A Categorical Framework for Learning Generalised Tree Automata

Gerco van Heerdt[1], Tobias Kappé[2], Jurriaan Rot[3], Matteo Sammartino[4,1], and Alexandra Silva[5,1]

[1] University College London
gerco.heerdt@ucl.ac.uk
[2] ILLC, University of Amsterdam
t.kappe@uva.nl
[3] Radboud University
jrot@cs.ru.nl
[4] Royal Holloway, University of London
matteo.sammartino@rhul.ac.uk
[5] Cornell University
alexandra.silva@cornell.edu

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 $L^\star$ 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.

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 [2], and network protocols [23]. 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 [25] and computational linguists [21]. The algorithm that caught the attention of the verification community is the one presented in Dana Angluin's seminal paper in 1987 [11]. She proves that it is possible to infer minimal deterministic automata in polynomial time using only so-called membership

and equivalence queries. Vaandrager's CACM article [45] provides an extensive review of the literature in automata learning and its applications to verification.

Angluin's algorithm, called $L^\star$, has served as a basis for many extensions that work for more expressive models than plain deterministic automata: I/O automata [4], weighted automata [16,31], register automata [34,40,1], nominal automata [39], and Büchi automata [12]. 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 $L^\star$ [35] and capture essential data structures abstractly, 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 [27], which then formed the basis of a wider project on developing a *Categorical Automata Learning Framework*—CALF.[6] CALF was described in the 2017 paper [32], but several problems were left open:

1. An abstract treatment of counterexamples: in the original $L^\star$ 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. Finding suitable constraints on the abstract framework to cover interesting examples, such as tree automata [20], that did not fit the constraints in [32].

In this paper, we resolve the open problems above, 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 $L^\star$ (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 $L^\star$ for models that are algebras of a given functor (Section 5).
3. An instantiation of the abstract algorithm to concrete categories (Section 6), providing the first abstractly derived learning algorithm for tree automata.

The present paper complements other recent work on abstract automata learning algorithms: Barlocco, Kupke, and Rot [15] gave an algorithm for coalgebras of a functor, whereas Urbat and Schröder [44] provided an algorithm for structures that can be represented as both algebras and coalgebras. More recently, Colcombet, Petrisan, and Stabile [19] gave an abstract learning algorithm based on modelling automata as functors. Our focus 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.

---

[6] `http://www.calf-project.org`

2   Preliminaries

We now introduce some categorical notions that we will need later in our technical development, and describe Angluin's original $\text{L}^\star$ algorithm. We assume some prior knowledge of category theory (categories, functors); see e.g., [14,36].

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 coproducts. When $f : X \to Z$ and $g : Y \to Z$, we write $[f, g]$ for the unique arrow from $X + Y$ to Z induced by the coproduct. We assume that $\mathbf{C}$ admits a fixed factorisation system $(\mathcal{E}, \mathcal{M})$, where $\mathcal{E}$ consists of epis and $\mathcal{M}$ consists of monos. We fix a *varietor* F in $\mathbf{C}$, that is, an endofunctor such that there is a free F-algebra monad (T, η, μ). We write γ_X for the F-algebra structure $FTX \to TX$, which is natural in X. Given an F-algebra (Y, y), we write $f^\sharp \colon (TX, \mu_X) \to (Y, y)$ for the extension of $f \colon X \to Y$ and denote $y^* = \text{id}_Y^\sharp \colon (TY, \mu_Y) \to (Y, y)$. We often implicitly apply forgetful functors. We fix an input object I and an output object O and write F_I for the functor $I + F(-)$. Lastly, we assume F preserves $\mathcal{E}$.

2.1   Abstract Automata

We recall the automaton definition from Arbib and Manes [13], which we will use in this paper, and its basic properties of accepted language and minimality.

Definition 1 (Automaton). *An* automaton *is a tuple* $\mathcal{A} = (Q, \delta, i, o)$ *consisting of a state space object Q, dynamics $\delta \colon FQ \to Q$, initial states $i \colon I \to Q$, and an output $o \colon Q \to O$. A* homomorphism *from $\mathcal{A}$ to $\mathcal{A}' = (Q', \delta', i', o')$ is an F-algebra morphism h from (Q, δ) to (Q', δ') — that is to say, a function $h \colon Q \to Q'$ 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.

Example 2. 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 function $1 \to Q$, and the classification of states into accepting and rejecting ones is represented by a function $Q \to 2$. 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 $\delta \colon Q \times A \to Q$ to $\delta^* \colon Q \times A^* \to Q$ is the usual one that lets the automaton read a word starting from a given state.

Definition 3 (Language). *A* language *is a morphism $TI \to O$. The language accepted by an automaton $\mathcal{A} = (Q, \delta, i, o)$ is given by $\mathcal{L}_\mathcal{A} = TI \xrightarrow{\text{reach}_\mathcal{A}} Q \xrightarrow{o} O$, where $\text{reach}_\mathcal{A} \colon TI \to Q$ is the* reachability map *of $\mathcal{A}$ given by $i^\sharp$.*

Algorithm 1 Make table closed and consistent	**Algorithm 2** L* algorithm
1: **function** FIX(S, E)	1: $S \leftarrow \{\varepsilon\}$
2: **while** T is not closed or not consistent **do**	2: $E \leftarrow \{\varepsilon\}$
3: **if** T is not closed **then**	3: $S, E \leftarrow$ FIX(S, E)
4: find $t \in S, a \in A$ such that $\forall s \in S. \ \mathsf{T}(ta) \neq \mathsf{T}(s)$	4: **while** EQ($\mathcal{H}_\mathsf{T}$) = c **do**
5: $S \leftarrow S \cup \{sa\}$	5: $S \leftarrow S \cup$ prefixes(c)
6: **else if** T is not consistent **then**	6: $S, E \leftarrow$ FIX(S, E)
7: find $s_1, s_2 \in S$, $a \in A$ and $e \in E$ such that $\mathsf{T}(s_1) = \mathsf{T}(s_2)$ and $\mathsf{T}(s_1 a)(e) \neq \mathsf{T}(s_2 a)(e)$	7: **return** $\mathcal{H}_\mathsf{T}$
8: $E \leftarrow E \cup \{ae\}$	
9: **return** S, E	

Definition 4 (Minimality [13]). *An automaton $\mathcal{A}$ is said to be* reachable *if* reach$_\mathcal{A} \in \mathcal{E}$. $\mathcal{A}$ *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 5. Recall the setting from Example 2. The reachability map reach$_\mathcal{A} : 1 \times A^* \to Q$ 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 $\mathcal{L}_\mathcal{A} : 1 \times A^* \to 2$ 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.

A general study of existence of minimal automata in this setting is given in [10]; see also [29].

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:

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 $\mathsf{T} : S \cup S \cdot A \to 2^E$, where $S, E \subseteq A^*$ are finite and $\mathsf{T}(s)(e) = \mathcal{L}(se)$. This function can be depicted as a table where elements of $S \cup S \cdot A$ label rows ($\cdot$ is pointwise concatenation) and elements of E label columns.

As an example, consider the table on the right, over the alphabet $A = \{a, b\}$, where $S = \{\varepsilon\}$ and $E = \{\varepsilon, b, ab\}$. 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.

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

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 $\mathcal{H}_\mathsf{T}$ from T: states of $\mathcal{H}_\mathsf{T}$ are distinct rows of T, corresponding to distinct Myhill–Nerode equivalence classes. Formally, $\mathcal{H}_\mathsf{T} = (Q, q_0, \delta, F)$ is defined as follows:

- $Q = \{\mathsf{T}(s) \mid s \in S\}$ is the set of states;
- $F = \{\mathsf{T}(s) \mid s \in S, \mathsf{T}(s)(\varepsilon) = 1\}$ is the set of final states;
- $q_0 = \mathsf{T}(\varepsilon)$ is the initial state;
- $\delta \colon Q \times A \to Q, (\mathsf{T}(s), a) \mapsto \mathsf{T}(sa)$ 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 $\mathsf{T}(sa) \in Q$ for all $sa \in S \cdot A$, and we must ensure that the choice of s to represent a row does not affect the transition. These constraints are captured in the following two properties.

Definition 6 (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* $\mathsf{T}(s) = \mathsf{T}(ta)$. *A table is* consistent *if for all* $s_1, s_2 \in S$ *with* $\mathsf{T}(s_1) = \mathsf{T}(s_2)$ *we have* $\mathsf{T}(s_1 a) = \mathsf{T}(s_2 a)$ *for any* $a \in A$.

Closedness and consistency form the core of $\mathsf{L}^\star$, 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). See Appendix A for an example.

3   The abstract data structures in CALF

We recall the basic notions underpinning CALF [32]: generalisations of the observation table, closedness, consistency and hypothesis. The generalised table is called a *wrapper*:

Definition 7 (Wrapper). *A* wrapper *for an object Q is a pair of morphisms*

$$\mathcal{W} = \left(S \xrightarrow{\alpha} Q \, , \, Q \xrightarrow{\beta} P \right)$$

We denote the factorisation of $\beta \circ \alpha$ by $S \xrightarrow{e_\mathcal{W}} H_\mathcal{W} \xrightarrowtail{m_\mathcal{W}} P$.

This will be instantiated with Q the state space of the target automaton, S a collection of row labels of an observation table, and P a collection of possible values of the rows. Then α *selects* states in Q, and β *classifies* them into P. We note that although such α and β underly the learning algorithm, they are not actually known to the learner, as they explicitly involve the unknown target automaton. However, we will see that we only need to represent certain compositions involving these morphisms, and that when α and β are chosen appropriately it will be possible to compute these compositions.

Example 8 (Observation table wrapper). Recall the DA setting from Example 2 and consider a DA $\mathcal{A} = (Q, \delta, i, o)$. For $S \subseteq A^*$ and $E \subseteq A^*$, we can define a wrapper $\mathcal{W} = \big(S \xrightarrow{\alpha_S} Q \ , \ Q \xrightarrow{\beta_E} 2^E \big)$ for Q as follows:

$$\alpha_S(w) = \mathsf{reach}_{\mathcal{A}}(*, w) \qquad\qquad \beta_E(q)(e) = (o \circ \delta^*)(q, e).$$

The composition $\beta_E \circ \alpha_S \colon S \to 2^E$ is precisely the top part of the observation table of $\mathsf{L}^\star$, with rows S and columns E. In fact, we have $(\beta_E \circ \alpha_S)(s)(e) = \mathcal{L}_{\mathcal{A}}(*, se)$. The image of $\beta_E \circ \alpha_S$ is the set of rows that appear in the table. In $\mathsf{L}^\star$, this set is used as states of the hypothesis, and in our setting can be obtained as $H_{\mathcal{W}}$, recalling that the (surjective, injective) factorisation system in **Set** gives factorisation through the image.

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

Definition 9 (Closedness and consistency). *Given a wrapper*

$$\mathcal{W} = \big(S \xrightarrow{\alpha} Q \ , \ Q \xrightarrow{\beta} P \big),$$

where Q is the state space of an automaton (Q, δ, i, o), we say that $\mathcal{W}$ is closed *if there exist morphisms $i_{\mathcal{W}} \colon I \to H_{\mathcal{W}}$ and $\mathsf{close}_{\mathcal{W}} \colon FS \to H_{\mathcal{W}}$ making the diagrams below commute.*

$$
\begin{array}{ccc}
I & \xrightarrow{\ i\ } & Q \\
{\scriptstyle i_{\mathcal{W}}}\big\downarrow & & \big\downarrow{\scriptstyle \beta} \\
H_{\mathcal{W}} & \xrightarrow[\ m_{\mathcal{W}}\ ]{} & P
\end{array}
\qquad\qquad
\begin{array}{ccccc}
FS & \xrightarrow{\ F\alpha\ } & FQ & \xrightarrow{\ \delta\ } & Q \\
{\scriptstyle \mathsf{close}_{\mathcal{W}}}\big\downarrow & & & & \big\downarrow{\scriptstyle \beta} \\
H_{\mathcal{W}} & & \xrightarrow[\ m_{\mathcal{W}}\ ]{} & & P
\end{array}
$$

Furthermore, we say that $\mathcal{W}$ is consistent *if there exist morphisms $o_{\mathcal{W}} \colon H_{\mathcal{W}} \to O$ and $\mathsf{cons}_{\mathcal{W}} \colon FH_{\mathcal{W}} \to P$ making the diagrams below commute.*

$$
\begin{array}{ccc}
S & \xrightarrow{\ e_{\mathcal{W}}\ } & H_{\mathcal{W}} \\
{\scriptstyle \alpha}\big\downarrow & & \big\downarrow{\scriptstyle o_{\mathcal{W}}} \\
Q & \xrightarrow[\ o\ ]{} & O
\end{array}
\qquad\qquad
\begin{array}{ccccc}
FS & \xrightarrow{\ Fe_{\mathcal{W}}\ } & & & FH_{\mathcal{W}} \\
{\scriptstyle F\alpha}\big\downarrow & & & & \big\downarrow{\scriptstyle \mathsf{cons}_{\mathcal{W}}} \\
FQ & \xrightarrow{\ \delta\ } & Q & \xrightarrow{\ \beta\ } & P
\end{array}
$$

Example 10. In the DA case, generalised closedness and consistency instantiate to the conditions allowing the hypothesis to be well-defined in $\mathsf{L}^\star$ (see Section 2.2):

Closedness: The wrapper (α_S, β_E) is closed if: (i) there exists $s \in S$ such that $(\beta_E \circ \alpha_S)(s) = (\beta_E \circ i)(*)$ and; (ii) for all $s \in S$ and $a \in A$ there exists $s_a \in S$ such that $(\beta_E \circ \alpha_S)(s_a) = (\beta_E \circ \delta)(\alpha_S(s), a)$. Condition (i) holds immediately if $\varepsilon \in S$—the function $(\beta_E \circ i)(*) \colon E \to 2$ maps $e \in E$ to $\mathcal{L}_{\mathcal{A}}(*, e)$. Condition (ii) corresponds to closedness in Definition 6. In fact, $\beta_E \circ \delta \circ (\alpha_S \times \mathrm{id}_A) \colon S \times A \to 2^E$ represents the lower part of the observation table associated with S and E.

Consistency: The wrapper (α_S, β_E) is consistent if: (iii) 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; (iv) for all $a \in A$ we have $(\beta_E \circ \delta)(\alpha_S(s_1), a) = (\beta_E \circ \delta)(\alpha_S(s_2), a)$. Condition (iii) holds immediately if $\varepsilon \in E$—the function $o \circ \alpha_S \colon S \to 2$ maps $s \in S$ to $\mathcal{L}_{\mathcal{A}}(*, s)$. Condition (iv) corresponds to consistency in Definition 6.

