

A Categorical Framework for Learning Generalised Tree Automata

Gerco van Heerdt
University College London
gerco.heerdt@ucl.ac.uk

Tobias Kappé
University College London
tkappe@cs.ucl.ac.uk

Jurriaan Rot
University College London
Radboud University
jrot@cs.ru.nl

Matteo Sammartino
Royal Holloway, University of
London
University College London
matteo.sammartino@rhul.ac.uk

Alexandra Silva
University College London
alexandra.silva@ucl.ac.uk

Abstract

Automata learning is a popular technique used to automatically construct an automaton model from queries. Much research went into devising ad hoc adaptations of algorithms for different types of automata. The CALF project seeks to unify these using category theory in order to ease correctness proofs and guide the design of new algorithms. In this paper, we extend CALF to cover learning of algebraic structures that may not have a coalgebraic presentation. Furthermore, we provide a detailed algorithmic account of an abstract version of the popular L^* algorithm, which was missing from CALF. We instantiate the abstract theory to a large class of **Set** functors, by which we recover for the first time practical tree automata learning algorithms from an abstract framework and at the same time obtain new algorithms to learn algebras of quotiented polynomial functors.

Keywords active learning, algebras, tree automata

1 Introduction

Automata learning—automated discovery of automata models from system observations—is emerging as a highly effective bug-finding technique with applications in verification of passports [3], bank cards [1], and basic network protocols [18]. The design of algorithms for automata learning of different models is a fundamental research problem, and in the last years much progress has been made in developing and understanding new algorithms. The roots of the area go back to the 50s, when Moore studied the problem of inferring deterministic finite automata. Later, the same problem, albeit under different names, was studied by control theorists [19] and computational linguists [16]. The algorithm that caught the attention of the verification community is the one presented in Dana Angluin’s seminal paper in 1987 [8]. She proves that it is possible to infer minimal

deterministic automata in polynomial time using only so-called membership and equivalence queries. Vaandrager’s recent CACM article [27] provides an extensive review of the literature in automata learning and its applications to verification.

Angluin’s algorithm, called L^* , has served as a basis for many extensions that work for more expressive models than plain deterministic automata: I/O automata [4], weighted automata [13, 29], register automata [2, 20, 24], nominal automata [23], and Büchi automata [9]. Many of these extensions were developed independently and, though they bear close resemblance to the original algorithm, arguments of correctness and termination had to be repeated every time. This motivated Silva and Jacobs to provide a categorical understanding of L^* [21] and capture essential data structures in an abstract way in the hope of developing a generic, modular, and parametric framework for automata learning based on (co)algebra. Their early work was taken much further in van Heerdt’s master thesis [28], which then formed the basis of a wider project on developing a *Categorical Automata Learning Framework*—CALF.¹ CALF was described in the 2017 paper [30], but several problems were left open:

1. An abstract treatment of counterexamples: in the original L^* algorithm, counterexamples are a core component, as they enable refinement of the state space of the learned automaton to ensure progress towards termination.
2. The development of a full abstract learning algorithm that could readily be instantiated for a given model: in essence CALF provided only the abstract data structures needed in the learning process, but no direct algorithm.
3. Identifying suitable constraints on the abstract framework to cover interesting examples, such as tree automata, that did not fit the constraints of the original paper.

¹<http://www.calf-project.org>

In this paper, we provide answers to all the above open problems and develop CALF further to provide concrete learning algorithms for models that are algebras for a given functor, which notably include tree automata. In a nutshell, the contributions and technical roadmap of the paper are as follows. After recalling some categorical notions, the basics of L^* (Section 2), and CALF (Section 3), we provide:

1. A general treatment of counterexamples (Section 4), together with an abstract analysis of progress, that enables termination analysis of a generic algorithm.
2. A step-by-step generalisation of all components of L^* for models that are algebras of a given functor (Section 5).
3. Instantiation of the abstract algorithm to concrete categories and functors (Section 6), providing the first learning algorithm for tree automata derived abstractly.

The work in the present paper complements other recent work on abstract automata learning algorithms: Barlocco, Kupke, and Rot [12] gave an algorithm for coalgebras of a functor, whereas Urbat and Schröder [26] provided an algorithm for structures that can be represented as both algebras and coalgebras. Our focus instead is on algebras, such as tree automata, that cannot be covered by the aforementioned frameworks. A detailed comparison is given in Section 7. We conclude with directions for future work in Section 8. Omitted proofs are included in appendices A and B.

2 Preliminaries

In this section, we introduce some categorical notions that we will need later in the technical development of the paper, and we describe Angluin's original L^* algorithm. We assume some prior knowledge of category theory (categories, functors); definitions can be found in [11].

Definition 2.1 (Factorisation). An $(\mathcal{E}, \mathcal{M})$ -factorisation system on a category \mathbf{C} consists of classes of morphisms \mathcal{E} and \mathcal{M} , closed under composition with isos, such that for every morphism f in \mathbf{C} there exist $e \in \mathcal{E}$ and $m \in \mathcal{M}$ with $f = m \circ e$, and we have a unique diagonal fill-in property.

Given a morphism f , we write f^\triangleright and f^\triangleleft for the \mathcal{E} -part and \mathcal{M} -part of its factorisation, respectively.

We work in a category \mathbf{C} with finite products and co-products and a fixed factorisation system $(\mathcal{E}, \mathcal{M})$, where \mathcal{E} consists of epis and \mathcal{M} consists of monos. We fix a variator F in \mathbf{C} , that is, a functor such that there is a free F -algebra monad (T, η, μ) . We write γ_X for the F -algebra structure $FTX \rightarrow TX$, which is natural in X . Given an F -algebra (Y, γ) , we write $f^\sharp: (TX, \mu_X) \rightarrow (Y, \gamma)$ for the extension of $f: X \rightarrow Y$ and denote $y^* = \text{id}_Y^\sharp: (TY, \mu_Y) \rightarrow (Y, \gamma)$. We often implicitly apply forgetful functors. We also fix an input object I and an output object O and write F_I for the functor $I + F(-)$. Lastly, we assume that F preserves \mathcal{E} .

2.1 Abstract Automata

We now recall the abstract automaton definition from Arbib and Manes [10], which we will use in this paper, and its basic properties of accepted language and minimality.

Definition 2.2 (Automaton). An *automaton* is a tuple $\mathcal{A} = (Q, \delta, i, o)$ consisting of a state space object Q , dynamics $\delta: FQ \rightarrow Q$, initial states $i: I \rightarrow Q$, and an output $o: Q \rightarrow O$. A *homomorphism* from \mathcal{A} to $\mathcal{A}' = (Q', \delta', i', o')$ is an F -algebra morphism h from (Q, δ) to (Q', δ') —i.e., $h: Q \rightarrow Q'$ with $\delta' \circ Fh = h \circ \delta$ —such that $h \circ i = i'$ and $o' \circ h = o$.

We will use the case of deterministic automata as a running example throughout this paper.

Example 2.3. If $\mathbf{C} = \mathbf{Set}$ with the (surjective, injective) factorisation system, $F = (-) \times A$ for a finite set A , $I = 1 = \{*\}$, and $O = 2 = \{0, 1\}$, we recover deterministic automata (DAs) as automata: the state space is a set Q , the transition function is the dynamics, the initial state is represented as a morphism $1 \rightarrow Q$, and the classification of states into accepting and rejecting ones is represented by a morphism $Q \rightarrow 2$. In this case we obtain the monad $T = (-) \times A^*$, with its unit pairing an element with the empty word ε and the multiplication concatenating words. The extension of $\delta: Q \times A \rightarrow Q$ to $\delta^*: Q \times A^* \rightarrow Q$ is the usual one that lets the automaton read a word starting from a given state.

Definition 2.4 (Language). A *language* is a morphism $TI \rightarrow O$. The language accepted by an automaton $\mathcal{A} = (Q, \delta, i, o)$ is given by $\mathcal{L}_{\mathcal{A}} = TI \xrightarrow{i^\sharp} Q \xrightarrow{o} O$.

Definition 2.5 (Minimality [10]). An automaton \mathcal{A} is said to be *reachable* if its *reachability map* i^\sharp is in \mathcal{E} . \mathcal{A} is *minimal* if it is reachable and every reachable automaton \mathcal{A}' s.t. $\mathcal{L}_{\mathcal{A}} = \mathcal{L}_{\mathcal{A}'}$ admits a (necessarily unique) homomorphism to \mathcal{A} .

Example 2.6. Recall the DA setting from Example 2.3. The reachability map $i^\sharp: 1 \times A^* \rightarrow Q$ for a DA $\mathcal{A} = (Q, \delta, i, o)$ assigns to each word the state reached after reading that word from the initial state. The language $\mathcal{L}_{\mathcal{A}}: 1 \times A^* \rightarrow 2$ accepted by \mathcal{A} is precisely the language accepted by \mathcal{A} in the traditional sense. Reachability of \mathcal{A} means that for every state $q \in Q$ there exists a word that leads to q from the initial state. If this is the case, the unique homomorphism into a language-equivalent minimal automaton identifies states that accept the same language. Here, minimality is equivalent to having a minimal number of states.

2.2 The L^* algorithm

In this section, we recall Angluin's algorithm L^* , which learns the *minimal* DFA accepting a given unknown regular language \mathcal{L} . The algorithm can be seen as a game between two players: a *learner* and a *teacher*. The learner can ask two types of *queries* to the teacher:

Algorithm 1 Make table closed and consistent

```

1: function Fix( $S, E$ )
2:   while  $T$  is not closed or not consistent do
3:     if  $T$  is not closed then
4:       find  $t \in S, a \in A$  such that  $\forall s \in S. T(ta) \neq T(s)$ 
5:        $S \leftarrow S \cup \{sa\}$ 
6:     else if  $T$  is not consistent then
7:       find  $s_1, s_2 \in S, a \in A$  and  $e \in E$  such that
            $T(s_1) = T(s_2)$  and  $T(s_1a)(e) \neq T(s_2a)(e)$ 
8:        $E \leftarrow E \cup \{ae\}$ 
9:   return  $S, E$ 

```

Algorithm 2 L^* algorithm

```

1:  $S \leftarrow \{\varepsilon\}$ 
2:  $E \leftarrow \{\varepsilon\}$ 
3:  $S, E \leftarrow \text{Fix}(S, E)$ 
4: while  $\text{EQ}(\mathcal{H}_T) = c$  do
5:    $S \leftarrow S \cup \text{prefixes}(c)$ 
6:    $S, E \leftarrow \text{Fix}(S, E)$ 
7: return  $\mathcal{H}_T$ 

```

Figure 1. Angluin's L^* algorithm

1. **Membership queries:** is a word $w \in A^*$ in \mathcal{L} ?
2. **Equivalence queries:** is a hypothesis DFA \mathcal{H} correct?
That is, is $\mathcal{L}_{\mathcal{H}} = \mathcal{L}$?

The teacher answers yes or no to these queries. Moreover, negative answers to equivalence queries are witnessed by a *counterexample*—a word classified incorrectly by \mathcal{H} .

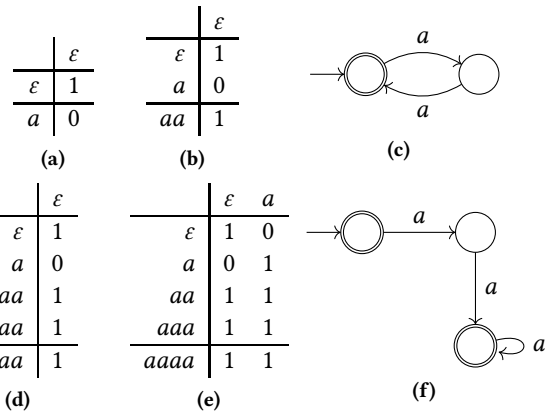
The learner gathers the results of queries into an *observation table*: a function $T: S \cup S \cdot A \rightarrow 2^E$, where $S, E \subseteq A^*$ are finite sets of words and $T(s)(e) = \mathcal{L}(se)$. The function T can be depicted as a table in which elements of S and $S \cdot A$ label rows (\cdot is pointwise concatenation) and elements of E label columns. As an example, consider the following table over the alphabet $\{a, b\}$, where $S = \{\varepsilon\}$ and $E = \{\varepsilon, b, ab\}$:

		E		
		ε	b	ab
S [ε	1	0	1
	a	0	1	0
	b	0	0	0
$S \cdot A$ [

This table approximates a language that contains ε, ab , but not a, b, bb, aab, bab . Following the visual intuition, we will refer to the part of the table indexed by S as the *top* part of the table, and the one indexed by $S \cdot A$ as the *bottom* part.

Intuitively, the content of each row labelled by a word s approximates the Myhill-Nerode equivalence class of s . This is in fact the main idea behind the construction of a hypothesis DFA \mathcal{H}_T from T : states of \mathcal{H}_T are distinct rows of T , corresponding to distinct Myhill-Nerode equivalence classes. Formally, $\mathcal{H}_T = (Q, q_0, \delta, F)$ is defined as follows:

- $Q = \{T(s) \mid s \in S\}$ is a finite set of states;
 - $F = \{T(s) \mid s \in S, T(s)(\varepsilon) = 1\}$ is the set of final states;
 - $q_0 = T(\varepsilon)$ is the initial state;
 - $\delta: Q \times A \rightarrow Q, (T(s), a) \mapsto T(sa)$ is the transition function.
- For F and q_0 to be well-defined we need ε in E and S respectively. Moreover, for δ to be well-defined we need $T(sa) \in Q$ for all $sa \in S \cdot A$, and we must ensure that the choice of s

**Figure 2.** Example run of L^* on $\mathcal{L} = \{w \in \{a\}^* \mid |w| \neq 1\}$.

to represent a row does not affect the transition. These constraints are captured in the following two properties.

Definition 2.7 (Closedness and consistency). A table T is *closed* if for all $t \in S$ and $a \in A$ there exists $s \in S$ such that $T(s) = T(ta)$. A table is *consistent* if for all $s_1, s_2 \in S$ such that $T(s_1) = T(s_2)$ we have $T(s_1a) = T(s_2a)$ for any $a \in A$.

Closedness and consistency form the core of L^* , described in Algorithm 2. The sets S and E are initialised with the empty word ε (lines 1 and 2), and extended as a closed and consistent table is built using the subroutine `Fix`, given in Algorithm 1. The main loop uses an equivalence query, denoted `EQ`, to ask the teacher whether the hypothesis induced by the table is correct. If the result is a counterexample c , the table is updated by adding all prefixes of c to S (line 5) and made closed and consistent again (line 6). Otherwise, the algorithm returns with the correct hypothesis (line 7).

Example 2.8. We now run Algorithm 2 with the target language $\mathcal{L} = \{w \in \{a\}^* \mid |w| \neq 1\}$. The algorithm starts with $S = E = \{\varepsilon\}$; the corresponding table is depicted in Figure 2a. It invokes Algorithm 1 to check closedness and

consistency of this table: it is not closed, because $T(a)$ does not appear in the top part of the table. To fix this, it adds the word a to the set S . The resulting table (Figure 2b) is closed and consistent, so the algorithm builds the hypothesis \mathcal{H} shown in Figure 2c and poses the equivalence query $\text{EQ}(\mathcal{H})$. The teacher returns the counterexample aaa , which should have been accepted, and all its prefixes are added to the set S . The next table (Figure 2d) is closed, but not consistent: $T(\varepsilon) = T(aa)$, but $T(\varepsilon \cdot a) \neq T(aa \cdot a)$. The algorithm adds $a \cdot \varepsilon = a$ to E so that $T(\varepsilon)$ and $T(aa)$ can be distinguished, as depicted in Figure 2e. The table is now closed and consistent, and the new hypothesis automaton is precisely the minimal DFA accepting \mathcal{L} (Figure 2f).

3 The abstract data structures in CALF

In this section we recall the basic notions underpinning CALF [30]: generalisations of the observation table, and the notions of closedness, consistency and hypothesis. We call the generalised table a *wrapper*:

Definition 3.1 (Wrapper). A *wrapper* for an object Q is a pair of morphisms $\mathcal{W} = (S \xrightarrow{\alpha} Q, Q \xrightarrow{\beta} P)$. We define $\tau_{\mathcal{W}} = \beta \circ \alpha$ and call the object through which it factorises $H_{\mathcal{W}}$ as in $\tau_{\mathcal{W}} = (S \xrightarrow{\tau_{\mathcal{W}}} H_{\mathcal{W}} \xrightarrow{\tau_{\mathcal{W}}} P)$.

