E}, \mathcal{M})$*-pseudo-factorizations.*
(iv) Reflect α *into the subcategory* $\mathbf{S}$ *and then apply the construction of Theorem 3.10 using* $(\mathcal{E}, \mathcal{M})$*-factorizations.*

Note that we do not have to require here that $\mathbf{C}$ is complete. As it is clear in the proof of Theorem 4.9 (see the extended version of this paper [2]) Construction 3.7 and the construction in Theorem 3.10 can be straightforwardly adapted to pseudo-factorizations instead of factorizations: The quotients E_i and the chain $e_{j,i}$ of connecting morphisms obtained in variants (i)–(iv) are identical and live in the subcategory $\mathbf{S}$. Since $\mathbf{S}$ is reflective in $\mathbf{C}$ we obtain the same results when taking the limit in $\mathbf{C}$ or in $\mathbf{S}$, respectively.

Variant (iii) allows to tightly integrate minimization with a determinization-like construction, i.e., to do both simultaneously instead of sequentially. For practical purposes it is usually the most efficient solution, since it avoids building the final chain of Construction 3.7 and the reflected coalgebra of Definition 4.3 which both usually involve significant combinatorial explosion.

Example 4.10. (NDA) Theorem 4.9 suggests two ways to build the minimization of an NDA (and thus checking the equivalence of its states). We first apply Construction (iv) to the NDA (X, α) in Example 2.4 and then we illustrate Construction (iii). Recall that the reflection of (X, α) into $\mathbf{Set}^{\mathbf{op}}$ is $(\mathcal{P}(X), \alpha')$ in Example 4.5. By applying Construction 3.7 (with the factorization structure of $\mathbf{Set}^{\mathbf{op}}$), we remove from $(\mathcal{P}(X), \alpha')$ the states that are not related to any word in the final coalgebra or, in other words, those states from which there is no path to the final state. Intuitively, we perform a backwards breadth-first search and the factorizations make sure that unreachable states are discarded. The resulting automaton is illustrated above.



Construction (iii) can be understood as an efficient implementation of Construction (iv): we do not build the entire $(\mathcal{P}(X), \alpha')$, but we construct directly the above automaton by iteratively adding states and transitions. We start with state 3, then we add 23 and $\emptyset$ and finally we add 123. All the details are shown in [2].

The minimized NDA can be thought of as a *canonical* representative of its equivalence class. The quest for canonical NDAs (also referred to as "universal") started in

the seventies and, recently, an interesting kind of canonical NDAs (called *átomata*) has been proposed in [8]. In [2] , we show that our minimized NDAs coincide with átomata of [8]. This provides a universal property that uniquely characterizes átomata (up to isomorphism), namely the átomaton of a regular language is the minimization of any NDA accepting the language.

It is worth noting that the automaton obtained above is precisely the automaton in the third step of the well-known Brzozowski algorithm for minimization of non-deterministic automata [7], which, in a nutshell, works as follows: 1) given an NDA reverse it, by reversing all arrows and exchanging final and initial states; 2) determinize it, using the subset construction, and remove unreachable states; 3) reverse it again; 4) determinize it, using the subset construction, and remove unreachable states. In our example, we are doing steps 1)–3) but without the explicit reversal. Our automata do not have initial states, but steps 1)–3) are independent on the specific choice of initial states, because of the two reversals.

Example 4.11. (CTS) Recall the coalgebraic description of CTS given in Example 2.5: the base category is $\mathcal{K}\ell(T)$, where T is the input monad and $\overline{F} = \mathcal{P}_c$ is the countable powerset functor. CTS (1) of the introduction is the coalgebra $\alpha\colon X \to \mathcal{P}_c(X)$ represented by the table in Example 2.5.

We describe the algorithm in Theorem 4.9(iii) with the pseudo-factorization of Example 4.7 (Construction (iv) only consists in the standard minimization of the reflected coalgebra α', that is CTS (2) of the introduction). We start by taking the unique morphism $d_0\colon X \to 1$ into the final object of $\mathcal{K}\ell(T)$, that is $1 = \{\bullet\}$. At the iteration i, we obtain e_i via the pseudo-factorization of $d_i = n_i \circ e_i$, and then we build $d_{i+1} = \overline{F}e_i \circ \alpha$. The iterations of the algorithm are shown in the following tables below.

$d_0\colon X \to 1 = \{\bullet\} = E_0$

d_0, e_0	1	2	3	4	5	6	7	8	9	10
a	•	•	•	•	•	•	•	•	•	•
$\bar{a}$	•	•	•	•	•	•	•	•	•	•

$d_1\colon X \to \mathcal{P}_c(E_0) = \{\emptyset, \{\bullet\}\} = E_1$

d_1, e_1	1	2	3	4	5	6	7	8	9	10
a	$\{\bullet\}$	$\{\bullet\}$	$\emptyset$	$\emptyset$	$\emptyset$	$\{\bullet\}$	$\{\bullet\}$	$\emptyset$	$\emptyset$	$\emptyset$
$\bar{a}$	$\{\bullet\}$	$\emptyset$	$\{\bullet\}$	$\emptyset$	$\emptyset$	$\{\bullet\}$	$\{\bullet\}$	$\emptyset$	$\emptyset$	$\emptyset$

