Available online at www.sciencedirect.com
ScienceDirect
Electronic Notes in Theoretical Computer Science
ELSEVIER
Electronic Notes in Theoretical Computer Science 203 (2008) 109–129
www.elsevier.com/locate/entcs

Coalgebraising Subsequential Transducers

Helle Hvid Hansen[1,2]

Section Theoretical Computer Science
Vrije Universiteit Amsterdam
Amsterdam, The Netherlands

Abstract

Subsequential transducers generalise both classic deterministic automata and Mealy/Moore type state machines by combining (input) language recognition with transduction. In this paper we show that normalisation and taking differentials of subsequential transducers and their underlying structures can be seen as coalgebraisation. More precisely, we show that the subclass of normalised subsequential structures is a category of coalgebras which is reflective in the category of coaccessible subsequential structures, and which has a final object. This object is then also final for coaccessible structures. The existence and properties of the minimal subsequential transducer realising a partial word function f can be derived from this result. We also show that subsequential structures in which all states are accepting, can be seen as coalgebras by taking differentials. The coalgebraic representation obtained in this way gives rise to an alternative method of deciding transducer equivalence.

Keywords: Subsequential transducer, word function, coalgebra, normalisation, differential.

1 Introduction

Subsequential transducers generalise deterministic finite automata (DFA's) as well as Mealy/Moore type state machines, by combining the notion of final state with output on transitions, and allowing for initial and terminal output. This combination of language recognition and transduction makes subsequential transducers useful in areas such as lexical analysis, coding theory, and more recently, in speech and language processing (cf. [9]). The semantics of subsequential transducers is given in terms of partial word functions $f : A^* \dashrightarrow B^*$. Subsequential transducers were introduced (cf. [17]) as a generalisation of sequential transducers. Sequential transducers and sequential functions were studied as early as the 1950's ([11]) and their theory is well established ([7]). Existing results on subsequential transducers include a characterisation of the functions which can be realised by a finite

[1] Supported by NWO grant 612.000.316.
[2] Email:hhhansen@few.vu.nl

doi:10.1016/j.entcs.2008.05.022

subsequential transducer ([5,4]), and algorithms for determinisation and minimisation ([2,6]).

Automata have traditionally been studied from an algebraic perspective ([7]) with focus on the notions of structure, congruence and initiality. Coalgebra, on the other hand provides abstract notions of behaviour, bisimilarity and minimality arising from finality, and it has proved to be an equally suitable mathematical framework for modelling and studying the behaviour of state-based systems ([15]). In particular, classic deterministic automata and Mealy/Moore machines can all be described coalgebraically in such a way that their traditional semantics coincides with the final semantics (cf. [13,8]). The question naturally arises of whether the coalgebraic modelling generalises to subsequential transducers.

Our motivation is the following. Firstly, we hope to provide a deeper understanding of existing notions and constructions on subsequential transducers by showing that they are instances of more general mathematical notions. Secondly, if we can establish that subsequential transducers, or more precisely, the underlying structures, can be seen as coalgebras, we obtain a range of results and techniques from the general theory of coalgebra which could be applied to them. We mention coalgebraic modal logic (cf. [10,16,19]) for purposes of logic specification and reasoning, and the very recent regular expressions for polynomial functors ([3]) which provide a Kleene-style theorem and a generic synthesis procedure.

In this paper, we will see that, in general, subsequential structures cannot be regarded as coalgebras. The reason is that their semantics requires a more general notion of morphism than is provided by coalgebra. However, we show that the so-called normalised subsequential structures can correctly be seen as coalgebras, and that there exists a final normalised subsequential structure. With respect to finality, the restriction to normalised structures is no real loss, as it is well-known that every subsequential transducer/structure can be normalised. We investigate this result in the coalgebraic setting and show that normalised subsequential structures are full and reflective in the category of coaccessible subsequential structures (the class of subsequential structures in which all states have a well-defined behaviour). This result parallels the fact that minimal DFA's form a reflective subcategory of all DFA's, and is an argument for saying that the coalgebraic description is the right way of thinking about subsequential structures.

Moreover, we show that the class of subsequential transducers in which all states are accepting (step-by-step subsequential structures) can be viewed as coalgebras. This transformation is essentially obtained by taking the differential of a function $f : A^* \dashrightarrow B^*$. The practical interest of this coalgebraic representation is that it provides us with an alternative method for deciding transducer equivalence, which does not require normalisation. We illustrate with an example in section 4.3. Normalisation and taking differentials are both known existing constructions (cf. [4,6]) that transform a subsequential structure or function into a more manageable (computationally, conceptually) representation. The results of this paper show that these transformations are a form of coalgebraisation.

The paper is structured as follows. In section 2 we introduce relevant notions

on sets, words and functions, and give a brief overview of basic coalgebraic notions. In section 3 we review the definition of subsequential transducers and some known results. Finally in section 4, we carry out the coalgebraic modelling of normalised, sequential and step-by-step subsequential transducers.

2 Preliminaries

2.1 Sets, words, functions.

Let X and Y be sets. A (partial) function from X to Y is denoted by $f : X \dashrightarrow Y$. We will write $f : X \to Y$ when f is a total function from X to Y. For a function $f : X \dashrightarrow Y$, and subsets $C \subseteq X$ and $D \subseteq Y$, the f-image of C is denoted $f(C)$, the inverse f-image of D is $f^{-1}(D)$, and the restriction of f to C is $f{\restriction_C}$. The domain and range of f are denoted $\mathrm{dom}(f)$ and $\mathrm{ran}(f)$, respectively. As is standard, we can view a partial function $f : X \dashrightarrow Y$ as a total function $f : X \to Y \cup \{\star\}$, where $\star$ is the *undefined value*, by letting $f(x) = \star$ for all $x \notin \mathrm{dom}(f)$.

The free monoid over a set X is the monoid $(X^*, \varepsilon, .)$ where X^* is the set of all words over X, ε is the empty word, and $u.w$, or simply uw, denotes the concatenation of two words $u, w \in X^*$. If $f, g : X \to B^*$, then $f.g : X \to B^*$ is the function defined by $(f.g)(x) = f(x).g(x)$. The free group over X is denoted by $X^{(*)}$, and the formal inverse of $x \in X$ is written $\overline{x}$. For $w \in X^*$, the inverse of $w = x_1 x_2 \dots x_k$ is $\overline{w} = \overline{x_k} \dots \overline{x_2}\,\overline{x_1}$, and $\overline{\varepsilon} = \varepsilon$. We will apply concatenation and inverse to obtain prefixes and suffixes of words: If $w = uv \in X^*$, then $\overline{u}.w = v$ and $w.\overline{v} = u$. In the case u is not a prefix of $w \in X^*$, then $\overline{u}.w$ is read as an element of $X^{(*)}$. For example, $\overline{aa}a.ab = \overline{aa}.b$. For all $u, w \in X^*$, we write $u \preceq w$ if u is a prefix of w. A subset $T \subseteq X^*$ is called *prefix-closed* if $u \preceq w \in T$ implies $u \in T$. A partial function $f : A^* \dashrightarrow B^*$ is *prefix-preserving* if $\mathrm{dom}(f)$ is prefix-closed, and for all $u, w \in \mathrm{dom}(f)$, if $u \preceq w$ then $f(u) \preceq f(w)$. For a set $S \subseteq A^*$ of words, we denote by $lcp(S)$ the longest common prefix of words in S with the convention that $lcp(\emptyset)$ is undefined.

Let $f : A^* \dashrightarrow B^*$ be a partial function, and $w \in A^*$. The *maximal output of f on input w* is given by

$$f[w] := lcp(f(wA^*)) = lcp(\{f(wu) \mid wu \in \mathrm{dom}(f)\}).$$

The *derivative of f with respect to w* is the partial function $f{\cdot}w : A^* \dashrightarrow B^*$ defined for all $u \in A^*$ by

$$(f{\cdot}w)(u) = \begin{cases} \overline{f[w]}.f(wu) & \text{if } wu \in \mathrm{dom}(f) \\ \star & \text{otherwise} \end{cases}$$

The derivative of f is sometimes called the *residual of f* in the literature on subsequential transducers.

2.2 Coalgebra and Automata

We now recall the basic coalgebraic definitions relevant for this paper, and fix notation. Our coalgebras will be based on Set, the category of sets and (total) functions. Given a functor $\mathsf{T} : \mathsf{Set} \to \mathsf{Set}$, a T-*coalgebra* is a pair $\mathbb{S} = (S, \sigma)$ where S is a set and $\sigma : S \to \mathsf{T}(S)$ is a function. A function $f : S_1 \to S_2$ is a T-*coalgebra morphism* from (S_1, σ_1) to (S_2, σ_2) (written: $f : (S_1, \sigma_1) \to (S_2, \sigma_2)$), if $\mathsf{T}(f) \circ \sigma_1 = \sigma_2 \circ f$. The category of T-coalgebras and T-coalgebra morphisms is denoted by $\mathsf{Coalg}(\mathsf{T})$. In this paper all functors considered are polynomial, i.e. they are constructed from constant sets, identity, product, coproduct and exponentiation.

A *pointed* T-*coalgebra* (S, σ, s_0) consists of a T-coalgebra (S, σ) and an initial state $s_0 \in S$. A *morphism of pointed* T-*coalgebras* from (S_1, σ_1, s_1) to (S_2, σ_2, s_2) is a T-coalgebra morphism $f : (S_1, \sigma_1) \to (S_2, \sigma_2)$ for which $f(s_1) = s_2$. Pointed T-coalgebras and their morphisms form a category $\mathsf{PtCoalg}(\mathsf{T})$. If (S, σ) is a T-coalgebra for a polynomial functor T, and $S' \subseteq S$, then $(S', \sigma{\restriction_{S'}})$ is a *subcoalgebra* of (S, σ) if the inclusion map $i : S' \to S$ is a T-coalgebra morphism. Given a point s in (S, σ), we denote by $\langle s \rangle$ the *subcoalgebra generated by* s *in* (S, σ) which is the least subcoalgebra (S', σ') (w.r.t. inclusion) that contains s. For polynomial functors such a least subcoalgebra always exists, and it can be obtained essentially by taking the transition closure of $\{s\}$.