Example 3.2. Recall the DA setting from Example 2.3 and consider a DA (Q, δ, i, o) . For each $S \subseteq A^*$ and $E \subseteq A^*$, we define $\alpha_S: S \rightarrow Q$ and $\beta_E: Q \rightarrow 2^E$ by

$$\alpha_S(w) = i^{\#}(w), \quad \beta_E(q) = (o \circ \delta^*)(q, e).$$

One can show that for all $S \subseteq A^*$ and $E \subseteq A^*$ we have

$$(\beta_E \circ \alpha_S)(s)(e) = \mathcal{L}_{\mathcal{A}}(s, se).$$

This composed function $S \rightarrow 2^E$ is precisely the upper part of the observation table with rows S and columns E in Angluin's algorithm for regular languages. The image of $\beta_E \circ \alpha_S$ is precisely the set of rows that appear in the table, which are used as states in the hypothesis, and can be obtained as $H_{\mathcal{W}}$ from the wrapper (α_S, β_E) .

Before we define hypotheses in this abstract framework, we need generalised notions of closedness and consistency.

Definition 3.3 (Closedness and consistency). Given a wrapper $\mathcal{W} = (S \xrightarrow{\alpha} Q, Q \xrightarrow{\beta} P)$, where Q is the state space of an automaton (Q, δ, i, o) , we say that \mathcal{W} is *closed* if there exist morphisms $i_{\mathcal{W}}: I \rightarrow H_{\mathcal{W}}$ and $\text{close}_{\mathcal{W}}: FS \rightarrow H_{\mathcal{W}}$ making the diagrams below commute.

$$\begin{array}{ccc} I & \xrightarrow{i} & Q \\ i_{\mathcal{W}} \downarrow & & \downarrow \beta \\ H_{\mathcal{W}} & \xrightarrow{\tau_{\mathcal{W}}} & P \end{array} \quad \begin{array}{ccc} FS & \xrightarrow{F\alpha} & FQ \xrightarrow{\delta} Q \\ \text{close}_{\mathcal{W}} \downarrow & & \downarrow \beta \\ H_{\mathcal{W}} & \xrightarrow{\tau_{\mathcal{W}}} & P \end{array}$$

Furthermore, we say that \mathcal{W} is *consistent* if there exist morphisms $o_{\mathcal{W}}: H_{\mathcal{W}} \rightarrow O$ and $\text{cons}_{\mathcal{W}}: FH_{\mathcal{W}} \rightarrow P$ making the

diagrams below commute.

$$\begin{array}{ccc} S & \xrightarrow{\tau_{\mathcal{W}}} & H_{\mathcal{W}} \\ \alpha \downarrow & & \downarrow o_{\mathcal{W}} \\ Q & \xrightarrow{o} & O \end{array} \quad \begin{array}{ccc} FS & \xrightarrow{F\tau_{\mathcal{W}}} & FH_{\mathcal{W}} \\ F\alpha \downarrow & & \downarrow \text{cons}_{\mathcal{W}} \\ FQ & \xrightarrow{\delta} Q & \xrightarrow{\beta} P \end{array}$$

If \mathcal{E} contains only regular epimorphisms, then $\tau_{\mathcal{W}}$ is the coequaliser of morphisms $f, g: X \rightarrow S$ and $F\tau_{\mathcal{W}}$ is the coequaliser of morphisms $p, q: Y \rightarrow FS$ for certain objects X and Y . A wrapper \mathcal{W} is consistent in this situation if and only if $o \circ \alpha \circ f = o \circ \alpha \circ g$ and $\beta \circ \delta \circ \alpha \circ p = \beta \circ \delta \circ \alpha \circ q$.

Example 3.4. Recall the morphisms defined in Example 3.2 and consider a DA (Q, δ, i, o) . Given $S \subseteq A^*$ and $E \subseteq A^*$, the wrapper (α_S, β_E) is closed if there exists $s \in S$ such that $(\beta_E \circ \alpha_S)(s) = (\beta_E \circ i)(*)$ and if for all $t \in S \times A$ there exists $s \in S$ such that $(\beta_E \circ \alpha_S)(s) = (\beta_E \circ \delta \circ (\alpha_S \times \text{id}_A))(t)$. The first condition holds immediately if $\varepsilon \in S$; in the second condition, $\beta_E \circ \delta \circ (\alpha_S \times \text{id}_A): S \times A \rightarrow 2^E$ represents the lower part of the observation table associated with S and E .

The wrapper (α_S, β_E) is consistent if for all $s_1, s_2 \in S$ such that $(\beta_E \circ \alpha_S)(s_1) = (\beta_E \circ \alpha_S)(s_2)$ we have $(o \circ \alpha_S)(s_1) = (o \circ \alpha_S)(s_2)$ and $(\beta_E \circ \delta \circ (\alpha_S \times \text{id}_A))(s_1) = (\beta_E \circ \delta \circ (\alpha_S \times \text{id}_A))(s_2)$. The first condition holds immediately if $\varepsilon \in E$; the second condition says that observations with the same behaviour (i.e., the same row) should lead to rows with the same content in the lower part of the table.

When a wrapper is closed and consistent, we can build the hypothesis automaton as follows.

Definition 3.5 (Hypothesis). A closed and consistent wrapper $\mathcal{W} = (S \xrightarrow{\alpha} Q, Q \xrightarrow{\beta} P)$ for the state space of an automaton (Q, δ, i, o) induces a *hypothesis* automaton $\mathcal{H}_{\mathcal{W}} = (H_{\mathcal{W}}, \delta_{\mathcal{W}}, i_{\mathcal{W}}, o_{\mathcal{W}})$, where $\delta_{\mathcal{W}}$ is the unique diagonal in the commutative square below.

$$\begin{array}{ccc} FS & \xrightarrow{F\tau_{\mathcal{W}}} & FH_{\mathcal{W}} \\ \text{close}_{\mathcal{W}} \downarrow & \delta_{\mathcal{W}} \swarrow & \downarrow \text{cons}_{\mathcal{W}} \\ H_{\mathcal{W}} & \xrightarrow{\tau_{\mathcal{W}}} & P \end{array}$$

4 Counterexamples, generalised

In this section, we provide a key missing element for the development and analysis of an abstract learning algorithm in CALF: treatment of counterexamples. In the original L^* algorithm counterexamples were used to refine the state space of the hypothesis—namely the representations of the Myhill-Nerode classes of the language being learned. A crucial property for termination, which we prove at a high level of generality in this section, is that adding counterexamples to a closed and consistent table results in a table which either fails to be closed or consistent, and hence needs to be extended. In turn, this results in progress being made in the algorithm. We will show how we can use *recursive coalgebras*

as witnesses for discrepancies—i.e., as counterexamples—between a hypothesis and the target language in our abstract approach. Let us first recall the definition.

Definition 4.1 (Recursive coalgebras). Recall that, for an input object I , $F_I = I + F(-)$. An F_I -coalgebra $\rho: S \rightarrow F_I S$ is recursive if for every algebra $x: F_I X \rightarrow X$ there is a unique morphism $x^\rho: S \rightarrow X$ making the diagram below commute.

$$\begin{array}{ccc} F_I S & \xrightarrow{F_I x^\rho} & F_I X \\ \rho \uparrow & & \downarrow x \\ S & \xrightarrow{x^\rho} & X \end{array}$$

The uniqueness property makes these morphisms commute with algebra homomorphisms. That is, if $\rho: S \rightarrow F_I S$ is recursive, (X, x) and (Y, y) are F_I -algebras, then for any F_I -algebra morphism $h: (X, x) \rightarrow (Y, y)$ we have $h \circ x^\rho = y^\rho$.

Example 4.2. A prefix-closed subset $S \subseteq A^*$ is easily equipped with a coalgebra structure $\rho: S \rightarrow 1 + S \times A$ that detaches the last letter from each non-empty word and assigns $*$ to the empty one. Such a coalgebra is recursive, where the unique map into an algebra is defined as a restricted reachability map. This prefix-closed set is used in Algorithm 2, for DA (see Example 2.3), to fix the counterexample (line 5).

Recursive F_I -coalgebras have the special property that using them as the state selector in a wrapper leads to a reachable hypothesis, as we show next. Here, and throughout the rest of this paper, we fix a target automaton $\mathcal{A}_t = (Q_t, \delta_t, i_t, o_t)$ of which we want to learn the language.

Proposition 4.3. Given a recursive $\rho: S \rightarrow F_I S$ and a closed and consistent wrapper for Q_t of the form $\mathcal{W} = ([i_t, \delta_t]^\rho, \beta)$, we have that $\tau_{\mathcal{W}}^\rho = [i_{\mathcal{W}}, \delta_{\mathcal{W}}]^\rho$ and $\mathcal{H}_{\mathcal{W}}$ is reachable.

Proof. We will first show that $\tau_{\mathcal{W}}^\rho = [i_{\mathcal{W}}, \delta_{\mathcal{W}}]^\rho$ by using the uniqueness property of the right hand side. This follows from the commutative diagram below as a result of $\tau_{\mathcal{W}}^\rho$ being a mono, together with the uniqueness property of $[i_{\mathcal{W}}, \delta_{\mathcal{W}}]^\rho$.

$$\begin{array}{ccccc} & & F_I S & \xrightarrow{F_I \tau_{\mathcal{W}}^\rho} & F_I H_{\mathcal{W}} \\ & \nearrow \rho & \downarrow F_I [i_t, \delta_t]^\rho & \searrow \textcircled{3} & \downarrow [i_{\mathcal{W}}, \delta_{\mathcal{W}}] \\ & & F_I Q_t & \xrightarrow{[i_{\mathcal{W}}, \text{close}_{\mathcal{W}}]} & H_{\mathcal{W}} \\ & & \downarrow [i_t, \delta_t] & \textcircled{2} & \downarrow \tau_{\mathcal{W}}^\rho \\ S & \xrightarrow{[i_t, \delta_t]^\rho} & Q_t & \xrightarrow{\beta} & P \\ & \searrow \tau_{\mathcal{W}}^\rho & \textcircled{4} & & \uparrow \tau_{\mathcal{W}}^\rho \\ & & H_{\mathcal{W}} & & \end{array}$$

- ① definition of $[i_t, \delta_t]^\rho$ ② closedness and def. of $i_{\mathcal{W}}$
 ③ definition of $\delta_{\mathcal{W}}$ ④ definition of τ

Now $i_{\mathcal{W}}^\# \circ [\eta_I, \gamma_I]^\rho = [i_{\mathcal{W}}, \delta_{\mathcal{W}}]^\rho = \tau_{\mathcal{W}}^\rho \in \mathcal{E}$, so $i_{\mathcal{W}}^\# \in \mathcal{E}$ by [6, Proposition 14.11 via duality]. Thus, $H_{\mathcal{W}}$ is reachable. \square

Definition 4.4 (Restricted language). Given a recursive coalgebra $\rho: S \rightarrow F_I S$, we define for any automaton $\mathcal{A} = (Q, \delta, i, o)$ its ρ -restricted language as the composition

$$\mathcal{L}_{\mathcal{A}}^\rho = S \xrightarrow{[i, \delta]^\rho} Q \xrightarrow{o} O.$$

Now we can define which recursive F_I -coalgebras are counterexamples for a given hypothesis.

Definition 4.5 (Counterexample). A closed and consistent wrapper \mathcal{W} is said to be correct up to a recursive $\rho: S \rightarrow F_I S$ if $\mathcal{L}_{\mathcal{H}_{\mathcal{W}}}^\rho = \mathcal{L}_{\mathcal{A}_t}^\rho$. A counterexample for \mathcal{W} (or $\mathcal{H}_{\mathcal{W}}$) is a recursive $\rho: S \rightarrow F_I S$ such that \mathcal{W} is not correct up to ρ .

The following result guarantees that a counterexample exists for any incorrect hypothesis.

Proposition 4.6 (Language equivalence via recursion). Given an automaton $\mathcal{A} = (Q, \delta, i, o)$, we have $\mathcal{L}_{\mathcal{A}_t} = \mathcal{L}_{\mathcal{A}}$ if and only if $\mathcal{L}_{\mathcal{A}_t}^\rho = \mathcal{L}_{\mathcal{A}}^\rho$ for all recursive $\rho: S \rightarrow F_I S$.

Proof. First assume that $\mathcal{L}_{\mathcal{A}_t}^\rho = \mathcal{L}_{\mathcal{A}}^\rho$ for all recursive $\rho: S \rightarrow F_I S$. Note that TI is the initial algebra of F_I ; thus $[\eta, \gamma]: F_I TI \rightarrow TI$ has an inverse. One easily sees that this inverse is recursive, with the corresponding unique maps into algebras being reachability maps. Thus, $\mathcal{L}_{\mathcal{A}_t} = o_t \circ i_t^\# = o \circ i^\# = \mathcal{L}_{\mathcal{A}}$.

Conversely, assume $\mathcal{L}_{\mathcal{A}_t} = \mathcal{L}_{\mathcal{A}}$. Given a recursive coalgebra $\rho: S \rightarrow F_I S$, we have that $[i_t, \delta_t]^\rho = i_t^\# \circ [\eta_I, \gamma_I]^\rho$ and $[i, \delta]^\rho = i^\# \circ [\eta_I, \gamma_I]^\rho$ by uniqueness. Thus, the diagram below commutes.

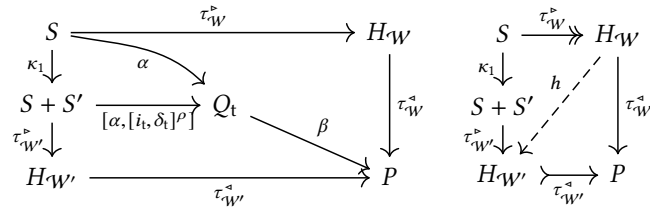
$$\begin{array}{ccccc} & & [i_t, \delta_t]^\rho & \rightarrow & Q_t & \xrightarrow{o_t} & O \\ & \nearrow [i, \delta]^\rho & \downarrow i_t^\# & & \downarrow i^\# & \nearrow o & \\ S & \xrightarrow{[\eta, \gamma]^\rho} & TI & \xrightarrow{\mathcal{L}_{\mathcal{A}_t} = \mathcal{L}_{\mathcal{A}}} & O & & \\ & \searrow [i, \delta]^\rho & \downarrow i^\# & & \downarrow o & & \end{array} \quad \square$$

Corollary 4.7 (Counterexample existence). Given a closed and consistent wrapper \mathcal{W} for Q_t , we have $\mathcal{L}_{\mathcal{H}_{\mathcal{W}}} \neq \mathcal{L}_{\mathcal{A}_t}$ if and only if there exists a counterexample for \mathcal{W} .

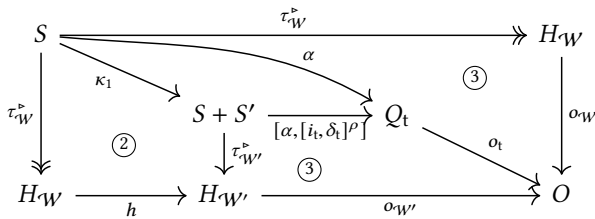
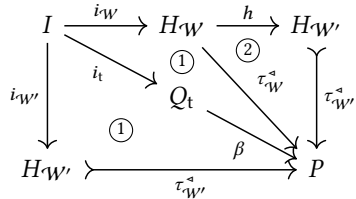
Given a counterexample, the algorithm should adjust its wrapper to accommodate the new information. The following guarantees that doing this will lead to either a closedness or a consistency defect.

Theorem 4.8 (Resolving counterexamples). Given a closed and consistent wrapper $\mathcal{W} = (S \xrightarrow{\alpha} Q_t, Q_t \xrightarrow{\beta} P)$ and a recursive coalgebra $\rho: S' \rightarrow F_I S'$, the following holds. If the wrapper $\mathcal{W}' = ([\alpha, [i_t, \delta_t]^\rho], \beta)$ is closed and consistent, then \mathcal{W} is correct up to ρ .

Proof. Since the diagram below on the left commutes, we obtain a unique diagonal $h: H_{\mathcal{W}} \rightarrow H_{\mathcal{W}'}$ on the right.

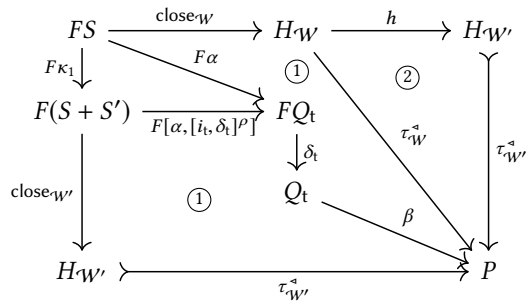


We will show that h is an automaton homomorphism. Noting that $\tau_{\mathcal{W}}^a$ is a mono and $\tau_{\mathcal{W}}^p$ is an epi, commutativity of the diagrams below shows that h commutes with the initial states ($h \circ i_{\mathcal{W}} = i_{\mathcal{W}'}$) and outputs ($o_{\mathcal{W}'} \circ h = o_{\mathcal{W}}$).



