

COALGEBRAS FOR BISIMULATION OF WEIGHTED AUTOMATA OVER SEMIRINGS

PURANDAR BHADURI 

Indian Institute of Technology Guwahati, Guwahati 781039, India
e-mail address: pbhaduri@iitg.ac.in

ABSTRACT. Weighted automata are a generalization of nondeterministic automata that associate a weight drawn from a semiring K with every transition and every state. Their behaviours can be formalized either as weighted language equivalence or weighted bisimulation. In this paper we explore the properties of weighted automata in the framework of coalgebras over (i) the category \mathbf{SMod} of semimodules over a semiring K and K -linear maps, and (ii) the category \mathbf{Set} of sets and maps. We show that the behavioural equivalences defined by the corresponding final coalgebras in these two cases characterize weighted language equivalence and weighted bisimulation, respectively. These results extend earlier work by Bonchi *et al.* using the category \mathbf{Vect} of vector spaces and linear maps as the underlying model for weighted automata with weights drawn from a field K . The key step in our work is generalizing the notions of linear relation and linear bisimulation of Boreale from vector spaces to semimodules using the concept of the kernel of a K -linear map in the sense of universal algebra. We also provide an abstract procedure for forward partition refinement for computing weighted language equivalence. Since for weighted automata defined over semirings the problem is undecidable in general, it is guaranteed to halt only in special cases. We provide sufficient conditions for the termination of our procedure. Although the results are similar to those of Bonchi *et al.*, many of our proofs are new, especially those about the coalgebra in \mathbf{SMod} characterizing weighted language equivalence.

1. INTRODUCTION

Bisimulation was introduced by Park and Milner [Par81, Mil89] for characterizing the equivalence of two processes specified by transition systems or process algebra terms. Over the years the notion has had an enduring impact on the study of the behaviour of systems, with ramifications in concurrency, automata theory, modal logic, coalgebras, games and formal verification. The basic notion of bisimulation for discrete systems has been extended to probabilistic, quantitative and even continuous systems.

Coalgebras are a category theoretic concept that enable a study of state transition systems and their behaviours in a unified setting [Rut00, Jac16]. The idea behind a coalgebraic theory of systems is the following. Given an endofunctor $\mathcal{F} : \mathcal{C} \rightarrow \mathcal{C}$ on a concrete category \mathcal{C} , a coalgebra $f : X \rightarrow \mathcal{F}X$ represents a transition system on the set of states X possibly with additional structure. For example, in \mathbf{Set} , the category of sets and maps, a coalgebra for the endofunctor $\mathcal{F}_d X = 2 \times X^A$ represents a deterministic automaton

Key words and phrases: Coalgebra; Bisimulation; Weighted Automata; Semirings; Semimodules.

with states X and alphabet A . This is because a coalgebra for \mathcal{F}_d can be seen as a map $\langle o, \delta \rangle : X \rightarrow 2 \times X^A$ where $o : X \rightarrow 2$ is the output map indicating whether a given state is accepting and $\delta : X \rightarrow X^A$ is the transition map where $\delta(x)(a)$ is the next state on reading the letter a in state x . Here $2 = \{0, 1\}$ is the set of truth values. Similarly a coalgebra for the endofunctor $\mathcal{F}_n(X) = 2 \times (\mathcal{P}_\omega X)^A$ on \mathbf{Set} , where $\mathcal{P}_\omega X$ is the finite powerset of X , represents a nondeterministic automaton.

Homomorphisms between coalgebras can be seen as behaviour-preserving maps. A final coalgebra for the functor $\mathcal{F} : \mathcal{C} \rightarrow \mathcal{C}$, if it exists, is the universe of all possible \mathcal{F} -behaviours. The unique arrow from any coalgebra to the final coalgebra maps a state to its behaviour. The final coalgebra for the functor \mathcal{F} naturally induces a notion of \mathcal{F} -behavioural equivalence, denoted by $\approx_{\mathcal{F}}$. Two states are behaviourally equivalent if they are mapped to the same element in the final coalgebra by the unique arrow. For example, for deterministic automata, the behaviour of a state is the language accepted by the automaton starting from that state; two states x and y satisfy $x \approx_{\mathcal{F}_d}$ if and only if they are language equivalent. For nondeterministic automata, two states are \mathcal{F}_n -behaviourally equivalent for the functor \mathcal{F}_n on \mathbf{Set} defined above if and only if they are bisimilar. However, if we consider the category \mathbf{Rel} of sets and relations, the behavioural equivalence for a suitable functor coincides with language equivalence [HJS07] of nondeterministic automata. Thus the notion of behavioural equivalence for the same computational object is relative to the underlying category and the associated endofunctor.

In this paper we focus on bisimulation for weighted automata [Buc08]. Weighted automata were introduced by Shützenburger [Sch61] and have found renewed interest in the last two decades [DKV09] as they arise in various contexts where quantitative modelling is involved. Intuitively, weighted automata generalize the notion of nondeterministic automata where each state and each transition has an associated weight valued in some semiring. Such a weight could represent a cost, reward or probability, or any other measure of interest. Like nondeterministic automata, the behaviour of weighted automata have two different characterizations, in terms of weighted language equivalence and weighted bisimulation.

In [BBB⁺12] Bonchi *et al.* have given a comprehensive account of both weighted-language equivalence and bisimilarity of weighted automata in terms of coalgebras of an endofunctor \mathcal{L} on \mathbf{Vect} (the category of vector spaces and linear maps) and an endofunctor \mathcal{W} on \mathbf{Set} , respectively. The characterization of weighted language equivalence, is based on an elegant notion of linear bisimulation given by Boreale [Bor09]. The idea here is that a weighted automaton is a vector space (of states) over a field K , with two linear maps: the output map from states to observations valued in K and an A -indexed family of transition maps from states to states, where A is the set of actions. Then, a *linear relation* over states is a set of pairs whose differences form a subspace of the vector space and a linear bisimulation is a linear relation that preserves the output map and (the corresponding subspace) is invariant under the transition maps. Boreale showed that this notion of linear bisimulation coincides with weighted language equivalence. Bonchi *et al.* [BBB⁺12] gave a coalgebraic formulation of these results using an endofunctor \mathcal{L} on \mathbf{Vect} .

Weighted automata were originally defined over semirings and not fields. Therefore, it is desirable that the results of Boreale [Bor09] and Bonchi *et al.* [BBB⁺12] related to linear bisimulation and language equivalence be generalized to semimodules over semirings. This is exactly what we accomplish in this paper, starting with the assumption that K is a semiring. Bonchi *et al.* had identified in their work the results which could be extended to semirings in a straightforward way, and the ones which could not, because the latter involve

Key Component	Boreale [Bor09] and Bonchi <i>et al.</i> [BBB ⁺ 12]	Our work
Domain of weights	Field K	Semiring K
State space	Vector Space V over K	Semimodule V over K
Foundation for K -linear relations	Subspace U of V	Linear map f from V
K -Linear Relation R	uRv iff $u - v \in U$	uRv iff $R = \ker(f)$
Linear extension R^ℓ of relation R	$uR^\ell v$ iff $u - v \in \text{span}(\ker(R))$	$R^\ell =$ smallest congruence containing R

TABLE 1. Correspondence between previous work and ours

the minus operator in fields. These are the the results of Boreale on linear bisimulation and their coalgebraic formulation which make essential use of the properties of fields and vector spaces. We show that there is an elegant characterization of these concepts in the semiring-semimodule setting, by leveraging the concept of *kernel* of a K -linear map in the universal algebraic sense, *i.e.*, $\ker(f) = \{(u, v) \mid f(u) = f(v)\}$ and its universal properties. We show that in the special case that K is a field, our results are identical to those of [Bor09] and [BBB⁺12].

Our coalgebraic characterization of linear bisimulation proceeds as follows. We define a K -linear relation on a semimodule V over the semiring K as the kernel of a K -linear map $f : V \rightarrow W$ with domain V . Clearly, this is an equivalence relation which is a congruence. In fact any relation R on V can be turned into a K -linear relation by considering the smallest congruence R^ℓ containing R and taking f to be the canonical map $V \rightarrow V/R^\ell$ that sends each element to its congruence class. A K -linear bisimulation on a weighted automaton is then defined as a K -linear relation which preserves the output map of the weighted automaton and is invariant under the transition map as in [BBB⁺12]. We show that this approach leads to all the results of [BBB⁺12] regarding the functor \mathcal{L} and weighted language equivalence although the proofs are new as we cannot assume the field properties of K , in particular the existence of the additive inverse. Table 1 is a summary of the correspondence between the key concepts in [Bor09, BBB⁺12] and the present work.

We briefly summarize our work highlighting the contributions. After introducing the basic definitions and notation we recall the notion of a K -weighted automaton (Section 2.4) and K -weighted bisimulation (Section 2.5). We use the definition of Bonchi *et al.* [BBB⁺12] but assume that the underlying set of weights is a semiring K . We use the terms K -weighted automaton and K -weighted bisimulation to distinguish our setting from that of [BBB⁺12]. The fact that K -weighted automata are \mathcal{W} -coalgebras for an endofunctor \mathcal{W} on Set follows, and so does the correspondence between K -weighted bisimulations and kernels of \mathcal{W} -homomorphisms (Section 2.6). This result uses our definition of kernels, so there is some novelty here. The proof that K -weighted bisimilarity \sim_w coincides with \mathcal{W} -behavioural equivalence $\approx_{\mathcal{W}}$ is the one in [BBB⁺12] presented in a different way.

The main focus of our work is on the coalgebras of the functor \mathcal{L} on SMod , the category of semimodules over the semiring K . We start by defining the functor \mathcal{L} and proceed to define the notions of K -linear automata as coalgebras for the functor \mathcal{L} . We define their behaviour in terms of weighted languages and present the final \mathcal{L} -coalgebra (Section 3.1).

This part is more or less similar to the treatment in [BBB⁺12] as it does not rely on any vector space property not enjoyed by a semimodule. The point of departure in our work from that of [BBB⁺12] is the definition of K -linear relations and K -linear bisimulations (Section 3.2) based on the kernel (in the universal algebraic sense) of K -linear maps as mentioned above. We prove the correspondence between K -linear bisimulations and kernels of \mathcal{L} -homomorphisms and establish the coincidence of the behavioural equivalence $\approx_{\mathcal{L}}$ and weighted language equivalence \sim_l (Section 3.2). Although these results mirror those of [BBB⁺12], the proofs, other than the one for the coincidence of $\approx_{\mathcal{L}}$ and \sim_l , are new and more general – they truly extend the results from the vector space setting to the semimodule setting in a non-trivial way. This is where the main contribution of this paper lies.