To determine these properties, we do not need the individual descriptions of α_S and β_E, which refer to the target automaton and are thus not available to the learner; we just need the compositions $\beta_E \circ \alpha_S$, $\beta_E \circ i$, $\beta_E \circ \delta \circ (\alpha_S \times \mathrm{id}_A)$, and $o \circ \alpha_S$, which can be determined using membership queries in this case. In general, for any instantiation of our abstract algorithm it will be important to show that these compositions (adapted to the wrapper and functor involved) can be determined and used concretely by the instantiated algorithm.

So far, we have used the wrapper to obtain the state space $\mathcal{H}_{\mathcal{W}}$ of the hypothesis. When a wrapper is closed and consistent, we can equip $\mathcal{H}_{\mathcal{W}}$ with a full automaton structure, leveraging the unique diagonal fill-in property of the factorisation.

Definition 11 (Hypothesis). *A closed and consistent wrapper*

$$\mathcal{W} = \left(S \xrightarrow{\alpha} Q \,,\, Q \xrightarrow{\beta} P \right)$$

for (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.*

4   Counterexamples, generalised

We now provide a key missing element for the development and analysis of an abstract learning algorithm in CALF: counterexamples. In the original $\mathrm{L}^{\star}$ algorithm, counterexamples are 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 is either not closed or not consistent, and hence needs to be extended. Such an extension, in turn, results in progress being

made in the algorithm. We show how we can use *recursive coalgebras* [41,43] as witnesses for discrepancies—i.e., as counterexamples—between a hypothesis and the target language in our abstract approach.[7] Here, and throughout the paper, we fix a *target* automaton $\mathcal{A}_t = (Q_t, \delta_t, i_t, o_t)$ whose language we want to learn.

Definition 12 (Recursive coalgebras). *An F-coalgebra $\rho\colon S \to FS$ is recursive if for every algebra $x\colon FX \to X$ there is a unique morphism $x^\rho\colon S \to X$ making the diagram below commute.*

$$
\begin{array}{ccc}
FS & \dashrightarrow^{Fx^\rho} & FX \\
\rho \uparrow & & \downarrow x \\
S & \dashrightarrow_{x^\rho} & X
\end{array}
$$

Example 13. A *prefix-closed* subset $S \subseteq A^*$ is easily equipped with a coalgebra structure $\rho\colon S \to 1 + S \times A$ that detaches the last letter from each non-empty word and assigns $*$ to the empty one. Such a coalgebra is recursive, with the unique map into an algebra being defined as a restricted reachability map. In fact, under certain conditions that are satisfied in the DA setting, recursivity of a coalgebra is equivalent to having a coalgebra homomorphism into the initial algebra [8, Corollary 5.6]. This means that every recursive coalgebra is isomorphic to one given by a prefix-closed multiset of words. If the unique morphism into the initial algebra is injective, then the multiset becomes a set.

Given an automaton $\mathcal{A} = (Q, \delta, i, o)$ and a recursive coalgebra $\rho\colon S \to F_I S$, the map $S \xrightarrow{[i,\delta]^\rho} Q$ can be seen as a generalised reachability map, allowing states in Q to be reached from S. We use this map to derive a notion of generalised language induced by a recursive coalgebra. This will be used to compare languages of the hypothesis and of the target automaton with respect to a specific recursive coalgebra, i.e., a specific counterexample.

Definition 14 (ρ-languages). *Given a recursive coalgebra $\rho\colon S \to F_I S$ and an automaton $\mathcal{A} = (Q, \delta, i, o)$, the ρ-language of $\mathcal{A}$ is $\mathcal{L}_{\mathcal{A}}^\rho = S \xrightarrow{[i,\delta]^\rho} Q \xrightarrow{o} O$.*

For instance, in the case of a DA $\mathcal{A}$ and a recursive coalgebra as in Example 13, $\mathcal{L}_{\mathcal{A}}^\rho$ is the restriction of the language of $\mathcal{A}$ to the prefix-closed set of words S.

In Algorithm 2, a counterexample is produced by the teacher (line 4) when the hypothesis does not agree with the target automaton. We now generalise counterexamples to wrappers: counterexamples are recursive coalgebras on which the languages of the hypothesis and of the target automaton disagree.

Definition 15 (Counterexample). *A closed and consistent wrapper $\mathcal{W}$ is said to be* correct up to *a recursive $\rho\colon S \to F_I S$ 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 $\rho\colon S \to F_I S$ such that $\mathcal{W}$ is not correct up to ρ.*

The following guarantees incorrect hypotheses yield counterexamples.

---

[7] Recursive coalgebras have been used to generalise prefix-closedness in an automata learning context in earlier work [28], as well as to generalise counterexamples [15,44].

Algorithm 3 Make wrapper closed and consistent

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

Algorithm 4 Abstract automata learning algorithm

1: $\alpha, \beta \leftarrow$ FIX$(!\colon 0 \to Q_\mathsf{t}, !\colon Q_\mathsf{t} \to 1)$
2: **while** $\mathsf{EQ}(\mathcal{H}_{(\alpha, \beta)}) = \rho\colon S \to F_I S$ **do**
3:    $\alpha \leftarrow \alpha'$ s.t. $\alpha'^{\lhd} = [\alpha, [i_\mathsf{t}, \delta_\mathsf{t}]^\rho]^{\lhd}$
4:    $\alpha, \beta \leftarrow$ FIX(α, β)
5: **return** $\mathcal{H}_{(\alpha, \beta)}$

Proposition 16 (Language equivalence via recursion). *Given an automaton $\mathcal{A} = (Q, \delta, i, o)$, we have $\mathcal{L}_{\mathcal{A}_\mathsf{t}} = \mathcal{L}_\mathcal{A}$ if and only if $\mathcal{L}_{\mathcal{A}_\mathsf{t}}^\rho = \mathcal{L}_\mathcal{A}^\rho$ for every recursive coalgebra $\rho\colon S \to F_I S$.*

Corollary 17 (Counterexample existence). *Given a closed and consistent wrapper $\mathcal{W}$ for Q_t, we have $\mathcal{L}_{\mathcal{H}_\mathcal{W}} \neq \mathcal{L}_{\mathcal{A}_\mathsf{t}}$ iff there exists a counterexample for $\mathcal{W}$.*

The next step in Algorithm 2 is to fix the table by adding all prefixes of the counterexample to S (line 5). We generalise this step by incorporating the counterexample given by a recursive coalgebra into the wrapper. In the DA case, this precisely corresponds to adding a prefix-closed subset to S. The following results say that doing so will lead to either a closedness or a consistency defect. In other words, we give theoretical guarantees that resolving counterexamples results in progress being made towards convergence.

Theorem 18 (Resolving counterexamples). *Given a closed and consistent wrapper $\mathcal{W} = \big(S \xrightarrow{\alpha} Q_\mathsf{t}, Q_\mathsf{t} \xrightarrow{\beta} P \big)$ and a recursive coalgebra $\rho\colon S' \to F_I S'$, the following holds. If the wrapper $\mathcal{W}' = ([\alpha, [i_\mathsf{t}, \delta_\mathsf{t}]^\rho], \beta)$ is closed and consistent, then $\mathcal{W}$ is correct up to ρ.*

This theorem is used contrapositively: given a closed and consistent wrapper, adding a counterexample yields a wrapper that is either not closed or inconsistent.

5   Generalised Learning Algorithm

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

algorithmic description operates on a wrapper (α, β), these individual morphisms will not be known to the learner. In fact, at this level of abstraction the descriptions should be seen as algorithmic templates rather than concrete algorithms. An instantiation must ensure that at least the compositions required to determine the closedness and consistency conditions and to construct the hypothesis can be maintained. These compositions are $\beta \circ \alpha$, $\beta \circ i_{\mathsf{t}}$, $\beta \circ \delta_{\mathsf{t}} \circ F\alpha$, and $o_{\mathsf{t}} \circ \alpha$. We have previously shown how these instantiate to recover $\mathsf{L}^\star$, and in Section 6 we will discuss the class of examples given by generalised tree automata.

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 (line 3, which will be explained in more detail later, when we define runs). The updated wrapper is passed on to the subroutine FIX (line 4) to be made closed and consistent.

A crucial point for Algorithm 3 is defining what it means to resolve the "current" closedness and consistency defects. We call these *local* defects, meaning the ones that can be directly detected in the current wrapper. For DAs, local closedness defects are rows from the bottom part missing in the top part, and the empty word row if it is missing. Local consistency defects are pairs of row labels which are distinguished by the target language, or with differing rows when the labels are extended with a single symbol.

We first introduce additional notions to formalise these ideas. We partially order the subobjects and quotients of the target automaton's state space Q_{t} in the usual way. Given two subobjects $j\colon J \hookrightarrow Q_{\mathsf{t}}$ and $k\colon K \hookrightarrow Q_{\mathsf{t}}$, we say $j \leq k$ if there is $f\colon J \to K$ such that $k \circ f = j$. Intuitively, j is "contained" in k. Given two quotients $x\colon Q_{\mathsf{t}} \twoheadrightarrow X$ and $y\colon Q_{\mathsf{t}} \twoheadrightarrow Y$, we say $x \leq y$ if there exists $g\colon X \to Y$ such that $y = g \circ x$. Intuitively, x is "finer" than y.

Now, consider a wrapper (α, β) for Q_{t}. We have that $\alpha^{\triangleleft}$ and $\beta^{\triangleright}$ are a subobject and a quotient of Q_{t}, respectively. For instance, in the DA case, $\alpha^{\triangleleft}$ is the set of states in Q_{t} currently represented by the table, and $\beta^{\triangleright}$ is the equivalence relation on states induced by the rows. We can now say another wrapper (α', β') is a locally closed extension of (α, β) if (a) it represents at least the same states of the target automaton as α, formalised $\alpha^{\triangleleft} \leq \alpha'^{\triangleleft}$, and; (b) it solves the closedness defects present in α. Local consistency is analogous: it requires the extended wrapper to distinguish at least the same states of Q_{t} as the original one.

Definition 19 (Local closedness and consistency). *Consider a wrapper*

$$\mathcal{W} = \left(\, S' \xrightarrow{\alpha'} Q_{\mathsf{t}} \, , \, Q_{\mathsf{t}} \xrightarrow{\beta'} P' \, \right)$$

We call $\mathcal{W}$ locally closed w.r.t. a morphism $\alpha\colon S \to Q_{\mathsf{t}}$ if $\alpha^{\triangleleft} \leq \alpha'^{\triangleleft}$ and there are morphisms $i_{\mathcal{W}}\colon I \to H_{\mathcal{W}}$ and $\mathsf{lclose}_{\mathcal{W},\alpha}\colon FS \to H_{\mathcal{W}}$ s.t. these diagrams commute:

$$
\begin{array}{ccc}
I & \xrightarrow{\;\; i_{\mathsf{t}} \;\;} & Q_{\mathsf{t}} \\
{\scriptstyle i_{\mathcal{W}}}\big\downarrow & & \big\downarrow{\scriptstyle \beta'} \\
H_{\mathcal{W}} & \xrightarrow{\;\; m_{\mathcal{W}} \;\;} & P'
\end{array}
\qquad\qquad
\begin{array}{ccccc}
FS & \xrightarrow{\;F\alpha\;} & FQ_{\mathsf{t}} & \xrightarrow{\;\delta_{\mathsf{t}}\;} & Q_{\mathsf{t}} \\
{\scriptstyle \mathsf{lclose}_{\mathcal{W},\alpha}}\big\downarrow & & & & \big\downarrow{\scriptstyle \beta'} \\
H_{\mathcal{W}} & & \xrightarrow{\;\; m_{\mathcal{W}} \;\;} & & P'
\end{array}
$$

Given $\beta\colon Q \to P$, we say that $\mathcal{W}$ is locally consistent *w.r.t. β if $\beta'^{\triangleright} \leq \beta^{\triangleright}$ and there exist morphisms $o_{\mathcal{W}}\colon H_{\mathcal{W}} \to O$ and $\mathsf{lcons}_{\mathcal{W},\beta}\colon FH_{\mathcal{W}} \to P$ making the diagrams below commute.*

$$
\begin{array}{ccc}
S' & \xrightarrow{\;e_{\mathcal{W}}\;} & H_{\mathcal{W}} \\
{\scriptstyle \alpha'}\downarrow & & \downarrow{\scriptstyle o_{\mathcal{W}}} \\
Q_{\mathsf{t}} & \xrightarrow{\;o_{\mathsf{t}}\;} & O
\end{array}
\qquad\qquad
\begin{array}{ccccc}
FS' & \xrightarrow{\;Fe_{\mathcal{W}}\;} & & & FH_{\mathcal{W}} \\
{\scriptstyle F\alpha'}\downarrow & & & & \downarrow{\scriptstyle \mathsf{lcons}_{\mathcal{W},\beta}} \\
FQ_{\mathsf{t}} & \xrightarrow{\;\delta_{\mathsf{t}}\;} & Q_{\mathsf{t}} & \xrightarrow{\;\beta\;} & P
\end{array}
$$

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 20. For the case of DAs, consider a wrapper $(\alpha_{S'}, \beta_{E'})$ representing an observation table (S', E') for the target DA $\mathcal{A}_{\mathsf{t}}$ (see Example 8):

Local closedness: Given $\alpha_S\colon S \to Q_{\mathsf{t}}$, $(\alpha_{S'}, \beta_{E'})$ is locally closed w.r.t. α_S if (1) $S \subseteq S'$ (to ensure $\alpha_S^{\triangleleft} \leq \alpha_{S'}^{\triangleleft}$); (2) S' contains the empty word (left diagram); and (3) any row in the bottom part of the table (S, E') occurs in the top part of (S', E') (right diagram).

Local consistency: Similarly, given $\beta_E\colon Q_{\mathsf{t}} \to 2^E$, $(\alpha_{S'}, \beta_{E'})$ is locally consistent w.r.t. β_E if (1) $E \subseteq E'$ (to ensure $\beta_{E'}^{\triangleright} \leq \beta_E^{\triangleright}$); (2) E' contains the empty word (left diagram); and (3) for all $s, s' \in S'$ and $a \in A$, if s and s' map to the same row in the top part of (S', E'), then the rows for sa and $s'a$ are the same in the bottom part of (S', E) (right diagram).

In Algorithm 3 we assume that we can always find locally closed and consistent wrappers (lines 4 and 6 respectively). This assumption holds in general for local closedness: for each wrapper (α, β) for Q_{t} we can always find α' such that (α', β) is locally closed w.r.t. α.

Lemma 21. *Given a wrapper (α, β) for Q_{t}, $([\alpha, [i_{\mathsf{t}}, \delta_{\mathsf{t}}] \circ F_I\alpha], \beta)$ is locally closed w.r.t. α.*

This result is enabled by the algebraic nature of automata. Local consistency is not inherently algebraic, so ensuring it takes more effort. We shall see in Section 6 that existence of locally closed/consistent extensions can be proved constructively for a broad class of automata.