- ① closedness
- ② definition of h
- ③ consistency

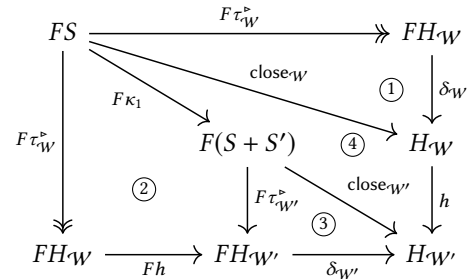
As for the transition functions, we use that $\tau_{\mathcal{W}}^a$ is a mono to show that $h \circ \text{close}_{\mathcal{W}} = \text{close}_{\mathcal{W}'} \circ F\kappa_1$ with the commutative diagram below.



- ① closedness
- ② definition of h

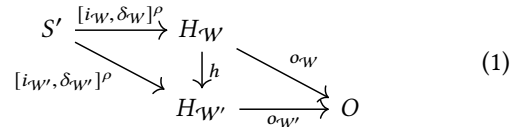
We are now ready to show that $h \circ \delta_{\mathcal{W}} = \delta_{\mathcal{W}'} \circ Fh$. This follows from commutativity of the diagram below using the

fact that $F\tau_{\mathcal{W}}^p$ is an epi.

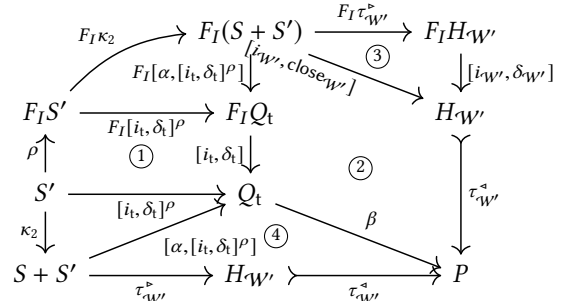


- ① definition of $\delta_{\mathcal{W}}$
- ② definition of h
- ③ definition of $\delta_{\mathcal{W}'}$
- ④ previous observation

Thus, h is an automaton homomorphism $\mathcal{H}_{\mathcal{W}} \rightarrow \mathcal{H}_{\mathcal{W}'}$. This implies in particular that $h \circ [i_{\mathcal{W}}, \delta_{\mathcal{W}}]^\rho = [i_{\mathcal{W}'}, \delta_{\mathcal{W}'}]^\rho$. It follows that the diagram below commutes.

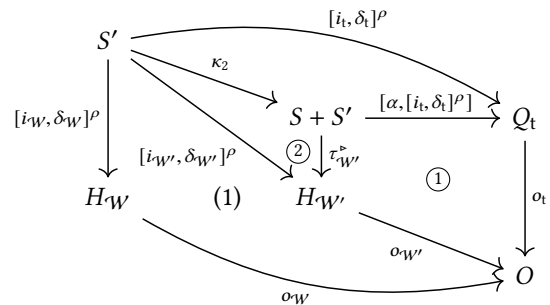


We now show that $\tau_{\mathcal{W}'}^p \circ \kappa_2 = [i_{\mathcal{W}'}, \delta_{\mathcal{W}'}]^\rho$. This follows by the uniqueness property of $[i_{\mathcal{W}'}, \delta_{\mathcal{W}'}]^\rho$ from commutativity of the diagram below, using that $\tau_{\mathcal{W}}^a$ is monic.



- ① definition of $[i_t, \delta_t]^\rho$
- ② closedness and def. of $i_{\mathcal{W}'}$
- ③ definition of $\delta_{\mathcal{W}'}$
- ④ definitions of $\tau_{\mathcal{W}}^p$ and $\tau_{\mathcal{W}}^a$

The commutative diagram below completes the proof.



- ① consistency
 - ② previous observation
-

Corollary 4.9 (Counterexample progress). *Given a closed and consistent wrapper $\mathcal{W} = (S \xrightarrow{\alpha} Q_t, Q_t \xrightarrow{\beta} P)$ and a recursive $\rho: S' \rightarrow F_I S'$ such that ρ is a counterexample for \mathcal{W} , $([\alpha, [i_t, \delta_t]^\rho], \beta)$ is either not closed or not consistent.*

5 Generalised Learning Algorithm

We are now in a position to describe our general algorithm. Similarly to L^* , which we described in Section 2.2, it is organised into two procedures: Algorithm 3, which contains the abstract procedure for making a table closed and consistent, and Algorithm 4, containing the learning iterations. These generalise the analogous procedures in L^* , Algorithm 1 and Algorithm 2, respectively.

The procedure in Algorithm 3 assumes that for each wrapper $\mathcal{W} = (S \xrightarrow{\alpha} Q_t, Q_t \xrightarrow{\beta} P)$ there exists $\beta' : Q_t \rightarrow P'$ such that (α, β') is locally consistent w.r.t. β . These are used in the steps in lines 4 and 6 to fix closedness and consistency defects. We will see later in Section 6 that existence of these maps ensuring local consistency can be proved constructively in concrete instances.

In Algorithm 4, the wrapper is initialised with trivial maps and extended to be closed and consistent using the subroutine Fix (line 1). The equivalence query for the main loop (line 2) returns a counterexample in the form of a recursive coalgebra, which is used to update the wrapper. The updated wrapper is then again passed on to the subroutine Fix (line 4) to be made closed and consistent.

A crucial point is defining what it means to resolve the “current” closedness and consistency defects. We refer to these as *local* defects, by which we mean the ones directly visible. For instance, in the DA example, the local closedness defects are the rows from the bottom part missing in the upper part, together with the empty word row if it is missing. The local consistency defects are the pairs of row labels that should be distinguished based on differing acceptance of those labels by the target, or differing rows when the labels are extended with a single symbol.

We first introduce some additional notions. We partially order the subobjects and quotients of Q_t in the usual way: for $j: J \rightarrow Q_t$ and $k: K \rightarrow Q_t$ in \mathcal{M} , we say $j \leq k$ if there exists $f: J \rightarrow K$ such that $k \circ f = j$; for $x: Q_t \rightarrow X$ and $y: Q_t \rightarrow Y$ in \mathcal{E} , we say $x \leq y$ if there exists $g: X \rightarrow Y$ such that $y = g \circ x$.

Definition 5.1 (Local closedness and consistency). Given a wrapper $\mathcal{W} = (S \xrightarrow{\alpha} Q_t, Q_t \xrightarrow{\beta} P)$ and $\alpha' : S' \rightarrow Q_t$, we say that \mathcal{W} is *locally closed* w.r.t. α' if $\alpha'^{\#} \leq \alpha^{\#}$ and there exist morphisms $i_{\mathcal{W}} : I \rightarrow H_{\mathcal{W}}$ and $\text{lclose}_{\mathcal{W}, \alpha'} : FS \rightarrow H_{\mathcal{W}}$ making the diagrams below commute.

$$\begin{array}{ccc} I & \xrightarrow{i_t} & Q_t \\ i_{\mathcal{W}} \downarrow & & \downarrow \beta \\ H_{\mathcal{W}} & \xrightarrow{\tau_{\mathcal{W}}^{\alpha}} & P \end{array} \quad \begin{array}{ccc} FS' & \xrightarrow{F\alpha'} & FQ_t \xrightarrow{\delta_t} Q_t \\ \text{lclose}_{\mathcal{W}, \alpha'} \downarrow & & \downarrow \beta \\ H_{\mathcal{W}} & \xrightarrow{\tau_{\mathcal{W}}^{\alpha}} & P \end{array}$$

Given $\beta' : Q \rightarrow P'$, we say that \mathcal{W} is *locally consistent* w.r.t. β' if $\beta^{\#} \leq \beta'^{\#}$ and there exist morphisms $o_{\mathcal{W}} : H_{\mathcal{W}} \rightarrow O$ and

$\text{lcons}_{\mathcal{W}, \beta'} : FH_{\mathcal{W}} \rightarrow P$ making the diagrams below commute.

$$\begin{array}{ccc} S & \xrightarrow{\tau_{\mathcal{W}}^{\alpha}} & H_{\mathcal{W}} \\ \alpha \downarrow & & \downarrow o_{\mathcal{W}} \\ Q_t & \xrightarrow{o_t} & O \end{array} \quad \begin{array}{ccc} FS & \xrightarrow{F\tau_{\mathcal{W}}^{\alpha}} & FH_{\mathcal{W}} \\ F\alpha \downarrow & & \downarrow \text{lcons}_{\mathcal{W}, \beta'} \\ FQ_t & \xrightarrow{\delta_t} & Q_t \xrightarrow{\beta'} P' \end{array}$$

Note that a wrapper (α, β) is closed if and only if it is locally closed w.r.t. α and consistent if and only if it is locally consistent w.r.t. β .

Example 5.2. Recall the morphisms defined in Example 3.2, for which we consider \mathcal{A}_t as the DA. Given $S, S' \subseteq A^*$ and $E \subseteq A^*$, the wrapper (α_S, β_E) is locally closed w.r.t. $\alpha_{S'}$ if (1) $S' \subseteq S$ and there exists $s \in S$ such that $(\beta_E \circ \alpha_S)(s) = (\beta_E \circ i_t)(*)$, and (2) for all $t \in S' \times A$ there exists $s \in S$ such that $(\beta_E \circ \alpha_S)(s) = (\beta_E \circ \delta_t \circ (\alpha_{S'} \times \text{id}_A))(t)$. The second condition is equivalent to the property that any row in the bottom part of the table (S', E) can be found in the top part of the table (S, E) .

Given $E' \subseteq A^*$, the wrapper (α_S, β_E) is locally consistent w.r.t. $\beta_{E'}$ if for all $s_1, s_2 \in S$ such that $(\beta_E \circ \alpha_S)(s_1) = (\beta_E \circ \alpha_S)(s_2)$ we have $(o_t \circ \alpha_S)(s_1) = (o_t \circ \alpha_S)(s_2)$ and $(\beta_{E'} \circ \delta_t \circ (\alpha_S \times \text{id}_A))(s_1) = (\beta_{E'} \circ \delta_t \circ (\alpha_S \times \text{id}_A))(s_2)$. This condition is equivalent to the property that for all $s, s' \in S$ mapping to the same row in the upper part of (S, E) , the rows for sa and $s'a$ are the same in the lower part of (S, E') for all $a \in A$.

The following shows that for each wrapper (α, β) for Q_t we can always find α' s.t. (α', β) is locally closed w.r.t. α .

Proposition 5.3. Given $\alpha : S \rightarrow Q_t$ and $\beta : Q_t \rightarrow P$, the wrapper $([\alpha, [i_t, \delta_t] \circ F_I \alpha], \beta)$ is locally closed w.r.t. α .

If α is the unique morphism induced by a recursive coalgebra, we can even simplify the above result and show that the α' found is also induced by a recursive coalgebra.

Proposition 5.4. Given a recursive $\rho : S \rightarrow F_I S$, the wrapper $([i_t, \delta_t] \circ F_I [i_t, \delta_t]^{\rho}, \beta)$ is locally closed w.r.t. $[i_t, \delta_t]^{\rho}$. Furthermore, $F_I \rho$ is also recursive and $[i_t, \delta_t] \circ F_I [i_t, \delta_t]^{\rho} = [i_t, \delta_t]^{F_I \rho}$.

Definition 5.5 (Run of the algorithm). A *run* of the algorithm is a stream of wrappers $\mathcal{W}_n = (\alpha_n, \beta_n)$ satisfying the following conditions:

1. $\alpha_0 : 0 \rightarrow Q_t$ and $\beta_0 : Q_t \rightarrow 1$ are the unique morphisms;
2. if \mathcal{W}_n is not closed, then $\beta_{n+1} = \beta_n$ and α_{n+1} is such that (α_{n+1}, β_n) is locally closed w.r.t. α_n ;
3. if \mathcal{W}_n is closed but not consistent, then $\alpha_{n+1} = \alpha_n$ and β_{n+1} is s.t. (α_n, β_{n+1}) is locally consistent w.r.t. β_n ;
4. if \mathcal{W}_n is closed and consistent and we obtain a counterexample $\rho : S \rightarrow F_I S$ for \mathcal{W}_n , then $\alpha_{n+1}^{\#} = [\alpha_n, [i_t, \delta_t]^{\rho}]^{\#}$ and $\beta_{n+1} = \beta_n$; and
5. if \mathcal{W}_n is closed and consistent and correct up to all recursive F_I -coalgebras, then $\mathcal{W}_{n+1} = \mathcal{W}_n$.

We have the following correspondence.

Algorithm 3 Make wrapper closed and consistent

```

1: function FIX( $\alpha, \beta$ )
2:   while ( $\alpha, \beta$ ) is not closed or not consistent do
3:     if ( $\alpha, \beta$ ) is not closed then
4:        $\alpha \leftarrow \alpha'$  such that ( $\alpha, \beta$ ) is loc. closed w.r.t.  $\alpha'$ 
5:     else if ( $\alpha, \beta$ ) is not consistent then
6:        $\beta \leftarrow \beta'$  such that ( $\alpha, \beta$ ) is loc. consistent w.r.t.  $\beta'$ 
7:   return  $\alpha, \beta$ 

```

Algorithm 4 Abstract automata learning algorithm

```

1:  $\alpha, \beta \leftarrow \text{FIX}(!: 0 \rightarrow Q_t, !: Q_t \rightarrow 1)$ 
2: while EQ( $\mathcal{H}_{(\alpha, \beta)}$ ) =  $\rho: S \rightarrow F_I S$  do
3:    $\alpha \leftarrow \alpha'$  s.t.  $\alpha'^a = [\alpha, [i_t, \delta_t]^\rho]^a$ 
4:    $\alpha, \beta \leftarrow \text{FIX}(\alpha, \beta)$ 
5: return  $\mathcal{H}_{(\alpha, \beta)}$ 

```

Figure 3. Generalised Learning Algorithm.

Proposition 5.6. *Algorithm 4 terminates if and only if for all runs $\{\mathcal{W}_n\}_{n \in \mathbb{N}}$ there exists $n \in \mathbb{N}$ such that $\mathcal{W}_{n+1} = \mathcal{W}_n$.*

Lemma 5.7. *Consider a run $\{\mathcal{W}_n = (\alpha_n, \beta_n)\}_{n \in \mathbb{N}}$ and $n \in \mathbb{N}$. We have $\alpha_n^a \leq \alpha_{n+1}^a$ and $\beta_{n+1}^a \leq \beta_n^a$ for all $n \in \mathbb{N}$. Moreover, if $\alpha_{n+1}^a \leq \alpha_n^a$, then $\alpha_{n+1} = \alpha_n$; if $\beta_n^a \leq \beta_{n+1}^a$, then $\beta_{n+1} = \beta_n$.*

Putting the above results together, we obtain the following theorem showing that the algorithm terminates. Moreover, it necessarily terminates with a correct automaton, for which we give conditions that guarantee minimality.

Theorem 5.8 (Termination). *If Q_t has finitely many subobject and quotient isomorphism classes, then for all runs $\{\mathcal{W}_n = (\alpha_n, \beta_n)\}_{n \in \mathbb{N}}$ there exists $n \in \mathbb{N}$ such that \mathcal{W}_n is closed and consistent and the corresponding hypothesis is correct. If \mathcal{A}_t is minimal and for all $k \in \mathbb{N}$ there exists a recursive $\rho_k: S_k \rightarrow F_I S_k$ such that $\alpha_k = [i_t, \delta_t]^{\rho_k}$, then the final hypothesis is minimal.*

Proof. We will show that $\{\mathcal{W}_n\}_{n \in \mathbb{N}}$ converges, for which it suffices to show that both $\{\alpha_n\}_{n \in \mathbb{N}}$ and $\{\beta_n\}_{n \in \mathbb{N}}$ converge. Suppose $\{\alpha_n\}_{n \in \mathbb{N}}$ does not converge. By Lemma 5.7 there exist $i_n \in \mathbb{N}$ for all $n \in \mathbb{N}$ such that $i_{n+1} > i_n$, $\alpha_{i_n}^a \leq \alpha_{i_{n+1}}^a$, and $\alpha_{i_{n+1}}^a \not\leq \alpha_{i_n}^a$ for all $n \in \mathbb{N}$. Note that isomorphic subobjects are ordered in both directions. Using transitivity of the order on subobjects we know that for all $m, n \in \mathbb{N}$ with $m \neq n$ we have that $\alpha_{i_m}^a$ and $\alpha_{i_n}^a$ are not isomorphic subobjects. This contradicts the fact that Q_t has finitely many subobject isomorphism classes. Thus, $\{\alpha_n\}_{n \in \mathbb{N}}$ must converge.