One desideratum is the existence of a partition refinement algorithm for computing the weighted language equivalence \sim_l for finitely generated semimodules. Unfortunately, the results from Bonchi *et al.* [BBB⁺12] do not carry over to our semimodule setting. First, even finitely generated semimodules do not have the descending chain property: they can have an infinite descending chain of submodules. Also, weighted language equivalence is known to be undecidable for finite-state weighted automata over the tropical semiring [Kro94, ABK20]. Instead, we offer an abstract procedure via the final sequence whose limit exists in \mathbf{SMod} . The limit of the final sequence is shown to be isomorphic to the final coalgebra for the functor \mathcal{L} , essentially following the reasoning in [BBB⁺12]. It is well-known that if the final sequence stabilizes at an object that object is isomorphic to the final coalgebra. For any K -linear weighted automaton there is a cone to the final sequence such that the kernel of the arrows in the cone constitute a sequence of K -linear relations. We show that these K -linear relations converge to \mathcal{L} -bisimilarity, *i.e.*, language equivalence, in a finite number of steps in case the state space of the automaton is a finitely generated *Artinian* semimodule (those satisfying the descending chain property) as well as when a weaker condition holds (Section 4.1). Unfortunately, this does not give us an algorithm for computing the bisimilarity relation in general, as this involves solving K -linear equations. Although this is possible for specific semirings, such as \mathbb{R} , \mathbb{Z} and \mathbb{N} , the problem is known to be undecidable for certain semirings [Nar96]. We conclude the paper by comparing our work with the only existing coalgebraic formulation of partition refinement for weighted automata over semirings, that of König and Küpper [KK18] (Section 4.2).

2. K -WEIGHTED AUTOMATA AND K -WEIGHTED BISIMULATION

This section is a mild generalization of the coalgebraic characterization of weighted automata and weighted bisimulation in Bonchi *et al.* [BBB⁺12] from the setting of vector spaces to semimodules. We start by fixing the notation and recalling the basic notions of semimodules and coalgebras. Then we show how weighted automata over a semiring K can be seen as a coalgebra over a functor $\mathcal{W} : \mathbf{Set} \rightarrow \mathbf{Set}$ and characterize weighted bisimilarity \sim_w as the behavioural equivalence $\approx_{\mathcal{W}}$ for the functor \mathcal{W} . The results and proofs are essentially identical to those in [BBB⁺12], as they do not use the additional properties satisfied by vector spaces. We include them for the sake of completeness. The only exception to this is the correspondence between K -weighted bisimulation and kernels of \mathcal{W} -homomorphisms in Section 2.6, which uses our definition of kernel and is therefore new.

2.1. Notation and Preliminaries. We denote sets by capital letters X, Y, Z, \dots and maps (*i.e.*, functions) by small letters f, g, h, \dots . We denote the identity map on a set X by id_X . Given two maps $f : X \rightarrow Y$ and $g : Y \rightarrow Z$, their composition is denoted $g \circ f : X \rightarrow Z$. The product of two sets is denoted $X \times Y$ with the projections $\pi_1 : X \times Y \rightarrow X$ and $\pi_2 : X \times Y \rightarrow Y$. The product of two maps $f_1 : X_1 \rightarrow Y_1$ and $f_2 : X_2 \rightarrow Y_2$ is $f_1 \times f_2 : X_1 \times X_2 \rightarrow Y_1 \times Y_2$ defined by $(f_1 \times f_2)(x_1, x_2) = (f_1(x_1), f_2(x_2))$. Given maps $f : X \rightarrow Y$ and $g : X \rightarrow Z$, $\langle f, g \rangle : X \rightarrow Y \times Z$ is the pairing map defined by $\langle f, g \rangle(x) = (f(x), g(x))$. We use \mathbb{N} for the set of natural numbers, \mathbb{Z} for the integers, \mathbb{R} for the set of reals and \mathbb{R}_+ for the set of non-negative reals.

The disjoint union of sets X_1 and X_2 is $X_1 + X_2$ with the injections $\iota_1 : X_1 \rightarrow X_1 + X_2$ and $\iota_2 : X_2 \rightarrow X_1 + X_2$. The sum of two maps $f_1 : X_1 \rightarrow Y_1$ and $f_2 : X_2 \rightarrow Y_2$ is $f_1 + f_2 : X_1 + X_2 \rightarrow Y_1 + Y_2$ defined by $(f_1 + f_2)(\iota_i(z)) = \iota_i(f_i(z))$ for $i = 1, 2$. We denote the set of maps from X to Y by Y^X . For a map $f : X_1 \rightarrow X_2$, the map $f^Y : X_1^Y \rightarrow X_2^Y$ is defined by $f^Y(g) = f \circ g$. This defines a functor $(_)^Y : \mathbf{Set} \rightarrow \mathbf{Set}$, called the *exponential functor*. The set of all finite subsets of X is denoted by $\mathcal{P}_\omega(X)$. For a finite set of letters A , A^* denotes the set of all finite words over A . We denote by ϵ the empty word, and by $w_1 w_2$ the concatenation of words $w_1, w_2 \in A^*$. The length of a word w is denoted by $|w|$.

If R is an equivalence relation on a set X we denote the set of equivalence classes of R by X/R and the equivalence class of an element $x \in X$ by $[x]_R$. The subscript is often dropped if the relation R is clear from the context. For a map $f : X \rightarrow Y$, the *kernel* of f is the equivalence relation $\ker(f) = \{(x_1, x_2) \mid f(x_1) = f(x_2)\}$. Further, f has a *unique* (up to isomorphism) factorization through X/R , $f = \mu_f \circ \varepsilon_f$ into a surjection $\varepsilon_f : X \rightarrow X/\ker(f)$ followed by an injection $\mu_f : X/\ker(f) \rightarrow Y$ defined by $\varepsilon_f(x) = [x]$ and $\mu_f([x]) = f(x)$.

2.2. Semirings and Semimodules. Semirings and semimodules generalize the notions of fields and vector spaces, respectively. A *semiring* $(K, +, \cdot, 0, 1)$ consists of a commutative monoid $(K, +, 0)$ and a monoid $(K, \cdot, 1)$ such that the product distributes over the sum on both sides and $k \cdot 0 = 0 \cdot k = 0$ for all $k \in K$. We will refer to the semiring K when the operations are understood. A *semiring module*, or simply a *semimodule* V over a semiring K is a commutative monoid $(V, +, 0)$ together with an action $\cdot : K \times V \rightarrow V$ such that for all $k, k_1, k_2 \in K$ and $v, v_1, v_2 \in V$:

$$(k_1 + k_2) \cdot v = k_1 \cdot v + k_2 \cdot v \quad (k_1 \cdot k_2) \cdot v = k_1 \cdot (k_2 \cdot v) \quad (2.1)$$

$$k \cdot (v_1 + v_2) = k \cdot v_1 + k \cdot v_2 \quad 1 \cdot v = v \quad (2.2)$$

$$0 \cdot v = 0 \quad (2.3)$$

A K -linear map between two semimodules V and W (over the semiring K) is a map $f : V \rightarrow W$ satisfying $f(v_1 + v_2) = f(v_1) + f(v_2)$ and $f(k \cdot v) = k \cdot f(v)$ for all $v, v_1, v_2 \in V$ and $k \in K$.

An equivalence relation R on a semimodule V is called a *congruence* if all the semimodule operations are compatible with R , *i.e.*, for all $k \in K$ and $u_1, u_2, v_1, v_2 \in V$, $u_1 R v_1$ and $u_2 R v_2$ imply $k \cdot u_1 R k \cdot u_2$ and $(u_1 + u_2) R (v_1 + v_2)$. It is easy to verify that the quotient V/R has a semimodule structure given by the operations $k \cdot [u] = [ku]$ and $[u] + [v] = [u + v]$. The fact that these are well-defined, *i.e.*, independent of the choice of representative of an equivalence class, is a consequence of R being a congruence.

For a K -linear map $f : V \rightarrow W$, the *kernel* of f is the equivalence relation $\ker(f) = \{(v_1, v_2) \mid f(v_1) = f(v_2)\}$. It can be shown that $\ker(f)$ is a congruence. It follows that

$V/\ker(f)$ is a semimodule. Further, every K -linear map f has a *unique* (up to isomorphism) factorization into two K -linear maps, a surjection $\varepsilon_f : V \rightarrow V/\ker(f)$ followed by an injection $\mu_f : V/\ker(f) \rightarrow W$ defined by $\varepsilon_f(u) = [u]$ and $\mu_f([u]) = f(u)$. In the following, when we refer to the kernel of a K -linear map f , we mean the kernel of f as defined above and *not* the set $\{u \mid f(u) = 0\}$. This usage is common in universal algebra [BS81].

Let V be a K -semimodule. A nonempty subset U of V is called a *subsemimodule* of V if U is closed under addition and scalar multiplication. The intersection $\bigcap_{i \in I} U_i$ of any family $\{U_i\}_{i \in I}$ of subsemimodules of V is clearly a subsemimodule of V . If U is any nonempty subset of V then the intersection of all subsemimodules of V containing U is called the subsemimodule *generated by U* , and is denoted by $\text{span}(U)$. It is easy to check that $\text{span}(U)$ is the set of finite linear combinations of elements of U

$$\text{span}(U) = \left\{ \sum_{i=1}^n k_i u_i \mid n \in \mathbb{N}, u_i \in U \text{ for } 1 \leq i \leq n \right\}.$$

If $V = \text{span}(U)$ then U is called a *generating set for V* . If V has a finite generating set it is called *finitely generated*. One can verify that given a K -linear map $f : V \rightarrow W$, $\ker(f)$ is a subsemimodule of $V \times V$.