Termination. To analyse termination of Algorithm 4, we introduce its *runs*.

Definition 22 (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. *$\alpha_0\colon 0 \to Q_{\mathsf{t}}$ and $\beta_0\colon Q_{\mathsf{t}} \to 1$ are the unique morphisms;*
2. *if $\mathcal{W}_n$ is not closed, then $\beta_{n+1} = \beta_n$ and α_{n+1} is s.t. (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 $\rho\colon S \to F_I S$ for $\mathcal{W}_n$, then $\alpha_{n+1}^{\triangleleft} = [\alpha_n, [i_{\mathsf{t}}, \delta_{\mathsf{t}}]^{\rho}]^{\triangleleft}$ 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 $\mathcal{W}_{n+1} = \mathcal{W}_n$.

Note that, in point 4 above and in line 3 of Algorithm 4, we admit a more general counterexample resolution than Theorem 18: we only require that α_{n+1} and $[\alpha_n, [i_t, \delta_t]^\rho]$ represent the same states of the target automaton. This captures how observation tables are updated in practice; for instance in $\mathsf{L}^\star$ a counterexample prefix already in the table is discarded.

Proposition 23. *Algorithm 4 halts if and only if for all runs* $\{\mathcal{W}_n\}_{n \in \mathbb{N}}$ *there is* n *with* $\mathcal{W}_{n+1} = \mathcal{W}_n$.

We can establish an invariant on the order of subsequent wrappers in runs.

Lemma 24. *Let* $\{\mathcal{W}_n = (\alpha_n, \beta_n)\}_{n \in \mathbb{N}}$ *be a run. For all* $n \in \mathbb{N}$, *we have* $\alpha_n^\triangleleft \leq \alpha_{n+1}^\triangleleft$ *and* $\beta_{n+1}^\triangleright \leq \beta_n^\triangleright$. *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* $\beta_{n+1} = \beta_n$.

   Putting these results together, we conclude that the algorithm terminates with a correct automaton, which is minimal under certain conditions. Satisfaction of the requirement on recursive coalgebras ρ_k depends on the implementation of counterexamples and closing of wrappers; for DAs, it suffices to keep the set of row labels S prefix-closed.

Theorem 25 (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 its hypothesis is correct. If* $\mathcal{A}_t$ *is minimal and for all* $k \in \mathbb{N}$ *there exists a recursive* $\rho_k \colon S_k \to F_I S_k$ *such that* $\alpha_k = [i_t, \delta_t]^{\rho_k}$, *then the final hypothesis is minimal.*

6   Generalised Tree Automata

We now 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—including bottom-up tree automata and unordered tree automata. We first introduce the running examples.

Example 26 (Tree automata). Let Γ be a ranked alphabet, i.e., a finite set where $\gamma \in \Gamma$ comes with $\mathsf{arity}(\gamma) \in \mathbb{N}$. 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 $t_1, \dots, t_{\mathsf{arity}(\gamma)} \in T_\Gamma(I)$ implies $(\gamma, t_1, \dots, t_{\mathsf{arity}(\gamma)}) \in T_\Gamma(I)$. The alphabet Γ gives rise to the polynomial functor $FX = \coprod_{\gamma \in \Gamma} X^{\mathsf{arity}(\gamma)}$. The 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 [20] is then an automaton over F, with finite I and $O = 2$.

Example 27 (Unordered tree automata). Consider the finite powerset functor $\mathcal{P}_f \colon \mathbf{Set} \to \mathbf{Set}$, mapping a set to its finite subsets. The corresponding free

$\mathcal{P}_\mathsf{f}$-monad maps a set X to the set of finitely-branching unordered trees with nodes in X. Automata over $\mathcal{P}_\mathsf{f}$, 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 $\Gamma = \{\mathfrak{s}_i \mid i \in \mathbb{N}\}$, where $\mathsf{arity}(\mathfrak{s}_i) = i$, 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* [9]—i.e., they are finitary and preserve finite sets—and they preserve weak pullbacks. We turn these into a global assumption, used in several places; in particular, that F is strongly finitary is used to guarantee the existence of finite counterexamples.

Assumption 28. *In the remainder we take* $\mathbf{C} = \mathbf{Set}$ *with the (surjective, injective) factorisation system, assume F is strongly finitary and preserves weak pullbacks, and I finite.*

If the target automaton $\mathcal{A}_\mathsf{t}$ is finite, the algorithm terminates by Theorem 25.

We start with the central notion of *contextual wrapper*, a specific form of wrapper using contexts to generalise string concatenation to trees. We then show that contextual wrappers enable effective procedures for local closedness and consistency, and for computing hypotheses. Moreover, they can always be updated via finite counterexamples. Altogether, this makes the ingredients of our abstract algorithm concrete for generalised tree automata.

6.1   Contextual wrappers

Denote by 1 the set $\{\square\}$. Given $x \in X$ for any set X, we write $\mathfrak{e}_x$ for the function $1 \to X$ that assigns x to $\square$. 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 Examples 26 and 27. In fact, it is well known that $T(I + (-))$ forms a monad with unit $\hat{\eta}_X$ turning each hole from X into a context, and multiplication $\hat{\mu}_X$ plugging a context into another context [38].

Definition 29 (Contextual wrapper). *Let $S \subseteq TI$ and $E \subseteq T(I+1)$. Now:*
  – $\alpha_S \colon S \to Q_\mathsf{t}$ is defined as the restriction of the reachability map of $\mathcal{A}_\mathsf{t}$ to S;
  – $\beta_E \colon Q_\mathsf{t} \to O^E$ is defined as the function given by $\beta_E(q)(e) = (o_\mathsf{t} \circ [i_\mathsf{t}, \mathfrak{e}_q]^\sharp)(e)$.
A wrapper is called contextual *if it is of the form (α_S, β_E) for some S and E.*

Intuitively, β_E classifies states by plugging them into every context in E and comparing the resulting outputs. In the DA case, contextual wrappers are equivalent to those of Example 8, where row labels are plugged into word contexts—i.e., words of the form $\square \cdot e$, with $e \in A^*$—to achieve string concatenation.

We now show how to compute several morphisms induced by a wrapper. These morphisms, intuitively, correspond to different parts of an observation table, and are used for (local) closedness and consistency, and to construct the hypothesis. In particular, we show that they can be computed concretely by querying the language $\mathcal{L}_{\mathcal{A}_\mathsf{t}}$, i.e., via membership queries.

Proposition 30 (Computing wrapper morphisms). *Given $S \subseteq TI$ with inclusion $j \colon S \to TI$ and $E \subseteq T(I+1)$ with inclusion $k \colon E \to T(I+1)$, we have:*
- *The* top observation table $\beta_E \circ \alpha_S \colon S \to O^E$, $s \mapsto \mathcal{L}_{\mathcal{A}_t} \circ \mu_I \circ T[\eta_I, j \circ \mathfrak{e}_s] \circ k$;
- *The* bottom observation table $\beta_E \circ \delta_t \circ F\alpha_S \colon FS \to O^E$, $t \mapsto \mathcal{L}_{\mathcal{A}_t} \circ \mu_I \circ T[\eta_I, \gamma_I \circ Fj \circ \mathfrak{e}_t] \circ k$;
- *The* input rows $\beta_E \circ i_t \colon I \to O^E$ *given by* $(\beta_E \circ i_t)(x) = \mathcal{L}_{\mathcal{A}_t} \circ T[\mathrm{id}_I, \mathfrak{e}_x] \circ k$;
- *The* row output $o_t \circ \alpha_S \colon S \to O$ *given by* $(o_t \circ \alpha_S)(s) = \mathcal{L}_{\mathcal{A}_t}(s)$.

Example 31. For tree automata, a contextual wrapper is as follows: $S \subseteq T_\Gamma(I)$ 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, or equivalently $\Gamma + \square$-trees, where $\Gamma + \square$ is the signature Γ extended with an additional constant $\square$. Plugging into a context intuitively amounts to replacing this leaf with a tree.

We now give the intuition behind the maps of Proposition 30:
- The top part of the observation table has rows labelled by trees in S, columns by contexts in E, and rows are computed by plugging their tree 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 [22,17].
- The bottom part contains rows labelled over elements of $FS = \coprod_{\gamma \in \Gamma} S^{\mathsf{arity}(\gamma)}$, i.e., trees obtained by adding a new root symbol to those from S. This generalises adding an alphabet symbol to row labels, as done in the bottom observation table of $\mathsf{L}^\star$. Rows are computed as in the top part, by plugging their tree labels into contexts E and querying.
- The input rows are those for the leaves I, and the row output function queries the language for each row label.

The case of unordered trees is analogous, with a key difference: wrapper maps are now up to equations, as both S and E are sets of unordered trees. The corresponding observation table can be understood as containing equivalence classes of rows and columns. For instance, the bottom part has only 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 $\mathsf{arity}(\gamma)$-list of trees from S.

Hypotheses. Recall that, given a closed and consistent wrapper (α_S, β_E), the state space of the associated hypothesis is given by the image of $\beta_E \circ \alpha_S \colon S \to O^E$. Since S and E are finite sets, we can compute the image of this function. For bottom-up and unordered tree automata, as in the DA case (see Example 8), this image consists of distinct rows. The initial states, outputs and dynamics of the hypothesis automaton are defined as follows:

$$i_{\mathcal{H}_\mathcal{W}}(x) = (\beta_E \circ i_t)(x) \qquad o_{\mathcal{H}_\mathcal{W}}(e_\mathcal{W}(s)) = (o_t \circ \alpha_S)(s)$$

$$\delta_{\mathcal{H}_\mathcal{W}}(F(e_\mathcal{W})(x)) = (\beta_E \circ \delta_t \circ F\alpha_S)(x).$$

Closedness and consistency ensure well-definedness. We know from Proposition 30 how to compute those functions via membership queries.

6.2   Witnessing local closedness and consistency

We now consider local closedness and consistency. In the current setting, these amount to equality checks on finite structures, which can be performed effectively.

Lemma 32 (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 $k\colon I \to S'$ and $\ell\colon FS \to S'$ such that (1) $\alpha_{S'} \circ k = i_{\mathsf{t}}$ and (2) $\alpha_{S'} \circ \ell = \delta_{\mathsf{t}} \circ F\alpha_S$.*

Example 33. For bottom-up tree automata, local closedness holds if the table (S', E) already contains each leaf row (equation 1), and it contains every successor row for S, namely $FS = \coprod_{\gamma \in \Gamma} S^{\mathsf{arity}(\gamma)}$ (equation 2). For unordered tree automata the condition is similar, and now involves successor trees in $\mathcal{P}_{\mathsf{f}}(S)$.

Lemma 34 (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: (1) $(o_{\mathsf{t}} \circ \alpha_S)(s) = (o_{\mathsf{t}} \circ \alpha_S)(s')$; and (2) $\beta_E \circ \delta_{\mathsf{t}} \circ F(\alpha_S \circ [\mathrm{id}_S, \mathfrak{e}_s]) = \beta_E \circ \delta_{\mathsf{t}} \circ F(\alpha_S \circ [\mathrm{id}_S, \mathfrak{e}_{s'}])$. Then $\mathcal{W} = (\alpha_S, \beta_{E'})$ is locally consistent w.r.t. β_E.*

Example 35. For bottom-up tree automata, local consistency amounts to require the following for the table for (S, E'). For all $s, s' \in S$ corresponding to the same row we must have: (1) s and s' are both accepted/rejected; (2) successor rows obtained by plugging s and s' into the same one-level context from $F(S+1) = \coprod_{\gamma \in \Gamma} (S + \{\square\})^{\mathsf{arity}(\gamma)}$ are equal.

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'.

We now develop procedures for fixing local closedness and consistency defects. First, we show that we can always extend S to make the wrapper locally closed.

Proposition 36. *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 $\rho\colon S \to F_I S$ such that $[\eta_I, \gamma_I]^\rho\colon S \to TI$ is the inclusion, then there exists a recursive $\rho'\colon S' \to F_I S'$ such that $[\eta_I, \gamma_I]^{\rho'}\colon S' \to TI$ is the inclusion.*

The condition of $[\eta_I, \gamma_I]^\rho\colon S \to TI$ being the inclusion map in the above result amounts to prefix closedness of S in tree automata, see Example 37 below. Further, under this condition we have that $\alpha_S = [i_{\mathsf{t}}, \delta_{\mathsf{t}}]^\rho$, since the reachability map is an algebra morphism, and similarly for $\alpha_{S'}$. This is crucial to satisfy the requirements for minimality of the termination theorem.

Example 37. To better understand the above proposition, it is worth describing what recursive coalgebras are for the automata of Examples 26 and 27. For bottom-up tree automata, they are coalgebras $\rho\colon S \to \coprod_{\gamma \in \Gamma} S^{\mathsf{arity}(\gamma)} + I$ satisfying suitable conditions. Subtree-closed subsets of $T_\Gamma(I)$ are sets of trees closed under taking subtrees. Every subtree-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, $\rho\colon S \to \mathcal{P}_{\mathsf{f}}S + I$ will just return the set of subtrees or a leaf.

The proof of Proposition 36, which can be found in Appendix D, is constructive and describes a naive procedure to make a table locally closed: adding all (finitely-many) successor rows to the table. For instance, in the case of tree automata, one adds rows obtained by adding a new root symbol to trees labelling rows in all possible ways, for each symbol in the alphabet. One may optimise the algorithm by instead adding only missing rows.

   We now show how to fix local consistency, by extending a finite set of column labels E to a finite set E' such that the resulting wrapper is locally consistent.

Proposition 38. *Given finite $S \subseteq TI$ and $E \subseteq T(I+1)$, define $E' \subseteq T(I+1)$ by $E' = E \cup \{(\eta_{I+1} \circ \kappa_2)(\square)\} \cup \{(\hat{\mu}_1 \circ T(\mathrm{id}_I + c_x))(e) \mid e \in E, x \in F(S+1)\}$ where $c_x\colon 1 \to T(I+1)$, with $c_x = \gamma_{I+1} \circ F[T\kappa_1 \circ j, \hat{\eta}_1] \circ \mathfrak{e}_x$, and $j\colon S \to TI$ is set inclusion. It holds that E' is finite and $(\alpha_S, \beta_{E'})$ is locally consistent w.r.t. β_E.*

   For tree automata, E' is E plus the empty context $\square$ and the trees obtained by plugging one-level contexts formed from the current row labels (see Example 35) into all contexts in E. This amounts to extending columns so that *all* consistency defects are fixed. One can optimise the procedure above by incrementally adding to E only those elements of E' that result in new pairs of rows being distinguished.

