An L^* Algorithm for Deterministic Weighted Regular Languages

Clemente Pasti **Talu Karagöz** **Anej Svete**
Franz Nowak **Reda Boumasmoud** **Ryan Cotterell**
{clemente.pasti, anej.svete, franz.nowak, ryan.cotterell}@inf.ethz.ch
reda.boumasmoud@math.ethz.ch, talukaragoz@gmail.com

ETH zürich

Abstract

Extracting finite state automata (FSAs) from black-box models offers a powerful approach to gaining interpretable insights into complex model behaviors. To support this pursuit, we present a weighted variant of Angluin's (1987) $\mathbf{L}^*$ algorithm for learning FSAs. We stay faithful to the original formulation, devising a way to exactly learn deterministic weighted FSAs whose weights support division. Furthermore, we formulate the learning process in a manner that highlights the connection with FSA minimization, showing how $\mathbf{L}^*$ directly learns a minimal automaton for the target language.

github.com/rycolab/weighted-angluin

1 Introduction

Learning formal languages from data is a classic problem in computer science. Unfortunately, learning only from positive examples is impossible, as shown by Gold (1978). However, by granting the learner access to more than just positive examples, Angluin (1987) introduced the *active* learning scheme $\mathbf{L}^*$, where the learner interacts with an oracle by asking it queries. Concretely, Angluin's (1987) $\mathbf{L}^*$ algorithm learns regular languages in the form of deterministic finite-state automata (DFSAs) from *membership* queries (analogous to asking for a ground truth label of a string in the training dataset) and *equivalence* queries (analogous to asking whether a hypothesis is correct).

Weighted formal languages, where strings are assigned weights such as probabilities or costs, naturally generalize membership-based (boolean) formal languages. Weighted languages, especially probabilistic languages, serve as a cornerstone in the conceptual framework of many NLP problems (Mohri, 1997). Their significance is twofold: first, in practical applications, where they underpin algorithms for tasks such as parsing (Goodman, 1996) and machine translation (Mohri, 1997), and second, as an analytical framework for better understanding modern language models (Weiss et al., 2018; Jumelet and Zuidema, 2023; Nowak et al., 2024, *inter alia*). The practical applications in NLP, coupled with theoretical interests in formal language theory, have motivated the development of various weighted extensions of Angluin's (1987) $\mathbf{L}^*$. For instance, Weiss et al. (2019) describes a generalization of $\mathbf{L}^*$ that (approximately) learns a probabilistic DFSA by querying a neural language model to interpret it. Additionally, and less faithfully to the original $\mathbf{L}^*$ algorithm, multiple algorithms for learning *non-deterministic* weighted FSAs have been proposed (Bergadano and Varricchio, 1996; Beimel et al., 2000; Balle and Mohri, 2012; Balle et al., 2014; Daviaud and Johnson, 2024).

In this paper, we present a novel weighted generalization of the $\mathbf{L}^*$ algorithm that learns *semifield*-weighted *deterministic* FSAs with membership and equivalence queries. In contrast to other algorithms inspired by $\mathbf{L}^*$, ours is a true generalization, i.e., we generalize Angluin's (1987) original algorithm, resulting in a familiar procedure that, just like the original, learns a deterministic FSA exactly in a finite number of steps if the automata can be determinized.[1] Additionally, we loosen the requirement for field-weighted FSAs made by spectral algorithms, e.g., Balle and Mohri's (2012) algorithm, that learn non-deterministic automata; our algorithm only requires a semifield-weighted FSA. Interestingly, our exposition further illuminates the connection between three important algorithms for weighted FSAs: minimization (Hopcroft and Ullman, 1979; Mohri, 1997), weight pushing (Mohri, 1997; Mohri and Riley, 2001) and $\mathbf{L}^*$.

2 Weighted Regular Languages

Semirings and Semifields. A **monoid** is a 3-tuple $(\mathbb{K}, \bullet, \mathbf{1})$, where $\mathbb{K}$ is a set, $\bullet : \mathbb{K} \times \mathbb{K} \to \mathbb{K}$ is an associative operator, and $\mathbf{1} \in \mathbb{K}$ is a distinguished identity element such that

[1] All boolean-weighted FSA can be determinized, which is why Angluin's (1987) $\mathbf{L}^*$ always halts.

Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing, pages 8197–8210
November 12-16, 2024

$\mathbf{1} \bullet w = w \bullet \mathbf{1} = w$ for any $w \in \mathbb{K}$. A monoid is **commutative** if $\bullet$ is a commutative operator. A **group** is a monoid $(\mathbb{K}, \bullet, \mathbf{1})$ where every element has an inverse: for every $w \in \mathbb{K}$ there exists a $w^{-1} \in \mathbb{K}$ such that $w \bullet w^{-1} = w^{-1} \bullet w = \mathbf{1}$; we use the notation w_1/w_2 as an alias for $w_2^{-1} \otimes w_1$. A **semiring** $(\mathbb{K}, \oplus, \otimes, \mathbf{0}, \mathbf{1})$ is a 5-tuple where $(\mathbb{K}, \oplus, \mathbf{0})$ is a commutative monoid, $(\mathbb{K}, \otimes, \mathbf{1})$ is a monoid, $\otimes$ distributes over $\oplus$ and $\mathbf{0}$ is an annihilator for $\otimes$. A **semifield** is a semiring $(\mathbb{K}, \oplus, \otimes, \mathbf{0}, \mathbf{1})$ where $(\mathbb{K}\backslash\{\mathbf{0}\}, \otimes, \mathbf{1})$ is a commutative group.

Strings and Languages. An **alphabet** Σ is a non-empty, finite set of symbols. A **string** is a finite sequence of symbols from an alphabet. We write $\boldsymbol{xy}$ to denote the concatenation of the strings $\boldsymbol{x}$ and $\boldsymbol{y}$. Let $\Sigma^{n+1} \stackrel{\text{def}}{=} \{\boldsymbol{y}a \mid \boldsymbol{y} \in \Sigma^n, a \in \Sigma\}$ and $\Sigma^0 \stackrel{\text{def}}{=} \{\varepsilon\}$, where ε is the empty string. The **Kleene closure** $\Sigma^* \stackrel{\text{def}}{=} \bigcup_{n=0}^{\infty} \Sigma^n$ of Σ is the set containing all strings made with symbols of Σ. We further introduce the set $\Sigma^{\leqslant k} = \bigcup_{n=0}^{k} \Sigma^k$. Given an alphabet Σ and a semiring $\langle \mathbb{K}, \oplus, \otimes, \mathbf{0}, \mathbf{1} \rangle$, a **weighted formal language** is a function $L \colon \Sigma^* \to \mathbb{K}$ that assigns weights $w \in \mathbb{K}$ to strings $\boldsymbol{y} \in \Sigma^*$. Unless differently specified, in this paper we will assume that all weighted languages are *semifield*-weighted.

Weighted Finite-state Automata. A **weighted finite-state automaton** (WFSA) $\mathcal{A}$ over a semifield $\langle \mathbb{K}, \oplus, \otimes, \mathbf{0}, \mathbf{1} \rangle$ is a 5-tuple $(\Sigma, Q, \delta, \lambda, \rho)$ where Σ is an alphabet, Q is a finite set of states, δ is a set of weighted arcs rendered as $p \xrightarrow{a/w} q$ with $p, q \in Q$, $a \in \Sigma$, and $w \in \mathbb{K}$,[2] and $\lambda \colon Q \to \mathbb{K}$ and $\rho \colon Q \to \mathbb{K}$ are the initial and final weight function, respectively. A **path** $\boldsymbol{\pi}$ in $\mathcal{A}$ is a finite sequence of contiguous arcs, denoted as

$$q_0 \xrightarrow{a_1/w_1} q_1, \cdots, q_{N-1} \xrightarrow{a_N/w_N} q_N. \quad (1)$$

We call $i(\boldsymbol{\pi}) = q_0$ the initial state of the path, and $f(\boldsymbol{\pi}) = q_N$ the final state of the path. The **weight** of $\boldsymbol{\pi}$ is $\boldsymbol{w}(\boldsymbol{\pi}) = w_1 \otimes \cdots \otimes w_N$ and its **yield** is $\sigma(\boldsymbol{\pi}) = a_1 \cdots a_N$. With $\Pi_{\mathcal{A}}$, we denote the set of all paths in $\mathcal{A}$, and with $\Pi_{\mathcal{A}}(\boldsymbol{p})$ the subset of all paths in $\mathcal{A}$ with yield $\boldsymbol{p}$. We say that a WFSA $\mathcal{A} = (\Sigma, Q, \delta, \lambda, \rho)$ is **deterministic** (a WDFSA) if, for every $p \in Q, a \in \Sigma$, there is at most one $q \in Q$ such that $p \xrightarrow{a/w} q \in \delta$ with $w > 0$, and there is a single state q_I with $\lambda(q_I) \neq 0$. In such case, we refer to q_I as the **initial state**. Naturally, a WDFSA can have at most one path yielding a string $\boldsymbol{y} \in \Sigma^*$ from the initial state q_I.

Weighted Regular Languages. Every WFSA $\mathcal{A}$ generates the weighted language

$$L_{\mathcal{A}}(\boldsymbol{p}) \stackrel{\text{def}}{=} \bigoplus_{\boldsymbol{\pi} \in \Pi_{\mathcal{A}}(\boldsymbol{p})} \lambda(i(\boldsymbol{\pi})) \otimes \boldsymbol{w}(\boldsymbol{\pi}) \otimes \rho(f(\boldsymbol{\pi})) \quad (2)$$

for $\boldsymbol{p} \in \Sigma^*$. We define the set $\text{supp}(L) = \{\boldsymbol{p} \in \Sigma^* \mid L(\boldsymbol{p}) \neq \mathbf{0}\}$ to be the **support** of L. A weighted language is said to be **regular** if there exists a WFSA that generates it. If two WFSAs generate the same language, they are said to be **equivalent**. Finally, a weighted regular language is said to be **deterministic** if there exists a WDFSA that generates it. In contrast to the boolean case, not every weighted regular language can be generated by a deterministic WFSA (Allauzen and Mohri, 2003), and, therefore, weighted deterministic regular languages are a strict subset of weighted regular languages. This distinction plays a critical role in our exposition—we develop a generalization of Angluin's (1987) algorithm that learns weighted *deterministic* regular languages.