We use the notion of T-bisimilarity of coalgebras (see e.g. [15]). Let two T-coalgebras $\mathbb{S}_1 = (S_1, \sigma_1)$ and $\mathbb{S}_2 = (S_2, \sigma_2)$ be given. A relation $Z \subseteq S_1 \times S_2$ is a T-*bisimulation between* $\mathbb{S}_1$ *and* $\mathbb{S}_2$, if Z can be equipped with coalgebraic structure $\zeta : Z \to \mathsf{T}(Z)$ such that the projections $\pi_i : Z \to S_i$, $i \in \{1, 2\}$ are T-coalgebra morphisms. Two states $s_1 \in S_1$ and $s_2 \in S_2$ are T-*bisimilar* (notation: $s_1 \sim s_2$) if there exists a T-bisimulation Z between $\mathbb{S}_1$ and $\mathbb{S}_2$ such that $\langle s_1, s_2 \rangle \in Z$.

A *final* T-*coalgebra* (Φ, ϕ) is a final object in the category $\mathsf{Coalg}(\mathsf{T})$. This means that for every T-coalgebra (S, σ) there exists a unique T-coalgebra morphism $h : (S, \sigma) \to (\Phi, \phi)$. In general, a final T-coalgebra may not exist, but if it does, then the final map h assigns to a state s in a T-coalgebra (S, σ) its *behaviour* $h(s)$. In a final T-coalgebra (Φ, ϕ), the *principle of coinduction* holds: For all $s, t \in \Phi$: $s \sim t$ iff $s = t$.

Many known structures are identified as being coalgebras (see e.g. [15,16]). We mention in particular two types of automata which are special instances of subsequential transducers. The first is classic deterministic finite automata over an alphabet A (see e.g. [13]). These are coalgebras for the functor $\mathsf{DFA}(X) = \mathbf{2} \times X^A$, where the output function $o : Q \to \mathbf{2} = \{0, 1\}$ defines whether a state $q \in Q$ is accepting ($o(q) = 1$) or not ($o(q) = 0$); and the transition function $d : Q \to Q^A$ defines for each $q \in Q$, the next state $d(q)(a)$ on input a. It is straightforward to show that DFA-coalgebra morphisms coincide with the well known morphisms of deterministic automata, the final DFA-coalgebra consists of the set of all languages $\mathcal{P}(A^*)$, and the final DFA-coalgebra morphism is the map that sends a state q to the set of words accepted from q. The second example is given by Mealy machines (cf. [8]). A Mealy machine with input in A and output in B is a coalgebra of the type $t : Q \to (B \times Q)^A$. In this case, the final Mealy coalgebra has as its carrier the

set of *causal stream functions*, that is, functions $f : A^\omega \to B^\omega$ where for any stream $\alpha \in A^\omega$ the n'th element of $f(\alpha)$ is determined by the first n elements of α.

3 Subsequential Transducers

In this section we fix notation, and review the basic definitions of subsequential transducers and their morphisms. These follow Choffrut [6] more or less, but the presentation here is slightly different in order to make the connection with coalgebra more clear.

3.1 Basic definitions

Definition 3.1 (subsequential structure & transducer) A *subsequential structure* is a triple $\mathbb{S} = (Q, t, r)$ where Q is a set of states, and t, r are maps of the following type:

$$t : Q \to (A \dashrightarrow (B^* \times Q)) \quad \text{(transition structure)}$$
$$r : Q \dashrightarrow B^* \quad \text{(terminal output function)}$$

A *subsequential transducer* is a 5-tuple $\mathbb{T} = (Q, t, r, i, m)$ where (Q, t, r) is a subsequential structure, $i \in Q$ is the *initial state*, and $m \in B^*$ is the *initial prefix*. The set of *final (or accepting)* states of $\mathbb{T}$ is $F := \mathrm{dom}(r)$. If $q \notin \mathrm{dom}(r)$ then q is called an *internal* state.

Remark 3.2 (notation) For all states $q \in Q$, we will write $\mathrm{supp}(q)$ instead of $\mathrm{dom}(t(q))$. The transition structure t will often be described in terms of an *output function* $o : Q \to (A \dashrightarrow B^*)$, and a *next-state function* $d : Q \to (A \dashrightarrow Q)$ where we quietly assume that for all $q \in Q$, $\mathrm{dom}(o(q)) = \mathrm{dom}(d(q))$.

We extend the definition of o and d from letters to words in the usual manner. For a state q, $a \in A$ and $w \in A^*$, we define $d(q)(\varepsilon) = q$ and $d(q)(wa) = d(d(q)(w))(a)$, similarly, $o(q)(\varepsilon) = \varepsilon$ and $o(q)(wa) = o(q)(w).o(d(q)(w))(a)$, with the proviso that the left-side is defined only if the right-side is.

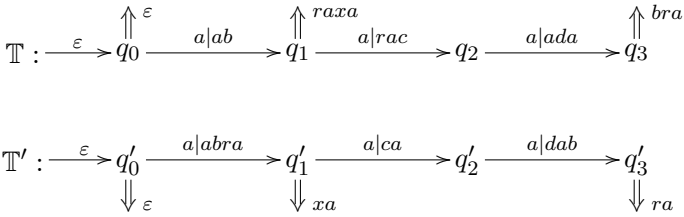
The usual notion of path in subsequential structures applies. A path is called *final* if it ends in a final state, and a state q *coaccessible* if there exists a final path starting in q. The set of coaccessible states of a subsequential structure $\mathbb{S}$ will be denoted by $Coacc(\mathbb{S})$. In a subsequential transducer a final path is *successful* if it starts in the initial state, and a state q is *accessible* if it is reachable from the initial state. A subsequential transducer is called *trim* if all states lie on a successful path.

Definition 3.3 (realisation, behaviour) Given a subsequential structure $\mathbb{S} = (Q, o, d, r)$ a state $q \in Q$ *realises* a partial function $[\![q]\!]_{\mathbb{S}} : A^* \dashrightarrow B^*$ defined for all $w \in A^*$ by: $[\![q]\!]_{\mathbb{S}}(w) = o(q)(w).r(d(q)(w))$. We call $[\![q]\!]_{\mathbb{S}}$ the *behaviour of q (in $\mathbb{S}$)*. Given two subsequential structures $\mathbb{S}$ and $\mathbb{S}'$, two states q in $\mathbb{S}$ and q' in $\mathbb{S}'$ are *equivalent* if $[\![q]\!]_{\mathbb{S}} = [\![q']\!]_{\mathbb{S}'}$. A subsequential transducer $\mathbb{T} = (\mathbb{S}, i, m)$ *realises* a partial function $[\![\mathbb{T}]\!] : A^* \dashrightarrow B^*$ defined as follows for $w \in A^*$: $[\![\mathbb{T}]\!](w) = m.[\![i]\!]_{\mathbb{S}}(w)$. We

refer to $[\![\mathbb{T}]\!]$ as the *behaviour of* $\mathbb{T}$. Two subsequential transducers $\mathbb{T}_1$ and $\mathbb{T}_2$ are *equivalent* if $[\![\mathbb{T}_1]\!] = [\![\mathbb{T}_2]\!]$.

For notational simplicity, we sometimes leave out the subscript when $\mathbb{S}$ is clear from the context, or we use some appropriate indexing, for example, $[\![q]\!]_1$ instead of $[\![q]\!]_{\mathbb{S}_1}$ etc. Before we give the definition of morphisms between subsequential transducers, we look at an example.

Example 3.4 Consider the subsequential transducers $\mathbb{T}$ and $\mathbb{T}'$ as illustrated below. The inital prefix in both transducers is the empty word, and the terminal output functions r and r' are indicated with double arrows. The states q_2 and q'_2 are internal.



We see that $\mathrm{dom}([\![\mathbb{T}]\!]) = \mathrm{dom}([\![\mathbb{T}']\!]) = \{\varepsilon, a, aaa\}$, and the two transducers compute the same partial function $f : \{a\}^* \dashrightarrow \{a,b,c\}^*$: $f(\varepsilon) = \varepsilon$, $f(a) = abraxa$, $f(aaa) = abracadabra$.

Intuitively, we would like the state map $\alpha(q) = q', q \in Q$, to be a morphism from $\mathbb{T}$ to $\mathbb{T}'$, since $\mathbb{T}'$ is just like $\mathbb{T}$ except that, internally, $\mathbb{T}'$ produces its output a bit faster than $\mathbb{T}$. In other words, we could shift some of the output letters "upstream" in $\mathbb{T}$ and obtain an obviously isomorphic copy of $\mathbb{T}'$.

The above example illustrates the idea behind the notion of subsequential morphism.

Definition 3.5 (subsequential morphism) Let $\mathbb{S}_j = (Q_j, o_j, d_j, r_j)$, $j = 1, 2$, be two subsequential structures. A partial function $\alpha : Q_1 \dashrightarrow Q_2$ is a *subsequential morphism* from $\mathbb{S}_1$ to $\mathbb{S}_2$, if there exists a function $\beta : Q_1 \to B^*$ such that the following conditions are satisfied for all $q \in \mathrm{dom}(\alpha)$:

(supp) $\quad \mathrm{supp}(q) = \mathrm{supp}(\alpha(q))$,

(next) $\quad \forall a \in \mathrm{supp}(q) : \alpha(d_1(q)(a)) = d_2(\alpha(q))(a)$,

(out) $\quad \forall a \in \mathrm{supp}(q) : \beta(q).o_2(\alpha(q))(a) = o_1(q)(a).\beta(d_1(q)(a))$,

(acc) $\quad q \in \mathrm{dom}(r_1) \iff \alpha(q) \in \mathrm{dom}(r_2)$, i.e., $\alpha^{-1}[F_2] = F_1$,

(term-out) $\forall q \in \mathrm{dom}(r_1) : \beta(q).r_2(\alpha(q)) = r_1(q)$.

We will often use the notation $(\alpha, \beta) : \mathbb{S}_1 \dashrightarrow \mathbb{S}_2$ to say that α is a subsequential morphism from $\mathbb{S}_1$ to $\mathbb{S}_2$ with witnessing function β.

Given two subsequential transducers $\mathbb{T}_j = (\mathbb{S}_j, i_j, m_j)$, $j = 1, 2$, and a subsequential morphism $(\alpha, \beta) : \mathbb{S}_1 \dashrightarrow \mathbb{S}_2$, then (α, β) is a *subsequential (transducer)*

morphism from $\mathbb{T}_1$ to $\mathbb{T}_2$ if α and β satisfy:

(init) $\alpha(i_1) = i_2$,

(ε-in) $m_2 = m_1.\beta(i_1)$.