6.3   Finite counterexamples

Finally, we show that the teacher can always supply a finite counterexample.

Proposition 39 (Language equivalence via finite recursion). *Given an automaton $\mathcal{A} = (Q, \delta, i, o)$, we have $\mathcal{L}_{\mathcal{A}_{\mathsf{t}}} = \mathcal{L}_{\mathcal{A}}$ iff $\mathcal{L}_{\mathcal{A}_{\mathsf{t}}}^{\rho} = \mathcal{L}_{\mathcal{A}}^{\rho}$ for all recursive $\rho\colon S \to F_I S$ such that S is finite.*

Corollary 40 (Finite counterexamples). *Given a closed and consistent wrapper $\mathcal{W}$ for Q_{t}, we have $\mathcal{L}_{\mathcal{H}_{\mathcal{W}}} \neq \mathcal{L}_{\mathcal{A}_{\mathsf{t}}}$ iff there exists a counterexample $\rho\colon S \to F_I S$ for $\mathcal{W}$ such that S is finite.*

Example 41. Recall from Example 37 that finite recursive coalgebras for bottom-up (resp. unordered) tree automata are coalgebras $\rho\colon S \to \coprod_{\gamma \in \Gamma} S^{\mathsf{arity}(\gamma)} + I$ (resp. $\rho\colon S \to \mathcal{P}_{\mathsf{f}}S + I$). Thus, finite counterexamples are recursive coalgebras of this form where S is finite or, more concretely, a finite subtree-closed set of trees.

Given a finite counterexample, if α_S arises from a recursive coalgebra ρ (e.g., when S is prefix-closed), updating the wrapper in line 3 of Algorithm 4 can be done as follows: (1) combine ρ with the recursive coalgebra in Corollary 40 via a coproduct (which preserves recursiveness); (2) take a suitable factorisation to make sure that there is an inclusion of S into TI, and thus that the updated $\alpha_{S'}$ forms a contextual wrapper (see Lemma 48 for a formal justification). Concretely, the latter step amounts to removing multiple copies of rows with the same label. Altogether, these steps take the union of the current rows with the (prefix-closed) counterexample, and guarantee that $\alpha_{S'}$ again arises from a recursive coalgebra.

6.4   Minimality

Theorem 25 gives sufficient conditions for minimality of the automaton obtained from the algorithm, namely: each α arises from a recursive coalgebra, and the target automaton should be minimal. For the first condition to hold, there are two parts of the algorithm that need to be implemented appropriately, as they change α: closing the table and adding counterexamples. This can always be done: for closing the table, this follows from Proposition 36; for counterexamples, the strategy outlined in the previous section yields a wrapper of the desired form. As for the second condition, a minimal automaton exists if the functor F preserves arbitrary cointersections, which is the case iff F is finitary [10].

7   Related work

This paper proceeds in the line of work on categorical automata learning started in [35], and further developed in the CALF framework [32,33]. CALF provides abstract definitions of closedness, consistency, and hypothesis and several techniques to analyse and guide the development of concrete learning algorithms. CALF operates at a high level of abstraction and previously did not include an explicit learning algorithm. We discuss two further recent categorical approaches to learning, which make stronger assumptions than CALF in order to allow for the definition of concrete algorithms. The present paper is a third such approach.

Barlocco et al. [15] proposed an abstract algorithm for learning coalgebras, where tests are 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 what enables to define an actual algorithm in [15]. The current work focuses on algebras rather than coalgebras, and is orthogonal. In particular, it covers (bottom-up) tree automata, which are outside the scope of [15].

Urbat and Schröder proposed another categorical approach to automata learning [44], which—similarly to the work of Barlocco et al.—makes stronger assumptions than 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 [44]. In the present paper, this is achieved by learning algebras directly.

Yet another categorical approach to learning was proposed recently by Colcombet, Petrisan, and Stabile [19]. Here, the way automata are modelled is rather

different: not as algebras or coalgebras within a category, but as functors from a structure category to an output category. So far this has led the authors to develop an abstract automata learning algorithm that generalises algorithms for DFAs, weighted automata, and subsequential transducers. However, as their structure category is built by generating morphisms representing words by starting with a morphism for each alphabet symbol and closing under composition, it is unclear whether this approach could cover tree automata.

Concrete algorithms for learning tree automata and languages have appeared in the literature. The inference of regular tree languages using membership and equivalence queries appeared in [22], extending earlier work of Sakakibara [42]. Later, [17] 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. 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

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. It remains an open challenge to extend the present algorithm to a setting with more general equations. For the concrete case of pomset languages [24,26] represented by *bimonoids* [37], we note that we have succesfully instantiated the abstract algorithm described in this paper, and augmented it to include optimisations specific to the equations that hold in that setting [30]. In future work, we aim to extend the ideas behind these optimisations to the abstract setting, as well.

Another direction is to extend the framework with side-effects, encoded by a monad, in the style of [33]. 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.

References

1. Aarts, F., Fiterău-Broștean, P., Kuppens, H., Vaandrager, F.W.: Learning register automata with fresh value generation. In: ICTAC. pp. 165–183 (2015). https://doi.org/10.1007/978-3-319-25150-9_11

2. Aarts, F., de Ruiter, J., Poll, E.: Formal models of bank cards for free. In: ICST. pp. 461–468 (2013). https://doi.org/10.1109/ICSTW.2013.60
3. Aarts, F., Schmaltz, J., Vaandrager, F.W.: Inference and abstraction of the biometric passport. In: ISoLA. pp. 673–686 (2010). https://doi.org/10.1007/978-3-642-16558-0_54
4. Aarts, F., Vaandrager, F.W.: Learning I/O automata. In: CONCUR. pp. 71–85 (2010). https://doi.org/10.1007/978-3-642-15375-4_6
5. Adámek, J.: Free algebras and automata realizations in the language of categories. Commentationes Mathematicae Universitatis Carolinae **15**(4), 589–602 (1974)
6. Adámek, J., Herrlich, H., Strecker, G.E.: Abstract and Concrete Categories - The Joy of Cats. Dover Publications (2009)
7. Adámek, J., Lücke, D., Milius, S.: Recursive coalgebras of finitary functors. RAIRO - TIA **41**(4), 447–462 (2007). https://doi.org/10.1051/ita:2007028
8. Adámek, J., Milius, S., Moss, L.S.: On well-founded and recursive coalgebras. In: FoSSaCS. pp. 17–36 (2020). https://doi.org/10.1007/978-3-030-45231-5_2
9. Adámek, J., Milius, S., Velebil, J.: Free iterative theories: a coalgebraic view. MFCS **13**(2), 259–320 (2003). https://doi.org/10.1017/S0960129502003924
10. Adámek, J., Trnková, V.: Automata and algebras in categories. Kluwer (1989)
11. Angluin, D.: Learning regular sets from queries and counterexamples. Inf. Comput. **75**(2), 87–106 (1987). https://doi.org/10.1016/0890-5401(87)90052-6
12. Angluin, D., Fisman, D.: Learning regular omega languages. Theor. Comput. Sci. **650**, 57–72 (2016). https://doi.org/10.1016/j.tcs.2016.07.031
13. Arbib, M.A., Manes, E.G.: A categorist's view of automata and systems. In: Category Theory Applied to Computation and Control. pp. 51–64 (1974). https://doi.org/10.1007/3-540-07142-3_61
14. Awodey, S.: Category theory. Oxford University Press (2010)
15. Barlocco, S., Kupke, C., Rot, J.: Coalgebra learning via duality. In: FoSSaCS. pp. 62–79 (2019). https://doi.org/10.1007/978-3-030-17127-8_4
16. Bergadano, F., Varricchio, S.: Learning behaviors of automata from multiplicity and equivalence queries. SIAM J. Comput. **25**(6), 1268–1280 (Dec 1996). https://doi.org/10.1137/S009753979326091X
17. Besombes, J., Marion, J.: Learning tree languages from positive examples and membership queries. Theor. Comput. Sci. **382**(3), 183–197 (2007). https://doi.org/10.1016/j.tcs.2007.03.038
18. Capretta, V., Uustalu, T., Vene, V.: Recursive coalgebras from comonads. Information and Computation **204**(4), 437–468 (2006). https://doi.org/10.1016/j.ic.2005.08.005
19. Colcombet, T., Petrisan, D., Stabile, R.: Learning automata and transducers: A categorical approach. In: CSL. vol. 183, pp. 15:1–15:17 (2021). https://doi.org/10.4230/LIPIcs.CSL.2021.15
20. Comon, H., Dauchet, M., Gilleron, R., Jacquemard, F., Lugiez, D., Löding, C., Tison, S., Tommasi, M.: Tree Automata Techniques and Applications (2008), https://hal.inria.fr/hal-03367725
21. de la Higuera, C.: Grammatical Inference: Learning Automata and Grammars. Cambridge University Press, USA (2010)
22. Drewes, F., Högberg, J.: Learning a regular tree language from a teacher. In: DLT. pp. 279–291 (2003). https://doi.org/10.1007/3-540-45007-6_22
23. Fiterău-Broştean, P., Janssen, R., Vaandrager, F.W.: Combining model learning and model checking to analyze TCP implementations. In: CAV. pp. 454–471 (2016). https://doi.org/10.1007/978-3-319-41540-6_25

24. Gischer, J.L.: The equational theory of pomsets. Theor. Comput. Sci. **61**, 199–224 (1988). https://doi.org/10.1016/0304-3975(88)90124-7

25. Gold, E.M.: System identification via state characterization. Automatica **8**(5), 621–636 (Sep 1972). https://doi.org/10.1016/0005-1098(72)90033-7

26. Grabowski, J.: On partial languages. Fundam. Inform. **4**(2), 427 (1981)

27. van Heerdt, G.: An Abstract Automata Learning Framework. Master's thesis, Radboud Universiteit Nijmegen (2016)

28. van Heerdt, G., Jacobs, B., Kappé, T., Silva, A.: Learning to coordinate. In: It's All About Coordination, pp. 139–159. Springer (2018). https://doi.org/10.1007/978-3-319-90089-6_10

29. van Heerdt, G., Kappé, T., Rot, J., Sammartino, M., Silva, A.: Tree automata as algebras: Minimisation and determinisation. In: CALCO. vol. 139, pp. 6:1–6:22 (2019). https://doi.org/10.4230/LIPIcs.CALCO.2019.6

30. van Heerdt, G., Kappé, T., Rot, J., Silva, A.: Learning pomset automata. In: FOSSACS. pp. 510–530 (2021). https://doi.org/10.1007/978-3-030-71995-1_26

31. van Heerdt, G., Kupke, C., Rot, J., Silva, A.: Learning weighted automata over principal ideal domains. In: FoSSaCS. pp. 602–621 (2020). https://doi.org/10.1007/978-3-030-45231-5_31

32. van Heerdt, G., Sammartino, M., Silva, A.: CALF: categorical automata learning framework. In: CSL. pp. 29:1–29:24 (2017). https://doi.org/10.4230/LIPIcs.CSL.2017.29

33. van Heerdt, G., Sammartino, M., Silva, A.: Learning automata with side-effects. In: CMCS. pp. 68–89 (2020). https://doi.org/10.1007/978-3-030-57201-3_5

34. Isberner, M., Howar, F., Steffen, B.: The open-source learnlib - A framework for active automata learning. In: CAV. pp. 487–495 (2015). https://doi.org/10.1007/978-3-319-21690-4_32

35. Jacobs, B., Silva, A.: 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. pp. 384–406 (2014). https://doi.org/10.1007/978-3-319-06880-0_20

36. Lane, S.M.: Categories for the Working Mathematician. Graduate Texts in Mathematics, Springer New York (1998)

37. Lodaya, K., Weil, P.: Series-parallel languages and the bounded-width property. Theoretical Computer Science **237**(1), 347–380 (2000). https://doi.org/10.1016/S0304-3975(00)00031-1

38. Lüth, C., Ghani, N.: Composing monads using coproducts. In: ICFP. pp. 133–144 (2002). https://doi.org/10.1145/581478.581492

39. Moerman, J., Sammartino, M., Silva, A., Klin, B., Szynwelski, M.: Learning nominal automata. In: POPL. pp. 613–625 (2017). https://doi.org/10.1145/3009837.3009879

40. Mues, M., Howar, F., Luckow, K.S., Kahsai, T., Rakamaric, Z.: Releasing the PSYCO: using symbolic search in interface generation for Java. ACM SIGSOFT Software Engineering Notes **41**(6), 1–5 (2016). https://doi.org/10.1145/3011286.3011298

41. Osius, G.: Categorical set theory: a characterization of the category of sets. Journal of Pure and Applied Algebra **4**(1), 79–119 (1974)

42. Sakakibara, Y.: Learning context-free grammars from structural data in polynomial time. Theor. Comput. Sci. **76**(2?3), 223–242 (Nov 1990). https://doi.org/10.1016/0304-3975(90)90017-C

43. Taylor, P.: Practical foundations of mathematics. Cambridge University Press (1999)

44. Urbat, H., Schröder, L.: Automata learning: An algebraic approach. In: LICS. pp. 900–914 (2020). https://doi.org/10.1145/3373718.3394775
45. Vaandrager, F.W.: Model learning. Commun. ACM **60**(2), 86–95 (2017). https://doi.org/10.1145/2967606

A   Example of execution of $L^\star$

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 1a. It invokes Algorithm 1 to check closedness and consistency of this table: it is not closed, because $\mathsf{T}(a)$ 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 1b) is closed and consistent, so the algorithm builds the hypothesis $\mathcal{H}$ shown in Figure 1c and poses the equivalence query $\mathsf{EQ}(\mathcal{H})$. 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 1d) is closed, but not consistent: $\mathsf{T}(\varepsilon) = \mathsf{T}(aa)$, but $\mathsf{T}(\varepsilon \cdot a) \neq \mathsf{T}(aa \cdot a)$. The algorithm adds $a \cdot \varepsilon = a$ to E so that $\mathsf{T}(\varepsilon)$ and $\mathsf{T}(aa)$ can be distinguished, as depicted in Figure 1e. The table is now closed and consistent, and the new hypothesis automaton is precisely the minimal DFA accepting $\mathcal{L}$ (Figure 1f).
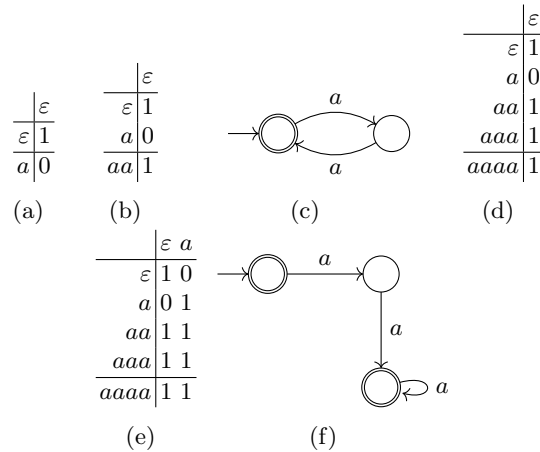
Fig. 1: Example run of $L^\star$ on $\mathcal{L} = \{w \in \{a\}^* \mid |w| \neq 1\}$.