Now suppose $\{\beta_n\}_{n \in \mathbb{N}}$ does not converge. By Lemma 5.7 there exist $i_n \in \mathbb{N}$ for all $n \in \mathbb{N}$ such that $i_{n+1} > i_n$, $\beta_{i_{n+1}}^a \leq \beta_{i_n}^a$, and $\beta_{i_n}^a \not\leq \beta_{i_{n+1}}^a$ for all $n \in \mathbb{N}$. Note that isomorphic quotients are ordered in both directions. Using transitivity of the order on quotients we know that for all $m, n \in \mathbb{N}$ with $m \neq n$ the quotients $\beta_{i_m}^a$ and $\beta_{i_n}^a$ are not isomorphic. This contradicts the fact that Q_t has finitely many quotient isomorphism classes. Thus, $\{\beta_n\}_{n \in \mathbb{N}}$ must converge. We conclude that $\{\mathcal{W}_n\}_{n \in \mathbb{N}}$ converges, and by Proposition 5.6 the algorithm terminates. By definition, it does so with a correct hypothesis.

Now assume that \mathcal{A}_t is minimal and that for all $k \in \mathbb{N}$ there exists $\rho_k: S_k \rightarrow F_I S_k$ such that $\alpha_k = [i_t, \delta_t]^{\rho_k}$. Let

$n \in \mathbb{N}$ be such that $\mathcal{W}_{n+1} = \mathcal{W}_n$, which by the above we know exists, and define $\mathcal{W} = \mathcal{W}_n$. We know from Proposition 4.3 that $\mathcal{H}_{\mathcal{W}}$ is reachable. Together with correctness of $\mathcal{H}_{\mathcal{W}}$ and minimality of \mathcal{A}_t there exists a unique automaton homomorphism $h: \mathcal{H}_{\mathcal{W}} \rightarrow \mathcal{A}_t$. We show that $\beta_n \circ h = \tau_{\mathcal{W}}^a$ with the commutative diagram below, where we precompose with the epi $\tau_{\mathcal{W}}^a$ and use that automaton homomorphisms commute with restricted reachability maps.

$$\begin{array}{ccc}
S_n & \xrightarrow{\tau_{\mathcal{W}}^a} & H_{\mathcal{W}} \\
\tau_{\mathcal{W}}^a = [i_{\mathcal{W}}, \delta_{\mathcal{W}}]^{\rho_n} \downarrow & \searrow \alpha_n = [i_t, \delta_t]^{\rho_n} & \downarrow \tau_{\mathcal{W}}^a \\
H_{\mathcal{W}} & \xrightarrow{h} & Q_t \xrightarrow{\beta_n} P_n
\end{array}$$

It follows that $h \in \mathcal{M}$ [6, Proposition 14.11]. Being an automaton homomorphism, h commutes with the reachability maps: $h \circ i_{\mathcal{W}}^\# = i_t^\#$. Because $i_{\mathcal{W}}^\# \in \mathcal{E}$ and $i_t^\# \in \mathcal{E}$, we have $h \in \mathcal{E}$ [6, Proposition 14.9 via duality]. Since $\mathcal{E} \cap \mathcal{M}$ contains only isomorphisms [6, Proposition 14.6], it follows that h is an isomorphism of automata and therefore that the hypothesis is minimal. \square

Computability. To be able to apply the algorithm in a concrete case one needs the following ingredients. For each automaton \mathcal{A}_t having finitely many subobject and quotient isomorphism classes there needs to be a run $\{\mathcal{W}_n\}_{n \in \mathbb{N}}$ such that we have a family of sets $\{D_n\}_{n \in \mathbb{N}}$ of data such that both D_0 and the function $D_n \mapsto D_{n+1}$ are computable, and such that we can determine from D_n whether \mathcal{W}_n is closed and consistent. Furthermore, we need to be able to compute the hypothesis \mathcal{H}_n whenever \mathcal{W}_n is closed and consistent. Usually D_n consists of a representation of the maps

$$\alpha_n \circ \beta_n \quad \beta_n \circ \delta_t \circ F\alpha_n \quad \beta_n \circ i_t \quad o_t \circ \alpha_n.$$

One then needs to show how to find local closedness and local consistency witnesses using these maps, and that the teacher can always choose a counterexample such that the data in the step after adding the counterexample can be computed. The teacher must also be restricted to return only such suitable counterexamples, rather than arbitrary ones.

6 Example: Generalised Tree Automata

In this section we instantiate the above development to a wide class of **Set** endofunctors. This yields an abstract algorithm for *generalised* tree automata—i.e., automata accepting sets of trees, possibly subject to equations—which include bottom-up tree automata and unordered tree automata. These are examples that were not in scope of any of the existing abstract learning frameworks in the literature.

We first introduce the running examples for this section.

Example 6.1 (Tree automata). Let Γ be a ranked alphabet, i.e., a finite set where $\gamma \in \Gamma$ comes with $\text{arity}(\gamma) \in \mathbb{N}$. The set of Γ -trees over a finite set of leaf symbols I is the smallest set $T_\Gamma(I)$ such that $I \subseteq T_\Gamma(I)$, and for all $\gamma \in \Gamma$ we have that $t_1, \dots, t_{\text{arity}(\gamma)} \in T_\Gamma(I)$ implies $(\gamma, t_1, \dots, t_{\text{arity}(\gamma)}) \in T_\Gamma(I)$. The alphabet Γ gives rise to the polynomial functor $FX = \coprod_{\gamma \in \Gamma} X^{\text{arity}(\gamma)}$. The corresponding free F -algebra monad is precisely T_Γ , where the unit turns elements into leaves, and the multiplication flattens nested trees into a tree. A bottom-up deterministic tree automaton is then an automaton over F with a finite input set I and output set $O = 2$.

Example 6.2 (Unordered tree automata). Consider the finite powerset functor $\mathcal{P}_f: \mathbf{Set} \rightarrow \mathbf{Set}$, mapping a set to its finite subsets. The corresponding free \mathcal{P}_f -monad maps a set X to the set of finitely-branching unordered trees with nodes in X . Automata over \mathcal{P}_f , with output set $O = 2$ and finite I , accept sets of such trees. Note that unordered trees can be seen as trees over a ranked alphabet $\Gamma = \{\bar{s}_i \mid i \in \mathbb{N}\}$, where $\text{arity}(\bar{s}_i) = i$, satisfying equations that collapse duplicate branches and identify lists of branches up to permutations.

Automata in these examples are algebras for endofunctors with the following properties: they are *strongly finitary* [7]—i.e., they preserve finite sets; and they preserve weak pullbacks. In this section we will show that the conditions listed at the end of Section 5 can be satisfied for all automata over strongly finitary, weak-pullback-preserving endofunctors F , with a finite input set I . For the rest of this section we assume these properties for F and I .

In particular, in Section 6.1 we instantiate wrappers to ones with a specific format, which make use of *contexts* to generalise string concatenation to trees, and we show how these wrapper can be computed, whereupon we develop procedures for local closedness (Section 6.2) and local consistency (Section 6.3). Section 6.4 covers the representation of the associated hypotheses, and we conclude by identifying a set of suitable (finite) counterexamples in Section 6.5.

6.1 Contextual wrappers

Denote by 1 the set $\{\square\}$. Given $x \in X$ for any set X , we write 1_x for the function $1 \rightarrow X$ that assigns x to \square . Note that for all functions $f: X \rightarrow Y$ we have

$$1_{f(x)} = f \circ 1_x. \quad (2)$$

We use the set 1 to define the set of *contexts* $T(I+1)$, where the *holes* \square occurring in a context $c \in T(I+1)$ can be used to plug in further data such as another context or a tree, e.g., in the case of Example 6.1. In fact, it is well known that $T(I+(-))$ forms a monad with unit $\hat{\eta}_X = T\kappa_2 \circ \eta_X: X \rightarrow T(I+X)$ and multiplication $\hat{\mu}_X: T(I+T(I+X)) \rightarrow T(I+X)$ [22].

We now introduce a class of wrappers where, intuitively, contexts are used to distinguish inequivalent states.

Definition 6.3 (Contextual wrappers). A contextual wrapper is a wrapper (α_S, β_E) for Q_t where:

- $\alpha_S: S \rightarrow Q_t$, for $S \subseteq TI$, is the restriction of the reachability map to S ;
- $\beta_E: Q_t \rightarrow O^E$ is the function given by

$$\beta_E(q) = \alpha_t \circ [i_t, 1_q]^\sharp.$$

for $E \subseteq T(I+1)$.

Example 6.4. In the case of DA, contextual wrappers are essentially those of Example 3.2. In fact, α_S is the restriction of $i^\sharp: A^* \cong 1 \times A^* \rightarrow Q_t$ to $S \subseteq A^*$. For β_E , a bit of care is required due to the generality of contexts. We have $E \subseteq \{*, \square\} \times A^*$, and

$$\beta_E(q)(x) = \begin{cases} (\alpha_t \circ \delta_t^*)(q, e) & \text{if } x = (\square, e) \\ (\alpha_t \circ \delta_t^*)(i_t(*), e) & \text{if } x = (*, e). \end{cases}$$

Note that the second case is not useful as a context distinguishing two states, because it does not depend on q .

Let us consider contextual wrappers for Example 6.1. We have that $S \subseteq T_\Gamma(I)$ is a set of Γ -trees over I , and $E \subseteq T_\Gamma(I+1)$ is formed by contexts, i.e., Γ -trees where a special leaf \square may occur. The function $\alpha_S(t)$ is the state reached after reading the tree t , and $\beta_E(q)(t)$ can be seen as a generalisation of the DA case: it is the output of the state reached via δ^* after replacing every occurrence of \square with q and $x \in I$ with $i(x)$ in t . Therefore $(\alpha_S \circ \beta_E): S \rightarrow O^E$ is the upper part of an observation table where rows are labelled by trees, columns by contexts, and rows are computed by plugging labels into each column context and querying the language. When E contains only contexts with exactly one instance of \square , this corresponds precisely to the observation tables of [14, 17].

We now show how to compute the morphisms induced by a wrapper that are used in the definition of closedness, consistency, and the hypothesis. In particular, we show that they can be computed by querying the language $\mathcal{L}_{\mathcal{A}}$.

Proposition 6.5 (Computing wrapper morphisms). *Given $S \subseteq TI$ with inclusion $j: S \rightarrow TI$ and $E \subseteq T(I+1)$, we have*

$$\begin{aligned} \beta_E \circ \alpha_S &: S \rightarrow O^E \\ (\beta_E \circ \alpha_S)(s) &= \mathcal{L}_{\mathcal{A}_t} \circ \mu_I \circ T[\eta_I, 1_s] \\ \beta_E \circ \delta_t \circ F\alpha_S &: FS \rightarrow O^E \\ (\beta_E \circ \delta_t \circ F\alpha_S)(f) &= \mathcal{L}_{\mathcal{A}_t} \circ \mu_I \circ T[\eta_I, \gamma_I \circ Fj \circ 1_f] \\ \beta_E \circ i_t &: I \rightarrow O^E \\ (\beta_E \circ i_t)(x) &= \mathcal{L}_{\mathcal{A}_t} \circ T[\text{id}_I, 1_x] \\ o_t \circ \alpha_S &: S \rightarrow O \\ (o_t \circ \alpha_S)(s) &= \mathcal{L}_{\mathcal{A}_t}(s). \end{aligned}$$

Example 6.6. As in the DA case, the maps of Proposition 6.5 correspond to the observation table. The proposition tells us how they can be computed by querying the language.

For bottom-up tree automata:

- $\beta_E \circ \alpha_S$ is the upper part of the observation table, as explained in Example 6.4;
- $\beta_E \circ \delta_t \circ F\alpha_S$ is the bottom part of the table. In fact, in this case the successor rows for S are labelled by $FS = \coprod_{\gamma \in \Gamma} S^{\text{arity}(\gamma)}$, i.e., by trees obtained by adding a new root symbol to those from S . Successor rows are then computed by plugging these trees into the contexts E , and querying the language. Note that this requires using the map γ to convert the additional root and its arguments into a tree before they are plugged into a context.
- $\beta_E \circ i_t$ returns the leaf rows, i.e., those labelled by the leaf symbols I ;
- $o_t \circ \alpha_S$ queries the language for each row label.

For unordered tree automata the maps are similar. The key difference is that now rows are labelled by trees and contexts up to equations. As a consequence, there is just one successor row for each set of trees in S , whereas in the previous case we have one successor row for each symbol $\gamma \in \Gamma$ and arity(γ)-list of trees from S .

6.2 Witnessing local closedness

We now show how the general notion of local closedness can be concretely instantiated for **Set** automata.

Lemma 6.7 (Local closedness for **Set** automata). *Given $S, S' \subseteq TI$ and $E \subseteq T(I+1)$ such that $S \subseteq S'$, $(\alpha_{S'}, \beta_E)$ is locally closed w.r.t. α_S if there exist $k: I \rightarrow S'$ and $\ell: FS \rightarrow S'$ such that*

$$\alpha_{S'} \circ k = i_t \quad \alpha_{S'} \circ \ell = \delta_t \circ F\alpha_S.$$

Example 6.8. For bottom-up tree automata, local closedness holds if the table (S', E) already contains each leaf row (left equation), and it contains every successor row for S , namely $FS = \coprod_{\gamma \in \Gamma} S^{\text{arity}(\gamma)}$ (right equation).

For unordered tree automata the condition is similar, and now involves successor trees in $\mathcal{P}_f(S)$.

The following proposition guarantees that we can always extend S to make the wrapper locally closed. Moreover, this can be done so that α'_S forms a proper contextual wrapper, i.e., it is again a restricted reachability map.

Proposition 6.9. *Given finite $S \subseteq TI$ and $E \subseteq T(I+1)$, there exists a finite $S' \subseteq TI$ such that $(\alpha_{S'}, \beta_E)$ is locally closed w.r.t. α_S . If there exists a recursive $\rho: S \rightarrow F_I S$ such that $\alpha_S = [i_t, \delta_t]^\rho$, then there exists a recursive $\rho': S' \rightarrow F_I S'$ such that $\alpha_{S'} = [i_t, \delta_t]^{\rho'}$.*

Proof. Let $j: S \rightarrow TI$ be the inclusion map and define

$$S' = S \cup \{\eta_I(x) \mid x \in I\} \cup \{\gamma_I \circ Fj(x) \mid x \in FS\}.$$

We choose $k: I \rightarrow S'$ and $\ell: FS \rightarrow S'$ by setting

$$k(x) = \eta_I(x) \quad \ell(x) = (\gamma_I \circ Fj)(x)$$

Note that k and ℓ are well-defined by construction of S' . Using the definitions of $\alpha_{S'}$ and k , we can then derive that

$$(\alpha_{S'} \circ k)(x) = i_t^\#(k(x)) = i_t^\#(\eta_I(x)) = i_t(x)$$

Furthermore, we find that

$$\begin{aligned} (\alpha_{S'} \circ \ell)(x) &= i_t^\#(\gamma_I(Fj(x))) && \text{(def. of } \ell) \\ &= \delta_t(F(i_t^\#(Fj(x)))) \\ & \quad (i_t^\# \text{ is an } F\text{-algebra homomorphism}) \\ &= \delta_t(F(i_t^\# \circ j)(x)) \\ &= (\delta_t \circ F\alpha_S)(x) && \text{(def. of } \alpha_S) \end{aligned}$$

Hence $(\alpha_{S'}, \beta_E)$ is locally closed w.r.t. α_S , by Lemma 6.7.

Given a recursive $\rho: S \rightarrow F_I S$ such that $\alpha_S = [i_t, \delta_t]^\rho$, define $\rho': S' \rightarrow F_I S'$ by

$$\rho'(s) = \rho(s) \quad \rho'([\eta_I, \gamma_I] \circ F_I j)(x) = F_I j(x).$$

Since both ρ and $[\eta_I, \gamma_I]^{-1}$ are recursive, so is ρ' \square

Example 6.10. To better understand this proposition, it is worth describing what recursive coalgebras are for the automata of Examples 6.1 and 6.2. For bottom-up tree automata, they are coalgebras $\rho: S \rightarrow \coprod_{\gamma \in \Gamma} S^{\text{arity}(\gamma)} + I$ satisfying suitable conditions. Prefix-closed subsets of $T_\Gamma(I)$ are sets of trees closed under taking subtrees. Every prefix-closed S can be made into a recursive coalgebra that returns the root symbol and its arguments, if applied to a tree of non-zero depth, and a leaf otherwise. For unordered tree automata, $\rho: S \rightarrow \mathcal{P}_f S + I$ will just return the set of subtrees or a leaf.