Stated informally, the map $\beta : Q_1 \to B^*$ in the above definition defines the speed-up in output which would synchronise $\mathbb{T}_1$ with $\mathbb{T}_2$ at all computation steps, including incomplete computations that have not yet reached a final state. We note that if we would allow β to be a function to the free group $B^{(*)}$ (as in [6]), then β would allow not only the speeding up of output, but also delaying output. The reader can now verify that in Example 3.4 the map $\alpha(q) = q'$, $q \in Q$, is a subsequential morphism by taking $\beta(q_0) = \varepsilon$, $\beta(q_1) = ra$, $\beta(q_2) = a$ and $\beta(q_3) = b$.

Looking closer at Definition 3.5 we note that a subsequential transducer morphism $\alpha : \mathbb{T}_1 \dashrightarrow \mathbb{T}_2$ must be defined on all accessible states due to the conditions (init) and (next). For subsequential structures (where the notion of accessibility does not apply), coaccessibility is the crucial property which makes a state interesting. The following lemma is well-known for deterministic transition structures.

Lemma 3.6 *If $\alpha : \mathbb{S}_1 \dashrightarrow \mathbb{S}_2$ is a subsequential morphism and $\mathbb{S}_1 = (Q_1, o_1, d_1, r_1)$ and $\mathbb{S}_2 = (Q_2, o_2, d_2, r_2)$, then for all $q \in \mathrm{dom}(\alpha)$ and all $w \in A^*$: $d_1(q)(w) \in F_1$ iff $d_2(\alpha(q))(w) \in F_2$. In particular, for all $q \in \mathrm{dom}(\alpha)$, q is coaccessible iff $\alpha(q)$ is coaccessible.*

Proof. By induction on the length of w, details are left to the reader. □

We observe that given $\alpha : Q_1 \dashrightarrow Q_2$, if a $\beta : Q_1 \to B^*$ exists which makes α a subsequential morphism, then there is only one possible definition of β on coaccessible states.

Lemma 3.7 *If $\alpha : \mathbb{S}_1 \dashrightarrow \mathbb{S}_2$ is a subsequential morphism with $\beta : Q_1 \to B^*$ as witnessing function, then $\beta{\restriction}_{Coacc(\mathbb{S}_1)}$ is uniquely defined.*

Proof. Let $\mathbb{S}_1 = (Q_1, o_1, d_1, r_1)$ and $\mathbb{S}_2 = (Q_2, o_2, d_2, r_2)$ be subsequential structures. If $q_0 \xrightarrow{a_1} q_1 \xrightarrow{a_2} \ldots \xrightarrow{a_n} q_n$, $n \geq 0$, is a final path in $\mathbb{S}_1$, then $q_n \in F_1$ and $\beta(q_n)$ is determined by (term-out). Now for $i = 0, \ldots, n-1$, $\beta(q_i)$ is defined by $\beta(q_{i+1})$ and (out):

$$\beta(q_i) = o_1(q_i)(a_{i+1}).\beta(q_{i+1}).\overline{o_2(\alpha(q_i))(a_{i+1})}. \tag{1}$$

The (out)-condition ensures that $o_2(\alpha(q_i))(a_{i+1})$ is a suffix of $o_1(q_i)(a_{i+1}).\beta(q_{i+1})$, so β indeed takes values in B^*. □

Subsequential morphisms do not preserve behaviour of states, i.e., $(\alpha, \beta) : \mathbb{S}_1 \dashrightarrow \mathbb{S}_2$ *does not* imply that for all q in $\mathrm{dom}(\alpha)$, $[\![q]\!]_1 = [\![\alpha(q)]\!]_2$. Instead we have:

Lemma 3.8 *If $(\alpha, \beta) : \mathbb{S}_1 \dashrightarrow \mathbb{S}_2$, then for all $q \in \mathrm{dom}(\alpha)$, $[\![q]\!]_1 = \beta(q).[\![\alpha(q)]\!]_2$.*

Proof. Let $(\alpha, \beta) : \mathbb{S}_1 \dashrightarrow \mathbb{S}_2$ be a subsequential morphism from $\mathbb{S}_1 = (Q_1, o_1, d_1, r_1)$ to $\mathbb{S}_2 = (Q_2, o_2, d_2, r_2)$. From Lemma 3.6 it follows immediately that $\mathrm{dom}([\![q]\!]_1) = \mathrm{dom}([\![\alpha(q)]\!]_2)$. Now let $w = a_1 a_2 \ldots a_n \in \mathrm{dom}([\![q]\!]_1)$, and let $q_0, q_1, \ldots, q_n$ be the

run on w in $\mathbb{S}_1$, i.e., $q_0 = q$ and for $j = 0, \ldots, n-1$, $q_{j+1} = d_1(q_j)(a_{j+1})$. We then have:

$$\begin{aligned}
[\![q]\!]_1(w) &= o_1(q_0)(a_1) \ldots o_1(q_{n-1})(a_n).r_1(q_n) \\
\text{(out), (term-out)} &= \beta(q_0).o_2(\alpha(q_0))(a_1)\overline{\beta(q_1)}.\beta(q_1).o_2(\alpha(q_1))(a_2)\overline{\beta(q_2)}. \ \ldots \\
&\qquad \ldots \beta(q_{n-1}).o_2(\alpha(q_{n-1}))(a_n).\overline{\beta(q_n)}.\beta(q_n).r_2(\alpha(q_n)) \\
&= \beta(q_0).o_2(\alpha(q_0))(a_1).\ldots.o_2(\alpha(q_{n-1}))(a_n).r_2(\alpha(q_n)) \\
\text{(next)} &= \beta(q).[\![\alpha(q)]\!]_2(w).
\end{aligned}$$

□

Proposition 3.9 *If there exists a subsequential transducer morphism $\alpha : \mathbb{T}_1 \dashrightarrow \mathbb{T}_2$, then $[\![\mathbb{T}_1]\!] = [\![\mathbb{T}_2]\!]$.*

Proof. Let $\mathbb{T}_1 = (\mathbb{S}_1, i_1, m_2)$, $\mathbb{T}_2 = (\mathbb{S}_2, i_2, m_2)$ and $(\alpha, \beta) : \mathbb{T}_1 \dashrightarrow \mathbb{T}_2$ be a subsequential transducer morphism. By definition, for $w \in A^*$: $[\![\mathbb{T}_1]\!](w) = m_1.[\![i_1]\!]_1(w)$. From Lemma 3.8 and (init), we get $[\![\mathbb{T}_1]\!](w) = m_1.\beta(i_1).[\![i_2]\!]_2(w)$, and finally from ($\varepsilon$-in), $[\![\mathbb{T}_1]\!](w) = m_2.[\![i_2]\!]_2(w) = [\![\mathbb{T}_2]\!](w)$. □

Subsequential morphisms can be composed as described in the following lemma (the straightforward proof is omitted).

Lemma 3.10 *If $(\alpha_1, \beta_1) : \mathbb{S}_1 \dashrightarrow \mathbb{S}_2$, and $(\alpha_2, \beta_2) : \mathbb{S}_2 \dashrightarrow \mathbb{S}_3$, are subsequential morphisms, then $(\alpha_2 \circ \alpha_1, \beta_1.(\beta_2 \circ \alpha_1)) : \mathbb{S}_1 \dashrightarrow \mathbb{S}_3$.*

It follows that subsequential structures and subsequential morphisms form a category SubSeq. Similarly, subsequential transducers and subsequential transducer morphisms form a category SubSeqTra.

3.2 Normalised Subsequential Transducers

Minimisation via quotienting with a suitable congruence is a standard automata-theoretic construction. A congruence is the kernel of a morphism, and for classical automata it is clear that morphisms preserve state behaviour. However, we also know that subsequential morphisms, in general, do not preserve state behaviour. The observation made by Choffrut [5] was that a subsequential transducer can be transformed into an equivalent one in which output is produced at maximal speed, and that in this *normalised* transducer, state equivalence can be determined much in the same way as for DFA's (as also remarked in [6]). In terms of subsequential morphisms (α, β), normalisation can be seen as an optimisation with respect to β, and quotienting with state equivalence as an optimisation with respect to α. A minimal subsequential transducer is optimal with respect to both.

Definition 3.11 (normalised, minimal) Let $\mathbb{S} = (Q, o, d, r)$ be a subsequential structure, and $q \in Q$. We define a function $\hat{\beta}_{\mathbb{S}} : Q \to \mathbf{1} + B^*$ by

$$\hat{\beta}_{\mathbb{S}}(q) = [\![q]\!]_{\mathbb{S}}[\varepsilon] = lcp(\{o(q)(w).r(d(q)(w)) \mid w \in A^*\}). \tag{2}$$

That is, $\hat{\beta}_{\mathbb{S}}(q)$ is the longest common prefix over all output words on final paths

starting in q. We say that a state $q \in Q$ *is normalised* if $\hat{\beta}_{\mathbb{S}}(q) = \varepsilon$. Note that q is coaccessible, iff $\hat{\beta}_{\mathbb{S}}(q) \in B^*$. A subsequential structure $\mathbb{S}$ is *normalised*, if all states in $\mathbb{S}$ are normalised, and $\mathbb{S}$ is *minimal*, if it is normalised, and no two states are equivalent. A subsequential transducer $\mathbb{T}$ is *normalised (minimal)* if the underlying subsequential structure is normalised (minimal).

Again, we may leave out the subscript in $\hat{\beta}_{\mathbb{S}}$ when $\mathbb{S}$ is clear from the context. In [6] an algorithm is given to compute $\hat{\beta}_{\mathbb{S}}$ for finite $\mathbb{S}$. A subsequential structure $\mathbb{S}$ can be normalised under the following transformation, which can be seen as the morphism $(id\restriction_{Coacc(\mathbb{S})}, \hat{\beta})$, cf. Theorem 4.5.

Definition 3.12 (normalisation) Let $\mathbb{T} = (Q, o, d, r, i, m)$ be a subsequential transducer. Define $\mathsf{N}(\mathbb{T}) = (Coacc(\mathbb{T}), o', d, r', i, m')$ where for all $q \in Q$, and all $a \in A$:

$$m' = m.\hat{\beta}(i),\ o'(q)(a) = \overline{\hat{\beta}(q)}.o(q)(a).\hat{\beta}(d(q)(a)),\ r'(q) = \overline{\hat{\beta}(q)}.r(q) \tag{3}$$

$\mathsf{N}(\mathbb{T})$ is called the normalisation of $\mathbb{T}$. Similarly, if $\mathbb{S} = (Q, o, d, r)$ is a subsequential structure, then the normalisation of $\mathbb{S}$ is $\mathsf{N}(\mathbb{S}) = (Coacc(\mathbb{S}), o', d, r')$, where o' and r' are defined as in (3).