B   Proofs for Section 4

Proposition 16 (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 $\mathcal{L}^\rho_{\mathcal{A}_t} = \mathcal{L}^\rho_{\mathcal{A}}$ for every recursive coalgebra $\rho \colon S \to F_I S$.*

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

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

Theorem 18 (Resolving counterexamples). *Given a closed and consistent wrapper $\mathcal{W} = \left(S \xrightarrow{\alpha} Q_t \,,\, Q_t \xrightarrow{\beta} P \right)$ and a recursive coalgebra $\rho \colon S' \to F_I S'$, 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 $h \colon H_{\mathcal{W}} \to H_{\mathcal{W}'}$ on the right.

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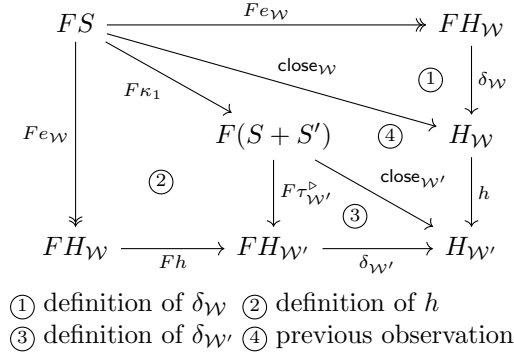
with the initial states ($h \circ i_{\mathcal{W}} = i_{\mathcal{W}'}$) and outputs ($o_{\mathcal{W}'} \circ h = o_{\mathcal{W}}$).

① closedness ② definition of h ③ consistency
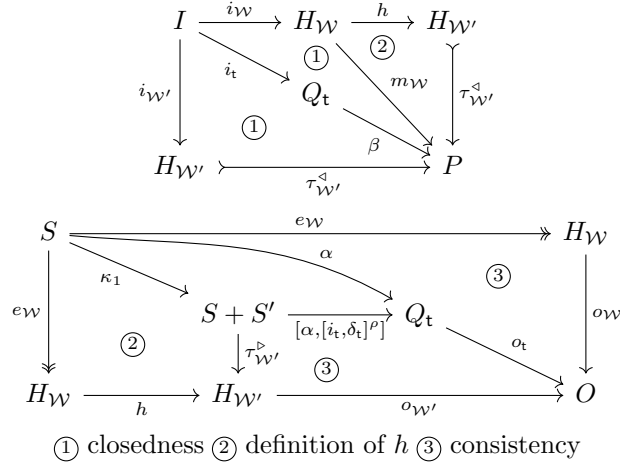
As for the transition functions, we use that $\tau_{\mathcal{W}'}^{\triangleleft}$ is a mono to show that $h \circ \mathsf{close}_{\mathcal{W}} = \mathsf{close}_{\mathcal{W}'} \circ F\kappa_1$ with the commutative diagram below.

① closedness ② definition of h
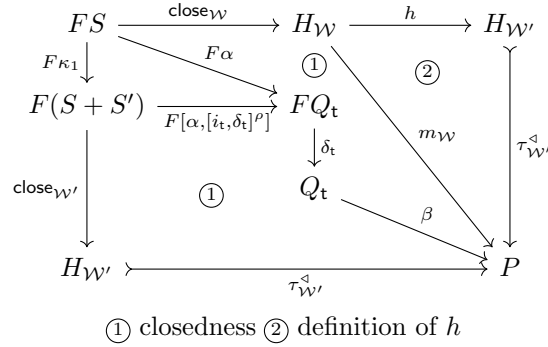
We are now ready to show that $h \circ \delta_{\mathcal{W}} = \delta_{\mathcal{W}'} \circ Fh$. This follows from commutativity of the diagram below using the fact that $Fe_{\mathcal{W}}$ is an epi.

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 $\mathcal{H}_{\mathcal{W}} \to \mathcal{H}_{\mathcal{W}'}$. 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.

$$\tag{1}$$

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

① definition of $[i_{\mathsf{t}}, \delta_{\mathsf{t}}]^\rho$ ② closedness and def. of $i_{\mathcal{W}}$
③ definition of $\delta_{\mathcal{W}'}$ ④ definitions of $\tau_{\mathcal{W}'}^\triangleright$ and $\tau_{\mathcal{W}'}^\triangleleft$

The commutative diagram below completes the proof.

① consistency ② previous observation □

C   Proofs for Section 5

Lemma 42. *For all morphisms $\alpha_1\colon S_1 \to Q_{\mathsf{t}}$, $\alpha_2\colon S_2 \to Q_{\mathsf{t}}$, and $f\colon S_1 \to S_2$ such that $\alpha_2 \circ f = \alpha_1$ we have $\alpha_1^\triangleleft \le \alpha_2^\triangleleft$.*

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

$$
\begin{array}{ccc}
S_1 & \xrightarrow{\alpha_1^{\triangleright}} & \bullet \\
{\scriptstyle f}\downarrow & & \downarrow{\scriptstyle \alpha_1^{\triangleleft}} \\
S_2 & & \\
{\scriptstyle \alpha_2^{\triangleright}}\downarrow & \xrightarrow{\alpha_2^{\triangleleft}} & \\
\star & \longrightarrow & Q_{\mathsf{t}}
\end{array}
$$
$\square$

Lemma 21. *Given a wrapper (α, β) for Q_{t}, $([\alpha, [i_{\mathsf{t}}, \delta_{\mathsf{t}}] \circ F_I \alpha], \beta)$ is locally closed w.r.t. α.*

Proof. Let $\mathcal{W} = ([\alpha, [i_{\mathsf{t}}, \delta_{\mathsf{t}}] \circ F_I \alpha], \beta)$. Note that $\alpha^{\triangleleft} \leq [\alpha, [i_{\mathsf{t}}, \delta_{\mathsf{t}}] \circ F_I \alpha]^{\triangleleft}$ by Lemma 42 (via $\kappa_1 \colon S \to S + F_I S$). We define

$$
i_{\mathcal{W}} = I \xrightarrow{\kappa_1} F_I S \xrightarrow{\kappa_2} S + F_I S \xrightarrow{e_{\mathcal{W}}} H_{\mathcal{W}}
$$
$$
\mathsf{lclose}_{\mathcal{W},\alpha} = FS \xrightarrow{\kappa_2} F_I S \xrightarrow{\kappa_2} S + F_I S \xrightarrow{e_{\mathcal{W}}} H_{\mathcal{W}}
$$

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

$$
\begin{array}{ccc}
I & & \\
{\scriptstyle \kappa_2 \circ \kappa_1}\downarrow & \searrow^{\kappa_2 \circ \kappa_1} & \xrightarrow{\ \ i_{\mathsf{t}}\ \ } \\
S + F_I S & \xrightarrow[\mathrm{id} + F_I \alpha]{} S + F_I Q_{\mathsf{t}} \xrightarrow[{[\alpha, [i_{\mathsf{t}}, \delta_{\mathsf{t}}]]}]{} & Q_{\mathsf{t}} \\
{\scriptstyle e_{\mathcal{W}}}\downarrow & & \downarrow{\scriptstyle \beta} \\
H_{\mathcal{W}} & \xrightarrow{\ \ m_{\mathcal{W}}\ \ } & P
\end{array}
$$

$$
\begin{array}{ccc}
FS & \xrightarrow{\ F_I \alpha\ } & FQ_{\mathsf{t}} \quad \overset{\delta_{\mathsf{t}}}{\searrow} \\
{\scriptstyle \kappa_2 \circ \kappa_2}\downarrow & \quad {\scriptstyle \kappa_2 \circ \kappa_2}\downarrow & \\
S + F_I S & \xrightarrow[\mathrm{id} + F_I \alpha]{} S + F_I Q_{\mathsf{t}} \xrightarrow[{[\alpha, [i_{\mathsf{t}}, \delta_{\mathsf{t}}]]}]{} & Q_{\mathsf{t}} \\
{\scriptstyle e_{\mathcal{W}}}\downarrow & & \downarrow{\scriptstyle \beta} \\
H_{\mathcal{W}} & \xrightarrow{\ \ m_{\mathcal{W}}\ \ } & P
\end{array}
$$
$\square$

Lemma 43. *For any run $\{\mathcal{W}_n = (\alpha_n, \beta_n)\}_{n \in \mathbb{N}}$ and $n \in \mathbb{N}$, if $\alpha_{n+1}^{\triangleleft} \leq \alpha_n^{\triangleleft}$, 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 $f_j \colon X_n \to H_n$

as the unique diagonal in the commutative square below.

$$
\begin{array}{ccc}
S_j & \xrightarrow{\ \alpha_j^{\triangleright}\ } & X_j \\
\ \downarrow{\scriptstyle \tau_{\mathcal{W}_j}^{\triangleright}} & {\scriptstyle f_j}\nearrow\!\!\!\!\!\!\!\!\!\searrow & \downarrow{\scriptstyle \alpha_j^{\triangleleft}} \\
& & Q_t \\
& & \downarrow{\scriptstyle \beta_j} \\
H_{\mathcal{W}_j} & \xrightarrowtail[\tau_{\mathcal{W}_j}^{\triangleleft}]{} & P_j
\end{array}
$$

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

    We write $v\colon X_{n+1} \to X_n$ for the witness of $\alpha_{n+1}^{\triangleleft} \le \alpha_n^{\triangleleft}$. 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 $i_{\mathcal{W}_n} = h \circ i_{\mathcal{W}_{n+1}}\colon I \to H_{\mathcal{W}_n}$ and $\mathsf{close}_{\mathcal{W}_n} = h \circ \mathsf{lclose}_{\mathcal{W}_{n+1},\alpha_n}$, where $i_{\mathcal{W}_{n+1}}$ and $\mathsf{lclose}_{\mathcal{W}_{n+1},\alpha_n}$ exist by local closedness and $h\colon H_{\mathcal{W}_{n+1}} \to H_{\mathcal{W}_n}$ is the unique diagonal in the commutative diagram below.

$$
\begin{array}{ccc}
X_{n+1} & \xrightarrow{\ f_{n+1}\ } & H_{\mathcal{W}_{n+1}} \\
{\scriptstyle v}\downarrow\ {\scriptstyle \alpha_{n+1}^{\triangleleft}}\searrow & \textcircled{1} & \downarrow{\scriptstyle \tau_{\mathcal{W}_{n+1}}^{\triangleleft}} \\
X_n \xrightarrow{\alpha_n^{\triangleleft}} Q_t & {\scriptstyle \beta_n=\beta_{n+1}}\searrow & \\
{\scriptstyle f_n}\downarrow\quad\textcircled{1} & & \\
H_{\mathcal{W}_n} & \xrightarrow[\tau_{\mathcal{W}_n}^{\triangleleft}]{} & P_n = P_{n+1}
\end{array}
\qquad
\begin{array}{ccc}
X_{n+1} & \xrightarrow{\ f_{n+1}\ } & H_{\mathcal{W}_{n+1}} \\
{\scriptstyle v}\downarrow & & \downarrow{\scriptstyle \tau_{\mathcal{W}_{n+1}}^{\triangleleft}} \\
X_n & {\scriptstyle h}\ \nearrow\!\!\!\!\!\!\searrow & \\
{\scriptstyle f_n}\downarrow & & \\
H_{\mathcal{W}_n} & \xrightarrowtail[\tau_{\mathcal{W}_n}^{\triangleleft}]{} & P_n = P_{n+1}
\end{array}
$$

    \textcircled{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.

$$
\begin{array}{ccc}
I & & \\
{\scriptstyle i_{\mathcal{W}_{n+1}}}\downarrow & {\scriptstyle i_t}\searrow & \\
H_{\mathcal{W}_{n+1}} & \textcircled{1}\quad & Q_t \\
{\scriptstyle h}\downarrow\ \ \textcircled{2} & \searrow{\scriptstyle \tau_{\mathcal{W}_{n+1}}^{\triangleleft}} & \downarrow{\scriptstyle \beta_n} \\
H_{\mathcal{W}_n} & \xrightarrow[\tau_{\mathcal{W}_n}^{\triangleleft}]{} & P_n = P_{n+1}
\end{array}
\qquad
\begin{array}{ccc}
FS_n & \xrightarrow{\ F\alpha_n\ } & FQ_t \\
{\scriptstyle \mathsf{lclose}_{\mathcal{W}_{n+1},\alpha_n}}\downarrow & \textcircled{1} & \downarrow{\scriptstyle \delta_t} \\
H_{\mathcal{W}_{n+1}} & {\scriptstyle \tau_{\mathcal{W}_{n+1}}^{\triangleleft}} & Q_t \\
{\scriptstyle h}\downarrow\ \ \textcircled{2} & \searrow & \downarrow{\scriptstyle \beta_n} \\
H_{\mathcal{W}_n} & \xrightarrow[\tau_{\mathcal{W}_n}^{\triangleleft}]{} & P_n = P_{n+1}
\end{array}
$$

    \textcircled{1} local closedness \textcircled{2} definition of h $\square$

Proposition 44. *Given a recursive $\rho\colon S \to F_I S$ and a closed and consistent wrapper for Q_t of the form $\mathcal{W} = ([i_t, \delta_t]^{\rho}, \beta)$, we have that $e_{\mathcal{W}} = [i_{\mathcal{W}}, \delta_{\mathcal{W}}]^{\rho}$ and $H_{\mathcal{W}}$ is reachable.*

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

result of $m_\mathcal{W}$ being a mono, together with the uniqueness property of $[i_\mathcal{W}, \delta_\mathcal{W}]^\rho$.
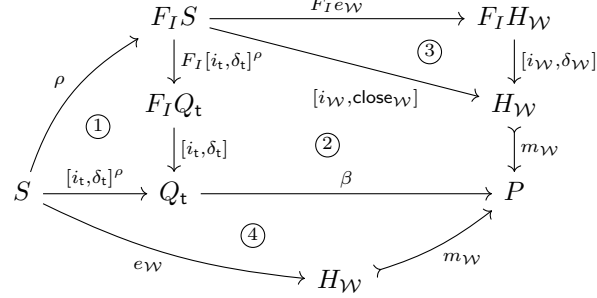

① definition of $[i_\mathsf{t}, \delta_\mathsf{t}]^\rho$ ② closedness and def. of $i_\mathcal{W}$
③ definition of $\delta_\mathcal{W}$ ④ definition of τ

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

Lemma 45. *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 $\beta_n^\triangleright \leq \beta_{n+1}^\triangleright$, 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 $f_j \colon H_{\mathcal{W}_j} \to X_j$ as the unique diagonal in the commutative square below.
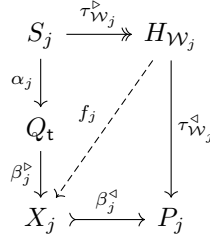
Note that $f_j \in \mathcal{M}$ because $\tau_{\mathcal{W}_j}^\triangleleft \in \mathcal{M}$.