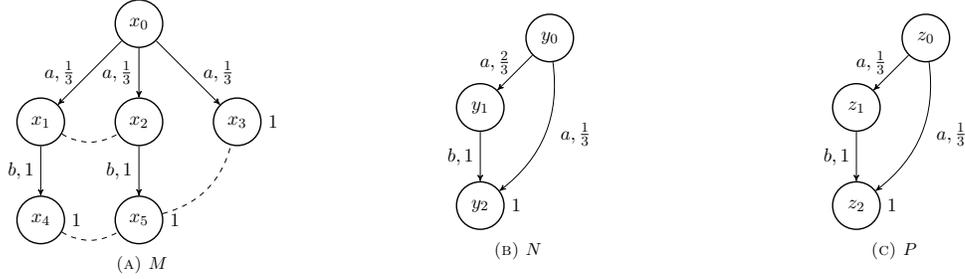
Semimodules over a semiring K and K -linear maps form the category \mathbf{SMod} . \mathbf{SMod} has products $V \times W$, and the set of all maps V^A from a set A to a semimodule V has a natural semimodule structure defined pointwise: for $f, g \in V^A$, $f + g \in V^A$ is defined by $(f + g)(a) = f(a) + g(a)$ and $(k.f)(a) = k.f(a)$. For a set X , the set of all maps $f : X \rightarrow K$ with *finite support*, i.e., the set $\{x \mid f(x) \neq 0\}$ is finite, is denoted $K(X)$. Its elements are conveniently represented as formal sums $\sum_{x \in X} k_x \cdot x$ by writing $k_x = f(x)$. In other words, $K(X) = \text{span}(X)$, where we identify an element $x \in X$ with the map $\eta_X(x) = \delta_x : X \rightarrow K$ where δ_x is the Kronecker delta that maps x to 1 and everything else to 0. Note that only a finite number of k_x are non-zero in the formal sum. $K(X)$ is called the *free semimodule generated by X* over K and satisfies the following universal property. Given any map $f : X \rightarrow V$ from a set X to a semimodule V there exists a unique K -linear map $f^\# : K(X) \rightarrow V$ which extends f . The map $f^\#$ is just the *linear extension* of f i.e., $f^\#(\sum_{x \in X} k_x \cdot x) = \sum_{x \in X} k_x \cdot f(x)$. This is shown in the commuting diagram below where $\eta_X : X \rightarrow K(X)$ is the inclusion map.

$$\begin{array}{ccc} X & \xleftarrow{\eta_X} & K(X) \\ & \searrow f & \downarrow f^\# \\ & & V \end{array} \quad (2.4)$$

2.3. Coalgebras. Given an endofunctor $\mathcal{F} : \mathcal{C} \rightarrow \mathcal{C}$ on a category \mathcal{C} , an \mathcal{F} -coalgebra is a \mathcal{C} -object X together with a \mathcal{C} -arrow $f : X \rightarrow \mathcal{F}X$. In many categories the pair (X, f) represents a transition system such as a deterministic, nondeterministic or probabilistic automaton [Rut00]. A *morphism of \mathcal{F} -coalgebras*, or an \mathcal{F} -homomorphism, between coalgebras (X, f) and (Y, g) is a \mathcal{C} -arrow $h : X \rightarrow Y$ such that the following diagram commutes.

$$\begin{array}{ccc} X & \xrightarrow{h} & Y \\ f \downarrow & & \downarrow g \\ \mathcal{F}X & \xrightarrow{\mathcal{F}h} & \mathcal{F}Y \end{array} \quad (2.5)$$

FIGURE 1. Bisimulation quotient of weighted automata over different semirings



An \mathcal{F} -coalgebra (Y, g) is called *final* if there is a *unique* \mathcal{F} -homomorphism $\llbracket - \rrbracket_X^{\mathcal{F}}$ from any \mathcal{F} -coalgebra (X, f) to (Y, g) . The final coalgebra represents the universe of all possible \mathcal{F} -behaviours and the arrow $\llbracket - \rrbracket_X^{\mathcal{F}}$ maps every element (or *state*) of a coalgebra X to its behaviour [Rut00]. Two states $x_1, x_2 \in X$ are said to be \mathcal{F} -*behaviourally equivalent*, denoted $\approx_{\mathcal{F}}$, iff $\llbracket x_1 \rrbracket_X^{\mathcal{F}} = \llbracket x_2 \rrbracket_X^{\mathcal{F}}$.

2.4. K -Weighted Automata. A weighted automaton [DKV09] is a generalization of a nondeterministic finite automaton where each transition and each state is assigned a weight in a semiring K . We follow the definition in [BBB⁺12]. Formally, for a semiring K , a K -*weighted automaton* (K -WA in short) with input alphabet A is a pair $(X, \langle o, t \rangle)$ where X is a set of states, $o : X \rightarrow K$ is an output map and $t : X \rightarrow (K^X)^A$ is the transition map. The state x can make a transition to state y on input $a \in A$ with weight $k \in K$ iff $t(x)(a)(y) = k$. A weight of zero means there is no transition. Note that the set of states X may be infinite in general. We often use X to refer to the K -WA $(X, \langle o, t \rangle)$ when the output and transition maps are clear from the context.

2.5. K -Weighted Bisimulation. The notion of weighted bisimulation [Buc08] generalizes the well-known notion from ordinary transition systems [Mil89] to finite-state weighted automata. We follow the definition in [BBB⁺12] which applies to infinite state spaces, but with *finite branching*: for all $x \in X, a \in A, t(x)(a)(y) \neq 0$ for only finitely many y . In the following we assume the finite branching condition for weighted automata without stating it explicitly.

Definition 2.1. Let $M = (X, \langle o, t \rangle)$ be a K -weighted automaton. An equivalence relation R on X is a K -*weighted bisimulation* on M if the following two conditions hold for all $x_1, x_2 \in X$:

- (1) $o(x_1) = o(x_2)$, and
- (2) for all $a \in A$ and $y \in X$, $\sum_{y' \in [y]_R} t(x_1)(a)(y') = \sum_{y' \in [y]_R} t(x_2)(a)(y')$.

The largest K -weighted bisimulation relation on M is called K -*weighted bisimilarity*, and is denoted by \sim_w . It exists because an arbitrary union of K -weighted bisimulation relations is a K -weighted bisimulation.

Example 2.2. This example illustrates bisimulation between weighted automata for different semirings, and is adapted from [Buc08]. Figure 1(A) shows a K -weighted automaton $M = (X_{\mathcal{A}}, \langle o_{\mathcal{A}}, t_{\mathcal{A}} \rangle)$ over the alphabet $A = \{a, b\}$ and the semiring $K = (\mathbb{R}_+, +, \cdot, 0, 1)$. It has the set of states $X_M = \{x_0, x_1, x_2, x_3, x_4, x_5\}$, the output map (shown in the figure as labels of the states when they are non-zero) $o_{\mathcal{A}}$ given by $\{x_0 \mapsto 0, x_1 \mapsto 0, x_2 \mapsto 0, x_3 \mapsto 1, x_4 \mapsto 1, x_5 \mapsto 1\}$ and the transition map t shown in the figure by the edges with their labels. For example, $t(x_0)(a)(x_1) = \frac{1}{3}$. All missing transitions have weight zero. Let R be the smallest equivalence relation on X_M containing the pairs $\{(x_1, x_2), (x_3, x_5), (x_4, x_5)\}$. R is shown in the figure as the dashed lines, which join states in the same equivalence class. It is easily checked that R is a bisimulation relation on M , and is in fact the largest such. The automaton N in Figure 1(B) is the one obtained from M by quotienting by R , where the quotient operation is defined in Section 2.6. Here the set of states is $\{y_0, y_1, y_2\}$ and the output map is given by $\{y_0 \mapsto 0, y_1 \mapsto 0, y_2 \mapsto 1\}$.

Now consider a different semiring $K' = ([0, 1], \max, \cdot, 0, 1)$ and consider M , the automaton on the left, as a K' -weighted automaton. The relation R above is again the largest bisimulation on M , but the quotient automaton P has different weights on transitions, and is shown in Figure 1(C). Here the set of states is $\{z_0, z_1, z_2\}$ and the output map is given by $\{z_0 \mapsto 0, z_1 \mapsto 0, z_2 \mapsto 1\}$. Notice the difference in the weights on the edges (y_0, y_1) and (z_0, z_1) . The former is obtained by addition whereas the latter by the max operation on the pair $(\frac{1}{3}, \frac{1}{3})$.

2.6. Coalgebraic Model for K -Weighted Automata and K -Weighted Bisimulation. Following [BBB⁺12], we now exhibit a functor $\mathcal{W} : \mathbf{Set} \rightarrow \mathbf{Set}$ such that a \mathcal{W} -coalgebra is just a K -weighted automaton and $\approx_{\mathcal{W}}$ is exactly K -weighted bisimilarity.

Definition 2.3. For a semiring K the *valuation functor* $K(_) : \mathbf{Set} \rightarrow \mathbf{Set}$ is defined by the mappings $X \mapsto K(X)$ on sets X and $X \xrightarrow{h} Y \mapsto K(X) \xrightarrow{K(h)} K(Y)$ on maps, where $K(h)$ sends $\sum_{x \in X} k_x x \in K(X)$ to $\sum_{y \in Y} k_y y \in K(Y)$ with $k_y = \sum_{x \in h^{-1}(y)} k_x$.

Recall that for a given set C , the functor $C \times _ : \mathbf{Set} \rightarrow \mathbf{Set}$ sends a set X to $C \times X$ and a map $f : X \rightarrow Y$ to the map $id_C \times f$. The functor $\mathcal{W} : \mathbf{Set} \rightarrow \mathbf{Set}$ is defined by $\mathcal{W} = K \times (K(_))^A$ where $(_)^A$ is the exponential functor defined earlier. Thus a \mathcal{W} -coalgebra $f : X \rightarrow \mathcal{W}X$ on a set X constitutes a pair of maps $\langle o, t \rangle$ where $o : X \rightarrow K$ and $t : X \rightarrow K(X)^A$. In other words, a \mathcal{W} -coalgebra is identical to a K -weighted automaton $(X, \langle o, t \rangle)$ and vice versa under the assumption of finite branching, since $K(X)$ is the set of maps $G : X \rightarrow K$ with finite support, which means t satisfies the finite branching property.

It is shown in Bonchi *et al.* [BBB⁺12] that the functor \mathcal{W} , being *bounded*, has a final coalgebra (Ω, ω) . Moreover, the behavioural equivalence $\approx_{\mathcal{W}}$ coincides with K -weighted bisimilarity \sim_w . It is also shown that K -weighted bisimilarity is strictly included in weighted language inclusion. For completeness we recall the proof by Bonchi *et al.* of the coincidence of the two relations $\approx_{\mathcal{W}}$ and \sim_w .