Note that the above proof leads to a rather inefficient algorithm that adds all successor rows to the table to make it locally closed. For instance, in the case of Example 6.4, it adds rows obtained by adding a new root symbol to existing row labels in all possible ways, for each symbol in the alphabet. One may optimise the algorithm by adding instead only missing rows, and one instance of each.

6.3 Witnessing local consistency

Analogously to the previous section, we now show how local consistency can be concretely instantiated for **Set** automata.

Lemma 6.11 (Local consistency for **Set** automata). *Let $S \subseteq TI$ and $E \subseteq E' \subseteq T(I+1)$, with S finite. Furthermore, suppose that for $s, s' \in S$ with $(\beta_{E'} \circ \alpha_S)(s) = (\beta_{E'} \circ \alpha_S)(s')$ we have*

$$(\alpha_t \circ \alpha_S)(s) = (\alpha_t \circ \alpha_S)(s')$$

$$\beta_E \circ \delta_t \circ F(\alpha_S \circ [\text{id}_S, 1_s]) = \beta_E \circ \delta_t \circ F(\alpha_S \circ [\text{id}_S, 1_{s'}])$$

Then $\mathcal{W} = (\alpha_S, \beta_E)$ is locally consistent w.r.t. β_E .

Example 6.12. For bottom-up tree automata, local consistency amounts to require the following for the table for (S, E') . For every pair of trees $s, s' \in S$ such that the corresponding rows are equal we must have:

- both s and s' are either accepted or rejected;
- successor rows obtained by extending s and s' in the same way are equal. Formally, comparable extensions of s and s' are obtained by plugging them into the same “one-level” context from $F(S+1) = \coprod_{\gamma \in \Gamma} (S + \{\square\})^{\text{arity}(\gamma)}$.

For unordered-tree automata, we need to compare s and s' only when they are equationally inequivalent. Note that one-level contexts are also up to equations, which means that the position of the hole in the context is irrelevant for computing extensions of s and s' .

The following ensures that we can always make the wrapper locally consistent by finding a suitable finite E' .

Proposition 6.13. *Given finite $S \subseteq TI$ and $E \subseteq T(I+1)$, the set $E' \subseteq T(I+1)$ is defined as*

$$E' = E \cup \{(\eta_{I+1} \circ \kappa_2)(\square)\} \\ \cup \{(\hat{\mu}_1 \circ T(\text{id}_I + c_x))(e) \mid e \in E, x \in F(S+1)\},$$

where $c_x: 1 \rightarrow T(I+1)$, with $c_x = \gamma_{I+1} \circ F[T\kappa_1 \circ j, \hat{\eta}_1] \circ 1_x$, where $j: S \rightarrow TI$ is set inclusion. It holds that E' is finite and $(\alpha_S, \beta_{E'})$ is locally consistent w.r.t. β_E .

We remark that the above definition of E' results in a highly inefficient procedure. One can optimise it by incrementally adding elements of the proposed E' to E that distinguish rows not distinguished by the current elements of E and that need to be added in order to satisfy the conditions of Lemma 6.11.

6.4 Representing hypotheses

Given finite $S \subseteq TI$ and $E \subseteq T(I+1)$, we consider the wrapper $\mathcal{W} = (\alpha_S, \beta_E)$. Assuming closedness and consistency, the state space of the associated hypothesis is given by the image of $\beta_E \circ \alpha_S: S \rightarrow O^E$. Since S and E are finite

we can represent this function and thus compute its image. The structure of the hypothesis is defined by

$$i_{\mathcal{H}_W}(x) = (\beta_E \circ i_t)(x) \quad o_{\mathcal{H}_W}(\tau_W^*(s)) = (\alpha_t \circ \alpha_S)(s)$$

$$\delta_{\mathcal{H}_W}(F(\tau_W^*(x))) = (\beta_E \circ \delta_t \circ F\alpha_S)(x).$$

We know from Proposition 6.5 how to compute those functions via membership queries. This can be done in finite time because E is finite.

The hypothesis automaton for bottom-up and unordered tree automata, as in the DA case (see Example 3.2), is obtained by taking distinct rows as states. See Example 6.6 for the description of the hypothesis input, output and transition maps for those automata types.

6.5 Finite counterexamples

Finally, we refine Proposition 4.6 to show that the teacher can always pick a finite counterexample.

Proposition 6.14 (Language equivalence via finite recursion). *Given an automaton $\mathcal{A} = (Q, \delta, i, o)$, we have $\mathcal{L}_{\mathcal{A}_t} = \mathcal{L}_{\mathcal{A}}$ if and only if $\mathcal{L}_{\mathcal{A}_t}^\rho = \mathcal{L}_{\mathcal{A}}^\rho$ for all recursive $\rho: S \rightarrow F_I S$ such that S is finite.*

Proof. Suppose that for all recursive coalgebras $\rho: S \rightarrow F_I S$ such that S is finite we have $\alpha_t \circ [i, \delta]^\rho = o \circ [i, \delta]^\rho$. Given $t \in TI$, note that $(TI, [\eta_I, \gamma_I])$ is the initial algebra of functor F_I , which by being finitary is also the colimit of the initial sequence of F_I [5] and hence isomorphic to $(\bigcup_{n \in \mathbb{N}} F_I^n \emptyset, a)$ for an initial algebra structure $a: F_I(\bigcup_{n \in \mathbb{N}} F_I^n \emptyset) \rightarrow \bigcup_{n \in \mathbb{N}} F_I^n \emptyset$. Let $\phi: (TI, [\eta_I, \gamma_I]) \rightarrow (\bigcup_{n \in \mathbb{N}} F_I^n \emptyset, a)$ be the isomorphism. There exists $n \in \mathbb{N}$ such that $\phi(t) \in F_I^n \emptyset$. The set $F_I^n \emptyset$ is finite by F_I preserving finite sets and the carrier of a recursive coalgebra $\rho: F_I^n \emptyset \rightarrow F_I^{n+1} \emptyset$ by [15, Proposition 6], with $a^\rho: F_I^n \emptyset \rightarrow \bigcup_{n \in \mathbb{N}} F_I^n \emptyset$ being the inclusion. Then $S = \{\phi^{-1}(x) \mid x \in F_I^n \emptyset\}$ is also finite and the carrier of a recursive coalgebra $\rho': S \rightarrow F_I S$, with $[\eta_I, \gamma_I]^{\rho'}: S \rightarrow TI$ being the inclusion. Moreover, $t \in S$. Thus,

$$\begin{aligned} \mathcal{L}_{\mathcal{A}_t}(t) &= (\mathcal{L}_{\mathcal{A}_t} \circ [\eta_I, \gamma_I]^{\rho'})(t) \\ &= (\alpha_t \circ i_t^\# \circ [\eta_I, \gamma_I]^{\rho'})(t) && \text{(definition of } \mathcal{L}_{\mathcal{A}_t}) \\ &= (\alpha_t \circ [i, \delta]^{\rho'})(t) && (i_t^\# \text{ is an } F_I\text{-algebra homomorphism}) \\ &= \mathcal{L}_{\mathcal{A}}^{\rho'}(t) && \text{(definition of } \mathcal{L}_{\mathcal{A}}^{\rho'}) \\ &= \mathcal{L}_{\mathcal{A}}^{\rho'}(t) && \text{(assumption)} \\ &= (o \circ [i, \delta]^{\rho'})(t) && \text{(definition of } \mathcal{L}_{\mathcal{A}}^{\rho'}) \\ &= (o \circ i^\# \circ [\eta_I, \gamma_I]^{\rho'})(t) && (i^\# \text{ is an } F_I\text{-algebra homomorphism}) \\ &= (\mathcal{L}_{\mathcal{A}} \circ [\eta_I, \gamma_I]^{\rho'})(t) && \text{(definition of } \mathcal{L}_{\mathcal{A}}) \\ &= \mathcal{L}_{\mathcal{A}}(t). \end{aligned}$$

The converse follows from Proposition 4.6. \square

Corollary 6.15 (Finite counterexample existence). *Given a closed and consistent wrapper \mathcal{W} for Q_t , we have $\mathcal{L}_{\mathcal{H}_W} \neq \mathcal{L}_{\mathcal{A}_t}$ if and only if there exists a counterexample $\rho: S \rightarrow F_I S$ for \mathcal{W} such that S is finite.*

Example 6.16. Recall from Example 6.10 that finite recursive coalgebras for bottom-up (resp. unordered) tree automata are coalgebras $\rho: S \rightarrow \coprod_{y \in \Gamma} S^{\text{arity}(y)} + I$ (resp. $\rho: S \rightarrow \mathcal{P}_f S + I$). Therefore, finite counterexamples are recursive coalgebras of this form such that S is finite or, more concretely, a finite subtree-closed set of trees.

7 Related work

This paper proceeds in the line of work on categorical automata learning started in [21], and further developed in the CALF framework [30, 31]. CALF provides abstract definitions of closedness, consistency and hypothesis, and several techniques to analyse and guide development of concrete learning algorithms. CALF operates at a high level of abstraction and does not include an explicit learning algorithm. We discuss two further recent categorical approaches to learning, which make stronger assumptions than in CALF that allow for the definition of concrete algorithms. The present paper can be thought of as a third such approach.

Barlocco et al. [12] proposed an abstract algorithm for learning coalgebras. It stipulates the tests to be formed by an abstract version of coalgebraic modal logic. On the one hand, the notion of wrapper and closedness from CALF essentially instantiate to that setting; on the other hand, the combination of logic and coalgebra is precisely what enables to define an actual learning algorithm in [12]. The current work focus on algebras rather than coalgebras, and is orthogonal. In particular, it covers (bottom-up) tree automata, which is outside the scope of [12].

Urbat and Schröder have recently proposed another categorical approach to automata learning [26], which—similarly to the work of Barlocco et al.—makes stronger assumptions than in CALF in order to define a learning algorithm. Their work focuses primarily on automata, assuming that the systems of interest can be viewed both as algebras and coalgebras, and the generality comes from allowing to instantiate these in various categories. Moreover, it allows covering algebraic recognisers in certain cases, through a reduction to automata over a carefully constructed alphabet; this (orthogonal) extension allows covering, e.g., ω -languages as well as tree languages. However, the reduction to automata makes this process quite different than the approach to tree learning in the present paper: it makes use of an automaton over all (flat) contexts, yielding an infinite alphabet, and therefore the algorithmic aspect is not clear. The extension to an actual algorithm for learning tree automata is mentioned as future work in [26]. In the present paper, this is achieved by learning algebras directly.

Concrete algorithms for learning tree automata and languages have appeared in the literature [14, 17, 25]. The inference of regular tree languages using membership and equivalence queries appeared in [17], who extended earlier work of Sakakibara [25]. Later, [14] provided a learning algorithm for regular tree automata using only membership queries. The instantiated algorithm in our paper has elements (such as the use of contexts) close to the concrete algorithms. However, the focus of the present paper is on presenting an algebraic framework that can effectively be instantiated to recover such concrete algorithms in a modular and canonical fashion, with proofs of correctness and termination stemming from the general framework.

8 Future Work

The work in this paper makes use of the free monad of a functor F in the formulation of the generalised learning algorithm and hence can only deal with quotienting in a restricted setting, namely by flat equations in the presentation of F . This excludes more complex models such as that of pomset automata, which feature languages of words modulo complex equations. Such richer equations can be captured by monads that are not necessarily free. It remains an open challenge to extend the present algorithm to this richer setting.

Another direction for future work is to extend the framework with side-effects, encoded by a monad, in the style of [31]. This would enable learning more compact automata—albeit with richer, monadic, transitions—representing languages and, as a concrete instance, provide an active learning algorithm for weighted tree automata.

Acknowledgments

This work was partially supported by the ERC Starting Grant ProFoundNet (grant code 679127), a Leverhulme Prize (PLP-2016-129), a Marie Curie Fellowship (grant code 795119), and the EPSRC Standard Grant CLever (EP/S028641/1).

References

- [1] Fides Aarts, Joeri de Ruiter, and Erik Poll. 2013. Formal Models of Bank Cards for Free. In *ICST*. 461–468. <https://doi.org/10.1109/ICSTW.2013.60>
- [2] Fides Aarts, Paul Fiterău-Broștean, Harco Kuppens, and Frits W. Vaandrager. 2015. Learning Register Automata with Fresh Value Generation. In *ICTAC*. 165–183. https://doi.org/10.1007/978-3-319-25150-9_11
- [3] Fides Aarts, Julien Schmaltz, and Frits W. Vaandrager. 2010. Inference and Abstraction of the Biometric Passport. In *ISO LA*. 673–686. https://doi.org/10.1007/978-3-642-16558-0_54
- [4] Fides Aarts and Frits W. Vaandrager. 2010. Learning I/O Automata. In *CONCUR*. 71–85. https://doi.org/10.1007/978-3-642-15375-4_6
- [5] Jiří Adámek. 1974. Free algebras and automata realizations in the language of categories. *Commentationes Mathematicae Universitatis Carolinae* 15, 4 (1974), 589–602.
- [6] Jiří Adámek, Horst Herrlich, and George E. Strecker. 2009. *Abstract and Concrete Categories - The Joy of Cats*. Dover Publications.

- [7] Jiří Adámek, Stefan Milius, and Jiří Velebil. 2003. Free iterative theories: a coalgebraic view. *Mathematical Structures in Computer Science* 13, 2 (2003), 259–320. <https://doi.org/10.1017/S0960129502003924>
- [8] Dana Angluin. 1987. Learning Regular Sets from Queries and Counterexamples. *Inf. Comput.* 75, 2 (1987), 87–106. [https://doi.org/10.1016/0890-5401\(87\)90052-6](https://doi.org/10.1016/0890-5401(87)90052-6)
- [9] Dana Angluin and Dana Fisman. 2016. Learning regular omega languages. *Theor. Comput. Sci.* 650 (2016), 57–72. <https://doi.org/10.1016/j.tcs.2016.07.031>
- [10] Michael A. Arbib and Ernest G. Manes. 1974. A categorist's view of automata and systems. In *Category Theory Applied to Computation and Control*. 51–64. https://doi.org/10.1007/3-540-07142-3_61
- [11] Steve Awodey. 2010. *Category theory*. Oxford University Press.
- [12] Simone Barlocco, Clemens Kupke, and Jurriaan Rot. 2019. Coalgebra Learning via Duality. In *FoSSaCS*. 62–79. https://doi.org/10.1007/978-3-030-17127-8_4
- [13] Francesco Bergadano and Stefano Varricchio. 1996. Learning Behaviors of Automata from Multiplicity and Equivalence Queries. *SIAM J. Comput.* 25, 6 (Dec. 1996), 1268–1280. <https://doi.org/10.1137/S009753979326091X>
- [14] Jérôme Besombes and Jean-Yves Marion. 2007. Learning tree languages from positive examples and membership queries. *Theor. Comput. Sci.* 382, 3 (2007), 183–197. <https://doi.org/10.1016/j.tcs.2007.03.038>
- [15] Venanzio Capretta, Tarmo Uustalu, and Varmo Vene. 2006. Recursive coalgebras from comonads. *Information and Computation* 204, 4 (2006), 437–468. <https://doi.org/10.1016/j.ic.2005.08.005>
- [16] Colin de la Higuera. 2010. *Grammatical Inference: Learning Automata and Grammars*. Cambridge University Press, USA.
- [17] Frank Drewes and Johanna Högberg. 2003. Learning a Regular Tree Language from a Teacher. In *DLT*. 279–291. https://doi.org/10.1007/3-540-45007-6_22
- [18] Paul Fiterău-Broștean, Ramon Janssen, and Frits W. Vaandrager. 2016. Combining Model Learning and Model Checking to Analyze TCP Implementations. In *CAV*. 454–471. https://doi.org/10.1007/978-3-319-41540-6_25
- [19] E. Mark Gold. 1972. System Identification via State Characterization. *Automatica* 8, 5 (Sept. 1972), 621–636. [https://doi.org/10.1016/0005-1098\(72\)90033-7](https://doi.org/10.1016/0005-1098(72)90033-7)
- [20] Malte Isberner, Falk Howar, and Bernhard Steffen. 2015. The Open-Source LearnLib - A Framework for Active Automata Learning. In *CAV*. 487–495. https://doi.org/10.1007/978-3-319-21690-4_32
- [21] Bart Jacobs and Alexandra Silva. 2014. Automata Learning: A Categorical Perspective. In *Horizons of the Mind. A Tribute to Prakash Panangaden - Essays Dedicated to Prakash Panangaden on the Occasion of His 60th Birthday*. 384–406. https://doi.org/10.1007/978-3-319-06880-0_20
- [22] Christoph Lüth and Neil Ghani. 2002. Composing monads using co-products. In *ICFP*. 133–144. <https://doi.org/10.1145/581478.581492>
- [23] Joshua Moerman, Matteo Sammartino, Alexandra Silva, Bartek Klin, and Michał Szynwelski. 2017. Learning nominal automata. In *POPL*. 613–625. <https://doi.org/10.1145/3009837.3009879>
- [24] Malte Mues, Falk Howar, Kasper Søe Luckow, Temesghen Kahsai, and Zvonimir Rakamaric. 2016. Releasing the PSYCO: Using Symbolic Search in Interface Generation for Java. *ACM SIGSOFT Software Engineering Notes* 41, 6 (2016), 1–5. <https://doi.org/10.1145/3011286.3011298>
- [25] Yasubumi Sakakibara. 1990. Learning Context-Free Grammars from Structural Data in Polynomial Time. *Theor. Comput. Sci.* 76, 2/23 (Nov. 1990), 223–242. [https://doi.org/10.1016/0304-3975\(90\)90017-C](https://doi.org/10.1016/0304-3975(90)90017-C)
- [26] Henning Urbat and Lutz Schröder. 2019. Automata Learning: An Algebraic Approach. (2019). arXiv:1911.00874
- [27] Frits W. Vaandrager. 2017. Model learning. *Commun. ACM* 60, 2 (2017), 86–95. <https://doi.org/10.1145/2967606>