3 Minimal Deterministic Automata

We now introduce the notion of **right language equivalence** between two strings. It provides the basis for active learning (Biermann and Feldman, 1972; Angluin, 1987) and minimization algorithms (Hopcroft and Ullman, 1979; Mohri, 1997), and with it, our weighted extension of $\mathbf{L}^*$. We begin by defining a notion of equivalence between two languages L_1 and L_2 with weights on the same semifield $\langle \mathbb{K}, \oplus, \otimes, \mathbf{0}, \mathbf{1} \rangle$. We write $L_1 \equiv L_2$ if there exists a $k \in \mathbb{K}\backslash\{\mathbf{0}\}$ such that $L_1(\boldsymbol{p}) = k \otimes L_2(\boldsymbol{p})$ for every $\boldsymbol{p} \in \Sigma^*$.[3] Next, given a weighted language L and $\boldsymbol{y} \in \Sigma^*$, we define $\boldsymbol{y}$'s **right language** $\boldsymbol{y}^{-1}L(\boldsymbol{p}) \stackrel{\text{def}}{=} L(\boldsymbol{yp})$ for $\boldsymbol{p} \in \Sigma^*$. The definition of right language naturally induces on Σ^* the **right language equivalence relation** $\boldsymbol{y} \sim_L \boldsymbol{x} \iff \boldsymbol{y}^{-1}L \equiv \boldsymbol{x}^{-1}L, \mathbf{x}, \mathbf{y} \in \Sigma^*$ (Mohri, 1997).[4] The core intuition behind both minimization (Hopcroft and Ullman, 1979; Mohri, 1997) and active learning algorithms (Angluin, 1987) is that of building a DFSA whose states correspond to the right language equivalence classes of L.

[2] We do not consider ε-transitions. This is without loss of generality; any regular language can be represented by an ε-free automaton (Mohri, 2009, Theorem 7.1).

[3] $\equiv$ is an equivalence relation; see App. A.

[4] The fact that $\sim_L$ is an equivalence relation follows directly from the fact that $\equiv$ is an equivalence relation.

Minimality. We say that a WDFSA is **minimal** if no other equivalent WDFSA has fewer states (Mohri, 1997). The Myhill–Nerode theorem binds right-equivalence classes of a regular language to the states of the minimal DFSA (Hopcroft and Ullman, 1979). We state its weighted version:

Theorem 1 (Myhill–Nerode)**.** *Let $\mathcal{A}$ be a semifield-weighted DFSA and $L_{\mathcal{A}}$ its weighted language. Then, $\sim_{L_{\mathcal{A}}}$ induces a finite number of equivalence classes on Σ^*, which equals the number of states of a minimal automaton for $L_{\mathcal{A}}$.*

Proof. See App. C.1. ■

In a DFSA $\mathcal{A}$, every state q can be uniquely identified as $q_{\boldsymbol{p}}$,[5] where $\boldsymbol{p}$ is the string that is read by traversing the non-zero-weight path from the initial state to q. Therefore, it is straightforward to extend the notion of right language equivalence from strings to states of $\mathcal{A}$ as $q_{\boldsymbol{p}} \sim_{L_{\mathcal{A}}} p_{\boldsymbol{y}} \iff \boldsymbol{p} \sim_{L_{\mathcal{A}}} \boldsymbol{y}$. Minimization algorithms for unweighted DFSAs are based on merging states that are right-equivalent (Hopcroft and Ullman, 1979). This strategy can be adapted to WDFSAs after *canonicalizing* the distribution of the weights over the WDFSA transitions with weight pushing (Mohri, 1997). [6]

4 A Weighted L* Algorithm

We now present our weighted $\mathbf{L}^*$ algorithm. Let $\langle \mathbb{K}, \oplus, \otimes, \mathbf{0}, \mathbf{1} \rangle$ be a semifield and $L^\star : \Sigma^* \to \mathbb{K}$ a deterministic regular language. As Angluin (1987), we assume that we have access to an **oracle** that can answer the following queries about $L^\star$:

(1) **Membership query**: What is the weight $L^\star(\boldsymbol{p})$ of the string $\boldsymbol{p} \in \Sigma^*$?

(2) **Equivalence query**: Does an hypothesis automaton $\mathcal{H}$ generate $L^\star$? If it does *not*, the oracle provides a **counterexample**, which is a string $\boldsymbol{t}$ such that $L_{\mathcal{H}}(\boldsymbol{t}) \neq L^\star(\boldsymbol{t})$.

Our algorithm reduces to Angluin's (1987) in the case of the boolean semifield.

[5] More precisely, this holds for DFSAs where every state is accessible—a condition we will assume throughout the paper without loss of generality.

[6] In the weighted case, equivalent automata with the same topology but different weight distribution along the transitions and the initial and final weights exist. Weight pushing (Mohri, 1997) re-distributes the weights while preserving the language generated by the automaton. It can be used to obtain a canonical weight distribution, which facilitates, for instance, WDFSA equivalence testing. In essence, our algorithm learns a minimal WDFSA of a weighted regular language.

4.1 The Hankel Matrix

Let L be a weighted language as above. L's **Hankel matrix** $\mathbf{H}$ is a bi-infinite matrix indexed by elements of Σ^* defined entry-wise as $\mathbf{H}(\boldsymbol{p}, \boldsymbol{s}) \stackrel{\text{def}}{=} L(\boldsymbol{ps})$ for $\boldsymbol{p}, \boldsymbol{s} \in \Sigma^*$. Additionally, for every $\boldsymbol{p} \in \Sigma^*$, we define the function $\mathbf{H}_{\boldsymbol{p}} : \Sigma^* \to \mathbb{K}$ as $\mathbf{H}_{\boldsymbol{p}}(\boldsymbol{s}) \stackrel{\text{def}}{=} \mathbf{H}(\boldsymbol{p}, \boldsymbol{s}) = \boldsymbol{p}^{-1} L(\boldsymbol{s})$ for $\boldsymbol{s} \in \Sigma^*$.

The $\mathbf{L}^*$ algorithm learns from an *empirical* Hankel matrix. Let $\widetilde{\mathrm{P}} \subseteq \Sigma^*$ be a prefix-closed set of prefixes and let $\widetilde{\mathrm{S}} \subseteq \Sigma^*$ be a suffix-closed set of suffixes.[7] The **empirical Hankel matrix** $\widetilde{\mathbf{H}}$ is a matrix of size $|\widetilde{\mathrm{P}} \circ \Sigma^{\leqslant 1}| \times |\widetilde{\mathrm{S}}|$ defined entry-wise as $\widetilde{\mathbf{H}}(\boldsymbol{p}, \boldsymbol{s}) \stackrel{\text{def}}{=} L(\boldsymbol{ps})$ for $\boldsymbol{p} \in \widetilde{\mathrm{P}} \circ \Sigma^{\leqslant 1}, \boldsymbol{s} \in \widetilde{\mathrm{S}}$. Analogously to the Hankel matrix, we define the function $\widetilde{\mathbf{H}}_{\boldsymbol{p}} \colon \widetilde{\mathrm{S}} \to \mathbb{K}$ as $\widetilde{\mathbf{H}}_{\boldsymbol{p}}(\boldsymbol{s}) \stackrel{\text{def}}{=} \widetilde{\mathbf{H}}(\boldsymbol{p}, \boldsymbol{s})$ for every $\boldsymbol{s} \in \widetilde{\mathrm{S}}$. Note that $\widetilde{\mathbf{H}}_{\boldsymbol{p}}$ is a restriction of $\mathbf{H}_{\boldsymbol{p}}$ to a finite domain. We define the equivalence relation $\equiv$ on the set $\{\widetilde{\mathbf{H}}_{\boldsymbol{p}} \mid \boldsymbol{p} \in \widetilde{\mathrm{P}} \circ \Sigma^{\leqslant 1}\}$ as we did for weighted languages (§3). We say that $\widetilde{\mathbf{H}}$ is:

(1) **closed** if, for every $\boldsymbol{p} \in \widetilde{\mathrm{P}}$ and $a \in \Sigma$, there exists $\boldsymbol{p}' \in \widetilde{\mathrm{P}}$ such that $\widetilde{\mathbf{H}}_{\boldsymbol{p}a} \equiv \widetilde{\mathbf{H}}_{\boldsymbol{p}'}$;

(2) **consistent** if, for every $\boldsymbol{p}, \boldsymbol{p}' \in \widetilde{\mathrm{P}}$ and $a \in \Sigma$ we have that $\widetilde{\mathbf{H}}_{\boldsymbol{p}} \equiv \widetilde{\mathbf{H}}_{\boldsymbol{p}'} \implies \widetilde{\mathbf{H}}_{\boldsymbol{p}a} \equiv \widetilde{\mathbf{H}}_{\boldsymbol{p}'a}$.

Given a closed and consistent empirical Hankel matrix $\widetilde{\mathbf{H}}$: $\widetilde{\mathrm{P}} \circ \Sigma^{\leqslant 1} \times \widetilde{\mathrm{S}} \to \mathbb{K}$, we define the following equivalence relation on $\widetilde{\mathrm{P}} \circ \Sigma^{\leqslant 1}$:

$$\boldsymbol{p} \sim_{\widetilde{\mathbf{H}}} \boldsymbol{p}' \iff \widetilde{\mathbf{H}}_{\boldsymbol{p}} \equiv \widetilde{\mathbf{H}}_{\boldsymbol{p}'}. \tag{3}$$

We note the similarity to the right language equivalence relation on Σ^* induced by a weighted language (cf. §3). Because $\widetilde{\mathrm{P}}$ is finite by assumption, the relation $\sim_{\widetilde{\mathbf{H}}}$ induces a finite number of equivalence classes $\widetilde{\mathrm{P}} \circ \Sigma^{\leqslant 1} / \sim_{\widetilde{\mathbf{H}}}$. We denote each equivalence class as $[\boldsymbol{p}] = \{\boldsymbol{p}' \in \widetilde{\mathrm{P}} \circ \Sigma^{\leqslant 1} \mid \boldsymbol{p}' \sim_{\widetilde{\mathbf{H}}} \boldsymbol{p}\}$.

4.2 The Empirical Hankel Automaton

We now introduce the conversion of an empirical Hankel matrix into a WDFSA, the **empirical Hankel automaton** $\mathcal{H} = (\Sigma, Q, \delta, \lambda, \rho)$. We first detail how to construct $\mathcal{H}$'s states, transitions, and initial and final weights:

(1) **States.** We define the states Q to be in a one-to-one correspondence with $\widetilde{\mathrm{P}} / \sim_{\widetilde{\mathbf{H}}}$: Each state $q_{[\boldsymbol{p}]} \in Q$ corresponds to an equivalence class $[\boldsymbol{p}] \in \widetilde{\mathrm{P}} / \sim_{\widetilde{\mathbf{H}}}$. Here, $\boldsymbol{p} \in \widetilde{\mathrm{P}}$ is a *representative* of the class, which **we assume to be fixed** throughout the construction of $\mathcal{H}$. We

[7] A set of strings is prefix-closed (suffix-closed) if it contains all prefixes (suffixes) of each of its elements.

also define the map $r\colon \widetilde{\mathrm{P}} \circ \Sigma^{\leqslant 1} \to \widetilde{\mathrm{P}}$ that assigns to each prefix the representative of its equivalence class; $r(\boldsymbol{p}') = \boldsymbol{p} \iff \boldsymbol{p}' \in [\boldsymbol{p}]$.