Recall that a map $h : X \rightarrow Y$ is a \mathcal{W} -homomorphism between weighted automata, *i.e.*, \mathcal{W} -coalgebras, $(X, \langle o_X, t_X \rangle)$ and $(Y, \langle o_Y, t_Y \rangle)$ when the following diagram commutes.

$$\begin{array}{ccc} X & \xrightarrow{h} & Y \\ \langle o_X, t_X \rangle \downarrow & & \downarrow \langle o_Y, t_Y \rangle \\ K \times K(X)^A & \xrightarrow{id_K \times K(h)^A} & K \times K(Y)^A \end{array} \quad (2.6)$$

In words, for all $x \in X$, $y \in Y$, $a \in A$

$$o_X(x) = o_Y(h(x)) \quad \text{and} \quad \sum_{x' \in h^{-1}(y)} t_X(x)(a)(x') = t_Y(h(x)(a)(y)).$$

For any \mathcal{W} -homomorphism $h : (X, \langle o_X, t_X \rangle) \rightarrow (Y, \langle o_Y, t_Y \rangle)$, the equivalence relation $\ker(h)$ is a weighted bisimulation since $h(x_1) = h(x_2)$ implies

$$o_X(x_1) = o_Y(h(x_1)) = o_Y(h(x_2)) = o_X(x_2)$$

and for all $a \in A$, for all $y \in Y$

$$\sum_{x'' \in h^{-1}(y)} t_X(x_1)(a)(x'') = t_Y(h(x_1))(a)(y) = t_Y(h(x_2))(a)(y) = \sum_{x'' \in h^{-1}(y)} t_X(x_2)(a)(x'')$$

which in turn implies that for all $x' \in X$

$$\sum_{(x', x'') \in \ker(h)} t_X(x_1)(a)(x'') = \sum_{(x', x'') \in \ker(h)} t_X(x_2)(a)(x'').$$

Conversely, every K -weighted bisimulation R on $(X, \langle o_X, t_X \rangle)$ induces a coalgebra structure $(X/R, \langle o_{X/R}, t_{X/R} \rangle)$ on the quotient set X/R where $o_{X/R} : X/R \rightarrow K$ and $t_{X/R} : X/R \rightarrow (X/R)^A$ are defined by

$$o_{X/R}[x] = o_X(x) \quad \text{and} \quad t_{X/R}[x_1](a)([x_2]) = \sum_{x' \in [x_2]} t_X(x_1)(a)(x').$$

As R is a K -weighted bisimulation, both $o_{X/R} : X/R \rightarrow K$ and $t_{X/R} : X/R \rightarrow (X/R)^A$ are well-defined, *i.e.*, independent of the choice of representative of an equivalence class. Now, the map $\varepsilon_R : X \rightarrow X/R$ which sends x to its equivalence class $[x]_R$ is a \mathcal{W} -homomorphism. Therefore we have the following commuting diagram, where the dashed arrows constitute the unique \mathcal{W} -homomorphisms to the final coalgebra (Ω, ω) .

$$\begin{array}{ccccc} & & \mathbb{[-]}_X^{\mathcal{W}} & & \\ & & \text{---} & & \\ & & \text{---} & & \\ X & \xrightarrow{\varepsilon_R} & X/R & \text{---} & \Omega \\ \langle o_X, t_X \rangle \downarrow & & \downarrow \langle o_{X/R}, t_{X/R} \rangle & & \downarrow \omega \\ \mathcal{W}(X) & \xrightarrow{\mathcal{W}(\varepsilon_R)} & \mathcal{W}(X/R) & \text{---} & \mathcal{W}(\Omega) \\ & & \text{---} & & \\ & & \mathcal{W}(\mathbb{[-]}_X^{\mathcal{W}}) & & \end{array} \quad (2.7)$$

Theorem 2.4. *Let $(X, \langle o, t \rangle)$ be a weighted automaton. Then for $x_1, x_2 \in X$, $x_1 \sim_w x_2$ iff $x_1 \approx_{\mathcal{W}} x_2$.*

Proof. The equality of the two relations \sim_w and $\approx_{\mathcal{W}}$ follows by diagram chasing. By definition, $\approx_{\mathcal{W}} = \ker(\llbracket _ \rrbracket_X^{\mathcal{W}})$ and this is a K -weighted bisimulation since $\llbracket _ \rrbracket_X^{\mathcal{W}}$ is a \mathcal{W} -homomorphism as witnessed by the curved and dashed arrows in the diagram, which implies $\approx_{\mathcal{W}} \subseteq \sim_w$. In the other direction,

$$\begin{aligned}
& x \sim_w x_2 \\
& \Rightarrow (x_1, x_2) \in R \text{ for a } K\text{-weighted bisimulation } R \\
& \Rightarrow \varepsilon_R(x_1) = \varepsilon_R(x_2), \text{ since } [x_1]_R = [x_2]_R \\
& \Rightarrow \llbracket \varepsilon_R(x_1) \rrbracket_{X/R}^{\mathcal{W}} = \llbracket \varepsilon_R(x_2) \rrbracket_{X/R}^{\mathcal{W}} \\
& \Rightarrow \llbracket x_1 \rrbracket_X^{\mathcal{W}} = \llbracket x_2 \rrbracket_X^{\mathcal{W}} \text{ from the diagram above} \\
& \Rightarrow x_1 \approx_{\mathcal{W}} x_2, \text{ by definition of } \approx_{\mathcal{W}}. \quad \square
\end{aligned}$$

We now define K -weighted language equivalence \sim_l for K -weighted automata and show that K -weighted bisimilarity is a refinement of K -weighted language equivalence. The proof is from [BBB⁺12].

A K -weighted language over an alphabet A and semiring K is a map $\sigma : A^* \rightarrow K$ that assigns to each word $w \in A^*$ a weight in K . For a K -WA $(X, \langle o, t \rangle)$ the map $\sigma_l : X \rightarrow K^{A^*}$ assigns to each state $x \in X$ the K -weighted language recognized by x and is defined by induction on w as follows.

$$\sigma_l(x)(w) = \begin{cases} o(x) & \text{if } w = \epsilon \\ \sum_{x' \in X} (t(x)(a)(x')) \cdot \sigma_l(x')(w') & \text{if } w = aw' \end{cases}$$

Two states $x_1, x_2 \in X$ are said to be weighted language equivalent, denoted $x_1 \sim_l x_2$, if $\sigma_l(x_1)(w) = \sigma_l(x_2)(w)$ for all $w \in A^*$.

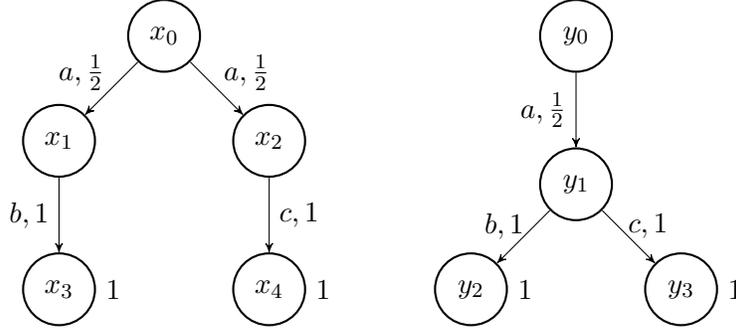
Proposition 2.5. *For K -weighted automata, $\sim_w \subseteq \sim_l$.*

Proof. We prove by induction on the length of w that if R is a K -weighted bisimulation on X then for all $x_1, x_2 \in X$ and all $w \in A^*$, $(x_1, x_2) \in R$ implies $\sigma_l(x_1)(w) = \sigma_l(x_2)(w)$. For the base case $w = \epsilon$, we have $\sigma_l(x_1)(w) = o(x_1)$ and $\sigma_l(x_2)(w) = o(x_2)$ and $o(x_1) = o(x_2)$ since R is a K -weighted bisimulation. For the inductive case, if $w = aw'$ then

$$\begin{aligned}
\sigma_l(x_1)(w) &= \sum_{x' \in X} (t(x_1)(a)(x')) \cdot \sigma_l(x')(w') \\
&= \sum_{[x']_R \in X/R} \left(\sum_{x'' \in [x']_R} t(x_1)(a)(x'') \cdot \sigma_l(x'')(w') \right) \text{ since by the induction hypothesis} \\
&\quad \text{for all } x'' \in [x']_R, \sigma_l(x'')(w') = \sigma_l(x')(w') \text{ and by grouping the states} \\
&\quad x'' \in [x'] \\
&= \sum_{[x']_R \in X/R} \left(\sum_{x'' \in [x']_R} t(x_2)(a)(x'') \cdot \sigma_l(x'')(w') \right) \text{ since } (x_1, x_2) \in R \text{ and } R \text{ is a } K\text{-} \\
&\quad \text{weighted bisimulation} \\
&= \sigma_l(x_2)(w) \text{ by an argument similar to the first two lines above.} \quad \square
\end{aligned}$$

Example 2.6. Figure 3 is an adaptation of a familiar example from the process algebra literature that shows that K -weighted bisimilarity strictly refines K -weighted language equivalence. Here $K = (\mathbb{R}_+, +, \cdot, 0, 1)$. The states x_0 and y_0 are language equivalent, since

FIGURE 3. Weighted language equivalence and weighted bisimulation



$\sigma_l(x_0)(ab) = \sigma_l(y_0)(ab) = \frac{1}{2}$ and $\sigma_l(x_0)(ac) = \sigma_l(y_0)(ac) = \frac{1}{2}$ and $\sigma_l(x_0)(w) = \sigma_l(y_0)(w) = 0$ for all other words w . But it is easily checked that x_0 and y_0 are not bisimilar.

3. K -LINEAR WEIGHTED AUTOMATA AS COALGEBRAS OVER SEMIMODULES

The goal of this section is to show that there is a functor $\mathcal{L} : \mathbf{SMod} \rightarrow \mathbf{SMod}$ for which behavioural equivalence $\approx_{\mathcal{L}}$ coincides with weighted language equivalence \sim_l of K -linear weighted automata, extending the results of Bonchi *et al.* [BBB⁺12] to the category of semimodules. Although the functor \mathcal{L} appears to be the same as in [BBB⁺12] the underlying details are different. The latter are based on a generalization of the notion of a *linear weighted automaton* in [Bor09, BBB⁺12] from the setting of vector spaces to semimodules. We propose definitions of a K -linear relation and a K -linear bisimulation that generalize the notions of a linear relation and a linear bisimulation from [Bor09]. This is the central part of the paper where the definitions and proofs do not mirror those in [Bor09, BBB⁺12]. In particular the notion of subspace of a vector space is replaced by that of the kernel (in the universal algebraic sense) of a K -linear map. But it is remarkable that all proofs go through and we obtain a true generalization of the concepts from vector spaces to semimodules.