- [28] Gerco van Heerdt. 2016. *An Abstract Automata Learning Framework*. Master's thesis. Radboud Universiteit Nijmegen.
- [29] Gerco van Heerdt, Clemens Kupke, Jurriaan Rot, and Alexandra Silva. 2020. Learning Weighted Automata over Principal Ideal Domains. (2020). arXiv:1911.04404 To appear.
- [30] Gerco van Heerdt, Matteo Sammartino, and Alexandra Silva. 2017. CALF: Categorical Automata Learning Framework. In *CSL*. 29:1–29:24. <https://doi.org/10.4230/LIPIcs.CSL.2017.29>
- [31] Gerco van Heerdt, Matteo Sammartino, and Alexandra Silva. 2017. Learning Automata with Side-Effects. (2017). arXiv:1704.08055

A Proofs for Section 5

Lemma A.1. *For all morphisms $\alpha_1 : S_1 \rightarrow Q_t$, $\alpha_2 : S_2 \rightarrow Q_t$, and $f : S_1 \rightarrow S_2$ such that $\alpha_2 \circ f = \alpha_1$ we have $\alpha_1^a \leq \alpha_2^a$.*

Proof. This follows directly from the unique diagonal obtained in the commutative diagram below.

$$\begin{array}{ccc}
 S_1 & \xrightarrow{\alpha_1^a} & \bullet \\
 f \downarrow & \nearrow & \downarrow \alpha_1^a \\
 S_2 & & \\
 \alpha_2^a \downarrow & \nearrow & \downarrow \alpha_2^a \\
 \star & \xrightarrow{\alpha_2^a} & Q_t
 \end{array}$$

□

Proposition 5.3. *Given $\alpha : S \rightarrow Q_t$ and $\beta : Q_t \rightarrow P$, the wrapper $([\alpha, [i_t, \delta_t] \circ F_I \alpha], \beta)$ is locally closed w.r.t. α .*

Proof. Let $\mathcal{W} = ([\alpha, [i_t, \delta_t] \circ F_I \alpha], \beta)$. Note that $\alpha^a \leq [\alpha, [i_t, \delta_t] \circ F_I \alpha]^a$ by Lemma A.1 (via $\kappa_1 : S \rightarrow S + F_I S$). We define

$$i_{\mathcal{W}} = I \xrightarrow{\kappa_1} F_I S \xrightarrow{\kappa_2} S + F_I S \xrightarrow{\tau_{\mathcal{W}}^a} H_{\mathcal{W}}$$

$$\text{lclose}_{\mathcal{W}, \alpha} = FS \xrightarrow{\kappa_2} F_I S \xrightarrow{\kappa_2} S + F_I S \xrightarrow{\tau_{\mathcal{W}}^a} H_{\mathcal{W}}$$

and note that the commutative diagrams below show that they satisfy the required properties.

$$\begin{array}{ccccc}
 I & & & & \\
 \kappa_2 \circ \kappa_1 \downarrow & \searrow^{i_t} & & & \\
 S + F_I S & \xrightarrow{\text{id} + F_I \alpha} & S + F_I Q_t & \xrightarrow{[\alpha, [i_t, \delta_t]]} & Q_t \\
 \tau_{\mathcal{W}}^a \downarrow & & & & \downarrow \beta \\
 H_{\mathcal{W}} & \xrightarrow{\tau_{\mathcal{W}}^a} & & & P
 \end{array}$$

$$\begin{array}{ccccc}
 FS & \xrightarrow{F_I \alpha} & F_I Q_t & & \\
 \kappa_2 \circ \kappa_2 \downarrow & & \kappa_2 \circ \kappa_2 \downarrow & \searrow^{\delta_t} & \\
 S + F_I S & \xrightarrow{\text{id} + F_I \alpha} & S + F_I Q_t & \xrightarrow{[\alpha, [i_t, \delta_t]]} & Q_t \\
 \tau_{\mathcal{W}}^a \downarrow & & & & \downarrow \beta \\
 H_{\mathcal{W}} & \xrightarrow{\tau_{\mathcal{W}}^a} & & & P
 \end{array}$$

□

Proposition 5.4. *Given a recursive $\rho : S \rightarrow F_I S$, the wrapper $([i_t, \delta_t] \circ F_I [i_t, \delta_t]^\rho, \beta)$ is locally closed w.r.t. $[i_t, \delta_t]^\rho$. Furthermore, $F_I \rho$ is also recursive and $[i_t, \delta_t] \circ F_I [i_t, \delta_t]^\rho = [i_t, \delta_t]^{F_I \rho}$.*

Proof. Let $\alpha = [i_t, \delta_t]^\rho$ and $\mathcal{W} = ([i_t, \delta_t] \circ F_I \alpha, \beta)$. Note that $\alpha^a \leq ([i_t, \delta_t] \circ F_I \alpha)^a$ by Lemma A.1 (via ρ).

We know from [15, Proposition 6] that $F_I \rho$ is recursive, so we have $[i_t, \delta_t] \circ F_I \alpha = [i_t, \delta_t]^{F_I \rho}$ by uniqueness from commutativity of the diagram below.

$$\begin{array}{ccccc} F_I F_I S & \xrightarrow{F_I F_I [i_t, \delta_t]^\rho} & F_I F_I Q_t & \xrightarrow{F_I [i_t, \delta_t]} & F_I Q_t \\ F_I \rho \uparrow & & \downarrow F_I [i_t, \delta_t] & & \downarrow [i_t, \delta_t] \\ F_I S & \xrightarrow{F_I [i_t, \delta_t]^\rho} & F_I Q_t & \xrightarrow{[i_t, \delta_t]} & Q_t \end{array} \quad \square$$

Lemma A.2. For any run $\{\mathcal{W}_n = (\alpha_n, \beta_n)\}_{n \in \mathbb{N}}$ and $n \in \mathbb{N}$, if $\alpha_{n+1}^a \leq \alpha_n^a$, then \mathcal{W}_n is closed.

Proof. For each $j \in \mathbb{N}$, denote by S_j the domain of α_j and by P_j the codomain of β_j , and let X_j be the object through which α_j factorises. We define $f_j: X_n \rightarrow H_n$ as the unique diagonal in the commutative square below.

$$\begin{array}{ccc} S_j & \xrightarrow{\alpha_j^a} & X_j \\ \tau_{\mathcal{W}_j}^a \downarrow & \searrow f_j & \downarrow \alpha_j^a \\ & & Q_t \\ & \swarrow \tau_{\mathcal{W}_j}^a & \downarrow \beta_j \\ H_{\mathcal{W}_j} & \xrightarrow{\tau_{\mathcal{W}_j}^a} & P_j \end{array}$$

Note that $f_j \in \mathcal{E}$ because $\tau_{\mathcal{W}_j}^a \in \mathcal{E}$.

We write $v: X_{n+1} \rightarrow X_n$ for the witness of $\alpha_{n+1}^a \leq \alpha_n^a$. Assume towards a contradiction that \mathcal{W}_n is not closed. By the definition of a run we then have that $\beta_{n+1} = \beta_n$ and \mathcal{W}_{n+1} is locally closed w.r.t. α_n . Define $i_{\mathcal{W}_n} = h \circ i_{\mathcal{W}_{n+1}}: I \rightarrow H_{\mathcal{W}_n}$ and $\text{close}_{\mathcal{W}_n} = h \circ \text{close}_{\mathcal{W}_{n+1}, \alpha_n}$, where $i_{\mathcal{W}_{n+1}}$ and $\text{close}_{\mathcal{W}_{n+1}, \alpha_n}$ exist by local closedness and $h: H_{\mathcal{W}_{n+1}} \rightarrow H_{\mathcal{W}_n}$ is the unique diagonal in the commutative diagram below.

$$\begin{array}{ccc} X_{n+1} & \xrightarrow{f_{n+1}} & H_{\mathcal{W}_{n+1}} \\ v \downarrow & \searrow \alpha_{n+1}^a & \downarrow \tau_{\mathcal{W}_{n+1}}^a \\ X_n & \xrightarrow{\alpha_n^a} & Q_t \\ f_n \downarrow & \searrow \beta_n = \beta_{n+1} & \downarrow \tau_{\mathcal{W}_n}^a \\ H_{\mathcal{W}_n} & \xrightarrow{\tau_{\mathcal{W}_n}^a} & P_n = P_{n+1} \end{array} \quad \textcircled{1}$$

$$\begin{array}{ccc} X_{n+1} & \xrightarrow{f_{n+1}} & H_{\mathcal{W}_{n+1}} \\ v \downarrow & \searrow h & \downarrow \tau_{\mathcal{W}_{n+1}}^a \\ X_n & \xrightarrow{h} & H_{\mathcal{W}_n} \\ f_n \downarrow & \searrow \tau_{\mathcal{W}_{n+1}}^a & \downarrow \tau_{\mathcal{W}_n}^a \\ H_{\mathcal{W}_n} & \xrightarrow{\tau_{\mathcal{W}_n}^a} & P_n = P_{n+1} \end{array} \quad \textcircled{2}$$

① definition of f_n or f_{n+1}

Now the diagrams below commute, leading to the desired contradiction that \mathcal{W}_n is closed.

$$\begin{array}{ccc} I & \xrightarrow{i_t} & Q_t \\ i_{\mathcal{W}_{n+1}} \downarrow & \searrow \alpha_{n+1}^a & \downarrow \tau_{\mathcal{W}_{n+1}}^a \\ H_{\mathcal{W}_{n+1}} & \xrightarrow{\tau_{\mathcal{W}_{n+1}}^a} & Q_t \\ h \downarrow & \searrow \beta_n & \downarrow \beta_n \\ H_{\mathcal{W}_n} & \xrightarrow{\tau_{\mathcal{W}_n}^a} & P_n = P_{n+1} \end{array} \quad \textcircled{1}$$

$$\begin{array}{ccc} F S_n & \xrightarrow{F \alpha_n} & F Q_t \\ \downarrow \text{close}_{\mathcal{W}_{n+1}, \alpha_n} & \searrow \alpha_n^a & \downarrow \delta_t \\ H_{\mathcal{W}_{n+1}} & \xrightarrow{\tau_{\mathcal{W}_{n+1}}^a} & Q_t \\ h \downarrow & \searrow \beta_n & \downarrow \beta_n \\ H_{\mathcal{W}_n} & \xrightarrow{\tau_{\mathcal{W}_n}^a} & P_n = P_{n+1} \end{array} \quad \textcircled{2}$$

① local closedness ② definition of h \square

Lemma A.3. For any run $\{\mathcal{W}_n = (\alpha_n, \beta_n)\}_{n \in \mathbb{N}}$ and $n \in \mathbb{N}$, if \mathcal{W}_n is closed and $\beta_n^a \leq \beta_{n+1}^a$, then \mathcal{W}_n is consistent.

Proof. For each $j \in \mathbb{N}$, denote by S_j the domain of α_j and by P_j the codomain of β_j , and let X_j be the object through which β_j factorises. We define $f_j: H_{\mathcal{W}_j} \rightarrow X_j$ as the unique diagonal in the commutative square below.

$$\begin{array}{ccc} S_j & \xrightarrow{\tau_{\mathcal{W}_j}^a} & H_{\mathcal{W}_j} \\ \alpha_j \downarrow & \searrow f_j & \downarrow \tau_{\mathcal{W}_j}^a \\ Q_t & \xrightarrow{f_j} & X_j \\ \beta_j^a \downarrow & \searrow \beta_j^a & \downarrow \beta_j^a \\ X_j & \xrightarrow{\beta_j^a} & P_j \end{array}$$

Note that $f_j \in \mathcal{M}$ because $\tau_{\mathcal{W}_j}^a \in \mathcal{M}$.

We write $v: X_n \rightarrow X_{n+1}$ for the witness of $\beta_n^a \leq \beta_{n+1}^a$. Assume towards a contradiction that \mathcal{W}_n is not consistent. By the definition of a run of the algorithm we then have that $\alpha_{n+1} = \alpha_n$ and \mathcal{W}_{n+1} is locally consistent w.r.t. β_n . Define $\sigma_{\mathcal{W}_n} = \sigma_{\mathcal{W}_{n+1}} \circ h: H_{\mathcal{W}_n} \rightarrow O$ and $\text{cons}_{\mathcal{W}_n} = \text{lcons}_{\mathcal{W}_{n+1}, \beta_n} \circ h$, where $h: H_{\mathcal{W}_n} \rightarrow H_{\mathcal{W}_{n+1}}$ is the unique diagonal in the commutative diagram below.

$$\begin{array}{ccc} S_n = S_{n+1} & \xrightarrow{\tau_{\mathcal{W}_n}^a} & H_{\mathcal{W}_n} \\ \tau_{\mathcal{W}_{n+1}}^a \downarrow & \searrow \alpha_n = \alpha_{n+1} & \downarrow f_n \\ & & Q_t \\ & \swarrow \beta_n^a & \downarrow \beta_n^a \\ H_{\mathcal{W}_{n+1}} & \xrightarrow{f_{n+1}} & X_{n+1} \end{array} \quad \textcircled{1}$$

$$\begin{array}{ccc} S_n = S_{n+1} & \xrightarrow{\tau_{\mathcal{W}_n}^a} & H_{\mathcal{W}_n} \\ \tau_{\mathcal{W}_{n+1}}^a \downarrow & \searrow h & \downarrow f_n \\ & & X_n \\ & \swarrow v & \downarrow v \\ H_{\mathcal{W}_{n+1}} & \xrightarrow{f_{n+1}} & X_{n+1} \end{array} \quad \textcircled{2}$$

① definition of f_n or f_{n+1}

Now the diagrams below commute, leading to the desired contradiction that \mathcal{W}_n is consistent.

$$\begin{array}{ccc} S_n = S_{n+1} & \xrightarrow{\tau_{\mathcal{W}_n}^a} & H_{\mathcal{W}_n} \\ \alpha_n \downarrow & \searrow \tau_{\mathcal{W}_{n+1}}^a & \downarrow h \\ Q_t & \xrightarrow{\tau_{\mathcal{W}_{n+1}}^a} & H_{\mathcal{W}_{n+1}} \\ \sigma_t \downarrow & \searrow \sigma_{\mathcal{W}_{n+1}} & \downarrow \sigma_{\mathcal{W}_{n+1}} \\ O & \xrightarrow{\sigma_{\mathcal{W}_{n+1}}} & O \end{array} \quad \textcircled{1}$$

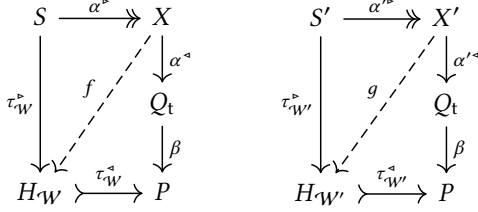
$$\begin{array}{ccc} F S_n = F S_{n+1} & \xrightarrow{F \tau_{\mathcal{W}_n}^a} & F H_{\mathcal{W}_n} \\ F \alpha_n \downarrow & \searrow F \tau_{\mathcal{W}_{n+1}}^a & \downarrow F h \\ F Q_t & \xrightarrow{F \tau_{\mathcal{W}_{n+1}}^a} & F H_{\mathcal{W}_{n+1}} \\ \delta_t \downarrow & \searrow \text{lcons}_{\mathcal{W}_{n+1}, \beta_n} & \downarrow \text{lcons}_{\mathcal{W}_{n+1}, \beta_n} \\ Q_t & \xrightarrow{\beta_n} & P_n \end{array} \quad \textcircled{2}$$

① local consistency ② definition of h \square

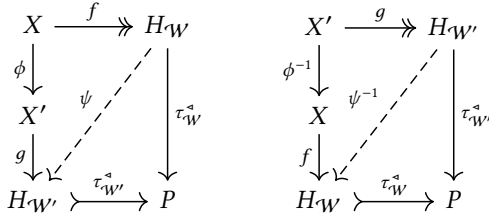
Lemma A.4. Let $\alpha: S \rightarrow Q_t$, $\alpha': S' \rightarrow Q_t$, and $\beta: Q_t \rightarrow P$ be such that α^a and α'^a are isomorphic subobjects. If (α, β) is closed and consistent, then so is (α', β) .