$d_2\colon X \to \mathcal{P}_c(E_1),\ E_2 = \{\emptyset, \{\emptyset\}, \{\emptyset, \{\bullet\}\}\}$

d_2, e_2	1	2	3	4	5	6	7	8	9	10
a	$\{\emptyset, \{\bullet\}\}$	$\{\emptyset\}$	$\emptyset$	$\emptyset$	$\emptyset$	$\{\emptyset, \{\bullet\}\}$	$\{\emptyset\}$	$\emptyset$	$\emptyset$	$\emptyset$
$\bar{a}$	$\{\emptyset, \{\bullet\}\}$	$\emptyset$	$\{\emptyset\}$	$\emptyset$	$\emptyset$	$\{\emptyset, \{\bullet\}\}$	$\{\emptyset\}$	$\emptyset$	$\emptyset$	$\emptyset$

$d_3\colon X \to \mathcal{P}_c(E_2),\ E_3 = \{\emptyset, \{\emptyset\}, \{\emptyset, \{\emptyset\}\}\}$

d_3, e_3	1	2	3	4	5	6	7	8	9	10
a	$\{\emptyset, \{\emptyset\}\}$	$\{\emptyset\}$	$\emptyset$	$\emptyset$	$\emptyset$	$\{\emptyset, \{\emptyset\}\}$	$\{\emptyset\}$	$\emptyset$	$\emptyset$	$\emptyset$
$\bar{a}$	$\{\emptyset, \{\emptyset\}\}$	$\emptyset$	$\{\emptyset\}$	$\emptyset$	$\emptyset$	$\{\emptyset, \{\emptyset\}\}$	$\{\emptyset\}$	$\emptyset$	$\emptyset$	$\emptyset$

Each table represents both d_i and $e_i\colon X \twoheadrightarrow E_i$ (the morphisms n_i such that $d_i = n_i \circ e_i$ are just the obvious injections). At the iterations 0 and 1, $E_0 = 1$ and $E_1 = \mathcal{P}_c(E_0)$. At the iteration 2 instead, $E_2 \neq \mathcal{P}_c(E_1)$, since nothing maps to $\{\{\bullet\}\} \in \mathcal{P}_c(E_1)$.

The algorithm reaches a fixed-point at iteration 3, since there is an iso $\iota\colon E_2 \to E_3$. The minimization $(E_3, \mathcal{P}_c(\iota) \circ n_3)$ is depicted below.

$$\{\emptyset, \{\emptyset\}\} \longrightarrow \{\emptyset\} \longrightarrow \emptyset$$

It is easy to see that the above transition system is isomorphic to the one from the introduction having states x, y, z. Moreover, the coalgebra morphism $e_3\colon (X, \alpha) \twoheadrightarrow (E_3, \mathcal{P}_c(\iota) \circ n_3)$, illustrated in the table above, corresponds to the dashed arrow of the introduction, where 2 is mapped to $\{\emptyset\}$ $(= y)$ if a holds, and to $\emptyset$ $(= z)$ if $\bar{a}$ holds.

5 Conclusion, Related and Future Work

In this work, we have introduced a notion of minimization, which encompasses several concepts of "canonical" systems in the literature, and abstract procedures to compute it. Our approach only relies on *(pseudo-)factorization structures* and it is completely independent of the base category and of the endofunctor F. Together with appropriate reflections, this allows to compute minimizations of interesting types of systems that, for the purpose of minimization, cannot be regarded as coalgebras over $\mathbf{Set}$, such as non-deterministic automataand conditional transition systems.

For non-deterministic automata, which we model as coalgebras in $\mathbf{Rel}$ following [13], the result of the proposed algorithm coincides with the one of the third step of Brzozowski's algorithm [7]. The resulting automata are not minimal in the number of states (it is well-known that there exists no unique minimal non-deterministic automata), but they correspond to átomata, recently introduced in [8].

The example of conditional transition systems is completely original, but it has been motivated by the work in [14,10], which introduces notions of bisimilarity depending on conditions (which are fixed once and for all). The setting of [10] is closer to ours, but no algorithm is given there. Our algorithm can be made more efficient by considering CTSs where conditions are boolean expressions. We already have a prototype implementation performing the fixed-point iteration based on binary decision diagrams. Moreover, our coalgebraic model of CTSs provides a notion of *quantitative bisimulations* that can be seen as a behavioural (pseudo-)metric. We plan to study how our approach can be integrated to define and compute behavioural metrics.

As related work, we should also mention that the notion of minimization generalizes simple [22] and minimal [12] coalgebras in the case where the base category is $\mathbf{Set}$ with epi-mono factorizations. Moreover, several previous studies (e.g. [17,9,23]) have pointed out the relationship between the construction of the final coalgebra (via the final chain [24,4]) and the minimization algorithm. For instance, in case of regular categories the chain of quotients $e_i \colon X \twoheadrightarrow E_i$ (Construction 3.7) corresponds to the chain $K_i \rightarrowtail X \times X$ of their kernel pairs, which is precisely the relation refinement sequence of Staton [23, Section 5.1]. However, none of these works employed reflections for determinization-like constructions, that is exactly what allows us to minimize coalgebras in categories not equipped with a proper factorization structure, such as non-deterministic automata and conditional transition systems.