Finally we recall the construction of a minimal subsequential transducer from an arbitrary function $f : A^* \dashrightarrow B^*$ (cf. [6]).

Definition 3.13 (minimal realisation) Let $f : A^* \dashrightarrow B^*$ be given. We define the subsequential transducer $\mathbb{T}_f = (\Delta_f, o, d, r, i, m)$ by taking:

$$\begin{aligned} \Delta_f &= \{f{\cdot}w \mid w \in A^*\}, & o(f{\cdot}w)(a) &= \overline{f[w]}.f[wa], \\ i &= f{\cdot}\varepsilon, & d(f{\cdot}w)(a) &= f{\cdot}wa, \\ m &= f[\varepsilon], & r(f{\cdot}w) &= \overline{f[w]}.f(w). \end{aligned}$$

It is straightforward to check that $\mathbb{T}_f$ realises f, and $\mathbb{T}_f$ is minimal.

4 Coalgebraic Modelling

Our aim is to find out whether subsequential transducers can be seen as coalgebras. It is easy to see that a subsequential structure (Q, t, r) has the type of a coalgebra for the Set-functor S defined by:

$$\begin{aligned} \mathsf{S}(X) &= (\mathbf{1} + B^* \times X)^A \times (\mathbf{1} + B^*), \\ \mathsf{S}(f : X \to Y) &= (\mathbf{1} + \mathsf{Id}_{B^*} \times f)^{\mathsf{Id}_A} \times (\mathbf{1} + \mathsf{Id}_{B^*}). \end{aligned}$$

Instantiating the definitions of S-coalgebra and S-coalgebra morphism yields:

Definition 4.1 ($\mathsf{Coalg}(\mathsf{S})$) An S-coalgebra consists of a carrier set Q and a map $\langle t, r\rangle : Q \to \mathsf{S}(Q)$, i.e.,

$$\begin{array}{rcl} \langle t, r\rangle & : & Q \to (\mathbf{1} + B^* \times Q)^A \times (\mathbf{1} + B^*) \\ & & q \mapsto \quad \langle \quad t(q) \quad , \quad r(q) \,\rangle \end{array}$$

Again, we will describe t in terms of an output function and a next-state function (cf. Remark 3.2), and $\mathrm{supp}(q) := \{a \in A \mid t(q)(a) \neq \star\}$, for all $q \in Q$.

If $\mathbb{S}_j = (Q_j, o_j, d_j, r_j)$, $j = 1, 2$, are S-coalgebras, a function $\alpha : Q_1 \to Q_2$ is an S-coalgebra morphism if for all $q \in Q_1$:

$$\begin{array}{ll} \text{(supp)} & \mathrm{supp}(q) = \mathrm{supp}(\alpha(q)), \\ \text{(next)} & \forall a \in \mathrm{supp}(q) : \alpha(d_1(q)(a)) = d_2(\alpha(q))(a), \\ \text{(out)}_n & \forall a \in \mathrm{supp}(q) : o_2(\alpha(q))(a) = o_1(q)(a), \\ \text{(acc)} & q \in \mathrm{dom}(r_1) \iff \alpha(q) \in \mathrm{dom}(r_2), \\ \text{(term-out)}_n & \forall q \in \mathrm{dom}(r_1) : r_2(\alpha(q)) = r_1(q). \end{array}$$

Lemma 4.2 *Let $\mathbb{S}_j = (Q_j, o_j, d_j, r_j)$, $j = 1, 2$, and be S-coalgebras. A total function $\alpha : Q_1 \to Q_2$ is an S-coalgebra morphism if and only if (α, ε) is a subsequential morphism.*

Proof. Follows almost immediately from Definitions 4.1 and 3.5. □

In other words, S-coalgebra morphisms do not allow the shifting of output letters via some function $\beta : Q \to B^*$. This means that, in general, subsequential structures can be seen as coalgebras only at the level of objects, but the coalgebraic notion of morphism is too strict with respect to the intended semantics of subsequential transducers. However, in the next section we show that normalisation is a natural transformation from subsequential structures to coalgebras.

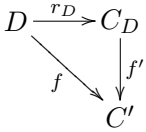
4.1 Normalisation is Coalgebraisation

From now on we will restrict ourselves to coaccessible subsequential structures, that is, subsequential structures in which all states are coaccessible. In the literature on automata, (co)accessibility is often assumed (cf. [6]). From an algorithmic point of view this is justified by the fact that it is straightforward to make a finite automaton (co)accessible. Our motivation is mainly technical, since coaccessibility allows us to work with total maps as morphisms, and it ensures that all states have non-empty behaviour.

Let $\mathsf{CSubSeq}$ be the (non-full) subcategory of SubSeq consisting of coaccessible subsequential structures and total subsequential morphisms between them. Let $\mathsf{NSubSeq}$ be the full subcategory of $\mathsf{CSubSeq}$ consisting of normalised subsequential structures and total subsequential morphisms. We will show that normalised subsequential structures are essentially coalgebras, and that $\mathsf{NSubSeq}$ is reflective in $\mathsf{CSubSeq}$.

We recall the definition of a reflective subcategory (see e.g. [1]). Let C be a subcategory of D, and D and object in D. A C-*reflection arrow* for D is a D-morphism $r_D : D \to C_D$ to some C-object C_D which has the following universal

property. For any $C' \in \mathsf{C}$ and any D-morphism $f : D \to C'$ there is a unique C-morphism $f' : C_D \to C'$ such that $f = f' \circ r_D$. That is, the following diagram commutes:



A subcategory C of D is *reflective in* D if every D-object has a C-reflection arrow. An equivalent formulation is the following: A subcategory C of D is *reflective in* D if the embedding functor $\mathsf{E} : \mathsf{C} \to \mathsf{D}$ has a left adjoint $\mathsf{R} : \mathsf{D} \to \mathsf{C}$. This left adjoint R is called a *reflector*.

First we make some observations regarding subsequential morphisms. Note that from Lemma 3.8 it follows that for any subsequential morphism $(\alpha, \beta) : \mathbb{S}_1 \dashrightarrow \mathbb{S}_2$, and all states q in $\mathrm{dom}(\alpha)$:

$$\hat{\beta}_1(q) = \beta(q).\hat{\beta}_2(\alpha(q)). \tag{4}$$

Using (4) and Lemma 4.2 it is easy to show the following (proof is left to the reader).

Lemma 4.3 (i) *If $(\alpha, \beta) : \mathbb{S}_1 \to \mathbb{S}_2$ is a* CSubSeq*-morphism, and $\mathbb{S}_2$ is normalised, then $\beta = \hat{\beta}_1$.*

(ii) *If $\mathbb{S}_1$ and $\mathbb{S}_2$ are normalised and $(\alpha, \beta) : \mathbb{S}_1 \to \mathbb{S}_2$ is a total subsequential morphism, then $\beta = \varepsilon$.*

(iii) NSubSeq *is a full subcategory of* Coalg(S).

Next we show that the normalisation operation N defined in Def. 3.12 can be extended to a functor from CSubSeq to NSubSeq.

Proposition 4.4 *For $\alpha : \mathbb{S}_1 \to \mathbb{S}_2$ in* CSubSeq *define $\mathsf{N}(\alpha) := \alpha$. The map N is a functor from* CSubseq *to* NSubSeq.

Proof. Assume $(\alpha, \beta) : \mathbb{S}_1 \to \mathbb{S}_2$ in CSubSeq, and $\mathsf{N}(\mathbb{S}_j) = (S_j, o'_j, d_j, r'_j)$, $j = 1, 2$. We must prove that $\alpha : \mathsf{N}(\mathbb{S}_1) \to \mathsf{N}(\mathbb{S}_2)$ in NSubSeq. By Lemma 4.3.(iii) this amounts to showing that $\alpha : \mathsf{N}(\mathbb{S}_1) \to \mathsf{N}(\mathbb{S}_2)$ is a Coalg(S)-morphism. Conditions (supp), (next) and (acc) follow from the assumption that $(\alpha, \beta) : \mathbb{S}_1 \to \mathbb{S}_2$ is a subsequential morphism. To see that $o'_1(q)(a) = o'_2(\alpha(q))(a)$ we have for $q \in \mathrm{dom}(\alpha)$ and $a \in A$ using (out) and (next) for (α, β), and equation (4):

$$\begin{aligned}
o'_1(q)(a) &= \overline{\hat{\beta}_1(q)}.o_1(q)(a).\hat{\beta}_1(d_1(q)(a)) \\
&\overset{(4)}{=} \overline{\hat{\beta}_2(\alpha(q)).\overline{\beta(q)}}.o_1(q)(a).\beta(d_1(q)(a)).\hat{\beta}_2(\alpha(d_1(q)(a))) \\
&\overset{(\mathrm{out})}{=} \overline{\hat{\beta}_2(\alpha(q))}.o_2(\alpha(q))(a).\hat{\beta}_2(\alpha(d_1(q)(a))) \\
&\overset{(\mathrm{next})}{=} o'_2(\alpha(q))(a).
\end{aligned}$$

Similarly, using (4) and (term-out) for (α, β) we have for $q \in \mathrm{dom}(\alpha)$:

$$r_1'(q) = \overline{\hat{\beta}_1(q)}.r_1(q) = \overline{\hat{\beta}_2(\alpha(q))}.\overline{\beta(q)}.r_1(q) = \overline{\hat{\beta}_2(\alpha(q))}.r_2(\alpha(q))$$
$$= r_2'(\alpha(q)).$$ □

We now show that normalised subsequential structures form a reflective subcategory of CSubSeq.

Theorem 4.5 NSubSeq *is a reflective subcategory of* CSubSeq.