3.1. K -Linear Weighted Automata. The following definition is a generalization of a linear weighted automaton of [Bor09] from the setting of vector spaces and linear maps to that of semimodules over a semiring K and K -linear maps. Note that we use the term “ K -linear weighted automaton” to distinguish it from the “linear weighted automaton” of [Bor09, BBB⁺12].

Definition 3.1. A K -linear weighted automaton (K -LWA in short) with input alphabet A over the semiring K is a coalgebra for the functor $\mathcal{L} = K \times (-)^A : \mathbf{SMod} \rightarrow \mathbf{SMod}$.

A K -LWA can be presented as a pair $(V, \langle o, t \rangle)$ where V is a semimodule over K whose elements are called *states*, $o : V \rightarrow K$ is a K -linear map assigning an *output weight* to every state and $t : V \rightarrow V^A$ is a K -linear *transition map* that, given a current state v and input a , assigns a new state $t(v)(a)$. We write $v_1 \xrightarrow{a} v_2$ for $t(v_1)(a) = v_2$. We often use V to refer to the K -LWA $(V, \langle o, t \rangle)$ when the output and transition maps are clear from the context.

The behaviour of K -LWA is described by weighted languages. The K -linear weighted language recognized by a state $v \in V$ of a K -LWA $(V, \langle o, t \rangle)$ is the map $\sigma_l(v) : A^* \rightarrow K$ defined by induction on words \mathcal{W} by:

$$\sigma_l(v)(w) = \begin{cases} o(v), & \text{if } w = \epsilon \\ \sigma_l(t(v)(a))(w') & \text{if } w = aw' \end{cases}$$

Two states $v_1, v_2 \in V$ are said to be weighted language equivalent, denoted $v_1 \sim_l v_2$, if $\sigma_l(v_1)(w) = \sigma_l(v_2)(w)$ for all $w \in A^*$. Note that we overload the symbol σ_l to denote weighted language equivalence for both K -weighted automata and K -linear weighted automata. The context disambiguates which concept the symbol denotes. Later in this section we show that $\sigma_l(v) = \llbracket v \rrbracket_V^\mathcal{L}$, the image of v under the unique \mathcal{L} -homomorphism from V into the final \mathcal{L} -coalgebra.

Given a K -WA $(X, \langle o, t \rangle)$ (see Section 2.4), we can construct a K -LWA $(K(X), \langle o^\#, t^\# \rangle)$, where $K(X)$ is the free semimodule generated by X and $o^\#$ and $t^\#$ are linear extensions of o and t . It can be shown that the above K -WA X and the K -LWA $K(X)$ have equivalent language behaviour, *i.e.*, the corresponding states x and $\eta_X(x)$ recognize the same weighted language for all $x \in X$.

Recall that a K -linear map $h : V \rightarrow W$ is an \mathcal{L} -homomorphism between K -LWA $(V, \langle o_V, t_V \rangle)$ and $(W, \langle o_W, t_W \rangle)$ when the following diagram commutes.

$$\begin{array}{ccc} V & \xrightarrow{h} & W \\ \langle o_V, t_V \rangle \downarrow & & \downarrow \langle o_W, t_W \rangle \\ K \times V^A & \xrightarrow{id_K \times h^A} & K \times W^A \end{array} \quad (3.1)$$

In words, for all $v \in V$, $a \in A$, $o_V(v) = o_W(h(v))$ and $h(t_V(v)(a)) = t_W(hv)(a)$.

For the special case when the K -LWA $V = K(X)$ and $W = K(Y)$ for given K -WA X and Y as above, we have the following situation. For a map $h : X \rightarrow Y$, the map $K(h) : K(X) \rightarrow K(Y)$ is the unique linear extension of $\eta_Y \circ h : X \rightarrow K(Y)$ and is hence K -linear. If h is a \mathcal{W} -homomorphism between the K -WA $(X, \langle o_X, t_X \rangle)$ and $(Y, \langle o_Y, t_Y \rangle)$ then $K(h)$ is an \mathcal{L} -homomorphism between the K -LWA $(K(X), \langle o_X^\#, t_X^\# \rangle)$ and $(K(Y), \langle o_Y^\#, t_Y^\# \rangle)$.

Bonchi *et al.* [BBB⁺12] showed that the final \mathcal{L} -coalgebra is defined on the set of all weighted languages K^{A^*} as follows. Consider the structure (K^{A^*}, ϵ, d) with the output map ϵ and the transition map d where $\epsilon : K^{A^*} \rightarrow K$, called the *empty map*, is defined by $\epsilon(\sigma) = \sigma(\epsilon)$ and $d : K^{A^*} \rightarrow (K^{A^*})^A$ is defined by $d(\sigma)(a) = \sigma_a$ where $\sigma_a : A^* \rightarrow K$ is the *a-derivative* of σ :

$$\sigma_a(w) = \sigma(aw).$$

We first show that the map d is K -linear. If σ_1 and σ_2 are two weighted languages in K^{A^*} , $k_1, k_2 \in K$, $a \in A$ and $w \in A^*$ then

$$\begin{aligned} d(k_1\sigma_1 + k_2\sigma_2)(a)(w) &= (k_1\sigma_1 + k_2\sigma_2)(aw) \\ &= (k_1\sigma_1)(aw) + (k_2\sigma_2)(aw) \\ &= k_1\sigma_1(aw) + k_2\sigma_2(aw) \\ &= k_1d(\sigma_1)(a)(w) + k_2d(\sigma_2)(a)(w) \end{aligned}$$

as desired. The proof of K -linearity of ϵ is similar. Hence (K^{A^*}, ϵ, d) is a coalgebra in \mathbf{SMod} . We now recall the proof from [BBB⁺12] that it is the final coalgebra in \mathbf{SMod} .

Theorem 3.2. *There exists a unique \mathcal{L} -homomorphism from any coalgebra $(V, \langle o, t \rangle)$ into the coalgebra (K^{A^*}, ϵ, d) .*

Proof. It is easy to check that the map $\llbracket - \rrbracket_V^{\mathcal{L}} = \sigma_l : V \rightarrow K^{A^*}$ which maps every state $v \in V$ to the weighted language $\sigma_l(v)$ is the only one that makes the following diagram commute in Set.

$$\begin{array}{ccc} V & \xrightarrow{\llbracket - \rrbracket_V^{\mathcal{L}}} & K^{A^*} \\ \langle o, t \rangle \downarrow & & \downarrow \langle \epsilon, d \rangle \\ \mathcal{L}(V) & \xrightarrow{\mathcal{L}(\llbracket - \rrbracket_V^{\mathcal{L}})} & \mathcal{L}(K^{A^*}) \end{array} \quad (3.2)$$

To show that $\llbracket - \rrbracket_V^{\mathcal{L}}$ is K -linear we prove that $\llbracket [k_1v_1 + k_2v_2] \rrbracket_V^{\mathcal{L}}(w) = \llbracket [k_1v_1] \rrbracket_V^{\mathcal{L}}(w) + \llbracket [k_2v_2] \rrbracket_V^{\mathcal{L}}(w)$ for all $w \in A^*$ by a routine induction on the length of the word w . \square

It follows that two states $v_1, v_2 \in V$ are \mathcal{L} -behaviourally equivalent, *i.e.*, $v_1 \approx_{\mathcal{L}} v_2$ iff they recognize the same weighted language.

3.2. K -Linear Bisimulation. In this section we generalize the definition of Boreale's linear weighted bisimulation [Bor09] from a field to a semiring K . We show that the two notions coincide in the special case when K is a field, for example $K = \mathbb{R}$, as in [Bor09]. Starting with this section almost all the results are our contribution and involve new concepts and proofs.

Definition 3.3. A binary relation R on a K -semimodule V is K -linear if there exists a K -semimodule W and a K -linear map $f : V \rightarrow W$ such that $R = \ker f = \{(u, v) \mid f(u) = f(v)\}$. Such a relation is denoted by R_f for the given f .

It is immediate that a K -linear relation on a K -semimodule V is an equivalence relation which, in addition, is a congruence. Moreover, there is a canonical way of turning *any* relation R on $K(X)$ into a K -linear relation R^ℓ as follows. Let R^ℓ be the least congruence relation on V containing R . R^ℓ is obtained by taking the intersection of all congruences on V that contain R and is well-defined since the universal relation is a congruence and the intersection of any family of congruences is a congruence. The quotient set V/R^ℓ has a K -semimodule structure given by $[u] + [v] = [u + v]$ and $k.[u] = [k.u]$, which are well-defined since R^ℓ is a congruence. Let $f = \varepsilon_{R^\ell} : X \rightarrow V/R^\ell$ be the map which sends any elements $v \in V$ to its equivalence class $[v]_{R^\ell}$. It is easy to check that f is a K -linear map and $R^\ell = \ker(f)$ by construction. Hence R^ℓ is a K -linear relation. The following lemma is an easy consequence of the definitions.

Lemma 3.4. *For any binary relation R on a K -module V , R^ℓ is the smallest K -linear relation containing R .*

Note that for a given K -linear relation R there may be two distinct K -linear maps $f, g : V \rightarrow W$ with the same codomain such that $R = \ker(f) = \ker(g)$. On the other hand, all such maps factor uniquely through the map $\varepsilon_f : V \rightarrow V/\ker f$ that sends $v \in V$ to its equivalence class $[v]$ in $\ker f$. Also, note that for any injective map $f : V \rightarrow W$, R_f is the identity relation on V . For the zero map $0_{V,W} : V \rightarrow W$ which maps every element in V to 0, $R_{0_{V,W}}$ is the universal relation on V .

We are now ready to define a K -linear bisimulation in analogy with linear bisimulation in [Bor09, BBB⁺12].

Definition 3.5. Let $(V, \langle o, t \rangle)$ be a K -LWA, for a semimodule K . A K -linear relation R on V is a K -linear bisimulation if for all (v_1, v_2) the following holds:

- (1) $o(v_1) = o(v_2)$, and
- (2) $\forall a \in A, t(v_1)(a) R t(v_2)(a)$.