(2) **Transitions.** For every state $q_{[\boldsymbol{p}]} \in Q$, and every symbol $a \in \Sigma$, let the transition $q_{[\boldsymbol{p}]} \xrightarrow{a/w} q_{[r(\boldsymbol{p}a)]}$ be in δ, where

$$w \stackrel{\text{def}}{=} \frac{\bigoplus_{\boldsymbol{s}\in\widetilde{\mathrm{S}}} \widetilde{\mathbf{H}}(\boldsymbol{p}a, \boldsymbol{s})}{\bigoplus_{\boldsymbol{s}\in\widetilde{\mathrm{S}}} \widetilde{\mathbf{H}}(\boldsymbol{p}, \boldsymbol{s})} \tag{4}$$

if $\bigoplus_{\boldsymbol{s}\in\widetilde{\mathrm{S}}} \widetilde{\mathbf{H}}(\boldsymbol{p}, \boldsymbol{s})$ is non-zero, and $w \stackrel{\text{def}}{=} \mathbf{0}$ else.

(3) **Initial weight.** For every state $q_{[\boldsymbol{p}]} \in Q$, we define the initial weighting function as

$$\lambda(q_{\boldsymbol{p}}) \stackrel{\text{def}}{=} \begin{cases} \bigoplus_{\boldsymbol{s}\in\widetilde{\mathrm{S}}} \widetilde{\mathbf{H}}(\varepsilon, \boldsymbol{s}) & \boldsymbol{p} = \varepsilon \\ \mathbf{0} & \boldsymbol{p} \neq \varepsilon \end{cases} \tag{5}$$

(4) **Final weights.** For every state $q_{[\boldsymbol{p}]}$, we define the final weight

$$\rho(q_{[\boldsymbol{p}]}) \stackrel{\text{def}}{=} \frac{\widetilde{\mathbf{H}}(\boldsymbol{p}, \varepsilon)}{\bigoplus_{\boldsymbol{s}\in\widetilde{\mathrm{S}}} \widetilde{\mathbf{H}}(\boldsymbol{p}, \boldsymbol{s})} \tag{6}$$

if $\bigoplus_{\boldsymbol{s}\in\widetilde{\mathrm{S}}} \widetilde{\mathbf{H}}(\boldsymbol{p}, \boldsymbol{s})$ is non-zero, and $\rho(q_{[\boldsymbol{p}]}) \stackrel{\text{def}}{=} \mathbf{0}$ else.

Theorem 2 (The empirical Hankel Automaton)**.** *Let $\widetilde{\mathbf{H}}\colon \widetilde{\mathrm{P}} \circ \Sigma^{\leqslant 1} \times \widetilde{\mathrm{S}} \to \mathbb{K}$ be a closed and consistent empirical Hankel matrix and let $\mathcal{H}$ be the empirical Hankel automaton induced by $\widetilde{\mathbf{H}}$. Then:*

(1) $\mathcal{H}$ is a well-defined WDFSA.

(2) $L_{\mathcal{H}}(\boldsymbol{p}\boldsymbol{s}) = \widetilde{\mathbf{H}}(\boldsymbol{p}, \boldsymbol{s})$ for all $\boldsymbol{p} \in \widetilde{\mathrm{P}}$ and $\boldsymbol{s} \in \widetilde{\mathrm{S}}$, meaning that $\mathcal{H}$ is consistent with $\widetilde{\mathbf{H}}$.

(3) $\mathcal{H}$ is minimal.

Proof. See App. C.2. ■

4.3 The Learning Algorithm

Our weighted $\mathbf{L}^*$ algorithm, with its main loop detailed in Alg. 1, employs the subroutines outlined in Alg. 2.

Initialization. The prefix and suffix sets $\widetilde{\mathrm{P}}$ and $\widetilde{\mathrm{S}}$ are initialized as $\{\varepsilon\}$ and the empirical Hankel matrix to the zero matrix.

Handling inconsistencies. The subroutine MAKECONSISTENT in (Line 7,Alg. 1, and Alg. 2) looks for rows $\boldsymbol{p}, \boldsymbol{p}' \in \widetilde{\mathrm{P}}$ that make $\widetilde{\mathbf{H}}$ non-consistent: $\widetilde{\mathbf{H}}_{\boldsymbol{p}} = \widetilde{\mathbf{H}}_{\boldsymbol{p}'}$, but $\widetilde{\mathbf{H}}_{\boldsymbol{p}a} \neq \widetilde{\mathbf{H}}_{\boldsymbol{p}'a}$ for some $a \in \Sigma$. To find the column(s) that make the relation $\widetilde{\mathbf{H}}_{\boldsymbol{p}a} \equiv \widetilde{\mathbf{H}}_{\boldsymbol{p}'a}$ not true, we normalize over the sum of row entries and compare each entry pair-wise

Algorithm 1 The Weighted $\mathbf{L}^*$ algorithm. Initially, the empirical Hankel matrix $\widetilde{\mathbf{H}}$ is set to the zero matrix and the sets $\widetilde{\mathrm{P}}, \widetilde{\mathrm{S}}$ to the empty string ε.

```
1.  def L* (𝕆):
2.    while true :
3.      while true :
4.        if H̃ is not consistent :
5.          MAKECONSISTENT(𝕆, H̃)
6.        else if H̃ is not closed :
7.          MAKECLOSED(𝕆, H̃)
8.        else : break
9.      H ← MAKEAUTOMATON(H̃)
10.     if EQUIVALENT(𝕆, H) : return H
11.     else :
12.       p ← COUNTEREXAMPLE(𝕆, H)
13.       for t = 1 to |p| + 1 :
14.         P̃ ← P̃ ∪ {p_<t}
15.       COMPLETE(𝕆, H̃)
```

(Alg. 2, Line 7).[8] If a column indexed by $\boldsymbol{s} \in \widetilde{\mathrm{S}}$ is found to make the empirical Hankel matrix not consistent, $a\boldsymbol{s}$ is added to $\widetilde{\mathrm{S}}$. This results in the new equivalence classes $[\boldsymbol{p}]$ and $[\boldsymbol{p}']$ because $\widetilde{\mathbf{H}}_{\boldsymbol{p}}$ and $\widetilde{\mathbf{H}}_{\boldsymbol{p}'}$ do not match anymore on the column indexed by $a\boldsymbol{s}$. See Lemma 5 in App. C for more details.

Closing $\widetilde{\mathbf{H}}$. MAKECLOSED (Alg. 1, Line 7; Alg. 2) adds to $\widetilde{\mathrm{P}}$ the missing prefixes required to make the empirical Hankel matrix closed. It searches for $\boldsymbol{p} \in \widetilde{\mathrm{P}}, a \in \Sigma$ such that $\widetilde{\mathbf{H}}_{\boldsymbol{p}a} \not\equiv \widetilde{\mathbf{H}}_{\boldsymbol{p}'}$ for every $\boldsymbol{p}' \in \widetilde{\mathrm{P}}$, and adds $\boldsymbol{p}a$ to $\widetilde{\mathrm{P}}$. This results in the new equivalence class $[\boldsymbol{p}a]$. See Lemma 5 in App. C for more details.

Filling out $\widetilde{\mathbf{H}}$. Finally, COMPLETE fills the empty entries of $\widetilde{\mathbf{H}}$ by asking membership queries of the oracle.

Handling inconsistencies, closing $\widetilde{\mathbf{H}}$, and filling $\widetilde{\mathbf{H}}$ is carried out by the inner while loop (Lines 3 to 8) of Alg. 1, which continues until $\widetilde{\mathbf{H}}$ is both closed and consistent.

Generating the hypothesis automaton. When $\widetilde{\mathbf{H}}$ is closed and consistent, Alg. 1 generates the hypothesis automaton $\mathcal{H}$ (Line 9) and submits an equivalence query to the oracle (Line 10). If the oracle answers positively, Alg. 1 halts and returns $\mathcal{H}$. Otherwise, the oracle provides a counterexam-

[8] Indeed, $\widetilde{\mathbf{H}}_{\boldsymbol{p}a} \equiv \widetilde{\mathbf{H}}_{\boldsymbol{p}'a}$ iff $\widetilde{\mathbf{H}}_{\boldsymbol{p}a}(\boldsymbol{s}) = k \otimes \widetilde{\mathbf{H}}_{\boldsymbol{p}'a}(\boldsymbol{s})$ for every $\boldsymbol{s} \in \widetilde{\mathrm{S}}$ and for some $k \in \mathbb{K}\backslash\{0\}$. Therefore $\widetilde{\mathbf{H}}_{\boldsymbol{p}a} \equiv \widetilde{\mathbf{H}}_{\boldsymbol{p}'a}$ entails that $\frac{\widetilde{\mathbf{H}}(\boldsymbol{p}a, \boldsymbol{s})}{\bigoplus_{\boldsymbol{s}'\in\widetilde{\mathrm{S}}} \widetilde{\mathbf{H}}(\boldsymbol{p}a, \boldsymbol{s}')} = \frac{k \otimes \widetilde{\mathbf{H}}(\boldsymbol{p}'a, \boldsymbol{s})}{k \otimes \bigoplus_{\boldsymbol{s}'\in\widetilde{\mathrm{S}}} \widetilde{\mathbf{H}}(\boldsymbol{p}'a, \boldsymbol{s}')}$ for every $\boldsymbol{s} \in \widetilde{\mathrm{S}}$.

Algorithm 2 Subroutines of Alg. 1.

```
1.  def MakeConsistent(𝕆, H̃):
2.    for ⟨p, p'⟩ ∈ P̃ × P̃ :
3.      if H̃_p ≡ H̃_p' :
4.        for ⟨a, s⟩ ∈ Σ × S̃ :
5.          Z_pa ← ⊕_{s'∈S̃} H̃(pa, s')
6.          Z_p'a ← ⊕_{s'∈S̃} H̃(p'a, s')
7.          if H̃(pa, s)/Z_pa ≠ H̃(p'a, s)/Z_p'a :
8.            S̃ ← S̃ ∪ {as}
9.    Complete(𝕆, H̃)
10. def MakeClosed(𝕆, H̃):
11.   for ⟨p, a⟩ ∈ P̃ × Σ :
12.     if ∃p' ∈ P̃ s.t. H̃_pa ≠ H̃_p' :
13.       P̃ ← P̃ ∪ {pa}
14.   Complete(𝕆, H̃)
15. def Complete(𝕆, H̃):
16.   for p ∈ P̃ ∘ Σ^{≤1} :
17.     for s ∈ S̃ :
18.       H̃(p, s) ← Membership(𝕆, ps)
```

ple $\boldsymbol{t}$, which is added to $\widetilde{\mathrm{P}}$ along with its prefixes. $\widetilde{\mathbf{H}}$ is then updated through membership queries (Lines 12 to 15). The algorithm continues until $\widetilde{\mathbf{H}}$ is closed and consistent again.

The correctness of Alg. 1 is shown by the following theorem.