Proof. For a coaccessible $\mathbb{S} = (Q, o, d, r)$, the map id_Q, is an NSubSeq-reflection arrow for $\mathbb{S}$: It is straightforward to check that $(id_Q, \hat{\beta}_{\mathbb{S}}) : \mathbb{S} \to \mathsf{N}(\mathbb{S})$ is a surjective subsequential morphism, and that for any normalised $\mathbb{S}' \in$ NSubSeq, and CSubSeq-morphism $(\alpha, \beta) : \mathbb{S} \to \mathbb{S}'$, the unique NSubSeq-morphism $\alpha' : \mathsf{N}(\mathbb{S}) \to \mathbb{S}'$ such that $(\alpha', \varepsilon) \circ (id_Q, \hat{\beta}_{\mathbb{S}}) = (\alpha, \beta)$ is just $\alpha' = \alpha$. □

The above theorem easily extends from structures to transducers.

Corollary 4.6 *Normalised subsequential transducers form a reflective subcategory of coaccessible subsequential tranducers.*

From Definition 3.11 it is clear that if a function $f : A^* \dashrightarrow B^*$ is the behaviour of some state q in a normalised subsequential structure, then $f[\varepsilon] = \varepsilon$, and hence $f \cdot \varepsilon = f$. We now show that the set $\Phi := \{f : A^* \dashrightarrow B^* \mid f[\varepsilon] = \varepsilon\}$ of all such (normalised) functions carries a normalised subsequential structure, and that this object is final in NSubSeq with the behaviour map $[\![\_]\!]$ as the unique subsequential morphism into Φ. The subsequential structure on Φ is obtained as follows: We define $\tau : \Phi \to (\mathbf{1} + B^* \times \Phi)^A$ and $\rho : \Phi \to \mathbf{1} + B^*$ by:

$$\begin{aligned} \tau(f) &= \langle f[a], f \cdot a \rangle \text{ if } \mathrm{dom}(f \cdot a) \neq \emptyset, \text{ otherwise } \tau(f) = \star, \\ \rho(f) &= f(\varepsilon). \end{aligned} \tag{5}$$

Theorem 4.7 *The subsequential structure $\Phi = (\Phi, \tau, \rho)$ is a final object in the category* NSubSeq *of normalised subsequential structures and total subsequential morphisms.*

Proof. We first show that derivatives are normalised. Let $f : A^* \dashrightarrow B^*$ and $a \in A$. By definition, $(f \cdot a)[\varepsilon] = lcp(\{\overline{f[a]}.f(aw) \mid aw \in \mathrm{dom}(f)\})$ and $f[a] = lcp(\{f(aw) \mid aw \in dom(f)\})$, hence $(f \cdot a)[\varepsilon] = \varepsilon$. It follows that τ is well-defined, and (Φ, τ, ρ) is a normalised subsequential structure.

In the rest of this proof we will denote the output and next-state components of τ by o_τ and d_τ, respectively. We show that if $\mathbb{S} = (Q, o, d, r)$ is a normalised subsequential structure, then the behaviour map $h := [\![\_]\!]_{\mathbb{S}} : Q \to \Phi$ is a subsequential morphism, i.e., an S-coalgebra morphism. Recall that by definition of normalised structures, all states in $\mathbb{S}$ are coaccessible, hence $[\![\_]\!]_{\mathbb{S}}$ is total.

Let $q \in Q$, and $a \in A$. We have: $a \in \mathrm{supp}(q)$ iff there exists a $q' \in Q$ such that $d(q)(a) = q'$. By coaccessibility, this is equivalent with the existence of a $w \in A^*$ such that $d(q)(aw) \in F$, which in turn is equivalent with the existence of a $w \in A^*$ such that $aw \in \mathrm{dom}(h(q))$, i.e., $\mathrm{dom}(h(q) \cdot a) \neq \emptyset$, i.e., $a \in \mathrm{supp}(h(q))$.

To see that $o_\tau(h(q))(a) = o(q)(a)$, we note that, again by coaccessibility, $o(q)(a) \preceq lcp(\llbracket q \rrbracket(aA^*)) = h(q)[a]$, and since in particular $d(q)(a)$ is normalised, we also have $h(q)[a] \preceq o(q)(a)$, hence $o(q)(a) = h(q)[a] = o_\tau(h(q))(a)$.

In order to show $h(d(q)(a)) = h(q)\cdot a$, let $w \in A^*$, then

$$\begin{aligned} h(d(q)(a))(w) &= o(d(q)(a))(w).r(d(d(q)(a))(w)) \\ &= \overline{o(q)(a)}.o(q)(aw).r(d(q)(aw)) \\ &= \overline{h(q)[a]}.h(q)(aw) \\ &= (h(q)\cdot a)(w). \end{aligned}$$

Also $\rho(h(q)) = h(q)(\varepsilon) = r(q)$. We leave it to the reader to verify that the map $h : (Q, o, d, r) \to (\Phi, o_\tau, d_\tau, \rho)$ is unique. □

Remark 4.8 (final object in $\mathsf{Coalg}(\mathsf{S})$ and SubSeq) Note that Φ is not final in $\mathsf{Coalg}(\mathsf{S})$, since the behaviour map will not in general be an S-coalgebra morphism (cf. Lemma 4.3.(i)). However, Φ is also not final in SubSeq, where non-trivial β's are allowed. The reason is that the behaviour map may fail to satisfy condition (supp) of Definition 3.5. For example, if a state q is coaccessible, but $d(q)(a)$ is not, yet still present, then $a \in \mathrm{supp}(q)$ but $a \notin \mathrm{supp}(\llbracket q \rrbracket)$. Adding a sink state (the empty map) to Φ will not solve this problem, since transitions between states with undefined behaviour cannot be matched (as required by (supp)) by transitions to/from the sink state.

Corollary 4.9 *The normalised subsequential structure $\Phi = (\Phi, \tau, \rho)$ (as defined in (5)) is a final object in* $\mathsf{CSubSeq}$.

Proof. This is a consequence of Theorems 4.5 and 4.7. There are several ways of formulating the proof, here is one: For any $\mathbb{S} \in \mathsf{CSubSeq}$, Theorem 4.5 tells us that there is a bijection of Hom-sets: $\mathsf{CSubSeq}(\mathbb{S}, \Phi) \cong \mathsf{NSubSeq}(\mathsf{N}(\mathbb{S}), \Phi)$. Hence the unique morphism $\llbracket \_ \rrbracket_{\mathsf{N}(\mathbb{S})} : \mathsf{N}(\mathbb{S}) \to \Phi$ corresponds to a unique morphism $h_{\mathbb{S}} : \mathbb{S} \to \Phi$. Concretely, one can show that $h_{\mathbb{S}} = (\llbracket \_ \rrbracket_{\mathsf{N}(\mathbb{S})}, \varepsilon) \circ (id_Q, \hat{\beta}_{\mathbb{S}}) = (\llbracket \_ \rrbracket_{\mathsf{N}(\mathbb{S})}, \hat{\beta}_{\mathbb{S}})$ is the unique subsequential morphism from $\mathbb{S}$ to Φ. □

Recall the minimal subsequential transducer $\mathbb{T}_f$ of a function $f : A^* \to B^*$ (cf. Definition 3.13). We now show that the existence and properties of $\mathbb{T}_f$ given in [6] are a consequence of Theorem 4.7. Recall the following notation. Given a function $f : A^* \dashrightarrow B^*$ in Φ, $\langle f \rangle_\Phi$ denotes the subcoalgebra generated by f in (Φ, τ, ρ).

Corollary 4.10 (i) *For any $f : A^* \dashrightarrow B^*$, $\mathbb{T}_f = (\langle f\cdot\varepsilon \rangle_\Phi, f\cdot\varepsilon, f[\varepsilon])$.*

(ii) *For any trim subsequential transducer $\mathbb{T} = (\mathbb{S}, i, m)$ and $f = \llbracket \mathbb{T} \rrbracket$, the behaviour map $\llbracket \_ \rrbracket_{\mathsf{N}(\mathbb{S})}$ is the unique subsequential transducer morphism (with witnessing function $\hat{\beta}_{\mathbb{S}}$) from $\mathbb{T}$ onto the minimal subsequential transducer $\mathbb{T}_f$.*

(iii) *Two subsequential transducers $\mathbb{T}_1$ and $\mathbb{T}_2$ are equivalent iff there exists a subsequential transducer $\mathbb{T}$ and subsequential morphisms $\alpha_j : \mathbb{T}_j \to \mathbb{T}$, for $j \in \{1, 2\}$.*

Proof. Item (i): Almost immediate, left to the reader. Item (ii): The proof can be found in [6], but it follows essentially from (Φ, τ, ρ) being final in $\mathsf{CSubSeq}$ (Corollary 4.9) and item (i). Item (iii): The direction from right to left follows from the fact that subsequential transducer morphisms preserve behaviour (Proposition 3.9); the other direction follows from (ii). □

4.2 Coalgebras for Sequential Transducers

A subsequential structure $\mathbb{S} = (Q, o, d, r)$ is *sequential* if $\mathrm{dom}(r) = Q$ and for all $q \in Q : r(q) = \varepsilon$. A *sequential transducer* $\mathbb{T} = (\mathbb{S}, i)$ consists of a sequential structure $\mathbb{S}$ and an initial state i in $\mathbb{S}$. Sequential transducers can thus be seen as the subsequential ones in which the initial prefix is the empty word ε, and all states are final with terminal output ε. We can therefore leave out m and r from the description of a sequential transducer, and simply write $\mathbb{T} = (Q, o, d, i)$. Moreover, from their definition it is immediate that a sequential structure is normalised, since for all states q, $\hat{\beta}(q) = r(q) = \varepsilon$.

Sequential transducers are treated in detail by Eilenberg [7] under the name *generalised sequential machines*. In particular, in Chapter XII of [7] the existence of a final sequential structure is proved, but without any mention of the words coalgebra, finality and bisimulation. In this section, we give a coalgebraic formulation of Eilenberg's [7] results, and relate them to the results of section 4.1.

First we look at the morphisms. By working out the details of Definition 3.5 we find that given two sequential structures $\mathbb{S}_1 = (Q_1, o_1, d_1)$ and $\mathbb{S}_2 = (Q_2, o_2, d_2)$, a function $\alpha : Q_1 \to Q_2$ is a subsequential morphism from $\mathbb{S}_1$ to $\mathbb{S}_2$ if for all $q \in Q_1$:

$$\begin{aligned} &(\mathrm{supp}) \;\; \mathrm{supp}(q) = \mathrm{supp}(\alpha(q)), \\ &(\mathrm{next}) \;\; \forall a \in \mathrm{supp}(q) : \alpha(d_1(q)(a)) = d_2(\alpha(q))(a), \\ &(\mathrm{out})_n \;\; \forall a \in \mathrm{supp}(q) : o_1(q)(a) = o_2(\alpha(q))(a). \end{aligned} \tag{6}$$

A subsequential morphism between sequential structures will simply be called a *sequential morphism*.