When V is a finite dimensional vector space over the field K the notions of K -linear relations and the linear relations of Boreale coincide. Consider a linear relation (as defined in [Bor09]) R over the vector space $V = K(X)$ with basis X where K is a field. By definition, there exists a subspace U of the vector space V over K such that uRv iff $u - v \in U$. It is easy to check that the equivalence relation $R = R_U = \{(u, v) \mid u - v \in U\}$ is a congruence. Then consider the canonical linear map $f_U : V \rightarrow V/R_U$ to the quotient space which maps an element $w \in V$ to $[w]_R$. Now $w \in U$ iff $w = u - v$ for some $u, v \in V$ with uRv . Therefore, $f_U(w) = f_U(u - v) = f_U(u) - f_U(v) = 0$ since $[u] = [v]$, i.e., f_U sends all elements in U to 0. It follows that $u - v \in U$ iff $f_U(u) = f_U(v)$ and thus R is a K -linear relation. Conversely, if R is a K -linear relation over V then $R = R_f$ for some $f : V \rightarrow W$. Let $U = \{w \mid f(w) = 0\}$. Clearly U is a subspace of V and $u - v \in U$ iff $f(u) = f(v)$, i.e., R is a linear relation. In addition, when V is a vector space over a field K it is routine to verify that the notions of a linear weighted bisimulation as in [Bor09, BBB⁺12] and a K -linear bisimulation coincide, as the two definitions are identical. Hence, K -linear bisimulation is a more general notion.

The following characterization of K -linear bisimulation is immediate from the definition.

Lemma 3.6. Let $(V, \langle o, t \rangle)$ be a K -LWA, where V is a K -semimodule and R a K -linear relation on V . Then R is a K -linear bisimulation iff

- (1) $R \subseteq \ker(o)$, and
- (2) R is t_a -invariant, i.e., uRv implies $t_a u R t_a v$, for each $a \in A$.

More generally, the kernel of a \mathcal{L} -homomorphism between two K -LWA is a K -linear bisimulation and conversely, for each K -linear bisimulation R there exists a \mathcal{L} -homomorphism between two K -LWA whose kernel is R . This result mirrors the one in [BBB⁺12] for linear weighted bisimulation between linear weighted automata, albeit with a different notion of kernel.

Proposition 3.7. Let $(V, \langle o_V, t_V \rangle)$ and $(W, \langle o_W, t_W \rangle)$ be two K -LWA and $h : V \rightarrow W$ an \mathcal{L} -homomorphism. Then $\ker(h)$ is a K -linear bisimulation on $(V, \langle o_V, t_V \rangle)$. Conversely, if R is a K -linear bisimulation on $(V, \langle o_V, t_V \rangle)$ then there exists a K -LWA $(W, \langle o_W, t_W \rangle)$ and a \mathcal{L} -homomorphism $h : V \rightarrow W$ such that $R = \ker(h)$.

Proof. Suppose $h : V \rightarrow W$ is an \mathcal{L} -homomorphism between $(V, \langle o_V, t_V \rangle)$ and $(W, \langle o_W, t_W \rangle)$. We show that $\ker(h)$ satisfies clauses (1) and (2) of Lemma 3.6. Take any $(v, w) \in \ker(h)$. Since by definition $h(v) = h(w)$, we have $o_W(h(v)) = o_W(h(w))$ and $t_W(h(v))(a) = t_W(h(w))(a)$ for all $a \in A$. Since h is an \mathcal{L} -homomorphism, we have (1) $o_V(v) = o_W(h(v)) = o_W(h(w)) = o_V(w)$, i.e., $\ker(h) \subseteq \ker(o_V)$ and (2) $h(t_V(v)(a)) = t_W(h(v))(a) = t_W(h(w))(a) = h(t_V(w)(a))$, which means $(t_V(v)(a), t_V(w)(a)) \in \ker(h)$ i.e., $\ker(h)$ is t_a -invariant.

In the other direction, let R be K -linear bisimulation on $(V, \langle o_V, t_V \rangle)$, where $R = R_f$ for the map $f : V \rightarrow W$. Let $f = \mu_f \circ \varepsilon_f$ be the unique factorization of f through $\varepsilon_f : V \rightarrow V/\ker(f)$ that sends each $v \in V$ to its equivalence class in $\ker(f)$. As we

observed earlier, $W = V/\ker(f)$ can be equipped with a K -LWA structure $(W, \langle o_W, t_W \rangle)$ as follows. The K -linear map $o_W : W \rightarrow K$ is defined as $o_W([v]) = o_V(v)$. The K -linear map $t_W : W \rightarrow W^A$ defined as $t_W([v])(a) = t_V(v)(a)$. These two maps are well-defined as $\ker(f)$ is a congruence. It is routine to verify that the K -linear map $h = \varepsilon_f : V \rightarrow W$ is an \mathcal{L} -homomorphism and $\ker(h) = \ker(\varepsilon_f)$, and therefore $R = \ker(h)$. \square

Theorem 3.8. *Let $(V, \langle o, t \rangle)$ be a K -LWA and let $\llbracket - \rrbracket_V^{\mathcal{L}} : V \rightarrow K^{A^*}$ be the unique \mathcal{L} -homomorphism into the final coalgebra. Then $\ker(\llbracket - \rrbracket_V^{\mathcal{L}})$ is the largest K -linear bisimulation on V .*

Proof. By Proposition 3.7, $\ker(\llbracket - \rrbracket_V^{\mathcal{L}})$ is a K -linear bisimulation. Suppose R is any K -linear bisimulation. Again by Proposition 3.7, there exists a K -LWA $(W, \langle o_W, t_W \rangle)$ and a \mathcal{L} -homomorphism $f : V \rightarrow W$ such that $R = \ker(f)$. Since $(W, \langle o_W, t_W \rangle)$ is an \mathcal{L} -coalgebra, there exists an \mathcal{L} -homomorphism $\llbracket - \rrbracket_W^{\mathcal{L}}$ from W to the final coalgebra K^{A^*} . Therefore $\llbracket - \rrbracket_W^{\mathcal{L}} \circ f : V \rightarrow K^{A^*}$, being the composition of two \mathcal{L} -homomorphisms, is an \mathcal{L} -homomorphism. But $\ker(\llbracket - \rrbracket_V^{\mathcal{L}})$ is the unique homomorphism from V to K^{A^*} by the finality of K^{A^*} and hence $\llbracket - \rrbracket_W^{\mathcal{L}} \circ f = \ker(\llbracket - \rrbracket_V^{\mathcal{L}})$. Then $R = \ker(f) \subseteq \ker(\llbracket - \rrbracket_W^{\mathcal{L}} \circ f) = \ker(\llbracket - \rrbracket_V^{\mathcal{L}})$. The set inclusion above is a consequence of the fact that $f(u) = f(v)$ implies $g(f(u)) = g(f(v))$ for all g composable with f . \square

Corollary 3.9. *The \mathcal{L} -behavioural equivalence relation $\approx_{\mathcal{L}}$ on a K -LWA V is the largest K -linear bisimulation.*

Proof. By definition, $\approx_{\mathcal{L}} = \{(v, w) \mid \llbracket v \rrbracket_V^{\mathcal{L}} = \llbracket w \rrbracket_V^{\mathcal{L}}\} = \ker(\llbracket - \rrbracket_V^{\mathcal{L}})$. The result follows from Theorem 3.8. \square

To summarize, for K -WA the largest K -weighted bisimulation \sim_w is strictly included in K -weighted language equivalence \sim_l as shown in Proposition 2.5. Corollary 3.9 shows that for K -LWA K -linear language equivalence coincides with the largest K -linear bisimulation. This raises the question: what is the relationship between K -weighted bisimulation and K -linear bisimulation? Again, our answer extends that of [BBB⁺12] to the semimodule setting.

Proposition 3.10. *Let $(X, \langle o, t \rangle)$ be a K -WA and $(K(X), \langle o^\sharp, t^\sharp \rangle)$ the K -LWA obtained from it as in Section 3.1. If R is a K -weighted bisimulation on X then R^ℓ is a K -linear bisimulation on $K(X)$.*

Proof. Recall the quotient weighted automaton $(X/R, \langle o_{X/R}, t_{X/R} \rangle)$ and the map $\varepsilon_R : X \rightarrow X/R$ from Section 2. From diagram 2.7 we have ε_R is a \mathcal{W} -homomorphism between $(X, \langle o, t \rangle)$ and $(X/R, \langle o_{X/R}, t_{X/R} \rangle)$. Earlier we have shown that $K(h)$ is an \mathcal{L} -homomorphism for every \mathcal{W} -homomorphism h . Therefore, $K(\varepsilon_R) : K(X) \rightarrow K(X/R)$ is an \mathcal{L} -homomorphism between $(K(X), \langle o^\sharp, t^\sharp \rangle)$ and $(K(X/R), \langle o_{X/R}^\sharp, t_{X/R}^\sharp \rangle)$. By Proposition 3.7, $\ker(K(\varepsilon_R))$ is a K -linear bisimulation on $K(X)$.

It remains to show that $\ker(K(\varepsilon_R)) = R^\ell$. Recall that $K(\varepsilon_R) : K(X) \rightarrow K(X/R)$ maps $k_i x_i$ to

$$\left(\sum_{x_j \in [x_i]_R} k_i \right) [x_i]_R$$

and hence by linearity maps any element $u = \sum_{i=1}^n k_i x_i$ of $K(X)$ to

$$\sum_{i=1}^n \left(\sum_{x_j \in [x_i]_R} k_j \right) [x_i]_R.$$

Therefore, for any element $v = \sum_{i=1}^m k'_i y_i$ of $K(X)$, $(u, v) \in \ker(K(\varepsilon_R))$ iff for all $x \in X$,

$$\sum_{x_j \in [x]_R} k_j = \sum_{y_\ell \in [x]_R} k'_\ell. \quad (3.3)$$

We show that $(u, v) \in R^\ell$ if the same condition holds. First, since R is an equivalence relation, and R^ℓ is the smallest congruence containing R , R^ℓ satisfies the following clauses (i) $xRy \Rightarrow k.xR^\ell k.y$ for all $k \in K$ and (ii) xRy and $x'Ry' \Rightarrow (k.x + k'.x')R^\ell(k.y + k'.y')$ for all $k_1, k_2 \in K$, and (iii) for any $u, v \in K(X)$, $uR^\ell v$ only if it can be derived from the rules (i) and (ii) above. It follows that for u and v as above, $uR^\ell v$ iff for all $x \in X$, Equation 3.3 holds. \square

4. K -LINEAR PARTITION REFINEMENT

We have shown that weighted language equivalence \sim_l coincides with the largest K -linear bisimulation $\approx_{\mathcal{L}}$ on a K -LWA. Now let us turn to the question of computing \sim_l , *i.e.*, given two states v_1 and v_2 in a K -LWA $(V, \langle o, t \rangle)$ the problem of deciding whether $v_1 \sim_l v_2$. For finite representability we assume that the submodule V is freely generated by a finite set X , and therefore o and t have finite representations as matrices.