Theorem 3. *Let $\langle \mathbb{K}, \oplus, \otimes, \mathbf{0}, \mathbf{1} \rangle$ be a semifield and Σ an alphabet. Let $\mathbb{O}$ be an oracle for the deterministic regular language $L^\star : \Sigma^* \to \mathbb{K}$, whose minimal WDFSA has N states. Then, Alg. 1 returns a minimal WDFSA $\mathcal{H}$ generating $L^\star$ in time $\mathcal{O}\left(N^4 M^2 |\Sigma|\right)$, where M is the length of the longest counterexample provided by $\mathbb{O}$.*

Proof. See App. C.3. ■

Relationship Angluin's (1987) algorithm. We strive to keep the weighted $\mathbf{L}^*$ algorithm as faithful as possible to Angluin's (1987). This brings with it the advantages of familiarity and ease of analysis. A closer look at the pseudocode in Alg. 1 and the proofs of Thms. 2 and 3 reveals that our algorithm should be familiar to anyone acquainted with the unweighted version: The learner uses the same learning loop, procedures of closing and completing the observation table, the same set of calls to the oracle, and an analogous construction of the hypothesis automaton.

$\mathbf{L}^*$ and Minimization. Building an automaton whose states correspond to right language equivalence classes (cf. §3) provides the basis for both minimization (Hopcroft and Ullman, 1979; Mohri, 1997) as well as the $\mathbf{L}^*$ algorithm. Both algorithms also rely on some sort of weight normalization to identify equivalent states—weight pushing (Mohri, 1997) normalizes the outgoing transition weights in minimization, while the $\mathbf{L}^*$ algorithm normalizes the rows of the empirical Hankel matrix to compute the transition weights (Eqs. (4) and (6)). The key distinction between the two algorithms lies in their utilization of the discovered equivalence classes: minimization *merges* equivalent states, whereas $\mathbf{L}^*$ incrementally *adds* states to the automaton as it identifies new equivalence classes until termination.

5 Conclusion

We introduce a weighted $\mathbf{L}^*$ algorithm, an oracle-based algorithm for learning weighted regular languages, building upon the paradigm pioneered by Angluin (1987). While similar methods have been proposed before, our method is novel in that it learns an *exact* deterministic WFSA, akin to the original Angluin's (1987) unweighted version. We highlight language model analysis as a possible application, given that language models describe probability distributions over strings (Icard, 2020; Nowak et al., 2023), i.e., weighted languages.

Limitations

One of the limitations of weighted $\mathbf{L}^*$ is that it requires an oracle capable of answering membership and equivalence queries. However, in the case we want to use $\mathbf{L}^*$ to study a language model, this is the ideal setting, as we can use the language model itself as the oracle (Weiss et al., 2018; Okudono et al., 2019; Weiss et al., 2019). Another limitation to the applications of our work is that not every language model is efficiently representable as a finite-state machine. For instance, Merrill (2019) shows that LSTMs are strictly more powerful than FSAs. Therefore, in practice, one may have to use a simplified abstraction of the model one aims to learn (Weiss et al., 2021), inevitably reducing the model's expressivity. Lastly, we note that $\mathbf{L}^*$ is not capable of learning *non-deterministic* regular languages, which, in the weighted case, can be a

strict subset of weighted regular languages.[9]

Acknowledgements

Ryan Cotterell acknowledges support from the Swiss National Science Foundation (SNSF) as part of the "The Nuts and Bolts of Language Models" project. Anej Svete is supported by the ETH AI Center Doctoral Fellowship.

References

Cyril Allauzen and Mehryar Mohri. 2003. Efficient algorithms for testing the twins property. *Journal of Automata, Languages and Combinatorics*, 8(2):117–144.

Dana Angluin. 1987. Learning regular sets from queries and counterexamples. *Information and Computation*, 75(2):87–106.

Borja Balle, Xavier Carreras, Franco M. Luque, and Ariadna Quattoni. 2014. Spectral learning of weighted automata. *Machine Learning*, 96(1):33–63.

Borja Balle and Mehryar Mohri. 2012. Spectral learning of general weighted automata via constrained matrix completion. In *Advances in Neural Information Processing Systems*, volume 25. Curran Associates, Inc.

Amos Beimel, Francesco Bergadano, Nader H. Bshouty, Eyal Kushilevitz, and Stefano Varricchio. 2000. Learning functions represented as multiplicity automata. *J. ACM*, 47(3):506–530.

Francesco Bergadano and Stefano Varricchio. 1996. Learning behaviors of automata from multiplicity and equivalence queries. *SIAM Journal on Computing*, 25(6):1268–1280.

A. W. Biermann and J. A. Feldman. 1972. On the synthesis of finite-state machines from samples of their behavior. *IEEE Transactions on Computers*, C-21(6):592–597.

Laure Daviaud and Marianne Johnson. 2024. Feasability of learning weighted automata on a semiring. *Preprint*, arXiv:2309.07806.

E Mark Gold. 1978. Complexity of automaton identification from given data. *Information and Control*, 37(3):302–320.

Joshua Goodman. 1996. Parsing algorithms and metrics. In *34th Annual Meeting of the Association for Computational Linguistics*, pages 177–183, Santa Cruz, California, USA. Association for Computational Linguistics.

[9]Precisely, this depends on the specific semifield we are considering. The boolean semifield reduces to the unweighted case thus every regular language is deterministic. Conversely, Allauzen and Mohri (2003) provide an example of a language in the real semifield which is not deterministic.

John E. Hopcroft and Jeffrey D. Ullman. 1979. *Introduction to Automata Theory, Languages, and Computation*. Addison-Wesley Publishing Company, Inc.

Thomas F. Icard. 2020. Calibrating generative models: The probabilistic chomsky–schützenberger hierarchy. *Journal of Mathematical Psychology*, 95:102308.

Jaap Jumelet and Willem Zuidema. 2023. Transparency at the source: Evaluating and interpreting language models with access to the true distribution. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 4354–4369, Singapore. Association for Computational Linguistics.

William Merrill. 2019. Sequential neural networks as automata. In *Proceedings of the Workshop on Deep Learning and Formal Languages: Building Bridges*, pages 1–13, Florence. Association for Computational Linguistics.

Mehryar Mohri. 1997. Finite-state transducers in language and speech processing. *Computational Linguistics*, 23(2):269–311.

Mehryar Mohri. 2009. *Weighted Automata Algorithms*, pages 213–254. Springer Berlin Heidelberg, Berlin, Heidelberg.

Mehryar Mohri and Michael Riley. 2001. A weight pushing algorithm for large vocabulary speech recognition. In *Proc. 7th European Conference on Speech Communication and Technology (Eurospeech 2001)*, pages 1603–1606.

Franz Nowak, Anej Svete, Alexandra Butoi, and Ryan Cotterell. 2024. On the representational capacity of neural language models with chain-of-thought reasoning. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Bangkok, Thailand. Association for Computational Linguistics.

Franz Nowak, Anej Svete, Li Du, and Ryan Cotterell. 2023. On the representational capacity of recurrent neural language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 7011–7034, Singapore. Association for Computational Linguistics.

Takamasa Okudono, Masaki Waga, Taro Sekiyama, and Ichiro Hasuo. 2019. Weighted automata extraction from recurrent neural networks via regression on state spaces. *Preprint*, arXiv:1904.02931.

Gail Weiss, Yoav Goldberg, and Eran Yahav. 2018. Extracting automata from recurrent neural networks using queries and counterexamples. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 5247–5256. PMLR.

Gail Weiss, Yoav Goldberg, and Eran Yahav. 2019. Learning deterministic weighted automata with queries and counterexamples. In *Advances in Neural Information Processing Systems*, volume 32.

Zarah Weiss, Xiaobin Chen, and Detmar Meurers. 2021. Using broad linguistic complexity modeling for cross-lingual readability assessment. In *Proceedings of the 10th Workshop on NLP for Computer Assisted Language Learning*, pages 38–54, Online. LiU Electronic Press.

A Language Equivalence

Lemma 1. *Let L_1 and L_1 be two languages with weights on the same semifield $\langle \mathbb{K}, \oplus, \otimes, \mathbf{0}, \mathbf{1} \rangle$. Further, let us write $L_1 \equiv L_2$ iff supp(L_1) = supp(L_2) and there exists a $k \in \mathbb{K} \backslash \{\mathbf{0}\}$ such that $L_1(\boldsymbol{p}) = k \otimes L_2(\boldsymbol{p})$ for every $\boldsymbol{p} \in$ supp(L_1). Then $\equiv$ is an equivalence relation.*

Proof. We show that $\equiv$ is:

(1) symmetric, since if $\text{supp}(L_1) = \text{supp}(L_2)$ and there exists $k \in \mathbb{K} \backslash \{\mathbf{0}\}$ such that $L_1(\boldsymbol{p}) = k \otimes L_2(\boldsymbol{p})$ for every $\boldsymbol{p} \in \text{supp}(L_1)$, then $\text{supp}(L_2) = \text{supp}(L_1)$ and we can write $L_2(\boldsymbol{p}) = k^{-1} \otimes L_1(\boldsymbol{p})$, for every $\boldsymbol{p} \in \text{supp}(L_2)$.

(2) transitive, since if $\text{supp}(L_1) = \text{supp}(L_2)$ and $L_1(\boldsymbol{p}) = k_1 \otimes L_2(\boldsymbol{p})$ for every $\boldsymbol{p} \in \text{supp}(L_1)$ and for some $k_1 \in \mathbb{K} \backslash \{\mathbf{0}\}$; and $\text{supp}(L_2) = \text{supp}(L_3)$, $L_2(\boldsymbol{p}) = k_2 \otimes L_2(\boldsymbol{p})$ for every $\boldsymbol{p} \in \text{supp}(L_2)$ and for some $k \in \mathbb{K} \backslash \{\mathbf{0}\}$ then $\text{supp}(L_1) = \text{supp}(L_2)$ and $L_1(\boldsymbol{p}) = k_1 \otimes k_2 \otimes L_3(\boldsymbol{p})$.

(3) reflexive, since $\text{supp}(L_1) = \text{supp}(L_1)$ and $L_1(\boldsymbol{p}) = \mathbf{1} \otimes L_1(\boldsymbol{p})$.

Note that if $\langle \mathbb{K}, \oplus, \otimes, \mathbf{0}, \mathbf{1} \rangle$ is a generic semiring that does not satisfy the semifield axioms, then $\equiv$ may not be symmetric, since an inverse for k may not be defined. ■

B Learning WDFSAs in a Semifields of Fractions

The algorithm presented in the main text learns a weighted deterministic finite-state automaton (WDFSA) over a *semifield* $\langle \mathbb{K}, \oplus, \otimes, \mathbf{0}, \mathbf{1} \rangle$. This is done mainly to allow for a more concise and approachable presentation of the algorithm. In this section, we present another framework that allows for learning WDFSAs over semirings that can be lifted into a *semifield of fractions*.