Let the functor $\mathsf{S}_0 : \mathsf{Set} \to \mathsf{Set}$ be defined by: $\mathsf{S}_0(X) = (\mathbf{1} + B^* \times X)^A$. It is easily seen that there is a 1-1 correspondence between sequential structures and S_0-coalgebras by taking the transition structure $t : Q \to (\mathbf{1} + B^* \times Q)^A$ as the coalgebra map (cf. Remark 3.2). Moreover, it follows almost immediately that a map α is a sequential morphism if and only if α is an S_0-coalgebra morphism. Denote by Seq the category of sequential structures and sequential morphisms; and by SeqTra the category of sequential transducers and sequential transducer morphisms.

Proposition 4.11 (i) Seq *is isomorphic to* $\mathsf{Coalg}(\mathsf{S}_0)$,

(ii) SeqTra *is isomorphic to* $\mathsf{PtCoalg}(\mathsf{S}_0)$.

As a consequence we also have that, up to isomorphism, $\mathsf{Coalg}(\mathsf{S}_0)$ is a full subcategory of $\mathsf{Coalg}(\mathsf{S})$, and Seq is a full subcategory of $\mathsf{NSubSeq}$. From now on we will use the words sequential structure and S_0-coalgebra interchangeably.

Eilenberg [7] proves that the class of prefix-preserving functions carries a final sequential structure. More precisely, he shows that for any sequential structure $\mathbb{S}$ and state q in $\mathbb{S}$, the behaviour of q, $[\![q]\!]_{\mathbb{S}} : A^* \dashrightarrow B^*$, is prefix-preserving, and $[\![q]\!]_{\mathbb{S}}(\varepsilon) = \varepsilon$. Let $\Psi \subseteq \Phi$ be the set $\Psi = \{f : A^* \dashrightarrow B^* \mid f \text{ is prefix-preserving } \& \ f(\varepsilon) = \varepsilon\}$. It is straightforward to check that Ψ is closed under taking derivatives, hence it follows that $\Psi = (\Psi, \tau\restriction_\Psi, \rho\restriction_\Psi)$, (with τ and ρ are defined as in (5)) is a subcoalgebra of Φ. By definition, $\rho(f) = f(\varepsilon) = \varepsilon$ for all $f \in \Psi$, hence Ψ is a sequential structure.

Theorem 4.12 (Eilenberg [7]) *The sequential structure Ψ is final in* Seq.

Proof. The proof can be found in Chapter XII of [7], but it also follows from the finality of Φ (Theorem 4.7). Let $\mathbb{S}$ be an arbitrary sequential structure. Viewing $\mathbb{S}$ as an object in NSubSeq, Theorem 4.7 tells us that the behaviour map $[\![\_]\!] : \mathbb{S} \to \Phi$ is the unique NSubSeq-morphism $[\![\_]\!]_{\mathbb{S}} : \mathbb{S} \to \Phi$. Since the image of $[\![\_]\!]_{\mathbb{S}}$ is contained in Ψ, it follows that $[\![\_]\!]_{\mathbb{S}} : \mathbb{S} \to \Psi$. Uniqueness in Seq follows from the uniqueness of $[\![\_]\!]_{\mathbb{S}}$ in NSubSeq and the fact that Seq is fully embedded into NSubSeq. □

4.3 Coalgebras for Differentials

The reason why subsequential structures, in general, cannot be seen as coalgebras essentially comes down to the fact that their semantics allows for asynchrony at internal computation steps, whereas the coalgebraic notion of equivalence requires synchrony at all steps. We have seen that normalisation is one way of eliminating internal asynchrony. In this section, we will see that for the class of subsequential structures which have no internal states, and therefore also no proper internal computations, there is an alternative coalgebraic representation which can be computed locally. This should be seen in contrast with normalisation which requires the computation of $\hat{\beta}$. This computation is a global fixpoint computation involving all states of the structure ([6]). Call a subsequential structure $\mathbb{S} = (Q, t, r)$ *step-by-step* if $\mathrm{dom}(r) = Q$ (i.e. all states are final). A subsequential transducer $(\mathbb{S}, i, m)$ is step-by-step if $\mathbb{S}$ is step-by-step. Let Step denote the full subcategory of CSubSeq which has step-by-step subsequential structures as its objects.

Note that step-by-step subsequential transducers need not to be normalised, hence two step-by-step subsequential transducers can realise the same function without being in perfect synchrony. The differential (cf. [7,12,4]) captures this equivalence notion, for the behaviour of step-by-step subsequential transducers on nonempty words. Let $f : A^* \dashrightarrow B^*$ be a function with prefix-closed domain. The *differential of f* is the partial function $D_f : A^+ \dashrightarrow B^{(*)}$ defined on $\mathrm{dom}(f) \setminus \{\varepsilon\}$ for all $a \in A, w \in A^*$ by

$$D_f(wa) = \overline{f(w)}.f(wa).$$

D_f describes f in a step by step manner, since for all $w = a_1 a_2 \dots a_n \in \mathrm{dom}(f)$, $n \geq 1$, we have

$$f(w) = f(\varepsilon) D_f(a_1) D_f(a_1 a_2) \dots D_f(a_1 a_2 \dots a_n). \tag{7}$$

We also have

$$D_{fw}(a) = \overline{(f\cdot w)(\varepsilon)}.(f\cdot w)(a) = \overline{\overline{f[w]}f(w)}.\overline{f[w]}f(wa) = \overline{f(w)}.f(wa) = D_f(wa) \quad (8)$$

from which it follows that for all $w = a_1a_2\dots a_n \in \mathrm{dom}(f)$, $n \geq 1$,

$$f(w) = f(\varepsilon)\, D_f(a_1)\, D_{fa_1}(a_2)\, D_{fa_1a_2}(a_3)\dots D_{fa_1\dots a_{n_1}}(a_n) \quad (9)$$

Proposition 4.13 *A function $f : A^* \dashrightarrow B^*$ is realised by a step-by-step subsequential transducer if and only if* $\mathrm{dom}(f)$ *is prefix-closed.*

Proof. Clearly, if f is realised by a step-by-step subsequential transducer, then $\mathrm{dom}(f)$ is prefix-closed. To prove the other direction, it suffices to show that the minimal realisation $\langle f\rangle_\Phi$ is step-by-step, that is, for all $w \in A^*$, if $w \in \mathrm{dom}(f)$ then $\varepsilon \in \mathrm{dom}(f\cdot w)$. But this is immediate from the definition of $f\cdot w$, since $(f\cdot w)(\varepsilon) = \overline{f[w]}.f(w)$. □

We observe that for any step-by-step subsequential structure $\mathbb{S} = (Q, o, d, r)$ and $q \in Q$, the differential of the function $[\![q]\!]_{\mathbb{S}}$ can be computed based on o, d and r. For all $a \in A$ and $w \in A^*$, letting $q_w = d(q)(w)$, we have:

$$D_{[\![q]\!]_{\mathbb{S}}}(wa) = \overline{[\![q]\!]_{\mathbb{S}}(w)}.[\![q]\!]_{\mathbb{S}}(wa) = \overline{r(q_w)}.o(q_w)(a).r(q_{wa}). \quad (10)$$

The above suggests that we can view step-by-step subsequential structures as coalgebras for the functor $\mathsf{D} : \mathsf{Set} \to \mathsf{Set}$ defined by:

$$\begin{aligned} \mathsf{D}(X) &= (\mathbf{1} + B^{(*)} \times X)^A \\ \mathsf{D}(f : X \to Y) &= (\mathbf{1} + \mathsf{Id}_{B^{(*)}} \times f)^{\mathsf{Id}_A} \end{aligned} \quad (11)$$

Definition 4.14 (differential structure) Let $\mathbb{S} = (Q, o, d, r)$ be a step-by-step subsequential structure. We define for $q \in Q$ and $a \in A$ the function $\partial_{\mathbb{S}} : Q \to (A \dashrightarrow B^{(*)})$ by $\partial_{\mathbb{S}}(q)(a) = \overline{r(q)}.o(q)(a).r(d(q)(a))$. The *differential structure of* $\mathbb{S}$ is the D-coalgebra $\mathsf{Dif}(\mathbb{S}) = (Q, \delta)$, where $\delta : Q \to (\mathbf{1} + B^{(*)} \times Q)^A$ is defined by

$$\delta(q)(a) = \begin{cases} \langle \partial_{\mathbb{S}}(q)(a), d(q)(a)\rangle & \text{if } a \in \mathrm{supp}(q), \\ \star & \text{otherwise.} \end{cases}$$

For a step-by-step subsequential transducer $\mathbb{T} = (\mathbb{S}, i, m)$ we define $\mathsf{Dif}(\mathbb{T}) = (\mathsf{Dif}(\mathbb{S}), i)$, which we will also refer to as the *differential automaton of the state i in* $\mathbb{S}$.

It should be clear that differential structures can be seen as sequential structures with output in $B^{(*)}$. The notions of behaviour and realisation from Definition 3.3 therefore apply, with identity taken in $B^{(*)}$ where appropriate. Also clear should be the fact that Seq and $\mathsf{Coalg}(\mathsf{S}_0)$ are full subcategories of Step and $\mathsf{Coalg}(\mathsf{D})$ respectively. We can extend Dif to a functor by letting Dif act as identity on morphisms.