We write $v \colon X_n \to X_{n+1}$ for the witness of $\beta_n^\triangleright \leq \beta_{n+1}^\triangleright$. 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 $o_{\mathcal{W}_n} = o_{\mathcal{W}_{n+1}} \circ h \colon H_{\mathcal{W}_n} \to O$ and $\mathsf{cons}_{\mathcal{W}_n} = \mathsf{lcons}_{\mathcal{W}_{n+1}, \beta_n} \circ h$, where

$h\colon H_{\mathcal{W}_n} \to H_{\mathcal{W}_{n+1}}$ is the unique diagonal in the commutative diagram below.

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.

① local consistency ② definition of h □

Lemma 46. *Let* $\alpha\colon S \to Q_{\mathsf{t}}$, $\alpha'\colon S' \to Q_{\mathsf{t}}$, *and* $\beta\colon Q_{\mathsf{t}} \to P$ *be such that* $\alpha^{\triangleleft}$ *and* $\alpha'^{\triangleleft}$ *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 $\phi\colon X \to X'$ the subobject isomorphism $(\alpha'^{\triangleleft} \circ \phi = \alpha^{\triangleleft})$. We define $f\colon X \to H_{\mathcal{W}}$ and $g\colon X' \to H_{\mathcal{W}'}$ as the unique diagonals in the diagrams below.

Note that $\alpha^{\triangleright}$ and $e_{\mathcal{W}}$ are in $\mathcal{E}$, and therefore so is f; similarly, since $\alpha'^{\triangleright}$ and $\tau^{\triangleright}_{\mathcal{W}'}$ are in $\mathcal{E}$, so is g [6, Proposition 14.9 via duality]. We now define $\psi\colon H_{\mathcal{W}} \to H_{\mathcal{W}'}$

and $\psi^{-1}\colon H_{\mathcal{W}} \to H_{\mathcal{W}'}$ as the unique diagonals in the diagrams below.

$$
\begin{array}{ccc}
X & \xrightarrow{\ f\ } & H_{\mathcal{W}} \\
{\scriptstyle \phi}\downarrow & & \downarrow{\scriptstyle m_{\mathcal{W}}} \\
X' & \overset{\psi}{\dashrightarrow} & \\
{\scriptstyle g}\downarrow & & \\
H_{\mathcal{W}'} & \overset{\tau^{\triangleleft}_{\mathcal{W}'}}{\rightarrowtail} & P
\end{array}
\qquad\qquad
\begin{array}{ccc}
X' & \xrightarrow{\ g\ } & H_{\mathcal{W}'} \\
{\scriptstyle \phi^{-1}}\downarrow & & \downarrow{\scriptstyle \tau^{\triangleleft}_{\mathcal{W}'}} \\
X & \overset{\psi^{-1}}{\dashrightarrow} & \\
{\scriptstyle f}\downarrow & & \\
H_{\mathcal{W}} & \overset{m_{\mathcal{W}}}{\rightarrowtail} & P
\end{array}
$$

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

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

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

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

$$
o_{\mathcal{W}} \circ f = o_{\mathsf{t}} \circ \alpha^{\triangleleft} \qquad\qquad m_{\mathcal{W}} \circ \delta_{\mathcal{W}} \circ Ff = \beta \circ \delta_{\mathsf{t}} \circ F\alpha^{\triangleleft}. \tag{2}
$$

Note that both $\alpha^{\triangleright}$ 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 (2) with the commutative diagrams below.

$$
\begin{array}{ccc}
S & \xrightarrow{\ \alpha^{\triangleright}\ } & X \\
{\scriptstyle \alpha^{\triangleright}}\downarrow & \textcircled{1} & {\scriptstyle f}\searrow \\
X & \xrightarrow{\ e_{\mathcal{W}}\ } & H_{\mathcal{W}} \\
{\scriptstyle \alpha^{\triangleleft}}\searrow & \textcircled{2} & \downarrow{\scriptstyle o_{\mathcal{W}}} \\
& Q_{\mathsf{t}} \xrightarrow{\ o_{\mathsf{t}}\ } & O
\end{array}
$$

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

$$
\begin{array}{ccc}
FS & \xrightarrow{\ F\alpha^{\triangleright}\ } & FX \\
{\scriptstyle F\alpha^{\triangleright}}\downarrow & \textcircled{1} & \downarrow{\scriptstyle Ff} \\
FX & \overset{Fe_{\mathcal{W}}}{\searrow} & FH_{\mathcal{W}} \\
{\scriptstyle F\alpha^{\triangleleft}}\downarrow & \textcircled{4} & \downarrow{\scriptstyle \delta_{\mathcal{W}}} \\
FQ_{\mathsf{t}} & \textcircled{3} \quad \mathsf{close}_{\mathcal{W}} & H_{\mathcal{W}} \\
{\scriptstyle \delta_{\mathsf{t}}}\downarrow & & \downarrow{\scriptstyle m_{\mathcal{W}}} \\
Q_{\mathsf{t}} & \xrightarrow{\ \beta\ } & P
\end{array}
$$

Now the diagrams below commute.

$$
\begin{array}{ccc}
& I & \xrightarrow{\ i_{\mathsf{t}}\ } Q_{\mathsf{t}} \\
{\scriptstyle i_{\mathcal{W}'}}\nearrow & \downarrow{\scriptstyle i_{\mathcal{W}}} \quad \textcircled{2} & \\
& H_{\mathcal{W}} & \downarrow{\scriptstyle \beta} \\
\textcircled{1}\searrow & \downarrow{\scriptstyle \psi} \quad m_{\mathcal{W}} \quad \textcircled{3} & \\
& H_{\mathcal{W}'} \xrightarrow{\ \tau_{\mathcal{W}'^{\triangleleft}}\ } & P
\end{array}
\qquad
\begin{array}{ccc}
& S' & \overset{\tau^{\triangleright}_{\mathcal{W}'}}{\searrow} \\
{\scriptstyle \alpha'^{\triangleright}}\downarrow & \textcircled{4} & \\
& X' \xrightarrow{\ g\ } & H_{\mathcal{W}'} \\
{\scriptstyle \alpha'^{\triangleleft}} & \downarrow{\scriptstyle \phi^{-1}} \ \textcircled{6}\ {\scriptstyle \psi^{-1}} & \searrow{\scriptstyle o_{\mathcal{W}'}} \\
& X \xrightarrow{\ f\ } & H_{\mathcal{W}} \\
& \textcircled{5}\downarrow{\scriptstyle \alpha^{\triangleleft}} \ (2)\ o_{\mathcal{W}} & \downarrow \textcircled{7} \\
& Q_{\mathsf{t}} \xrightarrow{\ o_{\mathsf{t}}\ } & O
\end{array}
$$

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

$$
\begin{array}{c}
FS' \xrightarrow{F\alpha'^{\rhd}} FX' \xrightarrow{\qquad F\alpha'^{\lhd} \qquad} \\
\quad ④ \quad \downarrow{Fg} \quad \overset{F\phi^{-1}}{\longrightarrow} ⑤ \\
F\tau^{\rhd}_{\mathcal{W}'} \searrow \\
FH_{\mathcal{W}'} \; ⑥ \quad FX \xrightarrow{F\alpha^{\lhd}} FQ_{\mathsf{t}} \xrightarrow{\delta_{\mathsf{t}}} Q_{\mathsf{t}} \\
\psi^{-1} \searrow \quad \downarrow{Ff} \qquad (2) \\
FH_{\mathcal{W}} \xrightarrow{\delta_{\mathcal{W}}} H_{\mathcal{W}} \qquad \downarrow{\beta} \\
d \searrow \quad ⑧ \quad \psi \downarrow ③ \quad \searrow^{m_{\mathcal{W}}} \\
\xrightarrow{\qquad} H_{\mathcal{W}'} \xrightarrow{\tau^{\lhd}_{\mathcal{W}'}} P
\end{array}
$$

Using [32, 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.

Lemma 24. *Let $\{\mathcal{W}_n = (\alpha_n, \beta_n)\}_{n\in\mathbb{N}}$ be a run. For all $n \in \mathbb{N}$, we have $\alpha_n^{\lhd} \leq \alpha_{n+1}^{\lhd}$ and $\beta_{n+1}^{\rhd} \leq \beta_n^{\rhd}$. Moreover, if $\alpha_{n+1}^{\lhd} \leq \alpha_n^{\lhd}$, then $\alpha_{n+1} = \alpha_n$; if $\beta_n^{\rhd} \leq \beta_{n+1}^{\rhd}$, 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 $\alpha_n^{\lhd} \leq \alpha_{n+1}^{\lhd}$ by the definition of local closedness. Supposing $\alpha_{n+1}^{\lhd} \leq \alpha_n^{\lhd}$ leads by Lemma 43 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 $\beta_{n+1}^{\rhd} \leq \beta_n^{\rhd}$ by the definition of local consistency. Supposing $\beta_n^{\rhd} \leq \beta_{n+1}^{\rhd}$ leads by Lemma 45 to the contradiction that $\mathcal{W}_n$ is consistent.

If $\mathcal{W}_n$ is closed and consistent and we obtain a counterexample $\rho\colon S \to F_I S$ for $\mathcal{W}_n$, then $\beta_{n+1} = \beta_n$ and $\alpha_{n+1}^{\lhd} = [\alpha_n, [i_{\mathsf{t}}, \delta_{\mathsf{t}}]^{\rho}]^{\lhd}$. We have $\alpha_n^{\lhd} \leq [\alpha_n, [i_{\mathsf{t}}, \delta_{\mathsf{t}}]^{\rho}]^{\lhd}$ using Lemma 42. Suppose $\alpha_{n+1}^{\lhd} \leq \alpha_n^{\lhd}$. Then $[\alpha_n, [i_{\mathsf{t}}, \delta_{\mathsf{t}}]^{\rho}]^{\lhd} = \alpha_{n+1}^{\lhd} \leq \alpha_n^{\lhd}$, so $\alpha_n^{\lhd}$ and $[\alpha_n, [i_{\mathsf{t}}, \delta_{\mathsf{t}}]^{\rho}]^{\lhd}$ are isomorphic subobjects. By Lemma 46 this implies that $([\alpha_n, [i_{\mathsf{t}}, \delta_{\mathsf{t}}]^{\rho}], \beta_n)$ is also closed and consistent, which by Theorem 18 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 $\beta_{n+1} = \beta_n$.

Theorem 25 (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 its hypothesis is correct. If $\mathcal{A}_{\mathsf{t}}$ is minimal and for all $k \in \mathbb{N}$ there exists a recursive $\rho_k\colon S_k \to F_I S_k$ such that $\alpha_k = [i_{\mathsf{t}}, \delta_{\mathsf{t}}]^{\rho_k}$, 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 24 there exist $i_n \in \mathbb{N}$ for all $n \in \mathbb{N}$ such that $i_{n+1} > i_n$, $\alpha_{i_n}^{\lhd} \leq \alpha_{i_n+1}^{\lhd}$, and $\alpha_{i_n+1}^{\lhd} \not\leq \alpha_{i_n}^{\lhd}$ 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 $\alpha_{i_m}^{\lhd}$ and $\alpha_{i_n}^{\lhd}$ 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 24 there exist $i_n \in \mathbb{N}$ for all $n \in \mathbb{N}$ such that $i_{n+1} > i_n$, $\beta_{i_n+1}^{\rhd} \leq \beta_{i_n}^{\rhd}$, and $\beta_{i_n}^{\rhd} \not\leq \beta_{i_n+1}^{\rhd}$ 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 $\beta_{i_m}^{\lhd}$ and $\beta_{i_n}^{\lhd}$ 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 23 the algorithm terminates. By definition, it does so with a correct hypothesis.

Now assume that $\mathcal{A}_{\mathsf{t}}$ is minimal and that for all $k \in \mathbb{N}$ there exists $\rho_k \colon S_k \to F_I S_k$ such that $\alpha_k = [i_{\mathsf{t}}, \delta_{\mathsf{t}}]^{\rho_k}$. 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 44 that $\mathcal{H}_{\mathcal{W}}$ is reachable. Together with correctness of $\mathcal{H}_{\mathcal{W}}$ and minimality of $\mathcal{A}_{\mathsf{t}}$ there exists a unique automaton homomorphism $h \colon \mathcal{H}_{\mathcal{W}} \to \mathcal{A}_{\mathsf{t}}$. We show that $\beta_n \circ h = m_{\mathcal{W}}$ with the commutative diagram below, where we precompose with the epi $e_{\mathcal{W}}$ and use that automaton homomorphisms commute with restricted reachability maps.

$$
\begin{array}{ccc}
S_n & \xrightarrow{\;e_{\mathcal{W}}\;} & H_{\mathcal{W}} \\
{\scriptstyle e_{\mathcal{W}} = [i_{\mathcal{W}}, \delta_{\mathcal{W}}]^{\rho_n}} \downarrow \quad {\scriptstyle \searrow \; \alpha_n = [i_{\mathsf{t}}, \delta_{\mathsf{t}}]^{\rho_n}} & & \downarrow {\scriptstyle m_{\mathcal{W}}} \\
H_{\mathcal{W}} \xrightarrow{\;h\;} Q_{\mathsf{t}} & \xrightarrow{\;\beta_n\;} & P_n
\end{array}
$$

It follows that $h \in \mathcal{M}$ [6, Proposition 14.11]. Being an automaton homomorphism, h commutes with the reachability maps: $h \circ i_{\mathcal{W}}^{\sharp} = i_{\mathsf{t}}^{\sharp}$. Because $i_{\mathcal{W}}^{\sharp} \in \mathcal{E}$ and $i_{\mathsf{t}}^{\sharp} \in \mathcal{E}$, 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.