Definition 1. *A **semi-integral domain** is a semiring $\langle \mathbb{K}, \oplus, \otimes, \mathbf{0}, \mathbf{1} \rangle$ where $(\mathbb{K}, \otimes, \mathbf{1})$ is a* commutative ***cancellative*** *monoid: for all $x, y \in \mathbb{K}$, if $x \otimes y = \mathbf{0}$, then $x = \mathbf{0}$ or $y = \mathbf{0}$.*

A semi-integral domain allows us to define a **semifield of fractions** as follows.

Definition 2. *Let $\langle \mathbb{K}, \oplus, \otimes, \mathbf{0}, \mathbf{1} \rangle$ be a semi-integral domain. Its **semifield of fractions** is the tuple $\langle \mathbb{S}, \underline{\oplus}, \underline{\otimes}, \underline{\mathbf{0}}, \underline{\mathbf{1}} \rangle$ where*

$$\mathbb{S} = \left\{ \frac{x}{y} \mid x, y \in \mathbb{K}, y \neq \mathbf{0} \right\} \tag{7}$$

equipped with the operations

$$\frac{x_1}{y_1} \underline{\oplus} \frac{x_2}{y_2} \stackrel{\text{def}}{=} \frac{x_1 \otimes y_2 \oplus x_2 \otimes y_1}{y_1 \otimes y_2} \tag{8a}$$

$$\frac{x_1}{y_1} \underline{\otimes} \frac{x_2}{y_2} \stackrel{\text{def}}{=} \frac{x_1 \otimes x_2}{y_1 \otimes y_2} \tag{8b}$$

$$\underline{\mathbf{0}} \stackrel{\text{def}}{=} \frac{\mathbf{0}}{\mathbf{1}} \tag{8c}$$

$$\underline{\mathbf{1}} \stackrel{\text{def}}{=} \frac{\mathbf{1}}{\mathbf{1}} \tag{8d}$$

where $\oplus$ and $\otimes$ are the operations of the semi-integral domain.

This definition induces a natural equivalence relation on the semifield of fractions:

$$\frac{x_1}{y_1} \sim \frac{x_2}{y_2} \iff x_1 \otimes y_2 = x_2 \otimes y_1. \tag{9}$$

We can then define the **equivalence class** of $\frac{x}{y}$ as $\left[\frac{x}{y}\right] = \left\{ \frac{x'}{y'} \mid \frac{x'}{y'} \sim \frac{x}{y} \right\}$. The **canonical form** of a fraction $\frac{x}{y}$ is the equivalence class $\left[\frac{x}{y}\right]$.

Alg. 1 generalizes over a semifield of fractions in the following sense.

1. We lift the weights of the semifield-weighted learned language to the semifield of fractions. This simply entails interpreting the string weights $L^\star(\boldsymbol{y}) \in \mathbb{K}$ as $\frac{L^\star(\boldsymbol{y})}{\mathbf{1}}$.

2. The WDFSA is then learned as in the semifield case, but with the weights in the semifield of fractions.

The initial, final, and transition weights of the automaton learned in this manner naturally fall in the semifield of fractions, and thus do not necessarily correspond to any specific element of the original semiring. However, the string weights defined by the learned automaton—and therefore its language—will be the same as in the ground-truth *semiring-weighted* WFSA.

Theorem 4. *Let $L^\star$ be a weighted regular language over a semi-integral domain $\langle \mathbb{K}, \oplus, \otimes, \mathbf{0}, \mathbf{1} \rangle$. Let $\mathcal{H}$ be the WDFSA learned by Alg. 1 over the corresponding semifield of fractions over $\mathbb{K}$, $\langle \mathbb{S}, \underline{\oplus}, \underline{\otimes}, \underline{\mathbf{0}}, \underline{\mathbf{1}} \rangle$. Then, $L_{\mathcal{H}}(\boldsymbol{y}) \sim \frac{L^\star(\boldsymbol{y})}{\mathbf{1}}$ for all $\boldsymbol{y} \in \Sigma^*$.*

Proof. The semifield of fractions is a semifield. Furthermore, since lifting the string weights does not affect the determinism of the language, the lifted language $L^\star$ satisfies the requirements for Alg. 1, which then correctly learns the field-of-fractions-weighted language: $L_{\mathcal{H}}(\boldsymbol{y}) \sim \frac{L^\star(\boldsymbol{y})}{\mathbf{1}}$. ■

C Proofs

C.1 Proof of Thm. 1

Theorem 1 (Myhill–Nerode)**.** *Let $\mathcal{A}$ be a semifield-weighted DFSA and $L_{\mathcal{A}}$ its weighted language. Then, $\sim_{L_{\mathcal{A}}}$ induces a finite number of equivalence classes on Σ^*, which equals the number of states of a minimal automaton for $L_{\mathcal{A}}$.*

Proof. Let $\mathcal{A} = (\Sigma, Q, \delta, \lambda, \rho)$ be a semifield-weighted DFSA. Since $\mathcal{A}$ is deterministic, there is at most one initial state q_I. Every string $s \in \Sigma^*$ can be mapped to at most one non-zero-weight path $\boldsymbol{\pi}_s$ starting in q_I and yielding s. Set $q_s = n(\pi_s)$.

We claim that

$$\begin{aligned} &\boldsymbol{x}^{-1}L_{\mathcal{A}} \equiv \boldsymbol{y}^{-1}L_{\mathcal{A}} \overset{1}{\iff} \forall \boldsymbol{z} \in \Sigma^*, \boldsymbol{xz} \in \operatorname{supp}(L_{\mathcal{A}}) \text{ iff} \\ &\quad \boldsymbol{yz} \in \operatorname{supp}(L_{\mathcal{A}}) \overset{2}{\iff} (q_{\boldsymbol{x}} = q_{\boldsymbol{y}} \text{ and } \boldsymbol{w}(\boldsymbol{\pi}_{\boldsymbol{x}}) \otimes \boldsymbol{w}(\boldsymbol{\pi}_{\boldsymbol{y}}) \neq 0) \text{ or } (\boldsymbol{w}(\boldsymbol{\pi}_{\boldsymbol{x}}) = \boldsymbol{w}(\boldsymbol{\pi}_{\boldsymbol{y}}) = 0) \end{aligned} \tag{10}$$

$\Rightarrow)^{1\&2}$ Both are straightforward. $\Leftarrow)$ If $q_{\boldsymbol{x}} = q_{\boldsymbol{y}}$. Accordingly, for any $\boldsymbol{z} \in \Sigma^*$, we have

$$\begin{aligned} \boldsymbol{x}^{-1}L_{\mathcal{A}}(\boldsymbol{z}) &= L_{\mathcal{A}}(\boldsymbol{xz}) \\ &= \lambda(q_I) \otimes \boldsymbol{w}(\boldsymbol{\pi}_{\boldsymbol{x}}) \otimes \boldsymbol{w}(\boldsymbol{\pi}_{\boldsymbol{z}}) \otimes \rho(q_{\boldsymbol{z}}) \\ &= \frac{\boldsymbol{w}(\boldsymbol{\pi}_{\boldsymbol{x}})}{\boldsymbol{w}(\boldsymbol{\pi}_{\boldsymbol{y}})} \otimes \boldsymbol{y}^{-1}L_{\mathcal{A}}(\boldsymbol{z}). \end{aligned}$$

And this shows that $\boldsymbol{x}^{-1}L_{\mathcal{A}} \equiv \boldsymbol{y}^{-1}L_{\mathcal{A}}$. In conclusion, the claim proves that $|\mathcal{A}/\sim_{L_{\mathcal{A}}}| \leqslant |Q|$. This shows that the equivalence classes induced by $L_{\mathcal{A}}$ are finite. The proof that their number is equal to $|Q^{\mathcal{M}}|$, where $Q^{\mathcal{M}}$ is the set of states of a minimal automaton for $L_{\mathcal{A}}$ is done by construction, for which we refer the reader to (Mohri, 1997, section 3.7). ■

C.2 Proof of Thm. 2

Theorem 2 (The empirical Hankel Automaton)**.** *Let $\widetilde{\mathbf{H}} \colon \widetilde{\mathrm{P}} \circ \Sigma^{\leqslant 1} \times \widetilde{\mathrm{S}} \to \mathbb{K}$ be a closed and consistent empirical Hankel matrix and let $\mathcal{H}$ be the empirical Hankel automaton induced by $\widetilde{\mathbf{H}}$. Then:*
(1) $\mathcal{H}$ is a well-defined WDFSA.
(2) $L_{\mathcal{H}}(\boldsymbol{ps}) = \widetilde{\mathbf{H}}(\boldsymbol{p}, \boldsymbol{s})$ for all $\boldsymbol{p} \in \widetilde{\mathrm{P}}$ and $\boldsymbol{s} \in \widetilde{\mathrm{S}}$, meaning that $\mathcal{H}$ is consistent with $\widetilde{\mathbf{H}}$.
(3) $\mathcal{H}$ is minimal.

Proof. Let $\widetilde{\mathbf{H}} \colon \Sigma^* \times \Sigma^* \to \mathbb{K}$ be a closed and consistent empirical Hankel matrix and let $\mathcal{H}$ be $\widetilde{\mathbf{H}}$'s induced WDFSA.

(1) The set of states Q is a well-defined finite set since, by assumption, the prefixes $\widetilde{\mathrm{P}}$ are a finite set and $\sim_{\widetilde{\mathbf{H}}}$ is a well-defined equivalence relation on $\widetilde{\mathrm{P}}$. The set of transitions (Eq. (4)) is well-defined since for any $q_{[\boldsymbol{p}]}$ and every symbol $a \in \Sigma$: $i)$ $q_{[r(\boldsymbol{p}a)]}$ is uniquely defined $ii)$ $q_{[r(\boldsymbol{p}a)]}$ is in Q since the empirical Hankel matrix is closed. Regarding the initial and final states, we just note that since $\widetilde{\mathrm{P}}$ and $\widetilde{\mathrm{S}}$ are respectively prefix and suffix closed, they always contain the empty string ε.

Determinism: To see that $\mathcal{H}$ is deterministic, note that:

(a) There is at most one state $q_{[\varepsilon]}$ with non-zero initial weight (Eq. (5)).

(b) Following Eq. (4), for every state $q_{[\boldsymbol{p}]}$ and for every input symbol $a \in \Sigma$, there is exactly one state $q_{[r(\boldsymbol{p}a)]}$ such that the transition $q_{[\boldsymbol{p}]} \xrightarrow{a/w} q_{[r(\boldsymbol{p}a)]}$ is in the set of transitions δ. We further, consistency ensures that the next state is independent of the specific choice of representatives for the equivalence classes.

(2) We begin by showing the following lemmata.