Proof. Write $\mathcal{W} = (\alpha, \beta)$ and $\mathcal{W}' = (\alpha', \beta)$. Let X and X' be the respective objects through which α and α' factorise, and denote by $\phi: X \rightarrow X'$ the subobject isomorphism $(\alpha'^a \circ$

$\phi = \alpha^\triangleleft$). We define $f: X \rightarrow H_{\mathcal{W}}$ and $g: X' \rightarrow H_{\mathcal{W}'}$ as the unique diagonals in the diagrams below.



Note that α^\triangleright and $\tau_{\mathcal{W}}^\triangleright$ are in \mathcal{E} , and therefore so is f ; similarly, since α^\triangleright and $\tau_{\mathcal{W}'}^\triangleright$ are in \mathcal{E} , so is g [6, Proposition 14.9 via duality]. We now define $\psi: H_{\mathcal{W}} \rightarrow H_{\mathcal{W}'}$ and $\psi^{-1}: H_{\mathcal{W}'} \rightarrow H_{\mathcal{W}}$ as the unique diagonals in the diagrams below.



It is a standard result that ψ and ψ^{-1} are inverse to each other [6, Proposition 14.7], as suggested by their names. We define

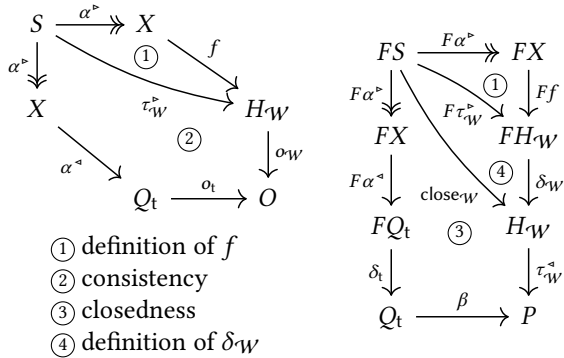
$$i_{\mathcal{W}'} = I \xrightarrow{i_{\mathcal{W}}} H_{\mathcal{W}} \xrightarrow{\psi} H_{\mathcal{W}'}, \quad o_{\mathcal{W}'} = H_{\mathcal{W}'} \xrightarrow{\psi^{-1}} H_{\mathcal{W}} \xrightarrow{o_{\mathcal{W}}} O$$

$$d = H_{\mathcal{W}'} \xrightarrow{F\psi^{-1}} FH_{\mathcal{W}} \xrightarrow{\delta_{\mathcal{W}}} H_{\mathcal{W}} \xrightarrow{\psi} H_{\mathcal{W}'}$$

To show closedness and consistency, we will need the following two equations.

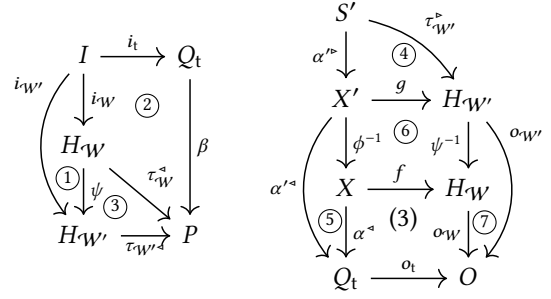
$$o_{\mathcal{W}} \circ f = o_t \circ \alpha^\triangleleft \quad \tau_{\mathcal{W}}^\triangleleft \circ \delta_{\mathcal{W}} \circ Ff = \beta \circ \delta_t \circ F\alpha^\triangleleft. \quad (3)$$

Note that both α^\triangleright and $F\alpha^\triangleright$ are in \mathcal{E} because F preserves \mathcal{E} , and that they are therefore both epis. We use this to prove (3) with the commutative diagrams below.

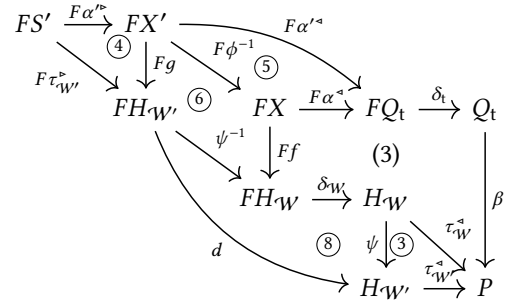


- ① definition of f
- ② consistency
- ③ closedness
- ④ definition of $\delta_{\mathcal{W}}$

Now the diagrams below commute.



- ① definition of $i_{\mathcal{W}'}$
- ② closedness
- ③ definition of ψ
- ④ definition of g
- ⑤ subobject morphism
- ⑥ definition of ψ^{-1}
- ⑦ definition of $o_{\mathcal{W}'}$
- ⑧ definition of d



Using [30, Theorem 9], the existence of a d making the last diagram above commute shows together with the other two commutative diagrams that \mathcal{W}' is closed and consistent. \square

Lemma 5.7. Consider a run $\{\mathcal{W}_n = (\alpha_n, \beta_n)\}_{n \in \mathbb{N}}$ and $n \in \mathbb{N}$. We have $\alpha_n^\triangleleft \leq \alpha_{n+1}^\triangleleft$ and $\beta_{n+1}^\triangleright \leq \beta_n^\triangleright$ for all $n \in \mathbb{N}$. Moreover, if $\alpha_{n+1}^\triangleleft \leq \alpha_n^\triangleleft$, then $\alpha_{n+1} = \alpha_n$; if $\beta_n^\triangleright \leq \beta_{n+1}^\triangleright$, then $\beta_{n+1} = \beta_n$.

Proof. We consider each of the cases listed in the definition of a run of the algorithm. If \mathcal{W}_n is not closed, then $\beta_{n+1} = \beta_n$ and $\alpha_n^\triangleleft \leq \alpha_{n+1}^\triangleleft$ by the definition of local closedness. Supposing $\alpha_{n+1}^\triangleleft \leq \alpha_n^\triangleleft$ leads by Lemma A.2 to the contradiction that \mathcal{W}_n is closed.

If \mathcal{W}_n is closed but not consistent, then $\alpha_{n+1} = \alpha_n$ and we have $\beta_{n+1}^\triangleright \leq \beta_n^\triangleright$ by the definition of local consistency. Supposing $\beta_n^\triangleright \leq \beta_{n+1}^\triangleright$ leads by Lemma A.3 to the contradiction that \mathcal{W}_n is consistent.

If \mathcal{W}_n is closed and consistent and we obtain a counterexample $\rho: S \rightarrow F_I S$ for \mathcal{W}_n , then $\beta_{n+1} = \beta_n$ and $\alpha_{n+1}^\triangleleft = [\alpha_n, [i_t, \delta_t]^\rho]^\triangleleft$. We have $\alpha_n^\triangleleft \leq [\alpha_n, [i_t, \delta_t]^\rho]^\triangleleft$ using Lemma A.1. Suppose $\alpha_{n+1}^\triangleleft \leq \alpha_n^\triangleleft$. Then $[\alpha_n, [i_t, \delta_t]^\rho]^\triangleleft = \alpha_{n+1}^\triangleleft \leq \alpha_n^\triangleleft$, so α_n^\triangleleft and $[\alpha_n, [i_t, \delta_t]^\rho]^\triangleleft$ are isomorphic subobjects. By Lemma A.4 this implies that $([\alpha_n, [i_t, \delta_t]^\rho], \beta_n)$ is also closed and consistent, which by Corollary 4.9 contradicts the fact that ρ is a counterexample for \mathcal{W}_n .

If \mathcal{W}_n is closed and consistent and correct up to all recursive F_I -coalgebras, then we immediately have $\alpha_{n+1} = \alpha_n$ and $\beta_{n+1} = \beta_n$. \square

B Proofs for Section 6

Lemma B.1. For all T -algebras (X, x) , $p: I \rightarrow X$, and $c \in T(I + X)$, the diagram below commutes.

$$\begin{array}{ccc} T(I + T(I + X)) & \xrightarrow{\hat{\mu}_X} & T(I + X) \\ T(\text{id}_I + 1_c) \uparrow & & \downarrow [p, \text{id}_X]^\# \\ T(I + 1) & \xrightarrow{[p, 1_{[p, \text{id}_X]^\#(c)}]^\#} & X \end{array}$$

Proof. Given any set Y and a $T(I + (-))$ -algebra (Z, z) , the extension of a morphism $f: Y \rightarrow Z$ to the $T(I + (-))$ -algebra homomorphism $f^\natural: T(I + Y) \rightarrow Z$ is given by $f^\natural = z \circ T(\text{id}_I + f)$. We can supply X with the $T(I + (-))$ -algebra structure $[p, \text{id}_X]^\#: T(I + X) \rightarrow X$. Thus,

$$\begin{aligned} [p, 1_{[p, \text{id}_X]^\#(c)}]^\# &= [p, 1_{\text{id}_X^\natural(c)}]^\# \\ &= [p, \text{id}_X]^\# \circ T(\text{id}_I + 1_{\text{id}_X^\natural(c)}) \\ &= 1_{\text{id}_X^\natural(c)}^\natural \\ &= (\text{id}_X^\natural \circ 1_c)^\natural & (2) \\ &= \text{id}_X^\natural \circ 1_c^\natural \\ &= [p, \text{id}_X]^\# \circ 1_c^\natural \\ &= [p, \text{id}_X]^\# \circ \hat{\mu}_X \circ T(\text{id}_I + 1_c). \quad \square \end{aligned}$$

Proposition 6.5 (Computing wrapper morphisms). Given $S \subseteq TI$ with inclusion $j: S \rightarrow TI$ and $E \subseteq T(I + 1)$, we have

$$\begin{aligned} \beta_E \circ \alpha_S: S &\rightarrow O^E \\ (\beta_E \circ \alpha_S)(s) &= \mathcal{L}_{\mathcal{A}_t} \circ \mu_I \circ T[\eta_I, 1_s] \\ \beta_E \circ \delta_t \circ F\alpha_S: FS &\rightarrow O^E \\ (\beta_E \circ \delta_t \circ F\alpha_S)(f) &= \mathcal{L}_{\mathcal{A}_t} \circ \mu_I \circ T[\eta_I, \gamma_I \circ Fj \circ 1_f] \\ \beta_E \circ i_t: I &\rightarrow O^E \\ (\beta_E \circ i_t)(x) &= \mathcal{L}_{\mathcal{A}_t} \circ T[\text{id}_I, 1_x] \\ \alpha_t \circ \alpha_S: S &\rightarrow O \\ (\alpha_t \circ \alpha_S)(s) &= \mathcal{L}_{\mathcal{A}_t}(s). \end{aligned}$$

Proof. We first claim that

$$(\beta_E \circ i_t^\#)(s) = \mathcal{L}_{\mathcal{A}_t} \circ \mu_I \circ T[\eta_I, 1_s]. \quad (4)$$

To see this, first note that

$$\begin{aligned} (\beta_E \circ i_t^\#)(s) &= \alpha_t \circ [i_t, 1_{i_t^\#(s)}]^\# \\ &= \alpha_t \circ \delta_t^* \circ T[i_t, 1_{i_t^\#(s)}]^\# \\ &= \alpha_t \circ \delta_t^* \circ T[i_t, i_t^\# \circ 1_s] \\ &= \alpha_t \circ \delta_t^* \circ T[i_t^\# \circ \eta_I, i_t^\# \circ 1_s] \\ &= \alpha_t \circ \delta_t^* \circ T i_t^\# \circ T[\eta_I, 1_s] \end{aligned}$$

It remains to show that $\alpha_t \circ \delta_t^* \circ T i_t^\# = \mathcal{L}_{\mathcal{A}_t} \circ \mu_I$, which follows by commutativity of the diagram below.

$$\begin{array}{ccccc} T^2 I & \xrightarrow{\mu_I} & T I & & \\ T i_t^\# \downarrow & \searrow^{T^2 i_t} & \downarrow^{T i_t} & & \downarrow^{\mathcal{L}_{\mathcal{A}_t}} \\ T^2 Q_t & \xrightarrow{\mu_{Q_t}} & T Q_t & & \\ T i_t^\# \downarrow & \searrow^{T^2 \delta_t^*} & \downarrow^{\delta_t^*} & & \downarrow^{\alpha_t} \\ T Q_t & \xrightarrow{\delta_t^*} & Q_t & \xrightarrow{\alpha_t} & O \end{array}$$

① property of $i_t^\#$ ② definition of $\mathcal{L}_{\mathcal{A}_t}$
③ naturality ④ (T, δ_t^*) is a T -algebra

For the first equation, we derive

$$\begin{aligned} (\beta_E \circ \alpha_S)(s) &= (\beta_E \circ i_t^\#)(s) & (\text{def. of } \alpha_S) \\ &= \mathcal{L}_{\mathcal{A}_t} \circ \mu_I \circ T[\eta_I, 1_s] & (\text{by (4)}) \end{aligned}$$

For the second equation, we derive

$$\begin{aligned} (\beta_E \circ \delta_t \circ F\alpha_S)(f) &= (\beta_E \circ \delta_t \circ F i_t^\# \circ Fj)(f) & (\text{def. of } \alpha_S) \\ &= (\beta_E \circ i_t^\# \circ \gamma_I \circ Fj)(f) & (i_t^\# \text{ is an } F\text{-algebra homomorphism}) \\ &= \mathcal{L}_{\mathcal{A}_t} \circ \mu_I \circ T[\eta_I, 1_{(\gamma_I \circ Fj)(f)}] & (\text{by (4)}) \\ &= \mathcal{L}_{\mathcal{A}_t} \circ \mu_I \circ T[\eta_I, \gamma_I \circ Fj \circ 1_f] & (\text{by (2)}) \end{aligned}$$

For the third equation, we derive

$$\begin{aligned} (\beta_E \circ i_t)(s) &= (\beta_E \circ i_t^\# \circ \eta_I)(s) & (\text{property of } i_t^\#) \\ &= \mathcal{L}_{\mathcal{A}_t} \circ \mu_I \circ T[\eta_I, 1_{\eta_I(s)}] & (\text{by (4)}) \\ &= \mathcal{L}_{\mathcal{A}_t} \circ \mu_I \circ T[\eta_I, \eta_I \circ 1_s] & (\text{by (2)}) \\ &= \mathcal{L}_{\mathcal{A}_t} \circ \mu_I \circ T \eta_I \circ T[\text{id}_I, 1_s] \\ &= \mathcal{L}_{\mathcal{A}_t} \circ T[\text{id}_I, 1_s] & (\text{monad law}) \end{aligned}$$

Finally, for the fourth equation, we derive:

$$\begin{aligned} (\alpha_t \circ \alpha_S)(s) &= (\alpha_t \circ i_t^\#)(s) & (\text{definition of } \alpha_S) \\ &= \mathcal{L}_{\mathcal{A}_t}(s) & (\text{definition of } \mathcal{L}_{\mathcal{A}_t}) \quad \square \end{aligned}$$

Lemma 6.7 (Local closedness for Set automata). Given $S, S' \subseteq TI$ and $E \subseteq T(I + 1)$ such that $S \subseteq S'$, $(\alpha_{S'}, \beta_E)$ is locally closed w.r.t. α_S if there exist $k: I \rightarrow S'$ and $\ell: FS \rightarrow S'$ such that

$$\alpha_{S'} \circ k = i_t \quad \alpha_{S'} \circ \ell = \delta_t \circ F\alpha_S.$$

Proof. Let $\mathcal{W} = (\alpha_{S'}, \beta_E)$, and choose

$$i_{\mathcal{W}} = \tau_{\mathcal{W}}^\triangleright \circ k \quad \text{lclose}_{\mathcal{W}, \alpha_S} = \tau_{\mathcal{W}}^\triangleright \circ \ell$$

The necessary diagrams now commute:

$$\begin{array}{ccc} I & \xrightarrow{i_t} & Q_t \\ \downarrow^{i_{\mathcal{W}}} & \searrow^k & \downarrow^\beta \\ & S' & \\ \downarrow^{\tau_{\mathcal{W}}^\triangleright} & \swarrow^{\alpha_{S'}} & \downarrow^\beta \\ H_{\mathcal{W}} & \xrightarrow{\tau_{\mathcal{W}}^\triangleright} & P \end{array} \quad \begin{array}{ccc} FS' & \xrightarrow{F\alpha'} & FQ_t \\ \downarrow^{\text{lclose}_{\mathcal{W}, \alpha'}} & \searrow^\ell & \downarrow^{\delta_t} \\ & S' & \\ \downarrow^{\tau_{\mathcal{W}}^\triangleright} & \swarrow^{\alpha_{S'}} & \downarrow^\beta \\ H_{\mathcal{W}} & \xrightarrow{\tau_{\mathcal{W}}^\triangleright} & P \end{array}$$

Note that $\alpha_S \leq \alpha_{S'}$ because $S \subseteq S'$. Thus, \mathcal{W} is locally closed w.r.t. α_S . \square

Lemma 6.11 (Local consistency for Set automata). *Let $S \subseteq TI$ and $E \subseteq E' \subseteq T(I+1)$, with S finite. Furthermore, suppose that for $s, s' \in S$ with $(\beta_{E'} \circ \alpha_S)(s) = (\beta_{E'} \circ \alpha_S)(s')$ we have*

$$(o_t \circ \alpha_S)(s) = (o_t \circ \alpha_S)(s')$$

$$\beta_E \circ \delta_t \circ F(\alpha_S \circ [\text{id}_S, 1_s]) = \beta_E \circ \delta_t \circ F(\alpha_S \circ [\text{id}_S, 1_{s'}])$$

Then $\mathcal{W} = (\alpha_S, \beta_{E'})$ is locally consistent w.r.t. β_E .