D Proofs for Section 6

In some of the proofs below we will use the basic property that for all functions $f \colon X \to Y$ and elements $x \in X$ we have:

$$\mathfrak{e}_{f(x)} = f \circ \mathfrak{e}_x. \tag{3}$$

Lemma 47. *For all T-algebras (X, x), $p \colon I \to X$, and $c \in T(I+X)$, the diagram below commutes.*

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

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

$$
\begin{aligned}
[p, \mathfrak{e}_{[p,\mathrm{id}_X]^\sharp(c)}]^\sharp &= [p, \mathfrak{e}_{\mathrm{id}^\natural_X(c)}]^\sharp \\
&= [p, \mathrm{id}_X]^\sharp \circ T(\mathrm{id}_I + \mathfrak{e}_{\mathrm{id}^\natural_X(c)}) \\
&= \mathfrak{e}^\natural_{\mathrm{id}^\natural_X(c)} \\
&= (\mathrm{id}^\natural_X \circ \mathfrak{e}_c)^\natural \qquad\qquad\qquad (3) \\
&= \mathrm{id}^\natural_X \circ \mathfrak{e}^\natural_c \\
&= [p, \mathrm{id}_X]^\sharp \circ \mathfrak{e}^\natural_c \\
&= [p, \mathrm{id}_X]^\sharp \circ \hat{\mu}_X \circ T(\mathrm{id}_I + \mathfrak{e}_c). \qquad\qquad \square
\end{aligned}
$$

Proposition 30 (Computing wrapper morphisms). *Given $S \subseteq TI$ with inclusion $j \colon S \to TI$ and $E \subseteq T(I+1)$ with inclusion $k \colon E \to T(I+1)$, we have:*
- *The top observation table $\beta_E \circ \alpha_S \colon S \to O^E$, $s \mapsto \mathcal{L}_{\mathcal{A}_\mathrm{t}} \circ \mu_I \circ T[\eta_I, j \circ \mathfrak{e}_s] \circ k$;*
- *The bottom observation table $\beta_E \circ \delta_\mathrm{t} \circ F\alpha_S \colon FS \to O^E$, $t \mapsto \mathcal{L}_{\mathcal{A}_\mathrm{t}} \circ \mu_I \circ T[\eta_I, \gamma_I \circ Fj \circ \mathfrak{e}_t] \circ k$;*
- *The input rows $\beta_E \circ i_\mathrm{t} \colon I \to O^E$ given by $(\beta_E \circ i_\mathrm{t})(x) = \mathcal{L}_{\mathcal{A}_\mathrm{t}} \circ T[\mathrm{id}_I, \mathfrak{e}_x] \circ k$;*
- *The row output $o_\mathrm{t} \circ \alpha_S \colon S \to O$ given by $(o_\mathrm{t} \circ \alpha_S)(s) = \mathcal{L}_{\mathcal{A}_\mathrm{t}}(s)$.*

Proof. We first claim that

$$
(\beta_E \circ i^\sharp_\mathrm{t})(s) = \mathcal{L}_{\mathcal{A}_\mathrm{t}} \circ \mu_I \circ T[\eta_I, \mathfrak{e}_s]. \qquad\qquad (4)
$$

(Here, and below, we omit the inclusion k.) To see this, first note that

$$
\begin{aligned}
(\beta_E \circ i^\sharp_\mathrm{t})(s) &= o_\mathrm{t} \circ [i_\mathrm{t}, \mathfrak{e}_{i^\sharp_\mathrm{t}(s)}]^\sharp \\
&= o_\mathrm{t} \circ \delta^*_\mathrm{t} \circ T[i_\mathrm{t}, \mathfrak{e}_{i^\sharp_\mathrm{t}(s)}] \\
&= o_\mathrm{t} \circ \delta^*_\mathrm{t} \circ T[i_\mathrm{t}, i^\sharp_\mathrm{t} \circ \mathfrak{e}_s] \\
&= o_\mathrm{t} \circ \delta^*_\mathrm{t} \circ T[i^\sharp_\mathrm{t} \circ \eta_I, i^\sharp_\mathrm{t} \circ \mathfrak{e}_s] \\
&= o_\mathrm{t} \circ \delta^*_\mathrm{t} \circ T i^\sharp_\mathrm{t} \circ T[\eta_I, \mathfrak{e}_s]
\end{aligned}
$$

It remains to show that $o_{\mathsf{t}} \circ \delta_{\mathsf{t}}^* \circ T i_{\mathsf{t}}^\sharp = \mathcal{L}_{\mathcal{A}_{\mathsf{t}}} \circ \mu_I$, which follows by commutativity of the diagram below.

$$
\begin{array}{ccc}
T^2 I & \xrightarrow{\;\;\mu_I\;\;} & TI \\
\end{array}
$$

① property of $i_{\mathsf{t}}^\sharp$ ② definition of $\mathcal{L}_{\mathcal{A}_{\mathsf{t}}}$
③ naturality   ④ $(T, \delta_{\mathsf{t}}^*)$ is a T-algebra

For the first equation, we derive

$$
\begin{aligned}
(\beta_E \circ \alpha_S)(s) &= (\beta_E \circ i_{\mathsf{t}}^\sharp)(s) && \text{(def. of } \alpha_S) \\
&= \mathcal{L}_{\mathcal{A}_{\mathsf{t}}} \circ \mu_I \circ T[\eta_I, \mathfrak{e}_s] && \text{(by (4))}
\end{aligned}
$$

For the second equation, we derive

$$
\begin{aligned}
(\beta_E \circ \delta_{\mathsf{t}} \circ F\alpha_S)(f) &= (\beta_E \circ \delta_{\mathsf{t}} \circ F i_{\mathsf{t}}^\sharp \circ Fj)(f) && \text{(def. of } \alpha_S) \\
&= (\beta_E \circ i_{\mathsf{t}}^\sharp \circ \gamma_I \circ Fj)(f) && \\
&&& (i^\sharp \text{ is an } F\text{-algebra homomorphism}) \\
&= \mathcal{L}_{\mathcal{A}_{\mathsf{t}}} \circ \mu_I \circ T[\eta_I, \mathfrak{e}_{(\gamma_I \circ Fj)(f)}] && \text{(by (4))} \\
&= \mathcal{L}_{\mathcal{A}_{\mathsf{t}}} \circ \mu_I \circ T[\eta_I, \gamma_I \circ Fj \circ \mathfrak{e}_f] && \text{(by (3))}
\end{aligned}
$$

For the third equation, we derive

$$
\begin{aligned}
(\beta_E \circ i_{\mathsf{t}})(s) &= (\beta_E \circ i_{\mathsf{t}}^\sharp \circ \eta_I)(s) && \text{(property of } i^\sharp) \\
&= \mathcal{L}_{\mathcal{A}_{\mathsf{t}}} \circ \mu_I \circ T[\eta_I, \mathfrak{e}_{\eta_I(s)}] && \text{(by (4))} \\
&= \mathcal{L}_{\mathcal{A}_{\mathsf{t}}} \circ \mu_I \circ T[\eta_I, \eta_I \circ \mathfrak{e}_s] && \text{(by (3))} \\
&= \mathcal{L}_{\mathcal{A}_{\mathsf{t}}} \circ \mu_I \circ T\eta_I \circ T[\mathrm{id}_I, \mathfrak{e}_s] && \\
&= \mathcal{L}_{\mathcal{A}_{\mathsf{t}}} \circ T[\mathrm{id}_I, \mathfrak{e}_s] && \text{(monad law)}
\end{aligned}
$$

Finally, for the fourth equation, we derive:

$$
\begin{aligned}
(o_{\mathsf{t}} \circ \alpha_S)(s) &= (o_{\mathsf{t}} \circ i_{\mathsf{t}}^\sharp)(s) && \text{(definition of } \alpha_S) \\
&= \mathcal{L}_{\mathcal{A}_{\mathsf{t}}}(s) && \text{(definition of } \mathcal{L}_{\mathcal{A}_{\mathsf{t}}}) \;\square
\end{aligned}
$$

Lemma 32 (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 $k\colon I \to S'$ and $\ell\colon FS \to S'$ such that (1) $\alpha_{S'} \circ k = i_{\mathsf{t}}$ and (2) $\alpha_{S'} \circ \ell = \delta_{\mathsf{t}} \circ F\alpha_S$.*

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

$$i_\mathcal{W} = e_\mathcal{W} \circ k \qquad\qquad \mathsf{lclose}_{\mathcal{W},\alpha_S} = e_\mathcal{W} \circ \ell$$

The necessary diagrams now commute:

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

Lemma 34 (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: (1) $(o_\mathsf{t} \circ \alpha_S)(s) = (o_\mathsf{t} \circ \alpha_S)(s')$; and (2) $\beta_E \circ \delta_\mathsf{t} \circ F(\alpha_S \circ [\mathrm{id}_S, \mathfrak{e}_s]) = \beta_E \circ \delta_\mathsf{t} \circ F(\alpha_S \circ [\mathrm{id}_S, \mathfrak{e}_{s'}])$. Then $\mathcal{W} = (\alpha_S, \beta_{E'})$ is locally consistent w.r.t. β_E.*

Proof. Since $e_\mathcal{W}$ is surjective, so is $Fe_\mathcal{W}$. We define the function

$$\mathsf{lcons}_{\mathcal{W},\beta_E} \colon FH_\mathcal{W} \to O^E, \qquad \mathsf{lcons}_{\mathcal{W},\beta_E}(Fe_\mathcal{W}(y)) = (\beta_E \circ \delta_\mathsf{t} \circ F\alpha_S)(y).$$

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

$$\{(s, s') \mid s, s' \in S, e_\mathcal{W}(s) = e_\mathcal{W}(s')\}$$

and let $j \colon K \to S \times S$ be the inclusion. Consider $y, z \in FS$ such that $Fe_\mathcal{W}(y) = Fe_\mathcal{W}(z)$. 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 $K = \{(s_1, s_1'), \ldots, (s_n, s_n')\}$. For all $1 \leq m \leq n$ we define $f_m \colon K \to S + 1$ 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

$$(\beta_E \circ \delta_\mathsf{t} \circ F(\alpha_S \circ [\mathrm{id}_S, \mathfrak{e}_{s_1}]))(c_1) = (\beta_E \circ \delta_\mathsf{t} \circ F(\alpha_S \circ [\mathrm{id}_S, \mathfrak{e}_{s_n}]))(c_n), \qquad (5)$$

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

$$(\beta_E \circ \delta_\mathsf{t} \circ F(\alpha_S \circ [\mathrm{id}_S, \mathfrak{e}_{s_m}]))(c_m)$$
$$= (\beta_E \circ \delta_\mathsf{t} \circ F(\alpha_S \circ [\mathrm{id}_S, \mathfrak{e}_{s_{m+1}}]))(c_{m+1}).$$

Note that
$$[\mathrm{id}_S, \mathfrak{e}_{s'_m}] \circ f_m = [\mathrm{id}_S, \mathfrak{e}_{s_{m+1}}] \circ f_{m+1}$$
by the definitions of f_m and f_{m+1}, so

$$
\begin{aligned}
&(\beta_E \circ \delta_{\mathsf{t}} \circ F(\alpha_S \circ [\mathrm{id}_S, \mathfrak{e}_{s_m}]))(c_m) \\
=\ &(\beta_E \circ \delta_{\mathsf{t}} \circ F(\alpha_S \circ [\mathrm{id}_S, \mathfrak{e}_{s'_m}]))(c_m) \\
&(\text{assumption}) \\
=\ &(\beta_E \circ \delta_{\mathsf{t}} \circ F(\alpha_S \circ [\mathrm{id}_S, \mathfrak{e}_{s'_m}] \circ f_m))(x) \\
&(\text{definition of } c_m) \\
=\ &(\beta_E \circ \delta_{\mathsf{t}} \circ F(\alpha_S \circ [\mathrm{id}_S, \mathfrak{e}_{s_{m+1}}] \circ f_{m+1}))(x) \\
=\ &(\beta_E \circ \delta_{\mathsf{t}} \circ F(\alpha_S \circ [\mathrm{id}_S, \mathfrak{e}_{s_{m+1}}]))(c_{m+1}) \\
&(\text{definition of } c_{m+1}).
\end{aligned}
$$

Then

$$
\begin{aligned}
(\beta_E \circ \delta_{\mathsf{t}} \circ F\alpha_S)(y) =\ &(\beta_E \circ \delta_{\mathsf{t}} \circ F(\alpha_S \circ \pi_1 \circ j))(x) \\
&(\text{definition of } x) \\
=\ &(\beta_E \circ \delta_{\mathsf{t}} \circ F(\alpha_S \circ [\mathrm{id}_S, \mathfrak{e}_{s_1}] \circ f_1))(x) \\
&(\text{definition of } f_1) \\
=\ &(\beta_E \circ \delta_{\mathsf{t}} \circ F(\alpha_S \circ [\mathrm{id}_S, \mathfrak{e}_{s_1}]))(c_1) \\
&(\text{definition of } c_1) \\
=\ &(\beta_E \circ \delta_{\mathsf{t}} \circ F(\alpha_S \circ [\mathrm{id}_S, \mathfrak{e}_{s_n}]))(c_n) \\
&(5) \\
=\ &(\beta_E \circ \delta_{\mathsf{t}} \circ F(\alpha_S \circ [\mathrm{id}_S, \mathfrak{e}_{s'_n}]))(c_n) \\
&(\text{assumption}) \\
=\ &(\beta_E \circ \delta_{\mathsf{t}} \circ F(\alpha_S \circ [\mathrm{id}_S, \mathfrak{e}_{s'_n}]) \circ f_n)(x) \\
&(\text{definition of } c_n) \\
=\ &(\beta_E \circ \delta_{\mathsf{t}} \circ F(\alpha_S \circ \pi_2 \circ j))(x) \\
&(\text{definition of } f_n) \\
=\ &(\beta_E \circ \delta_{\mathsf{t}} \circ F\alpha_S)(z) \\
&(\text{definition of } x).
\end{aligned}
$$

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

We define $o_{\mathcal{W}} \colon H_{\mathcal{W}} \to O$ by

$$o_{\mathcal{W}}(e_{\mathcal{W}}(s)) = (o_{\mathsf{t}} \circ \alpha_S)(s).$$

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 $e_{\mathcal{W}}(s_1) = e_{\mathcal{W}}(s_2)$.

Then

$$(\beta_{E'} \circ \alpha_S)(s_1) = (m_{\mathcal{W}} \circ e_{\mathcal{W}})(s_1)$$
$$= (m_{\mathcal{W}} \circ e_{\mathcal{W}})(s_2)$$
$$= (\beta_{E'} \circ \alpha_S)(s_2),$$

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.

Proposition 36. *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 $\rho\colon S \to F_I S$ such that $[\eta_I, \gamma_I]^\rho\colon S \to TI$ is the inclusion, then there exists a recursive $\rho'\colon S' \to F_I S'$ such that $[\eta_I, \gamma_I]^{\rho'}\colon S' \to TI$ is the inclusion.*

Proof. Let $j\colon S \to TI$ be the inclusion map and define

$$S' = S \cup \{([\eta_I, \gamma_I] \circ F_I j)(x) \mid x \in F_I S\} \subseteq TI.$$

Since S and I are finite and F preserves finite sets, S' is also finite. We choose $k\colon I \to S'$ and $\ell\colon FS \to S'$ by setting

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

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

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

Furthermore, we find that

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

Hence $(\alpha_{S'}, \beta_E)$ is locally closed w.r.t. α_S, by Lemma 32.