Boreale [Bor09] and Bonchi *et al.* [BBB⁺12] had proposed two versions of a partition refinement algorithm for finding the largest linear weighted bisimulation on a linear weighted automaton $(V, \langle o, t \rangle)$. Their algorithms were based on the identification of linear bisimulations with certain subspaces of the vector space of a linear weighted automaton over a field. Both algorithms start from the kernel of the output map o represented as a subspace of V , and successively obtain smaller and smaller subspaces by requiring invariance under the transitions until a fixed point is reached. For termination the algorithm relies on the fact that there can only be a finite descending chain of subspaces for a finite-dimensional vector space. Moreover, the algorithm computes a basis for each of the subspaces by solving systems of linear equations, using the matrix representations of the maps o and t .

Unfortunately, these algorithms cannot, in general, be lifted to the semiring-semimodule framework even for finitely generated semimodules. First, unlike vector spaces, finitely generated semimodules are not necessarily *Artinian*. An Artinian semimodule is one that satisfies the descending chain property, *i.e.*, there is no infinite descending chain of subsemimodules ordered by inclusion. For example, the semimodule $(\mathbb{N}, +, 0)$ over the semiring $(\mathbb{N}, +, \cdot, 0, 1)$, although finitely generated, is *not* Artinian since the subsemimodules $2\mathbb{N} \supset 4\mathbb{N} \supset 8\mathbb{N} \dots$ form an infinite descending chain. Second, even with an Artinian semimodule the above procedures are not effective in general, as they depend on solving linear equations in K , a problem that is undecidable for certain semirings [Nar96]. It is also known that weighted language equivalence is undecidable for finite-state weighted automata over the tropical semiring $(\mathbb{N} \cup \{\infty\}, \min, +, \infty, 0)$ [Kro94, ABK20].

But all is not lost as far as partition refinement is concerned. We can generalize the forward algorithm of [Bor09] and [BBB⁺12] to a construction of the final coalgebra in SMod based on a method of of Adámek and Koubek [AK95] that uses the notion of the

It follows from Proposition 4.1 that when V is a finite dimensional vector space then for some $n \in \mathbb{N}$, $\ker(!_{n+1}) = \ker(!_n)$ *i.e.*, the procedure terminates after a finite number of steps. This is because for vector spaces and linear maps, kernels (in the sense of [BBB⁺12]) correspond to subspaces and for a finite dimensional space there can only be a finite chain of subspace containment; see [BBB⁺12] for the details. For the more general case of finitely generated Artinian semimodules over a semiring, we can also guarantee stabilization of the cochain. This is shown in the following proposition.

Proposition 4.2. *Let $(V, \langle o, t \rangle)$ be a K -LWA where V is a finitely generated Artinian semimodule. Consider the sequence of relations over V defined inductively by*

$$R_0 = \ker(o) \quad R_{i+1} = R_i \cap \bigcap_{a \in A} \{(u, v) \mid (t(u)(a), t(v)(a)) \in R_i\}.$$

Then there is a j such that $R_j = R_{j+1}$. The largest K -linear bisimulation $\approx_{\mathcal{L}}$ on V is then identical to R_j .

Proof. We have shown that $R_n = \ker(!_{n+1})$ for each n . Since each R_i is linear it is a subsemimodule of the product $V \times V$. Since V is Artinian so is $V \times V$ as the projection of a subsemimodule of $V \times V$ is a subsemimodule of V . Since $R_i \supseteq R_{i+1}$, and we cannot have an infinite descending chain of subsemimodule inclusions there exists a j such that $R_j = R_{j+1}$.

We now show that R_j is a K -linear bisimulation by applying Lemma 3.6. Since $R_j \subseteq R_0 = \ker(o)$, condition (1) of the lemma is satisfied. Since $R_j = R_{j+1}$, we have $R_j = R_j \cap \bigcap_{a \in A} \{(u, v) \mid (t(u)(a), t(v)(a)) \in R_j\}$, *i.e.*, uR_jv implies $t_a u R_j t_a v$, for each $a \in A$, which means condition (2) is also satisfied.

Finally, we must show that any K -linear bisimulation R is included in R_j . We do this by showing $R \subseteq R_i$ for all i by induction. By Lemma 3.6, $R \subseteq \ker(o) = R_0$. Now assume $R \subseteq R_i$. By Lemma 3.6 again, uRv implies $t_a u R t_a v$ for all $a \in A$ and hence, $t_a u R_i t_a v$ for all $a \in A$. By definition of R_{i+1} it follows that $uR_{i+1}v$ and hence $R \subseteq R_{i+1}$. \square

We now show that the termination condition in Proposition 4.2, namely V is Artinian, can be weakened. This weaker condition is mentioned in [DK12, KK18] but in a somewhat different setting. To state the condition, let $V_n \subseteq V = K(X)$ for a finite set X be defined by $V_n = \{\sigma_l(\cdot)(w) : X \rightarrow K \mid w \in A_n^*\}$ and let $V_* = \bigcup_{n=0}^{\infty} V_n$. Then the following proposition states that the procedure described in Proposition 4.2 terminates if $\text{span}(V_*)$ is a finitely generated semimodule.

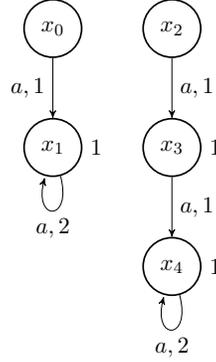
Proposition 4.3. *Let $(V, \langle o, t \rangle)$ be a K -LWA where $V = K(X)$ for a finite set X , such that $\text{span}(V_*)$ is finitely generated. Consider the sequence of relations over V defined inductively by*

$$R_0 = \ker(o) \quad R_{i+1} = R_i \cap \bigcap_{a \in A} \{(u, v) \mid (t(u)(a), t(v)(a)) \in R_i\}.$$

Then there is a j such that $R_j = R_{j+1}$. The largest K -linear bisimulation $\approx_{\mathcal{L}}$ on V is then identical to R_j .

Proof. Suppose $\text{span}(V_*)$ is finitely generated. Since $V_n \subseteq V_{n+1}$ for all n , and $V_* = \bigcup_{n=0}^{\infty} V_n$, we have, for some j , $\text{span}(V_*) = \text{span}(V_j) = \text{span}(V_{j+1})$. Below we show that $\ker(!_j) = \ker(!_{j+1})$. It follows from Proposition 4.1 that $R_j = R_{j+1}$. The rest of the proof is identical to the one for Proposition 4.2.

It is immediate from Proposition 4.1 that $\ker(!_{j+1}) \subseteq \ker(!_j)$, as $R_{j+1} \subseteq R_j$. To prove the containment in the other direction, suppose $(u, v) \in \ker(!_j)$. Then $\sigma_l(u)(w) = \sigma_l(v)(w)$ for all

FIGURE 4. A K -linear weighted automaton over the ring \mathbb{Z} 

$w \in A_j^*$ from Equation 4.2. Now, since $\text{span}(V_j) = \text{span}(V_{j+1})$, any element of V_{j+1} is a linear combination of elements of V_j , *i.e.*, for any $w' \in A_{j+1}^*$ we have, $\sigma_l(-)(w') = \sum_{i=1}^n \sigma_l(-)(w_i)$ for some $w_i \in A_j^*$, $1 \leq i \leq n$. Therefore,

$$\begin{aligned}
 \sigma_l(u)(w') &= \sum_{i=1}^n \sigma_l(u)(w_i) \text{ for some } w_i \in A_j^*, 1 \leq i \leq n \\
 &= \sum_{i=1}^n \sigma_l(v)(w_i) \text{ since } \sigma_l(u)(w) = \sigma_l(v)(w) \text{ for all } w \in A_j^* \\
 &= \sigma_l(v)(w'),
 \end{aligned}$$

i.e., $(u, v) \in \ker(!_{j+1})$. □

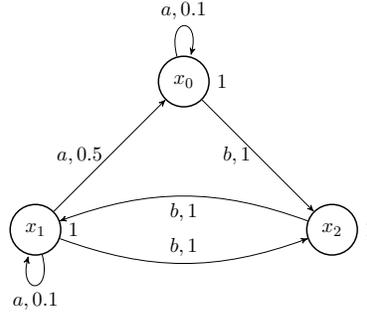
However, even when the final sequence stabilizes after a finite number of steps, in general there is no effective procedure to compute the kernel of a K -linear map for a semiring K . This is because unlike in \mathbb{R} , \mathbb{Z} and \mathbb{N} , for semirings in general we do not have a procedure for linear equation solving; the problem is undecidable for certain semirings [Nar96]. We conclude this section by presenting a couple of examples from different semimodules to illustrate how our procedure works.

Example 4.4. Figure 5 shows a K -LWA $(V, \langle o, t \rangle)$, adapted from [Sak16], over the alphabet $A = \{a\}$. The semiring K is the ring \mathbb{Z} of integers. Here $V = \mathbb{Z}(X)$ where $X = \{x_0, x_1, x_2, x_3, x_4\}$. In the example, the maps o and t are generated from their values shown against the nodes and edges, respectively, by linear extension. Missing values against nodes are assumed to be zero. Note that V is *not* Artinian but $\text{span}(V_*)$ is indeed finitely generated. To see this, V_3 consists of the set containing the following maps from X to \mathbb{Z}

$$\begin{aligned}
 \sigma_l(-)(\epsilon) &= \{x_0 \mapsto 0, x_1 \mapsto 1, x_2 \mapsto 0, x_3 \mapsto 1, x_4 \mapsto 1\} \\
 \sigma_l(-)(a) &= \{x_0 \mapsto 1, x_1 \mapsto 2, x_2 \mapsto 1, x_3 \mapsto 1, x_4 \mapsto 2\} \\
 \sigma_l(-)(aa) &= \{x_0 \mapsto 2, x_1 \mapsto 4, x_2 \mapsto 1, x_3 \mapsto 2, x_4 \mapsto 4\}.
 \end{aligned}$$

Now, it is easily seen that for all $x \in X$ and $n \geq 2$, $\sigma_l(x)(a^n) = 2^{n-2}\sigma_l(x)(aa)$. Therefore, $\text{span}(V_*)$ is generated by V_3 . To apply the partition refinement procedure to this K -LWA,