Proposition 4.15 *The map* $\mathsf{Dif}(\_)$ *is a functor from* Step *to* $\mathsf{Coalg}(\mathsf{D})$*, and also from the category of step-by-step subsequential transducers to pointed* D*-coalgebras.*

Proof. Let $\mathbb{S}_j = (Q_j, o_j, d_j, r_j)$, $j = 1, 2$, be step-by-step subsequential structures. We first claim that a function $\alpha : Q_1 \to Q_2$ is a subsequential morphism from $\mathbb{S}_1$ to $\mathbb{S}_2$ iff $\mathrm{dom}(\alpha) = Q_1$ and for all $q \in Q_1$ the following hold:

$$
\begin{array}{ll}
\text{(supp)} & \mathrm{supp}(q) = \mathrm{supp}(\alpha(q)), \\
\text{(next)} & \forall a \in \mathrm{supp}(q) : \alpha(d_1(q)(a)) = d_2(\alpha(q))(a), \\
(\mathrm{out})_s & \forall a \in \mathrm{supp}(q) : \\
 & \overline{r_1(q)}.o_1(q)(a).r_1(d_1(q)(a)) = \overline{r_2(\alpha(q))}.o_2(\alpha(q))(a).r_2(d_2(\alpha(q))(a)), \\
(\text{term-out})_s & r_1(q).\overline{r_2(\alpha(q))} \in B^*.
\end{array}
$$

To prove this claim, assume $(\alpha, \beta) : \mathbb{S}_1 \to \mathbb{S}_2$ is a subsequential morphism (cf. Def. 3.5). Since $\mathrm{dom}(r_2) = Q_2$, the condition (acc) implies that α must be a total function. Furthermore, condition (term-out) implies that

$$\beta(q) = r_1(q).\overline{r_2(\alpha(q))}. \tag{12}$$

One can now easily verify that (out) reduces to $(\mathrm{out})_s$. Conversely, for any total function α which satisfies the above requirements, we can define $\beta : Q \to B^*$ using (12), since condition $(\text{term-out})_s$ guarantees that $\beta(q) \in B^*$. It is now straightforward to verify that (α, β) is a subsequential morphism. From the above characterisation, it clearly follows that $\alpha : \mathbb{S}_1 \to \mathbb{S}_2$ is a subsequential morphism if and only if $\alpha : \mathsf{Dif}(\mathbb{S}_1) \to \mathsf{Dif}(\mathbb{S}_2)$ is a $\mathsf{Coalg}(\mathsf{D})$-morphism.

Similarly, it is easy to see that $\alpha : \mathbb{S}_1 \to \mathbb{S}_2$ is a subsequential transducer morphism between step-by-step $(\mathbb{S}_1, i_1, m_1)$ and $(\mathbb{S}_2, i_2, m_2)$ if and only if

$$
\begin{array}{ll}
\text{(init)} & \alpha(i_1) = i_2, \\
(\varepsilon\text{-in})_s & m_1.r_1(i_1) = m_2.r_2(i_2).
\end{array}
$$

In particular, α is a morphism of pointed D-coalgebras. □

We note that Dif is not full. For step-by-step subsequential transducers this is clear, since the condition $(\varepsilon\text{-in})_s$ is not guaranteed by $\mathsf{PtCoalg}(\mathsf{D})$-morphisms. But also $\mathsf{Dif} : \mathsf{Step} \to \mathsf{Coalg}(\mathsf{D})$ is not full. The following example shows this.

Example 4.16 Consider the following two simple step-by-step subsequential structures.

It is easy to see that $\mathsf{Dif}(\mathbb{S}_1)$ and $\mathsf{Dif}(\mathbb{S}_2)$ are both isomorphic to the following

D-coalgebra:

$$\mathbb{S}: \qquad q \xrightarrow{a|\bar{b}aab} s$$

Hence if we define $\alpha(q_1) = q_2$ and $\alpha(s_1) = s_2$, then $\alpha : \mathsf{Dif}(\mathbb{S}_1) \to \mathsf{Dif}(\mathbb{S}_2)$ is a $\mathsf{Coalg}(\mathsf{D})$-morphism, however α is not a subsequential morphism between $\mathbb{S}_1$ and $\mathbb{S}_2$, since condition $(\text{term-out})_s$ fails at q_1.

Proposition 4.15 tells us that step-by-step subsequential structures can be properly viewed as coalgebras without having to normalise. We now show that D-bisimilarity in the differential structure exactly captures equality of differentials, and hence equivalence of step-by-step transducers modulo the behaviour on ε. The definition of D-bisimulation amounts to the following. Let $(Q_1, \langle \partial_1, d_1 \rangle)$ and $(Q_2, \langle \partial_2, d_2 \rangle)$ be D-coalgebras. A relation $R \subseteq Q_1 \times Q_2$ is a D-*bisimulation* if and only if for all $\langle q_1, q_2 \rangle \in R$, we have for all $a \in A$, $\partial_1(q_1)(a) = \partial_2(q_2)(a)$ (in the free group $B^{(*)}$), and $\langle d_1(q_1)(a), d_2(q_2)(a) \rangle \in R$.

Proposition 4.17 *Let $\mathbb{T}_1 = (\mathbb{S}_1, q_1, m_1)$ and $\mathbb{T}_2 = (\mathbb{S}_2, q_2, m_2)$ be step-by-step subsequential transducers, where $\mathbb{S}_1 = (Q_1, t_1, r_1)$ and $\mathbb{S}_2 = (Q_2, t_2, r_2)$. We have:*

(i) $D_{[\![q_1]\!]_1} = D_{[\![q_2]\!]_2}$ *iff* $\mathsf{Dif}(\mathbb{S}_1), q_1 \sim \mathsf{Dif}(\mathbb{S}_2), q_2$.

(ii) $[\![\mathbb{T}_1]\!] = [\![\mathbb{T}_2]\!]$ *iff* $m_1.r_1(q_1) = m_2.r_2(q_2)$ *and* $\mathsf{Dif}(\mathbb{S}_1), q_1 \sim \mathsf{Dif}(\mathbb{S}_2), q_2$.

Proof. Item (i), sketch only: ($\Rightarrow$) Show that $R = \{\langle d_1(q_1)(w), d_2(q_2)(w) \rangle \mid w \in A^*\}$ is a D-bisimulation. ($\Leftarrow$) Use (10). Item (ii): By definition, for $j = 1, 2$ and all $w = a_1 \dots a_n \in A^*$ (i.e. $n \geq 0$) we have:

$$[\![\mathbb{T}_j]\!](w) \;=\; m_j.[\![q_j]\!]_j(w) \;\overset{(7)}{=}\; m_j.r_j(q_j).D_{[\![q_j]\!]_j}(a_1) \dots D_{[\![q_j]\!]_j}(a_1 \dots a_n).$$

Item (ii) now follows easily from (i). □

The main advantage of using the differential structures to decide equivalence is probably that in order to check D-bisimilarity it is not be necessary to store a representation of the differential structures, since for any step-by-step $\mathbb{S}$, $\partial_{\mathbb{S}}$ can be computed on the fly from a representation of $\mathbb{S}$ (cf. Def. 4.14). Checking S-bisimilarity of the normalisations, however, requires explicit computation of $\hat{\beta}$, and explicit storage of $\hat{\beta}$ or the normalised structures themselves.

The final result of this section states that bisimilarity in the differential structures coincides with bisimilarity in the normalised structures. We start with the following easy observation (the proof is obtained by writing out the details).

Lemma 4.18 *For any step-by-step subsequential structure $\mathbb{S}$, $\mathsf{Dif}(\mathbb{S}) = \mathsf{Dif}(\mathsf{N}(\mathbb{S}))$.*

Theorem 4.19 *If $\mathbb{S}_1, \mathbb{S}_2$ are step-by-step subsequential structures, and q_1, q_2 are states in $\mathbb{S}_1$, respectively $\mathbb{S}_2$, then*

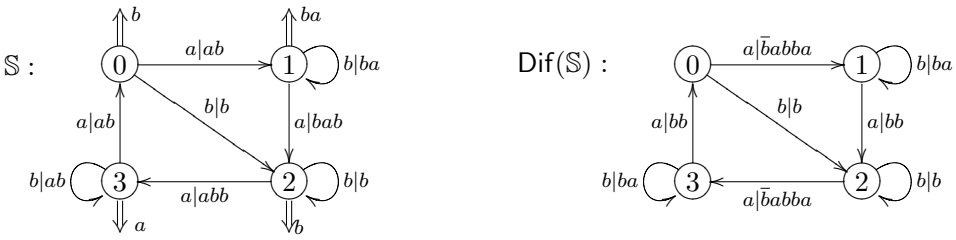
$$\mathsf{N}(\mathbb{S}_1), q_1 \sim \mathsf{N}(\mathbb{S}_2), q_2 \quad \textit{iff} \quad \mathsf{Dif}(\mathbb{S}_1), q_1 \sim \mathsf{Dif}(\mathbb{S}_2), q_2.$$

Proof. Let $\mathbb{S}_1 = (Q_1, t_1, r_1)$ and $\mathbb{S}_2 = (Q_2, t_2, r_2)$ be step-by-step subsequential structures; $q_1 \in Q_1$, $q_2 \in Q_2$, and let u_1 and u_2 denote the terminal output of

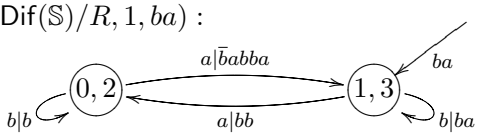
q_1 and q_2 in $\mathsf{N}(\mathbb{S}_1)$ and $\mathsf{N}(\mathbb{S}_2)$, respectively. We have: $\mathsf{N}(\mathbb{S}_1), q_1 \sim \mathsf{N}(\mathbb{S}_2), q_2$ iff (by Thm. 4.7) $[\![q_1]\!]_{\mathsf{N}(\mathbb{S}_1)} = [\![q_2]\!]_{\mathsf{N}(\mathbb{S}_2)}$ iff $[\![(\mathsf{N}(\mathbb{S}_1), q_1, \varepsilon)]\!] = [\![(\mathsf{N}(\mathbb{S}_2), q_2, \varepsilon)]\!]$ iff (Prop. 4.17.(ii) and Lemma 4.18) $u_1 = u_2$ and $\mathsf{Dif}(\mathbb{S}_1), q_1 \sim \mathsf{Dif}(\mathbb{S}_2), q_2$. It therefore suffices to show that $u_1 = u_2$ follows from $\mathsf{Dif}(\mathbb{S}_1), q_1 \sim \mathsf{Dif}(\mathbb{S}_2), q_2$. So assume $\mathsf{Dif}(\mathbb{S}_1), q_1 \sim \mathsf{Dif}(\mathbb{S}_2), q_2$. If $\mathrm{supp}(q_1) = \mathrm{supp}(q_2) = \emptyset$. then $\hat{\beta}_1(q_1) = r_1(q_1)$ and $\hat{\beta}_2(q_2) = r_2(q_2)$, hence $u_1 = \varepsilon = u_2$. Now suppose $a \in \mathrm{supp}(q_1) = \mathrm{supp}(q_2)$. We then have that $r_1(q_1) = \hat{\beta}_1(q_1)u_1$, $r_2(q_2) = \hat{\beta}_2(q_2)u_2$; and $o_1(q_1)(a) = \hat{\beta}_1(q_1)w_1$, $o_2(q_2)(a) = \hat{\beta}_2(q_2)w_2$ for some $w_1, w_2 \in A^*$. From $\mathsf{Dif}(\mathbb{S}_1), q_1 \sim \mathsf{Dif}(\mathbb{S}_2), q_2$ it follows that $\partial_{\mathbb{S}_1}(q_1)(a) = \partial_{\mathbb{S}_2}(q_2)(a)$, that is, $\overline{u_1}w_1r_1(s_1) = \overline{u_2}w_2r_2(s_2)$. Since $\hat{\beta}_1(q_1)$ and $\hat{\beta}_2(q_2)$ are maximal, it follows that $u_1 = u_2$. □