Given a recursive $\rho\colon S \to F_I S$ such that $[\eta_I, \gamma_I]^\rho\colon S \to TI$ is the inclusion map, define $\rho'\colon S' \to F_I S'$ for all $s \in S$ and $x \in F_I S$ by

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

To see that this is well-defined, note that $[\eta_I, \gamma_I] \circ F_I j$ is injective. Moreover, if $s \in S$ and $x \in F_I S$ are such that

$$s = ([\eta_I, \gamma_I] \circ F_I j)(x),$$

we have

$$([\eta_I, \gamma_I]^{-1} \circ j)(s) = F_I j(x).$$

Because the inclusion map $j = [\eta_I, \gamma_I]^\rho$ by assumption, it follows that

$$(F_I j \circ \rho)(s) = ([\eta_I, \gamma_I]^{-1} \circ j)(s) = F_I j(x).$$

This completes the proof of ρ' being well-defined.

Note that ρ is a subcoalgebra of $[\eta_I, \gamma_I]^{-1}$ via the inclusion map j by assumption. The definition of ρ' makes ρ' also a subcoalgebra of $[\eta_I, \gamma_I]^{-1}$ via the inclusion map $S' \to TI$. This implies that it is recursive [7, Theorem 3.17],[8] and thus its inclusion map must be $[\eta_I, \gamma_I]^{\rho'} \colon S' \to TI$ by uniqueness.

Proposition 38. *Given finite $S \subseteq TI$ and $E \subseteq T(I+1)$, define $E' \subseteq T(I+1)$ by $E' = E \cup \{(\eta_{I+1} \circ \kappa_2)(\square)\} \cup \{(\hat{\mu}_1 \circ T(\mathrm{id}_I + c_x))(e) \mid e \in E, x \in F(S+1)\}$ where $c_x \colon 1 \to T(I+1)$, with $c_x = \gamma_{I+1} \circ F[T\kappa_1 \circ j, \hat{\eta}_1] \circ \mathfrak{e}_x$, and $j \colon S \to TI$ 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

$$\begin{aligned}
(o_{\mathsf{t}} \circ \alpha_S)(s) &= (o_{\mathsf{t}} \circ \mathfrak{e}_{\alpha_S(s)})(\square) \\
&= (o_{\mathsf{t}} \circ [i_{\mathsf{t}}, \mathfrak{e}_{\alpha_S(s)}] \circ \kappa_2)(\square) \\
&= (o_{\mathsf{t}} \circ [i_{\mathsf{t}}, \mathfrak{e}_{\alpha_S(s)}]^\sharp \circ \eta_{I+1} \circ \kappa_2)(\square) \\
&= (\beta_{E'} \circ \alpha_S)(s)((\eta_{I+1} \circ \kappa_2)(\square)) \\
&\qquad \text{(definition of } \beta_{E'}),
\end{aligned}$$

so

$$\begin{aligned}
(o_{\mathsf{t}} \circ \alpha_S)(s_1) &= (\beta_{E'} \circ \alpha_S)(s_1)((\eta_{I+1} \circ \kappa_2)(\square)) \\
&= (\beta_{E'} \circ \alpha_S)(s_2)((\eta_{I+1} \circ \kappa_2)(\square)) \\
&= (o_{\mathsf{t}} \circ \alpha_S)(s_2).
\end{aligned}$$

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

$$\begin{aligned}
&F[Ti_{\mathsf{t}} \circ j, T\mathfrak{e}_{\alpha_S(s)} \circ \eta_1] \\
&= F[T[i_{\mathsf{t}}, \mathfrak{e}_{\alpha_S(s)}] \circ T\kappa_1 \circ j, T[i_{\mathsf{t}}, \mathfrak{e}_{\alpha_S(s)}] \circ T\kappa_2 \circ \eta_1] \\
&= FT[i_{\mathsf{t}}, \mathfrak{e}_{\alpha_S(s)}] \circ F[T\kappa_1 \circ j, T\kappa_2 \circ \eta_1] \\
&= FT[i_{\mathsf{t}}, \mathfrak{e}_{\alpha_S(s)}] \circ F[T\kappa_1 \circ j, \hat{\eta}_1] \\
&\qquad \text{(definition of } \hat{\eta}_1),
\end{aligned}$$

---

[8] As stated in the introduction of [7], preservation of inverse images is implied by preservation of weak pullbacks.

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

$$
(\delta_{\mathsf{t}} \circ F[\alpha_S, \mathfrak{e}_{\alpha_S(s)}])(x)
$$
$$
= (\delta_{\mathsf{t}} \circ F[\delta_{\mathsf{t}}^* \circ Ti_{\mathsf{t}} \circ j, \delta_{\mathsf{t}}^* \circ T\mathfrak{e}_{\alpha_S(s)} \circ \eta_1])(x)
$$
$$
\text{(definition of } \alpha_S)
$$
$$
= (\delta_{\mathsf{t}} \circ F\delta_{\mathsf{t}}^* \circ F[Ti_{\mathsf{t}} \circ j, T\mathfrak{e}_{\alpha_S(s)} \circ \eta_1])(x)
$$
$$
= (\delta_{\mathsf{t}}^* \circ \gamma_Q \circ F[Ti_{\mathsf{t}} \circ j, T\mathfrak{e}_{\alpha_S(s)} \circ \eta_1])(x)
$$
$$
(\delta_{\mathsf{t}}^* \text{ is an } F\text{-algebra homomorphism})
$$
$$
= (\delta_{\mathsf{t}}^* \circ \gamma_Q \circ FT[i_{\mathsf{t}}, \mathfrak{e}_{\alpha_S(s)}] \circ F[T\kappa_1 \circ j, \hat{\eta}_1])(x)
$$
$$
\text{(shown above)}
$$
$$
= (\delta_{\mathsf{t}}^* \circ T[i_{\mathsf{t}}, \mathfrak{e}_{\alpha_S(s)}] \circ \gamma_{I+1} \circ F[T\kappa_1 \circ j, \hat{\eta}_1])(x)
$$
$$
= (\delta_{\mathsf{t}}^* \circ T[i_{\mathsf{t}}, \mathfrak{e}_{\alpha_S(s)}] \circ \gamma_{I+1} \circ F[T\kappa_1 \circ j, \hat{\eta}_1] \circ \mathfrak{e}_x)(\square)
$$
$$
= (\delta_{\mathsf{t}}^* \circ T[i_{\mathsf{t}}, \mathfrak{e}_{\alpha_S(s)}] \circ c_x)(\square)
$$
$$
\text{(definition of } c_x)
$$
$$
= ([i_{\mathsf{t}}, \mathfrak{e}_{\alpha_S(s)}]^{\sharp} \circ c_x)(\square)
$$
$$
= [i_{\mathsf{t}}, \mathrm{id}_Q]^{\sharp}((T(\mathrm{id}_I + \mathfrak{e}_{\alpha_S(s)}) \circ c_x)(\square)).
$$

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

$$
\hat{\mu}_Q \circ T(\mathrm{id}_I + \mathfrak{e}_{(T(\mathrm{id}_I + \mathfrak{e}_{\alpha_S(s)}) \circ c_x)(\square)})
$$
$$
= \hat{\mu}_Q \circ T(\mathrm{id}_I + (T(\mathrm{id}_I + \mathfrak{e}_{\alpha_S(s)}) \circ \mathfrak{e}_{c_x(\square)}))
$$
$$
(3)
$$
$$
= \hat{\mu}_Q \circ T(\mathrm{id}_I + (T(\mathrm{id}_I + \mathfrak{e}_{\alpha_S(s)}) \circ c_x))
$$
$$
\text{(definition of } \mathfrak{e}_{c_x(\square)})
$$
$$
= \hat{\mu}_Q \circ T(\mathrm{id}_I + (T(\mathrm{id}_I + \mathfrak{e}_{\alpha_S(s)}))) \circ T(\mathrm{id}_I + c_x)
$$
$$
= T(\mathrm{id}_I + \mathfrak{e}_{\alpha_S(s)}) \circ \hat{\mu}_1 \circ T(\mathrm{id}_I + c_x),
$$

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

$$(\beta_E \circ \delta_{\mathsf{t}} \circ F(\alpha_S \circ [\mathrm{id}_S, \mathfrak{e}_s]))(x)(e)$$
$$= (\beta_E \circ \delta_{\mathsf{t}} \circ F[\alpha_S, \mathfrak{e}_{\alpha_S(s)}])(x)(e)$$
$$(3)$$
$$= (o_{\mathsf{t}} \circ [i_{\mathsf{t}}, \mathfrak{e}_{(\delta_{\mathsf{t}} \circ F[\alpha_S, \mathfrak{e}_{\alpha_S(s)}])(x)}]^{\sharp})(e)$$
$$(\text{definition of } \beta_E)$$
$$= (o_{\mathsf{t}} \circ [i_{\mathsf{t}}, \mathfrak{e}_{[i_{\mathsf{t}}, \mathrm{id}_Q]^{\sharp}((T(\mathrm{id}_I + \mathfrak{e}_{\alpha_S(s)}) \circ c_x)(\Box))}]^{\sharp})(e)$$
$$(\text{shown earlier})$$
$$= (o_{\mathsf{t}} \circ [i_{\mathsf{t}}, \mathrm{id}_Q]^{\sharp} \circ \hat{\mu}_Q \circ T(\mathrm{id}_I + \mathfrak{e}_{(T(\mathrm{id}_I + \mathfrak{e}_{\alpha_S(s)}) \circ c_x)(\Box)}))(e)$$
$$(\text{Lemma 47})$$
$$= (o_{\mathsf{t}} \circ [i_{\mathsf{t}}, \mathrm{id}_Q]^{\sharp} \circ T(\mathrm{id}_I + \mathfrak{e}_{\alpha_S(s)}) \circ \hat{\mu}_1 \circ T(\mathrm{id}_I + c_x))(e)$$
$$(\text{shown above})$$
$$= (o_{\mathsf{t}} \circ [i_{\mathsf{t}}, \mathfrak{e}_{\alpha_S(s)}]^{\sharp} \circ \hat{\mu}_1 \circ T(\mathrm{id}_I + c_x))(e)$$
$$= (\beta_{E'} \circ \alpha_S)(s)((\hat{\mu}_1 \circ T(\mathrm{id}_I + c_x))(e))$$
$$(\text{definition of } \beta_{E'}),$$

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

$$(\beta_E \circ \delta_{\mathsf{t}} \circ F(\alpha_S \circ [\mathrm{id}_S, \mathfrak{e}_{s_1}]))(x)(e)$$
$$= (\beta_{E'} \circ \alpha_S)(s_1)((\hat{\mu}_1 \circ T(\mathrm{id}_I + c_x))(e))$$
$$= (\beta_{E'} \circ \alpha_S)(s_2)((\hat{\mu}_1 \circ T(\mathrm{id}_I + c_x))(e)) \qquad\qquad (\text{assumption})$$
$$= (\beta_E \circ \delta_{\mathsf{t}} \circ F(\alpha_S \circ [\mathrm{id}_S, \mathfrak{e}_{s_2}]))(x)(e).$$

Thus, it follows from Lemma 34 that $(\alpha_S, \beta_{E'})$ is locally consistent w.r.t. β_E.

Proposition 39 (Language equivalence via finite recursion). *Given an automaton $\mathcal{A} = (Q, \delta, i, o)$, we have $\mathcal{L}_{\mathcal{A}_{\mathsf{t}}} = \mathcal{L}_{\mathcal{A}}$ iff $\mathcal{L}_{\mathcal{A}_{\mathsf{t}}}^{\rho} = \mathcal{L}_{\mathcal{A}}^{\rho}$ for all recursive $\rho \colon S \to F_I S$ such that S is finite.*

Proof. Suppose that for all recursive coalgebras $\rho \colon S \to F_I S$ such that S is finite we have $o_{\mathsf{t}} \circ [i_{\mathsf{t}}, \delta]^{\rho} = o \circ [i, \delta]^{\rho}$. 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 $a \colon F_I \left(\bigcup_{n \in \mathbb{N}} F_I^n \emptyset \right) \to \bigcup_{n \in \mathbb{N}} F_I^n \emptyset$. Let $\phi \colon (TI, [\eta_I, \gamma_I]) \to (\bigcup_{n \in \mathbb{N}} F_I^n \emptyset, a)$ 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 $\rho \colon F_I^n \emptyset \to F_I^{n+1} \emptyset$ by [18, Proposition 6], with $a^{\rho} \colon F_I^n \emptyset \to \bigcup_{n \in \mathbb{N}} F_I^n \emptyset$ 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 $\rho' \colon S \to F_I S$, with $[\eta_I, \gamma_I]^{\rho'} \colon S \to TI$ being the inclusion.

Moreover, $t \in S$. Thus,

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

The converse follows from Proposition 16. $\qquad\square$

The following useful lemma shows how contextual wrappers arise from recursive coalgebras.

Lemma 48. *Let $\rho\colon S \to F_I S$ be a recursive coalgebra, and let $\iota^{\lhd}\colon S' \rightarrowtail TI$ be the image of the unique coalgebra-to-algebra map $\iota\colon S \to TI$ to the initial algebra. Then S' is again a recursive coalgebra, $\iota^{\lhd}$ is the unique coalgebra-to-algebra morphism, and $\mathsf{reach}_{\mathcal{A}} \circ \iota^{\lhd} = \alpha_{S'}$.*

Proof. We show the announced recursiveness property. Consider the factorisation $\iota^{\lhd} \circ \iota^{\rhd}$ of the unique coalgebra-to-algebra map ι in:

$$
\begin{array}{ccccc}
F_I S & \xrightarrow{F_I \iota^{\rhd}} & F_I S' & \xrightarrow{F_I \iota^{\lhd}} & F_I T I \\
\big\uparrow\rho & & \big\uparrow d & & \big\downarrow\cong \\
S & \xrightarrow{\iota^{\rhd}} & S' & \xrightarrow{\iota^{\lhd}} & T I
\end{array}
$$

Our functor F_I preserves weak pullbacks (since F does), and hence monos in particular. Thus $F_I(\iota^{\lhd})$ is monic. Since the initial algebra is an isomorphism, the composition with its structure map is monic as well and we get the map d by the fill-in property of the factorisation system.

Finally, again since F_I preserves weak pullbacks it preserves intersections; and as a consequence, the coalgebra $d\colon S' \to F_I(S')$ is recursive by [8, Corollary 5.6]. $\qquad\square$