FIGURE 5. A K -linear weighted automaton over the semiring $K = ([0, 1], \max, \cdot, 0, 1)$



we use Propositions 4.1 and 4.3. We have

$$R_0 = \ker(o) = \{(u, v) \mid u = \sum_{i=0}^4 c_i x_i, v = \sum_{i=0}^4 d_i x_i, \text{ where } c_i, d_i \in \mathbb{Z} \text{ for } i \in [0, 4]\}$$

and

$$c_1 + c_3 + c_4 = d_1 + d_3 + d_4\},$$

as x_1 , x_3 and x_4 are the only states with nonzero weights and all of them have weight one. In the next iteration, we have

$$R_1 = R_0 \cap \{(u, v) \mid u = \sum_{i=0}^4 c_i x_i, v = \sum_{i=0}^4 d_i x_i, \text{ where } c_i, d_i \in \mathbb{Z} \text{ for } i \in [1, 4]\}$$

and

$$c_0 + 2c_1 + c_2 + c_3 + 2c_4 = d_0 + 2d_1 + d_2 + d_3 + 2d_4\},$$

as all the edges have weight one except the loops on x_1 and x_4 , which have weight two. It is clear that $R_2 = R_1$, since we take the intersection with the same set in obtaining R_1 from R_0 as in obtaining R_2 from R_1 . Therefore, the K -linear bisimilarity relation, *i.e.*, weighted language equivalence, is given by the set of all pairs (u, v) with $u = \sum_{i=0}^4 c_i x_i$, $v = \sum_{i=0}^4 d_i x_i$ satisfying the integer equations

$$c_1 + c_3 + c_4 = d_1 + d_3 + d_4$$

$$c_0 + 2c_1 + c_2 + c_3 + 2c_4 = d_0 + 2d_1 + d_2 + d_3 + 2d_4.$$

When $V = \mathbb{Z}(X)$ for a finite set X , it can be shown that the number of iterations of the loop computing R_{i+1} from R_i is bounded by $|X|$ and termination then follows from the decidability of linear integer arithmetic. A more efficient algorithm for deciding language equivalence between states in X (and not the entire $V(X)$ as in our case) appears in [BLS05]. In fact, this problem is decidable for any semiring that is a subsemiring of a field [Sak09].

Example 4.5. Figure 5 shows a K -LWA $(V, \langle o, t \rangle)$, adapted from [KK18], over the alphabet $A = \{a, b\}$ and semiring $K = ([0, 1], \max, \cdot, 0, 1)$. Here $V = K(X)$ where $X = \{x_0, x_1, x_2\}$. It can be checked that V is not an Artinian semimodule but V_* is finitely generated. As

in the example above, we use Propositions 4.1 and 4.3 to apply the partition refinement procedure. We have

$$R_0 = \ker(o) = \{(u, v) \mid u = \sum_{i=0}^2 c_i x_i, v = \sum_{i=0}^2 d_i x_i, \text{ where } c_i, d_i \in [0, 1] \text{ for } i \in \{0, 1, 2\}\}$$

and

$$\max\{c_0, c_1, c_2\} = \max\{d_0, d_1, d_2\}.$$

In the next iteration we have,

$$R_1 = R_0 \cap \{(u, v) \mid u = \sum_{i=0}^2 c_i x_i, v = \sum_{i=0}^2 d_i x_i, \text{ where } c_i, d_i \in [0, 1] \text{ for } i \in \{0, 1, 2\}\}$$

and $(t(u)(a), t(v)(a)) \in R_0$

and $(t(u)(b), t(v)(b)) \in R_0\}$.

In the above expression, $t(u)(a) = \max\{0.1c_0, 0.5c_1, 0.1c_2\}$, $t(v)(a) = \max\{0.1d_0, 0.5d_1, 0.1d_2\}$, $t(u)(b) = \max\{c_1, c_2\}$ and $t(v)(b) = \max\{d_1, d_2\}$. Again, we can check that $R_2 = R_1$. The resulting linear equations over K can be solved by using the theory of l -monoids and residuation [KK18].

4.2. Related Partition Refinement Algorithms. For specific semirings, algorithms have been proposed in the case of probabilities [KMO⁺11], fields [Bor09], rings [DK12], division rings [FL97] and principal ideal domains [BLS05].

The approach to partition refinement which has some similarity with ours is by König and Küpper [KK18]. This section is a brief summary of their work followed by a comparison with the current paper. The paper generalizes the partition refinement algorithm to a coalgebraic setting, just as this paper. It proposes a generic procedure for language equivalence for transition systems, of which weighted automata over arbitrary semirings form an important special case. However, unlike the usual partition refinement approach, [KK18] does not provide a unique or canonical representative for the weighted language accepted, in general. The procedure based on the approach is not guaranteed to terminate in all cases, but does so for particular semirings, just as in our case. It is based on a notion of equivalence classes of arrows and involves solving linear equations for a given semiring.

In more detail, [KK18] uses $\mathcal{M}(K)$, the category of finite sets and matrices over the semiring K with matrix multiplication as composition, as the base category. This is just the Kleisli category of the free semimodule monad and is equivalent to the category of free semimodules over finite sets that we use. The endofunctor F over $\mathcal{M}(K)$ defining a weighted automaton is given by $FX = 1 + A \times X$. This can be seen as being equivalent to our \mathcal{L} , as the paper [KK18] uses matrices as arrows rather than maps.

The paper presents two generic procedures for partition refinement, where the second is just an optimized version of the first. The basic ingredient in the theory which is new is a preorder on objects and arrows of a concrete category \mathcal{C} . For objects X and Y in \mathcal{C} , this is defined by $X \leq Y$ if there is an arrow $f : X \rightarrow Y$. The relation $X \equiv Y$ holds when $X \leq Y$ and $Y \leq X$. For arrows $f : X \rightarrow Y$ and $g : X \rightarrow Z$ with the same domain, $f \leq^X g$ if there exists an arrow $h : Y \rightarrow Z$ such that $g = h \circ f$. Similarly, $f \equiv^X g$ if $f \leq^X g$ and $g \leq^X f$. It is easy to check that \leq and \leq^X are preorders and \equiv and \equiv^X are equivalence relations.

with that in [KK18], which provides another partition refinement algorithm for weighted automata over semirings in the coalgebraic framework.

ACKNOWLEDGMENT

The author thanks David Benson for introducing him to category theory 35 years ago. He also wishes to acknowledge the anonymous referees for their help in substantially refining the contents of Section 4 of the paper on partition refinement.

REFERENCES

- [ABK20] Shaull Almagor, Udi Boker, and Orna Kupferman. What’s decidable about weighted automata? *Information and Computation*, 2020. doi:10.1016/j.ic.2020.104651.
- [AK95] Jiří Adámek and Václav Koubek. On the greatest fixed point of a set functor. *Theoretical Computer Science*, 150(1):57–75, 1995.
- [BBB⁺12] Filippo Bonchi, Marcello Bonsangue, Michele Boreale, Jan Rutten, and Alexandra Silva. A coalgebraic perspective on linear weighted automata. *Information and Computation*, 211:77–105, 2012.
- [BLS05] Marie-Pierre Béal, Sylvain Lombardy, and Jacques Sakarovitch. On the equivalence of \mathbb{Z} -automata. In *International Colloquium on Automata, Languages, and Programming*, pages 397–409. Springer, 2005.
- [Bor09] Michele Boreale. Weighted bisimulation in linear algebraic form. In *International Conference on Concurrency Theory*, pages 163–177. Springer, 2009.
- [BS81] Stanley Burris and H.P. Sankappanavar. *A Course in Universal Algebra*, volume 78 of *Graduate Texts in Mathematics*. Springer, 1981.
- [Buc08] Peter Buchholz. Bisimulation relations for weighted automata. *Theoretical Computer Science*, 393(1-3):109–123, 2008.
- [DK12] Manfred Droste and Dietrich Kuske. Weighted automata. In *Automata: From Mathematics to Applications*. European Mathematical Society, 2012.
- [DKV09] Manfred Droste, Werner Kuich, and Heiko Vogler. *Handbook of weighted automata*. Springer Science & Business Media, 2009.
- [FL97] Marianne Flouret and E Laugerotte. Noncommutative minimization algorithms. *Information Processing Letters*, 64(3):123–126, 1997.
- [HJS07] Ichiro Hasuo, Bart Jacobs, and Ana Sokolova. Generic trace semantics via coinduction. *Logical Methods in Computer Science*, 3, 2007.
- [Jac16] Bart Jacobs. *Introduction to Coalgebra: Towards Mathematics of States and Observation*. Cambridge Tracts in Theoretical Computer Science; 59. Cambridge University Press, 2016.
- [KK18] Barbara König and Sebastian Küpper. A generalized partition refinement algorithm, instantiated to language equivalence checking for weighted automata. *Soft Computing*, 22(4):1103–1120, 2018.
- [KMO⁺11] Stefan Kiefer, Andrzej S Murawski, Joël Ouaknine, Björn Wachter, and James Worrell. Language equivalence for probabilistic automata. In *International Conference on Computer Aided Verification*, pages 526–540. Springer, 2011.
- [Kro94] Daniel Kroh. The equality problem for rational series with multiplicities in the tropical semiring is undecidable. *International Journal of Algebra and Computation*, 4:405–425, 1994.
- [Mil89] Robin Milner. *Communication and concurrency*, volume 84. Prentice Hall, 1989.
- [Nar96] Paliath Narendran. Solving linear equations over polynomial semirings. In *Proceedings 11th Annual IEEE Symposium on Logic in Computer Science*, pages 466–472. IEEE, 1996.
- [Par81] David Park. Concurrency and automata on infinite sequences. In *Theoretical Computer Science*, LNCS, vol. 104, pages 167–183. Springer, 1981.
- [Rut00] Jan Rutten. Universal coalgebra: a theory of systems. *Theoretical Computer Science*, 249(1):3–80, 2000.
- [Sak09] Jacques Sakarovitch. Rational and recognisable power series. In *Handbook of Weighted Automata*, pages 105–174. Springer, 2009.

- [Sak16] Jacques Sakarovitch. Introduction to weighted automata theory. Lectures slides, 19th Estonian Winter School in Computer Science, 2016. Available at <https://cs.ioc.ee/ewscs/2014/sakarovitch/sakarovitch-slides.pdf>.
- [Sch61] Marcel Paul Schützenberger. On the definition of a family of automata. *Information and Control*, 4(2-3):245–270, 1961.