In future work we will study general conditions ensuring finite convergence: it is immediate to see that for any functor on $\mathbf{Set}$ with epi-mono factorizations, the sequence E_i of a finite coalgebra converges in a finite number of iterations. However, discovering general conditions encompassing all the examples of this paper seems to be non-trivial.

Preliminary research suggests that by integrating our approach with well-pointed coalgebras [5], we might obtain an explicit account of initial states. Indeed, given the reachable part of a pointed coalgebra for a set functor (which is defined through the canonical graph of Gumm [11]), the result of its minimization is a well-pointed coalgebra, i. e., a pointed coalgebra with no proper subcoalgebra and no proper quotient.

In addition we plan to study how our work is related to [20], which also recovers Brzozowski's algorithm in an abstract categorical setting.

Acknowledgements. We would like to thank Ana Sokolova, Paolo Baldan and Walter Tholen for answering our questions and giving generous feedback. We also thank the reviewers for their valuable comments.

References

1. J. Adámek. Free algebras and automata realizations in the language of categories. *Comment. Math. Univ. Carolin.*, 15:589–602, 1974.
2. J. Adámek, F. Bonchi, M. Hülsbusch, B. König, S. Milius, and A. Silva. A coalgebraic perspective on minimization and determinization (extended version). Available from `http://alexandrasilva.org/files/fossacs12-extended.pdf`.
3. J. Adámek, H. Herrlich, and G.E. Strecker. *Abstract and Concrete Categories – The Joy of Cats*. Wiley, 1990.
4. J. Adámek and V. Koubek. On the greatest fixed point of a set functor. *TCS*, 150:57–75, 1995.
5. J. Adámek, S. Milius, L. S. Moss, and L. Sousa. Well-pointed coalgebras. In *Proc. of FOSSACS '12*. Springer, 2012. LNCS, to appear.
6. M. Boreale. Weighted bisimulation in linear algebraic form. In *Proc. of CONCUR '09*, pages 163–177. Springer, 2009. LNCS 5710.
7. J. A. Brzozowski. Canonical regular expressions and minimal state graphs for definite events. In *Mathematical Theory of Automata*, volume 12(6), pages 529–561. Polytechnic Press, NY, 1962.
8. J.A. Brzozowski and H. Tamm. Theory of átomata. In *Proc. of DLT '11*, pages 105–116. Springer, 2011. LNCS 6795.
9. G.L. Ferrari, U. Montanari, and E.Tuosto. Coalgebraic minimization of HD-automata for the pi-calculus using polymorphic types. *TCS*, 331(2–3):325–365, 2005.
10. M. Fitting. Bisimulations and boolean vectors. In *Advances in Modal Logic*, volume 4, pages 1–29. World Scientific Publishing, 2002.
11. H.P. Gumm. From T-coalgebras to filter structures and transition systems. In *Proc. of CALCO '05*, pages 194–212. Springer, 2005. LNCS 3629.
12. H.P. Gumm. On minimal coalgebras. *Applied Categorical Structures*, 16:313–332, 2008.
13. I. Hasuo, B. Jacobs, and A. Sokolova. Generic trace semantics via coinduction. *LMCS*, 3(4:11):1–36, 2007.
14. M. Hennessy and H. Lin. Symbolic bisimulations. *TCS*, 138(2):353–389, 1995.
15. C. Hermida and B. Jacobs. Structural induction and coinduction in a fibrational setting. *Information and Computation*, 145:107–152, 1998.
16. J.E. Hopcroft, R. Motwani, and J.D. Ullman. *Introduction to Automata Theory, Languages, and Computation (3rd Edition)*. Wesley, 2006.
17. A. Kurz. *Logics for Coalgebras and Applications to Computer Science*. PhD thesis, Ludwigs-Maximilians-Universität München, 2000.
18. S. Mac Lane. *Categories for the Working Mathematician*. Springer, 1971.
19. P.S. Mulry. Lifting theorems for Kleisli categories. In *Proc. of MFPS '93*, pages 304–319. Springer, 1993. LNCS 802.
20. P. Panangaden. Duality in probabilistic automata. Available from `http://www.cs.mcgill.ca/~prakash/Talks/duality_talk.pdf`. Slides, 19th May 2011.
21. J. Power and D. Turi. A coalgebraic foundation for linear time semantics. In *Proc. of CTCS '99*, volume 29 of *ENTCS*, pages 259–274, 1999.
22. J.J.M.M. Rutten. Universal coalgebra: a theory of systems. *TCS*, 249:3–80, 2000.
23. S. Staton. Relating coalgebraic notions of bisimulation. *LMCS*, 7(1), 2011.
24. J. Worrell. On the final sequence of a finitary set functor. *TCS*, 338(1-3):184–199, 2005.