Theorem 4.19 implies that we can obtain a minimal $\mathsf{Coalg}(\mathsf{D})$-representation of a step-by-step $\mathbb{T}$ by quotienting $\mathsf{Dif}(\mathbb{T})$ with D-bisimilarity. This could be an interesting alternative to computing the minimal realisation of $[\![\mathbb{T}]\!]$ via normalisation and quotienting $\mathsf{N}(\mathbb{T})$ with S-bisimilarity. Of course, an actual implementation of a step-by-step $\mathbb{T}$ should be based on the minimal realisation of $[\![\mathbb{T}]\!]$, since the output produced by $\mathsf{Dif}(\mathbb{T})$ would have to be reduced in the free group in order to obtain the behaviour. The following example illustrates the difference between the two types of minimal realisations.

Example 4.20 (minimal realisations) Consider the following transition diagram of a step-by-step subsequential structure $\mathbb{S} = (Q, o, d, r)$, and its differential structure $\mathsf{Dif}(\mathbb{S})$:

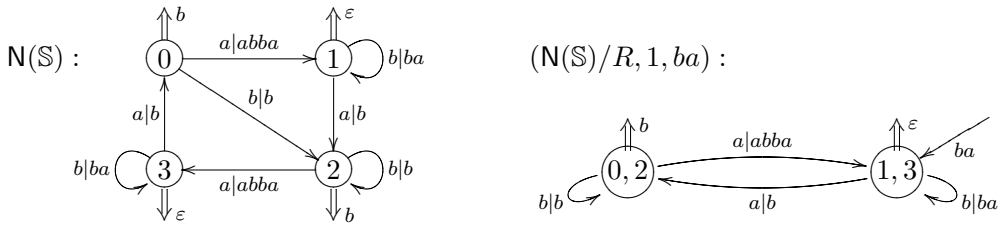


For $i \in \{0, 1, 2, 3\}$, let $f_i = [\![i]\!]_{\mathbb{S}}$. We then find that: $f_0(\varepsilon) = r(0) = b$, $f_1(\varepsilon) = r(1) = ba$, $f_2(\varepsilon) = r(2) = b$, and $f_3(\varepsilon) = r(3) = a$. It can easily be checked that the relation $R = \{\langle 0, 2\rangle; \langle 1, 3\rangle; \langle 0, 0\rangle; \langle 1, 1\rangle; \langle 2, 2\rangle; \langle 3, 3\rangle\}$ is the maximal D-bisimulation on $\mathsf{Dif}(\mathbb{S})$. This tells us that $D_{f_0} = D_{f_2}$ and $D_{f_1} = D_{f_3}$. Furthermore, since $f_0(\varepsilon) = f_2(\varepsilon)$, we can conclude that $f_0 = f_2$. We can obtain a minimal sequential transducer with output in $B^{(*)}$ which realises f_1 by quotienting $\mathsf{Dif}(\mathbb{S})$ with R and initialising this structure with the macro state containing 1, and adding initial prefix $f_1(\varepsilon) = ba$. (Similarly for the functions f_0, f_2 and f_3):



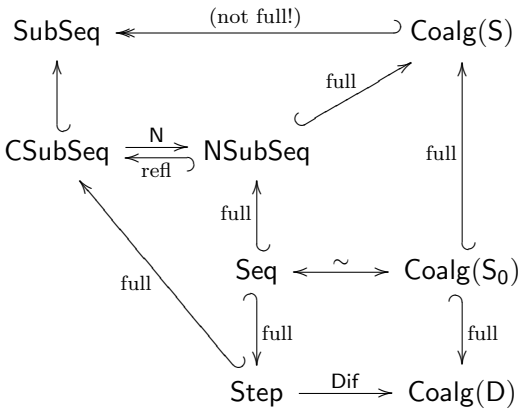
Alternatively, we could compute and minimise $\mathsf{N}(\mathbb{S})$. It can easily be verified that:

$\hat{\beta}(0) = \varepsilon$, $\hat{\beta}(1) = ba$, $\hat{\beta}(2) = \varepsilon$, $\hat{\beta}(3) = a$. We now obtain a minimal realisation of f_1 by quotienting $\mathsf{N}(\mathbb{S})$ with R, initialising with the macro state $\{1, 3\}$ and adding the initial prefix $\hat{\beta}(1) = ba$. $\mathsf{N}(\mathbb{S})$ and $(\mathsf{N}(\mathbb{S})/R, 1, ba)$ are illustrated below:



5 Summary and Discussion

In the diagram below right we provide an overview of the relationships between the various classes of subsequential structures and coalgebras that have been studied in this paper. The inclusion arrows indicate embeddings of categories; a double-headed arrow indicates that the embedding is surjective on objects; and the labels 'full', 'refl' and $\sim$ indicate whether the embedding is full, reflective or an isomorphism, respectively. The functors N (normalisation) and Dif (differential) are also indicated.



We note that the coalgebraic modelling of normalised and step-by-step subsequential structures does not capture the initial prefix. However, this does not cause any essential problems. For example, when deciding equivalence of step-by-step subsequential transducers, the initial prefix only requires one comparison (constant time) in addition to the time to decide bisimilarity equivalence.

The presence of internal states in a non-normalised subsequential transducer is the main reason why the coalgebraic notions of morphism and bisimilarity are not appropriate in such structures. Rather than transforming the subsequential transducer, one could perhaps try to look for alternative equivalence notions along the lines of weak bisimilarity. Unfortunately, weak bisimilarity in coalgebras is not very well understood, although some results may be found in [14,18]. As another possible direction for future work, we mention applying the regular expressions for polynomial coalgebras of [3] to normalised and step-by-step structures. These regular expressions provide a formal language for reasoning about and specifying coalgebras for polynomial functors, which include both normalised and step-by-step subsequential structures. As the main application of [3], we would get a symbolic synthesis method which constructs from a regular expression (of the relevant type) a normalised or step-by-step subsequential transducer. Moreover, the normalisation

procedure of [3] could perhaps be refined to deal with associativity of the join such that transducer equivalence can be determined symbolically by comparing normal forms of regular expressions.

References

[1] Adámek, J., H. Herrlich and G. Strecker, "Abstract and Concrete Categories: The Joy of Cats," J. Wiley and Sons, 1990, online version: http://katmat.math.uni-bremen.de/acc.

[2] Béal, M.-P. and O. Carton, *Determinization of transducers over finite and infinite words.*, Theoretical Computer Science **289** (2002), pp. 225–251.

[3] Bonsangue, M. M., J. J. M. M. Rutten and A. M. Silva, *Regular expressions for polynomial coalgebras*, Technical Report SEN-E0703, Centrum voor Wiskunde en Informatica (CWI) (2007).

[4] Bruyère, V. and C. Reutenauer, *A proof of Choffrut's theorem on subsequential functions*, Theoretical Computer Science **215** (1999), pp. 329–335.

[5] Choffrut, C., *A generalisation of Ginsburg and Rose's characterization of g-s-m mappings*, in: *Proceedings ICALP 1979*, LNCS **71** (1979), pp. 88–103.

[6] Choffrut, C., *Minimizing subsequential transducers: A survey*, Theoretical Computer Science **292** (2003), pp. 131–143.

[7] Eilenberg, S., "Automata, Languages and Machines (Vol. A)," Academic Press, 1974.

[8] Hansen, H., D. Costa and J. Rutten, *Synthesis of Mealy machines using derivatives*, in: *Proceedings of the 8th Workshop on Coalgebraic Methods in Computer Science (CMCS 2006), Vienna, Austria*, ENTCS **164(1)** (2006), pp. 27–45.

[9] Mohri, M., *Finite-state-transducers in language and speech processing*, Computational Linguistics **23** (1997), pp. 269–311.

[10] Pattinson, D., *Coalgebraic modal logic: Soundness, completeness and decidability of local consequence*, Theoretical Computer Science **309** (2003), pp. 177–193.

[11] Raney, G., *Sequential functions*, Journal of the ACM **5** (1958), pp. 177–180.

[12] Reutenauer, C., *Subsequential functions: Characterizations, minimization, examples.*, in: *IMYCS*, LNCS **464**, 1990, pp. 62–79.

[13] Rutten, J., *Automata and coinduction (an exercise in coalgebra)*, in: D. Sangiorgi and R. de Simone, editors, *Proceedings CONCUR'98*, LNCS **1466** (1998), pp. 194–218.

[14] Rutten, J., *A note on coinduction and weak bisimilarity for while programs*, Technical Report SEN-R9826, Centrum voor Wiskunde en Informatica (CWI) (1998).

[15] Rutten, J., *Universal coalgebra: a theory of systems*, Theoretical Computer Science **249** (2000), pp. 3–80.

[16] Schröder, L. and D. Pattinson, *Pspace bounds for rank-1 modal logics*, in: *Proceedings 21st Annual IEEE Symposium on Logic in Computer Science (LICS 2006)*, 2006, pp. 231–242, extended version to appear in ACM Transactions on Computational Logics.

[17] Schützenberger, M., *Sur un variante des fonctions sequentielles*, Theoretical Computer Science **4** (1977), pp. 47–57.

[18] Sokolova, A., E. de Vink and H. Woracek, *Weak bisimulation for action-type coalgebras*, in: *Proceedings of Category Theory and Computer Science (CTCS'04)*, 2005, pp. 211–228.

[19] Venema, Y., *Automata and fixed point logic: a coalgebraic perspective*, Information and Computation **204** (2006), pp. 637–678.