Proof. Since $\tau_{\mathcal{W}}^*$ is surjective, so is $F\tau_{\mathcal{W}}^*$. We define the function $\text{lcons}_{\mathcal{W}, \beta_E}: FH_{\mathcal{W}} \rightarrow O^E$ by

$$\text{lcons}_{\mathcal{W}, \beta_E}(F\tau_{\mathcal{W}}^*(y)) = (\beta_E \circ \delta_t \circ F\alpha_S)(y).$$

By definition this satisfies the local consistency condition. It remains to show that the function is well-defined. Denote by K the kernel

$$\{(s, s') \mid s, s' \in S, \tau_{\mathcal{W}}^*(s) = \tau_{\mathcal{W}}^*(s')\}$$

and let $j: K \rightarrow S \times S$ be the inclusion. Consider $y, z \in FS$ such that $F\tau_{\mathcal{W}}^*(y) = F\tau_{\mathcal{W}}^*(z)$. Because F preserves weak pullbacks we can find $x \in FK$ such that $F(\pi_1 \circ j)(x) = y$ and $F(\pi_2 \circ j)(x) = z$. Using that S is finite, write $K = \{(s_1, s'_1), \dots, (s_n, s'_n)\}$. For all $1 \leq m \leq n$ we define $f_m: K \rightarrow S+1$ by

$$f_m(s_k, s'_k) = \begin{cases} \kappa_1(s'_k) & \text{if } k < m \\ \kappa_2(\square) & \text{if } k = m \\ \kappa_1(s_k) & \text{if } k > m. \end{cases}$$

Furthermore, let $c_m = F(f_m)(x) \in F(S+1)$. We will prove that

$$(\beta_E \circ \delta_t \circ F(\alpha_S \circ [\text{id}_S, 1_{s_1}]))(c_1) = (\beta_E \circ \delta_t \circ F(\alpha_S \circ [\text{id}_S, 1_{s_n}]))(c_n), \quad (5)$$

for which it suffices by induction to prove for all $1 \leq m < n$ that

$$\begin{aligned} & (\beta_E \circ \delta_t \circ F(\alpha_S \circ [\text{id}_S, 1_{s_m}]))(c_m) \\ &= (\beta_E \circ \delta_t \circ F(\alpha_S \circ [\text{id}_S, 1_{s_{m+1}}]))(c_{m+1}). \end{aligned}$$

Note that

$$[\text{id}_S, 1_{s'_m}] \circ f_m = [\text{id}_S, 1_{s_{m+1}}] \circ f_{m+1}$$

by the definitions of f_m and f_{m+1} , so

$$\begin{aligned} & (\beta_E \circ \delta_t \circ F(\alpha_S \circ [\text{id}_S, 1_{s_m}]))(c_m) \\ &= (\beta_E \circ \delta_t \circ F(\alpha_S \circ [\text{id}_S, 1_{s'_m}]))(c_m) \\ & \quad (\text{assumption}) \\ &= (\beta_E \circ \delta_t \circ F(\alpha_S \circ [\text{id}_S, 1_{s'_m}] \circ f_m))(x) \\ & \quad (\text{definition of } c_m) \\ &= (\beta_E \circ \delta_t \circ F(\alpha_S \circ [\text{id}_S, 1_{s_{m+1}}] \circ f_{m+1}))(x) \\ &= (\beta_E \circ \delta_t \circ F(\alpha_S \circ [\text{id}_S, 1_{s_{m+1}}]))(c_{m+1}) \\ & \quad (\text{definition of } c_{m+1}). \end{aligned}$$

Then

$$\begin{aligned} (\beta_E \circ \delta_t \circ F\alpha_S)(y) &= (\beta_E \circ \delta_t \circ F(\alpha_S \circ \pi_1 \circ j))(x) \\ & \quad (\text{definition of } x) \\ &= (\beta_E \circ \delta_t \circ F(\alpha_S \circ [\text{id}_S, 1_{s_1}] \circ f_1))(x) \\ & \quad (\text{definition of } f_1) \\ &= (\beta_E \circ \delta_t \circ F(\alpha_S \circ [\text{id}_S, 1_{s_1}]))(c_1) \\ & \quad (\text{definition of } c_1) \\ &= (\beta_E \circ \delta_t \circ F(\alpha_S \circ [\text{id}_S, 1_{s_n}]))(c_n) \\ & \quad (5) \\ &= (\beta_E \circ \delta_t \circ F(\alpha_S \circ [\text{id}_S, 1_{s'_n}]))(c_n) \\ & \quad (\text{assumption}) \\ &= (\beta_E \circ \delta_t \circ F(\alpha_S \circ [\text{id}_S, 1_{s'_n}] \circ f_n))(x) \\ & \quad (\text{definition of } c_n) \\ &= (\beta_E \circ \delta_t \circ F(\alpha_S \circ \pi_2 \circ j))(x) \\ & \quad (\text{definition of } f_n) \\ &= (\beta_E \circ \delta_t \circ F\alpha_S)(z) \\ & \quad (\text{definition of } x). \end{aligned}$$

We conclude that $\text{lcons}_{\mathcal{W}, \beta_E}$ is well-defined.

We define $o_{\mathcal{W}}: H_{\mathcal{W}} \rightarrow O$ by

$$o_{\mathcal{W}}(\tau_{\mathcal{W}}^*(s)) = (o_t \circ \alpha_S)(s).$$

Again the local consistency condition is satisfied by definition, but we need to show that the function is well-defined. Consider $s_1, s_2 \in S$ such that $\tau_{\mathcal{W}}^*(s_1) = \tau_{\mathcal{W}}^*(s_2)$. Then

$$\begin{aligned} (\beta_{E'} \circ \alpha_S)(s_1) &= (\tau_{\mathcal{W}}^* \circ \tau_{\mathcal{W}}^*)(s_1) \\ &= (\tau_{\mathcal{W}}^* \circ \tau_{\mathcal{W}}^*)(s_2) \\ &= (\beta_{E'} \circ \alpha_S)(s_2), \end{aligned}$$

so $(o_t \circ \alpha_S)(s_1) = (o_t \circ \alpha_S)(s_2)$. Note that $\beta_{E'} \leq \beta_E$ because $E \subseteq E'$. Thus, \mathcal{W} is locally consistent w.r.t. β_E . \square

Proposition 6.13. *Given finite $S \subseteq TI$ and $E \subseteq T(I+1)$, the set $E' \subseteq T(I+1)$ is defined as*

$$\begin{aligned} E' &= E \cup \{(\eta_{I+1} \circ \kappa_2)(\square)\} \\ & \quad \cup \{(\hat{\mu}_1 \circ T(\text{id}_I + c_x))(e) \mid e \in E, x \in F(S+1)\}, \end{aligned}$$

where $c_x: 1 \rightarrow T(I+1)$, with $c_x = \gamma_{I+1} \circ F[T\kappa_1 \circ j, \hat{\eta}_1] \circ 1_x$, where $j: S \rightarrow TI$ is set inclusion. It holds that E' is finite and $(\alpha_S, \beta_{E'})$ is locally consistent w.r.t. β_E .

Proof. Note that since S is finite and F preserves finite sets we have that $F(S+1)$ is also finite. Together with the fact that E is finite it follows that E' is finite. Suppose $s_1, s_2 \in S$ are such that $(\beta_{E'} \circ \alpha_S)(s_1) = (\beta_{E'} \circ \alpha_S)(s_2)$. For all $s \in S$ we

have

$$\begin{aligned}
(o_t \circ \alpha_S)(s) &= (o_t \circ 1_{\alpha_S(s)})(\square) \\
&= (o_t \circ [i_t, 1_{\alpha_S(s)}] \circ \kappa_2)(\square) \\
&= (o_t \circ [i_t, 1_{\alpha_S(s)}]^\# \circ \eta_{I+1} \circ \kappa_2)(\square) \\
&= (\beta_{E'} \circ \alpha_S)(s)((\eta_{I+1} \circ \kappa_2)(\square)) \\
&\quad \text{(definition of } \beta_{E'}),
\end{aligned}$$

so

$$\begin{aligned}
(o_t \circ \alpha_S)(s_1) &= (\beta_{E'} \circ \alpha_S)(s_1)((\eta_{I+1} \circ \kappa_2)(\square)) \\
&= (\beta_{E'} \circ \alpha_S)(s_2)((\eta_{I+1} \circ \kappa_2)(\square)) \\
&= (o_t \circ \alpha_S)(s_2).
\end{aligned}$$

Furthermore, for all $s \in S$ we have

$$\begin{aligned}
&F[Ti_t \circ j, T1_{\alpha_S(s)} \circ \eta_1] \\
&= F[T[i_t, 1_{\alpha_S(s)}] \circ T\kappa_1 \circ j, T[i_t, 1_{\alpha_S(s)}] \circ T\kappa_2 \circ \eta_1] \\
&= FT[i_t, 1_{\alpha_S(s)}] \circ F[T\kappa_1 \circ j, T\kappa_2 \circ \eta_1] \\
&= FT[i_t, 1_{\alpha_S(s)}] \circ F[T\kappa_1 \circ j, \hat{\eta}_1] \\
&\quad \text{(definition of } \hat{\eta}_1),
\end{aligned}$$

so for all $s \in S$ and $x \in F(S+1)$ we have

$$\begin{aligned}
&(\delta_t \circ F[\alpha_S, 1_{\alpha_S(s)}])(x) \\
&= (\delta_t \circ F[\delta_t^* \circ Ti_t \circ j, \delta_t^* \circ T1_{\alpha_S(s)} \circ \eta_1])(x) \\
&\quad \text{(definition of } \alpha_S) \\
&= (\delta_t \circ F\delta_t^* \circ F[Ti_t \circ j, T1_{\alpha_S(s)} \circ \eta_1])(x) \\
&= (\delta_t^* \circ \gamma_Q \circ F[Ti_t \circ j, T1_{\alpha_S(s)} \circ \eta_1])(x) \\
&\quad (\delta_t^* \text{ is an } F\text{-algebra homomorphism}) \\
&= (\delta_t^* \circ \gamma_Q \circ FT[i_t, 1_{\alpha_S(s)}] \circ F[T\kappa_1 \circ j, \hat{\eta}_1])(x) \\
&\quad \text{(shown above)} \\
&= (\delta_t^* \circ T[i_t, 1_{\alpha_S(s)}] \circ \gamma_{I+1} \circ F[T\kappa_1 \circ j, \hat{\eta}_1])(x) \\
&= (\delta_t^* \circ T[i_t, 1_{\alpha_S(s)}] \circ \gamma_{I+1} \circ F[T\kappa_1 \circ j, \hat{\eta}_1] \circ 1_x)(\square) \\
&= (\delta_t^* \circ T[i_t, 1_{\alpha_S(s)}] \circ c_x)(\square) \\
&\quad \text{(definition of } c_x) \\
&= ([i_t, 1_{\alpha_S(s)}]^\# \circ c_x)(\square) \\
&= [i_t, \text{id}_Q]^\#((T(\text{id}_I + 1_{\alpha_S(s)}) \circ c_x)(\square)).
\end{aligned}$$

Note that for all $s \in S$, $x \in F(S+1)$ we have

$$\begin{aligned}
&\hat{\mu}_Q \circ T(\text{id}_I + 1_{(T(\text{id}_I + 1_{\alpha_S(s)}) \circ c_x)(\square)}) \\
&= \hat{\mu}_Q \circ T(\text{id}_I + (T(\text{id}_I + 1_{\alpha_S(s)}) \circ 1_{c_x(\square)})) \\
&\quad (2) \\
&= \hat{\mu}_Q \circ T(\text{id}_I + (T(\text{id}_I + 1_{\alpha_S(s)}) \circ c_x)) \\
&\quad \text{(definition of } 1_{c_x(\square)}) \\
&= \hat{\mu}_Q \circ T(\text{id}_I + (T(\text{id}_I + 1_{\alpha_S(s)}))) \circ T(\text{id}_I + c_x) \\
&= T(\text{id}_I + 1_{\alpha_S(s)}) \circ \hat{\mu}_1 \circ T(\text{id}_I + c_x),
\end{aligned}$$

so for all $s \in S$, $x \in F(S+1)$, and $e \in E$,

$$\begin{aligned}
&(\beta_E \circ \delta_t \circ F(\alpha_S \circ [\text{id}_S, 1_s]))(x)(e) \\
&= (\beta_E \circ \delta_t \circ F[\alpha_S, 1_{\alpha_S(s)}])(x)(e) \\
&\quad (2) \\
&= (o_t \circ [i_t, 1_{(\delta_t \circ F[\alpha_S, 1_{\alpha_S(s)}])(x)}]^\#)(e) \\
&\quad \text{(definition of } \beta_E) \\
&= (o_t \circ [i_t, 1_{[i_t, \text{id}_Q]^\#((T(\text{id}_I + 1_{\alpha_S(s)}) \circ c_x)(\square))}]^\#)(e) \\
&\quad \text{(shown earlier)} \\
&= (o_t \circ [i_t, \text{id}_Q]^\# \circ \hat{\mu}_Q \circ T(\text{id}_I + 1_{(T(\text{id}_I + 1_{\alpha_S(s)}) \circ c_x)(\square)}))(e) \\
&\quad \text{(Lemma B.1)} \\
&= (o_t \circ [i_t, \text{id}_Q]^\# \circ T(\text{id}_I + 1_{\alpha_S(s)}) \circ \hat{\mu}_1 \circ T(\text{id}_I + c_x))(e) \\
&\quad \text{(shown above)} \\
&= (o_t \circ [i_t, 1_{\alpha_S(s)}]^\# \circ \hat{\mu}_1 \circ T(\text{id}_I + c_x))(e) \\
&= (\beta_{E'} \circ \alpha_S)(s)((\hat{\mu}_1 \circ T(\text{id}_I + c_x))(e)) \\
&\quad \text{(definition of } \beta_{E'}),
\end{aligned}$$

and therefore for all $x \in F(S+1)$ and $e \in E$,

$$\begin{aligned}
&(\beta_E \circ \delta_t \circ F(\alpha_S \circ [\text{id}_S, 1_{s_1}]))(x)(e) \\
&= (\beta_{E'} \circ \alpha_S)(s_1)((\hat{\mu}_1 \circ T(\text{id}_I + c_x))(e)) \\
&= (\beta_{E'} \circ \alpha_S)(s_2)((\hat{\mu}_1 \circ T(\text{id}_I + c_x))(e)) \quad \text{(assumption)} \\
&= (\beta_E \circ \delta_t \circ F(\alpha_S \circ [\text{id}_S, 1_{s_2}]))(x)(e).
\end{aligned}$$

Thus, it follows from Lemma 6.11 that $(\alpha_S, \beta_{E'})$ is locally consistent w.r.t. β_E . \square