Lemma 2. *Let $\widetilde{\mathbf{H}}$ be a closed and consistent empirical Hankel matrix, with rows $\widetilde{\mathrm{P}} \circ \Sigma^{\leqslant 1}$ and columns $\widetilde{\mathrm{S}}$. Then, for every two $\boldsymbol{p}, \boldsymbol{p}' \in \widetilde{\mathrm{P}}$ such that $\widetilde{\mathbf{H}}(\boldsymbol{p}, \boldsymbol{s}) = k \otimes \widetilde{\mathbf{H}}(\boldsymbol{p}', \boldsymbol{s})\ \forall \boldsymbol{s} \in \widetilde{\mathrm{S}}$, we also have that $\widetilde{\mathbf{H}}(\boldsymbol{p}a, \boldsymbol{s}) = k \otimes \widetilde{\mathbf{H}}(\boldsymbol{p}'a, \boldsymbol{s})$, $\forall \boldsymbol{s} \in \widetilde{\mathrm{S}}$ and for $a \in \Sigma$.*

Proof. Since the table is consistent, we know that if for two $\boldsymbol{p}, \boldsymbol{p}' \in \widetilde{\mathrm{P}}$, $\widetilde{\mathbf{H}}(\boldsymbol{p}, \boldsymbol{s}) = k_1 \otimes \widetilde{\mathbf{H}}(\boldsymbol{p}', \boldsymbol{s})$ for every $\boldsymbol{s} \in \widetilde{\mathrm{S}}$ and for some $k_1 \in \mathbb{K}\backslash\{\mathbf{0}\}$, then $\widetilde{\mathbf{H}}(\boldsymbol{p}a, \boldsymbol{s}) = k_2 \otimes \widetilde{\mathbf{H}}(\boldsymbol{p}'a, \boldsymbol{s})$ for $\boldsymbol{s} \in \widetilde{\mathrm{S}}, a \in \Sigma$ and for some $k_2 \in \mathbb{K}\backslash\{\mathbf{0}\}$. Now, since $\widetilde{\mathrm{S}}$ is suffix closed, we can chose $\boldsymbol{s} \in \widetilde{\mathrm{S}}$ such that we can write $\boldsymbol{s} = a\boldsymbol{s}'$ for some $\boldsymbol{s}'$ in $\widetilde{\mathrm{S}}$, and therefore:

$$\widetilde{\mathbf{H}}(\boldsymbol{p}, \boldsymbol{s}) = k_1 \otimes \widetilde{\mathbf{H}}(\boldsymbol{p}', \boldsymbol{s}) \qquad \text{(11a, By hypotheis)}$$

$$\widetilde{\mathbf{H}}(\boldsymbol{p}, a\boldsymbol{s}') = k_1 \otimes \widetilde{\mathbf{H}}(\boldsymbol{p}', a\boldsymbol{s}') \qquad \text{(11b, Rewriting } \boldsymbol{s} \text{ as } a\boldsymbol{s}')$$

$$\widetilde{\mathbf{H}}(\boldsymbol{p}a, \boldsymbol{s}') = k_1 \otimes \widetilde{\mathbf{H}}(\boldsymbol{p}'a, \boldsymbol{s}') \qquad \text{(11c, Skew diagonals of the Hankel matrix are constant)}$$

and hence $k_1 = k_2$. ■

Lemma 3. *Let $\widetilde{\mathbf{H}}$ be a closed and consistent empirical Hankel matrix with rows $\widetilde{\mathrm{P}} \circ \Sigma^{\leqslant 1}$ and columns $\widetilde{\mathrm{S}}$, and let $\mathcal{H}$ be the empirical hankel automaton generated from $\widetilde{\mathbf{H}}$. Then for every $\boldsymbol{p} \in \widetilde{\mathrm{P}}$ let us denote with $\boldsymbol{\pi}_{\boldsymbol{p}}$ the path in $\mathcal{H}$ with initial state $q_{[\varepsilon]}$ and yield $\boldsymbol{p}$. The following equation holds:*

$$\lambda\left(q_{[\varepsilon]}\right) \otimes w(\boldsymbol{\pi}_{\boldsymbol{p}}) = \bigoplus_{\boldsymbol{s} \in \widetilde{\mathrm{S}}} \widetilde{\mathbf{H}}(\boldsymbol{p}, \boldsymbol{s}) \tag{12}$$

Proof. We proceed by induction on the string length.

Base Case. Let $\boldsymbol{p} = \varepsilon$. Then

$$\lambda\left(q_{[\varepsilon]}\right) = \bigoplus_{\boldsymbol{s} \in \widetilde{\mathrm{S}}} \widetilde{\mathbf{H}}(\varepsilon, \boldsymbol{s}) \tag{13}$$

by the definition of initial weight.

Induction Step. Recall that $\widetilde{\mathrm{P}}$ is prefix-closed. Let $\boldsymbol{p} \in \widetilde{\mathrm{P}}$ be a string with length $i \geqslant 1$ so that we can write $\boldsymbol{p} = \boldsymbol{p}_{<i}a$ for some $a \in \Sigma$ and some $\boldsymbol{p} \in \widetilde{\mathrm{P}}$. Then our induction hypothesis is that:

$$\lambda\left(q_{[\varepsilon]}\right) \otimes w(\boldsymbol{\pi}_{\boldsymbol{p}_{<i}}) = \bigoplus_{\boldsymbol{s} \in \widetilde{\mathrm{S}}} \widetilde{\mathbf{H}}(\boldsymbol{p}_{<i}, \boldsymbol{s}) \tag{14}$$

And then we can write

$$\lambda\left(q_{[\varepsilon]}\right) \otimes w(\boldsymbol{\pi}_{\boldsymbol{p}}) = \lambda\left(q_{[\varepsilon]}\right) \otimes w(\boldsymbol{\pi}_{\boldsymbol{p}_{<i}}) \otimes \frac{\bigoplus_{\boldsymbol{s}' \in \widetilde{\mathrm{S}}} \widetilde{\mathbf{H}}(r(\boldsymbol{p}_{<i})a, \boldsymbol{s})}{\bigoplus_{\boldsymbol{s} \in \widetilde{\mathrm{S}}} \widetilde{\mathbf{H}}(r(\boldsymbol{p}_{<i}), \boldsymbol{s})} \tag{15a}$$

$$= \bigoplus_{\boldsymbol{s} \in \widetilde{\mathrm{S}}} \widetilde{\mathbf{H}}(\boldsymbol{p}_{<i}, \boldsymbol{s}) \otimes \frac{\bigoplus_{\boldsymbol{s}' \in \widetilde{\mathrm{S}}} \widetilde{\mathbf{H}}(r(\boldsymbol{p}_{<i})a, \boldsymbol{s})}{\bigoplus_{\boldsymbol{s} \in \widetilde{\mathrm{S}}} \widetilde{\mathbf{H}}(r(\boldsymbol{p}_{<i}), \boldsymbol{s})} \tag{15b, Induction Hypothesis}$$

$$= \bigoplus_{\boldsymbol{s} \in \widetilde{\mathrm{S}}} \widetilde{\mathbf{H}}(\boldsymbol{p}_{<i}, \boldsymbol{s}) \otimes \frac{\cancel{k} \otimes \left(\bigoplus_{\boldsymbol{s}' \in \widetilde{\mathrm{S}}} \widetilde{\mathbf{H}}(\boldsymbol{p}_{<i}a, \boldsymbol{s})\right)}{\cancel{k} \otimes \left(\bigoplus_{\boldsymbol{s} \in \widetilde{\mathrm{S}}} \widetilde{\mathbf{H}}(\boldsymbol{p}_{<i}, \boldsymbol{s})\right)} \tag{15c, Lemma 2}$$

$$= \bigoplus_{\boldsymbol{s} \in \widetilde{\mathrm{S}}} \widetilde{\mathbf{H}}(\boldsymbol{p}_{<i}, \boldsymbol{s}) \otimes \frac{\bigoplus_{\boldsymbol{s}' \in \widetilde{\mathrm{S}}} \widetilde{\mathbf{H}}(\boldsymbol{p}_{<i}a, \boldsymbol{s})}{\bigoplus_{\boldsymbol{s} \in \widetilde{\mathrm{S}}} \widetilde{\mathbf{H}}(\boldsymbol{p}_{<i}, \boldsymbol{s})} \tag{15d, Lemma 2}$$

$$= \bigoplus_{\boldsymbol{s}' \in \widetilde{\mathrm{S}}} \widetilde{\mathbf{H}}(\boldsymbol{p}_{<i}a, \boldsymbol{s}) \tag{15e}$$

$$= \bigoplus_{\boldsymbol{s}' \in \widetilde{\mathrm{S}}} \widetilde{\mathbf{H}}(\boldsymbol{p}, \boldsymbol{s}) \tag{15f}$$

■

Lemma 4. *Let $\widetilde{\mathbf{H}}$ be a closed and consistent empirical Hankel matrix with rows $\widetilde{\mathrm{P}} \circ \Sigma^{\leqslant 1}$ and columns $\widetilde{\mathrm{S}}$, and let $\mathcal{H}$ be the empirical hankel automaton generated from $\widetilde{\mathbf{H}}$. Then for every $\boldsymbol{p} \in \widetilde{\mathrm{P}}$ and $\boldsymbol{s} \in \widetilde{\mathrm{S}}$, let us denote with $\boldsymbol{\pi}_{\boldsymbol{ps}}$ the path in $\mathcal{H}$ with initial state $q_{[\varepsilon]}$ and yield $\boldsymbol{ps}$. The following equation holds:*

$$\lambda\left(q_{[\varepsilon]}\right) \otimes w(\boldsymbol{\pi}_{\boldsymbol{p}}) \otimes \rho(n(\boldsymbol{\pi}_{\boldsymbol{ps}})) = \widetilde{\mathbf{H}}(\boldsymbol{p}, \boldsymbol{s}) \tag{16}$$

where we recall that $\rho(n(\boldsymbol{\pi}_{\boldsymbol{ps}}))$ denotes the final state of $\boldsymbol{\pi}_{\boldsymbol{ps}}$.

Proof. **Base Case:** for every $\boldsymbol{p} \in \widetilde{\mathrm{P}}$ we can write.

$$\lambda\left(q_{[\varepsilon]}\right) \otimes w(\boldsymbol{\pi}_{\boldsymbol{p}}) \otimes \rho(q_{r(p)}) = \lambda\left(q_{[\varepsilon]}\right) \otimes w(\boldsymbol{\pi}_{\boldsymbol{p}}) \otimes \frac{\widetilde{\mathbf{H}}(r(\boldsymbol{p}), \varepsilon)}{\bigoplus_{\boldsymbol{s} \in \widetilde{\mathrm{S}}} \widetilde{\mathbf{H}}(r(\boldsymbol{p}), \boldsymbol{s})} \tag{17a}$$

$$= \lambda\left(q_{[\varepsilon]}\right) \otimes w(\boldsymbol{\pi}_{\boldsymbol{p}}) \otimes \frac{\widetilde{\mathbf{H}}(\boldsymbol{p}, \varepsilon)}{\bigoplus_{\boldsymbol{s} \in \widetilde{\mathrm{S}}} \widetilde{\mathbf{H}}(\boldsymbol{p}, \boldsymbol{s})} \tag{17b, Since $\boldsymbol{p} \sim_{\widetilde{\mathbf{H}}} r(\boldsymbol{p})$:}$$

$$= \bigoplus_{\boldsymbol{s} \in \widetilde{\mathrm{S}}} \widetilde{\mathbf{H}}(\boldsymbol{p}, \boldsymbol{s}) \otimes \frac{\widetilde{\mathbf{H}}(\boldsymbol{p}, \varepsilon)}{\bigoplus_{\boldsymbol{s} \in \widetilde{\mathrm{S}}} \widetilde{\mathbf{H}}(\boldsymbol{p}, \boldsymbol{s})} \tag{17c, By Lemma 3:}$$

$$= \widetilde{\mathbf{H}}(\boldsymbol{p}, \varepsilon) \tag{17d}$$

Induction Step: Assume that:

$$\lambda\left(q_{[\varepsilon]}\right) \otimes w(\boldsymbol{\pi}_{\boldsymbol{ps}'}) \otimes \rho(n(\boldsymbol{\pi}_{\boldsymbol{ps}'})) = \widetilde{\mathbf{H}}(\boldsymbol{p}, \boldsymbol{s}') \tag{18}$$

$\forall \boldsymbol{p} \in \widetilde{\mathrm{P}}$ and forall $\boldsymbol{s}' \in \widetilde{\mathrm{S}}$, such that $|\boldsymbol{s}'| < k$. Now , let us assume that $\boldsymbol{s} \in \widetilde{\mathrm{S}}$ is a suffix with $|\boldsymbol{s}| = k$;

since $\widetilde{\mathrm{S}}$ is suffix closed, we can always write $\boldsymbol{s} = a\boldsymbol{s}'$ for some $a \in \Sigma$ and $\boldsymbol{s}' \in \widetilde{\mathrm{S}}$. Then, we write:

$$\lambda\left(q_{[\varepsilon]}\right) \otimes w(\boldsymbol{\pi}_{\boldsymbol{ps}}) \otimes \rho(n(\boldsymbol{\pi}_{\boldsymbol{ps}})) \tag{19a}$$

$$= \lambda\left(q_{[\varepsilon]}\right) \otimes w(\boldsymbol{\pi}_{\boldsymbol{pas}'}) \otimes \rho(n(\boldsymbol{\pi}_{\boldsymbol{pas}'})) \tag{19b}$$

$$= \lambda\left(q_{[\varepsilon]}\right) \otimes w(\boldsymbol{\pi}_{\boldsymbol{pa}}) \otimes w(\boldsymbol{\pi}_{\boldsymbol{s}'}) \otimes \rho(n(\boldsymbol{\pi}_{\boldsymbol{pas}'}))$$

(19c, where $\boldsymbol{\pi}_{\boldsymbol{s}'}$ is the path with initial state $q_{[r(\boldsymbol{pa})]}$ and yield $\boldsymbol{s}'$)

$$= \frac{\lambda\left(q_{[\varepsilon]}\right) \otimes w(\boldsymbol{\pi}_{\boldsymbol{pa}})}{\lambda\left(q_{[\varepsilon]}\right) \otimes w(\boldsymbol{\pi}_{r(\boldsymbol{pa})})} \otimes w(\boldsymbol{\pi}_{r(\boldsymbol{pa})}) \otimes w(\boldsymbol{\pi}_{\boldsymbol{s}'}) \otimes \rho(n(\boldsymbol{\pi}_{\boldsymbol{pas}'})) \tag{19d, Since } \widetilde{\mathbf{H}} \text{ is closed}$$

$$= \frac{\lambda\left(q_{[\varepsilon]}\right) \otimes w(\boldsymbol{\pi}_{\boldsymbol{pa}})}{\lambda\left(q_{[\varepsilon]}\right) \otimes w(\boldsymbol{\pi}_{r(\boldsymbol{pa})})} \otimes \widetilde{\mathbf{H}}(r(\boldsymbol{pa}), \boldsymbol{s}') \tag{19e, By induction hypothesis}$$

$$= \frac{\bigoplus_{\boldsymbol{s}_i \in \widetilde{\mathrm{S}}} \widetilde{\mathbf{H}}(\boldsymbol{pa}, \boldsymbol{s}_i)}{\bigoplus_{\boldsymbol{s}_i \in \widetilde{\mathrm{S}}} \widetilde{\mathbf{H}}(r(\boldsymbol{pa}), \boldsymbol{s}_i)} \otimes \widetilde{\mathbf{H}}(r(\boldsymbol{pa}), \boldsymbol{s}') \tag{19f, By Lemma 3}$$

$$= \frac{\widetilde{\mathbf{H}}(\boldsymbol{pa}, \boldsymbol{s}')}{\widetilde{\mathbf{H}}(r(\boldsymbol{pa}), \boldsymbol{s}')} \otimes \widetilde{\mathbf{H}}(r(\boldsymbol{pa}), \boldsymbol{s}') \tag{19g, Since } \boldsymbol{pa} \in [r(\boldsymbol{pa})]$$

$$= \widetilde{\mathbf{H}}(\boldsymbol{pa}, \boldsymbol{s}') = \widetilde{\mathbf{H}}(\boldsymbol{p}, a\boldsymbol{s}') \tag{19h, The emprical Hankel matrix is Skew diagnoal}$$

■

This lemma entails the proof of the second point of the theorem.

(3) By 2) of this theorem, we know that $\widetilde{\mathbf{H}}(\boldsymbol{p}, \boldsymbol{s}) = L_{\mathcal{H}}(\boldsymbol{ps})$ for every $\boldsymbol{p} \in \widetilde{\mathrm{P}}$ and $\boldsymbol{s} \in \widetilde{\mathrm{S}}$. Hence $\widetilde{\mathbf{H}}_{\boldsymbol{p}} : \widetilde{\mathrm{S}} \to \mathbb{K}$ is a restriction of $L_{\mathcal{H}} : \Sigma^* \to \mathbb{K}$ from Σ^* to $\widetilde{\mathrm{S}}$, and following the definition of the relations $\sim_{\widetilde{\mathbf{H}}}$ and $\sim_{L_{\mathcal{H}}}$, we have that $|\widetilde{\mathrm{P}}/\sim_{\widetilde{\mathbf{H}}}| \leqslant |\Sigma^*/\sim_{L_{\mathcal{H}}}|$.

Now, let us denote by N the number of states of $\mathcal{H}$, which by construction we know is equal to $|\widetilde{\mathrm{P}}/\sim_{\widetilde{\mathbf{H}}}|$. Therefore we can write $|\Sigma/\sim_{L_{\mathcal{H}}}| \geqslant |\widetilde{\mathrm{P}}/\sim_{\widetilde{\mathbf{H}}}| = N$. Then by the Myhill–Nerode theorem (Thm. 1), we have the number of states of the minimal automaton for $L_{\mathcal{H}}$ is at least N. But $\mathcal{H}$ has N states and hence $\mathcal{H}$ is minimal.

■

C.3 Proof of Thm. 3

In order to prove Thm. 3, we first give the following sequence of lemmata. Throughout the section, $L^\star$ denotes an unknown deterministic semifield-weighted regular language. Further, we write $\mathcal{A}^\star$ to denote an arbitrary, but fixed, minimal WFSA generating $L^\star$ with N states.

Lemma 5 (Equivalence classes increase #1)**.** *Let $\widetilde{\mathrm{P}}$ be a prefix-closed set of prefixes and $\widetilde{\mathrm{S}}$ a suffix-closed set of suffixes. Let $\widetilde{\mathbf{H}}$ be the empirical Hankel matrix of $L^\star$, whose rows and columns are indexed with $\widetilde{\mathrm{P}} \circ \Sigma^{\leqslant 1}$ and $\widetilde{\mathrm{S}}$ respectively. Then, the following statements hold:*

(1) If $\widetilde{\mathbf{H}}$ is not consistent, Alg. 1 adds a suffix to $\widetilde{\mathrm{S}}$ and the number of equivalence classes of $\widetilde{\mathrm{P}}$ increases.

(2) If $\widetilde{\mathbf{H}}$ is not closed, Alg. 1 adds a prefix to $\widetilde{\mathrm{P}}$ and the number of equivalence classes of $\widetilde{\mathrm{P}}$ increases.

Proof. We prove each of the statements in turn.

(1) If the empirical Hankel matrix is not consistent, MAKECONSISTENT (Alg. 1, Line 5) finds two prefixes $\boldsymbol{p}, \boldsymbol{p}' \in \widetilde{\mathrm{P}}$ such that $\widetilde{\mathbf{H}}_{\boldsymbol{p}} \equiv \widetilde{\mathbf{H}}_{\boldsymbol{p}'}$ but $\widetilde{\mathbf{H}}_{\boldsymbol{p}}(a\boldsymbol{s}) \not\equiv \widetilde{\mathbf{H}}_{\boldsymbol{p}'}(a\boldsymbol{s})$ for some $a \in \Sigma$ and $\boldsymbol{s} \in \widetilde{\mathrm{S}}$, and adds $a\boldsymbol{s}$ to $\widetilde{\mathrm{S}}$. After adding the suffix $\boldsymbol{s}$ to $\widetilde{\mathrm{S}}$, we have that $\widetilde{\mathbf{H}}_{\boldsymbol{p}} \not\equiv \widetilde{\mathbf{H}}_{\boldsymbol{p}'}$, and therefore an equivalence class has been divided in two.

(2) If $\widetilde{\mathbf{H}}$ is not closed, MAKECLOSED (Alg. 1, Line 7) finds $\boldsymbol{p} \in \widetilde{\mathrm{P}}$ and $a \in \Sigma$ such that $\widetilde{\mathbf{H}}_{\boldsymbol{pa}} \not\equiv \widetilde{\mathbf{H}}_{\boldsymbol{p}'}$ for every $\boldsymbol{p}' \in \widetilde{\mathrm{P}}$ and adds $\boldsymbol{pa}$ to $\widetilde{\mathrm{P}}$. Since there is no $\boldsymbol{p}' \neq \boldsymbol{pa}$ in $\widetilde{\mathrm{P}}$, such that $\boldsymbol{p}' \sim_{\widetilde{\mathbf{H}}} \boldsymbol{pa}$, it follows that a new equivalence class $[\boldsymbol{pa}]$ is added to $\widetilde{\mathrm{P}}/\sim_{\widetilde{\mathbf{H}}}$.

■

Lemma 6 (Equivalence classes increase #2)**.** *Let $\widetilde{\mathrm{P}}$ be a prefix-closed set of prefixes and $\widetilde{\mathrm{S}}$ a suffix-closed set of suffixes. Let $\widetilde{\mathbf{H}}$ be the empirical Hankel matrix of $L^\star$, whose rows and columns are indexed with $\widetilde{\mathrm{P}} \circ \Sigma^{\leqslant 1}$ and $\widetilde{\mathrm{S}}$ respectively. Then, each time the oracle replies with a counterexample $\boldsymbol{t}$, the number of equivalence classes of $\widetilde{\mathrm{P}}$ increases.*

Proof. When we add the counterexample $\boldsymbol{t}$ and its prefixes to $\widetilde{\mathrm{P}}$ (Alg. 1, Line 13) three outcomes are possible:

(1) The new empirical Hankel matrix $\widetilde{\mathbf{H}}'$ is not closed and/or consistent. In this case, new rows or columns are added, which, as shown by Lemma 5, increases the number of equivalence classes in $\widetilde{\mathrm{P}}$.

(2) The new empirical Hankel matrix $\widetilde{\mathbf{H}}'$ is closed and consistent and $\widetilde{\mathrm{P}}$ has at least one more equivalence class than in $\widetilde{\mathbf{H}}$.

(3) The new empirical Hankel matrix $\widetilde{\mathbf{H}}'$ is closed and consistent and $\widetilde{\mathrm{P}}$ has the same number of equivalence classes as in $\widetilde{\mathbf{H}}$. This implies that $\boldsymbol{t}_{[1:i]} \in [\boldsymbol{p}_i]$ for $i = 1, \ldots, |\boldsymbol{t}|$, and for some $\boldsymbol{p}_i \in \widetilde{\mathrm{P}}$, that was already in $\widetilde{\mathrm{P}}$ before the counterexample and its prefixes were added. Since the choice of representatives for the construction of the hypothesis automaton is fixed, this entails that the automaton $\mathcal{H}'$ generated from $\widetilde{\mathbf{H}}'$ is equal to the automaton $\mathcal{H}$ generated from $\widetilde{\mathbf{H}}$. However, this implies that $L_{\mathcal{H}} = L_{\mathcal{H}'}$ and therefore $\boldsymbol{t}$ was not a counterexample.

Therefore, both in case 1) and 2), the number of equivalence classes of $\widetilde{\mathrm{P}}$ increases, and case 3) cannot occur, as it would contradict the hypothesis that $\boldsymbol{t}$ is a counterexample. ■

Lemma 7 (Upper bound on the number of equivalence classes)**.** *Let L be a semifield-weighted deterministic regular language, and let $\widetilde{\mathbf{H}}$ be an empirical Hankel matrix with prefixes $\widetilde{\mathrm{P}}$ and suffixes $\widetilde{\mathrm{S}}$, containing observations from L. Then the number of equivalence classes induced by $\sim_L$ on Σ^* upper bounds the number of equivalence classes induced by $\sim_{\widetilde{\mathbf{H}}}$ on $\widetilde{\mathrm{P}}$.*

Proof. For every $\boldsymbol{p}_1, \boldsymbol{p}_2 \in \widetilde{\mathrm{P}}$, we show that that $\boldsymbol{p}_1 \sim_L \boldsymbol{p}_2 \implies \boldsymbol{p}_1 \sim_{\widetilde{\mathbf{H}}} \boldsymbol{p}_2$. Indeed $\boldsymbol{p}_1 \sim_L \boldsymbol{p}_2$ means that $\boldsymbol{p}_1^{-1}L \equiv \boldsymbol{p}_2^{-1}L$ and $\mathrm{supp}(\boldsymbol{p}_1^{-1}L) = \mathrm{supp}(\boldsymbol{p}_2^{-1}L)$. Following the definition of the relation $\equiv$ between weighted languages (cf. §3), we then have that:

$$\mathrm{supp}(\widetilde{\mathbf{H}}_{\boldsymbol{p}_1}) = \mathrm{supp}(\boldsymbol{p}_1^{-1}L) \cap \widetilde{\mathrm{S}} \qquad \text{since } \widetilde{\mathbf{H}}_{\boldsymbol{p}_1} : \widetilde{\mathrm{S}} \to \mathbb{K} \text{ and } \widetilde{\mathbf{H}}_{\boldsymbol{p}_1} = \boldsymbol{p}_1^{-1}L(\boldsymbol{s}), \forall \boldsymbol{s} \in \widetilde{\mathrm{S}} \tag{20a}$$

$$= \mathrm{supp}(\boldsymbol{p}_2^{-1}L) \cap \widetilde{\mathrm{S}} \qquad \text{since } \mathrm{supp}(\boldsymbol{p}_1^{-1}L) = \mathrm{supp}(\boldsymbol{p}_2^{-1}L) \tag{20b}$$

$$= \mathrm{supp}(\widetilde{\mathbf{H}}_{\boldsymbol{p}_2}) \tag{20c}$$

and further, $\forall \boldsymbol{s} \in \mathrm{supp}(\widetilde{\mathbf{H}}_{\boldsymbol{p}_1})$

$$\widetilde{\mathbf{H}}_{\boldsymbol{p}_1}(\boldsymbol{s}) = \boldsymbol{p}_1^{-1}L(\boldsymbol{s}) \tag{21a}$$

$$= k \otimes \boldsymbol{p}_2^{-1}L(\boldsymbol{s}) \text{ since } \boldsymbol{p}_1^{-1}L(\boldsymbol{s}) = k \otimes \boldsymbol{p}_2^{-1}L(\boldsymbol{s}) \text{ for some } k \in \mathbb{K}\backslash\{\mathbf{0}\}, \forall \boldsymbol{s} \in \mathrm{supp}(\boldsymbol{p}_1^{-1}L) \tag{21b}$$

$$= k \otimes \widetilde{\mathbf{H}}_{\boldsymbol{p}_2}(\boldsymbol{s}) \tag{21c}$$

For a fixed $k \in \mathbb{K}/\{\mathbf{0}\}$. Therefore $\widetilde{\mathbf{H}}_{\boldsymbol{p}_1} \equiv \widetilde{\mathbf{H}}_{\boldsymbol{p}_2}$ and $\boldsymbol{p}_1 \sim_{\widetilde{\mathbf{H}}} \boldsymbol{p}_2$. Since $\boldsymbol{p}_1 \sim_L \boldsymbol{p}_2 \implies \boldsymbol{p}_1 \sim_{\widetilde{\mathbf{H}}} \boldsymbol{p}_2$, it follows that $|\widetilde{\mathrm{P}}/\sim_{\widetilde{\mathbf{H}}}| \leqslant |\Sigma^*/\sim_L|$ ■

Theorem 3. *Let $\langle \mathbb{K}, \oplus, \otimes, \mathbf{0}, \mathbf{1} \rangle$ be a semifield and Σ an alphabet. Let $\mathbb{O}$ be an oracle for the deterministic regular language $L^\star : \Sigma^* \to \mathbb{K}$, whose minimal WDFSA has N states. Then, Alg. 1 returns a minimal WDFSA $\mathcal{H}$ generating $L^\star$ in time $\mathcal{O}\left(N^4 M^2 |\Sigma|\right)$, where M is the length of the longest counterexample provided by $\mathbb{O}$.*

Proof. We split the remainder of the proof into the proof of termination and the runtime analysis.

Termination. We will show that, at any iteration of the loop on Line 2 in Alg. 1, the algorithm either terminates or gets strictly closer to termination. Concretely, at each execution of the loop, three events can occur:

(1) The empirical Hankel matrix $\widetilde{\mathbf{H}}$ is not consistent and/or closed, resulting in the execution of the inner while loop (Alg. 1, Line 3).

(2) The oracle $\mathbb{O}$ replies positively to the equivalence query positively on Line 10, halting the execution.

(3) The oracle $\mathbb{O}$ replies with a counterexample.

Clearly, events *(2)* and *(3)* do not occur at the same time. By Lemmata 5 and 6, we know that *(1)* and *(3)* result an increase of the number of equivalence classes of $\widetilde{\mathrm{P}}$.

Assume that after some number of iterations of the outer while loop, the prefixes $\widetilde{\mathrm{P}}$ are partitioned into N equivalence classes and that the algorithm has not terminated yet. In the next iteration, events *(1)* and *(3)* would increase the number of equivalence classes of $\widetilde{\mathrm{P}}$ to at least $N+1$. Let $\mathcal{H}$ be the empirical Hankel automaton constructed during this iteration of the algorithm and let $L_{\mathcal{H}}$ be its language. By Thm. 2, we know that the $\mathcal{H}$ is consistent with the empirical Hankel matrix $\widetilde{\mathbf{H}}$, meaning that $\sim_{\widetilde{\mathbf{H}}}$ and $\sim_{L_{\mathcal{H}}}$ induce isomorphic equivalence classes. Now, by Lemma 7 we know that

$$N+1 \leqslant |\Sigma^*/\sim_{L_{\mathcal{H}}}| = |\widetilde{\mathrm{P}}/\sim_{\widetilde{\mathbf{H}}}| \leqslant |\Sigma^*/L^\star| = N. \tag{23}$$

Another execution therefore brings us to a contradiction with the Myhill–Nerode theorem (Thm. 1) since the number of states of the minimal automaton for $L^\star$ is N. To summarize, since event *(3)* results in a contradiction, event *(2)* must occur, meaning that the algorithm halts and return an automaton generating $L^\star$ after at most N iterations of the outer loop. The minimality of the returned automaton follows from Thm. 2.

Note that the same argument can be applied to show that the inner while loop (Alg. 1, Line 3) can be executed at most N times throughout the entire execution of $\mathbf{L}^*$, since by Lemma 5 at each iteration, the number of equivalence classes increases.

Runtime Analysis. First, we analyze the sizes of $\widetilde{\mathrm{P}}$ and $\widetilde{\mathrm{S}}$. The size of $\widetilde{\mathrm{P}}$ is upper bounded by $|\widetilde{\mathrm{P}}| \leqslant N + MN$: The empirical Hankel matrix can be found to be not closed at most N times—and in such case one string is added to $\widetilde{\mathrm{P}}$—and the oracle can reply with a counterexample at most N times—and in any such case, at most M strings are added to the matrix. The size of $\widetilde{\mathrm{S}}$ is at most N, since the empirical Hankel matrix can be found to be not consistent at most N times.

Now, we know that the outer while loop of Alg. 1 (Line 2) can be repeated at most N times, and similarly the inner while loop (Line 3) can be repeated at most N times during the entire execution of the outer loop. Of all the operations performed during the execution of the outer loop, the most expensive is MAKECLOSED, as it requires a double pass over $\widetilde{\mathrm{P}}$ to check if the table is consistent. Therefore the total runtime is:

$$\mathcal{O}\left(\left((N+MN)^2\,|\Sigma|N\right)N\right) = \mathcal{O}\left(N^4M^2|\Sigma|\right) \tag{24}$$

Where $\mathcal{O}\left((N+MN)^2\,|\Sigma|N\right)$ is the cost of MAKECLOSED, and N is the maximum number of calls to MAKECLOSED.

■