Weighted automata coalgebraically

Filippo Bonchi[4] Michele Boreale[5] Marcello Bonsangue[1,2]
Jan Rutten[1,3] Alexandra Silva[1]

[1]Centrum Wiskunde en Informatica
[2]LIACS - Leiden University
[3]Radboud Universiteit Nijmegen
[4]INRIA Saclay - LIX, École Polytechnique
[5]Università di Firenze

Braunschweig, August 2010

Motivation

- Weighted automata are transition systems with many applications: speech processing, image recognition, information theory, . . .
- Interesting questions: correct notion of equivalence, algorithms for minimization, . . .

Motivation

- Weighted automata are transition systems with many applications: speech processing, image recognition, information theory, . . .
- Interesting questions: correct notion of equivalence, algorithms for minimization, . . .

Motivation (cont'd)

- Modelling systems as coalgebras:
  - The type of the system determines a canonical notion of equivalence . . .
  - . . . and a universe of behaviours (final coalgebra).

Goals of this talk:

1. Show how to model weighted automata as coalgebras in two different settings;
2. Show the canonical equivalences derived in each;
3. An algorithm for minimization.

Motivation (cont'd)

- Modelling systems as coalgebras:
  - The type of the system determines a canonical notion of equivalence ...
  - ...and a universe of behaviours (final coalgebra).

Goals of this talk:

1. Show how to model weighted automata as coalgebras in two different settings;
2. Show the canonical equivalences derived in each;
3. An algorithm for minimization.

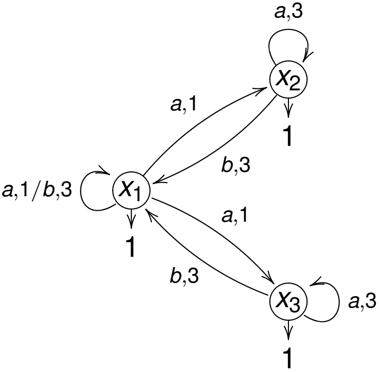
Weighted automata

A *weighted automaton* with input alphabet A is a pair $(X, \langle o, t \rangle)$, where

- X is a set of states;
- $o \colon X \to \mathbb{K}$ is an output function associating to each state its output weight and
- $t \colon X \to (\mathbb{K}^X)^A$ is the transition relation that associates a weight to each transition.

Example

$\mathbb{K} = \mathbb{R}$, $A = \{a, b\}$

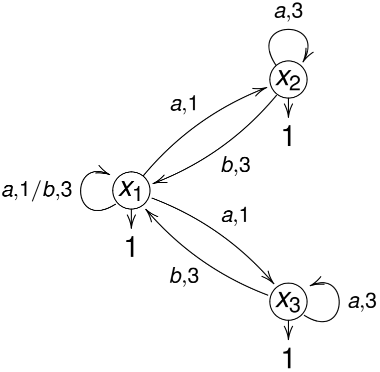


Convenient matrix representation (for X finite):

$$O_X = \begin{pmatrix} 1 & 1 & 1 \end{pmatrix} \quad (x_1\ x_2\ x_3) \qquad T_{X_a} = \begin{pmatrix} 1 & 0 & 0 \\ 1 & 3 & 0 \\ 1 & 0 & 3 \end{pmatrix} \qquad T_{X_b} = \begin{pmatrix} 3 & 3 & 3 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}$$

Example

$\mathbb{K} = \mathbb{R}$, $A = \{a, b\}$

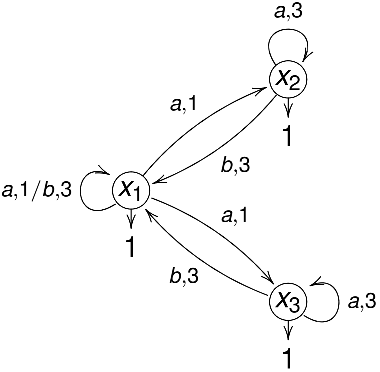


Convenient matrix representation (for X finite):

$$O_X = \begin{pmatrix} \overset{x_1}{1} & \overset{x_2}{1} & \overset{x_3}{1} \end{pmatrix} \quad T_{X_a} = \begin{pmatrix} 1 & 0 & 0 \\ 1 & 3 & 0 \\ 1 & 0 & 3 \end{pmatrix} \quad T_{X_b} = \begin{pmatrix} 3 & 3 & 3 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}$$

Example

$\mathbb{K} = \mathbb{R}$, $A = \{a, b\}$

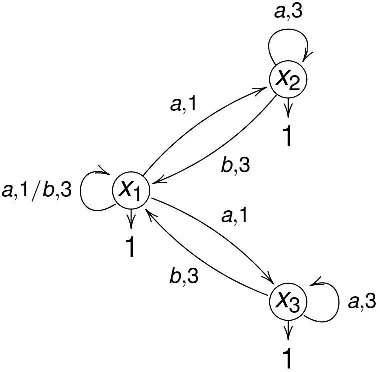


Convenient matrix representation (for X finite):

$$O_X = \begin{matrix} x_1 & x_2 & x_3 \\ (1 & 1 & 1) \end{matrix} \quad T_{X_a} = \begin{pmatrix} 1 & 0 & 0 \\ 1 & 3 & 0 \\ 1 & 0 & 3 \end{pmatrix} \quad T_{X_b} = \begin{pmatrix} 3 & 3 & 3 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}$$

Example

$\mathbb{K} = \mathbb{R}$, $A = \{a, b\}$



Convenient matrix representation (for X finite):

$$O_X = \begin{matrix} x_1 & x_2 & x_3 \\ (\ 1 & 1 & 1\ ) \end{matrix} \quad T_{X_a} = \begin{pmatrix} 1 & 0 & 0 \\ 1 & 3 & 0 \\ 1 & 0 & 3 \end{pmatrix} \quad T_{X_b} = \begin{pmatrix} 3 & 3 & 3 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}$$

Equivalence I : weighted bisimilarity

Definition

Let $(X, \langle o, t \rangle)$ be a weighted automaton. An equivalence relation $R \subseteq X \times X$ is a *weighted bisimulation* if for all $(x_1, x_2) \in R$, it holds that:

1. $o(x_1) = o(x_2)$,
2. $\forall a \in A,\ x' \in X,\ \sum_{x'' \in [x']_R} t(x_1)(a)(x'') = \sum_{x'' \in [x']_R} t(x_2)(a)(x'')$.

$[x]_R$: equivalence class of x with respect to R.

The definition only makes sense if the automaton has finite branching.

Equivalence I : weighted bisimilarity

Definition

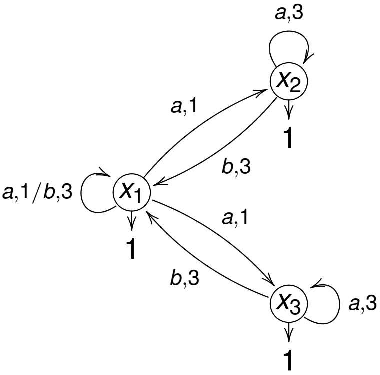
Let $(X, \langle o, t \rangle)$ be a weighted automaton. An equivalence relation $R \subseteq X \times X$ is a *weighted bisimulation* if for all $(x_1, x_2) \in R$, it holds that:

1. $o(x_1) = o(x_2)$,
2. $\forall a \in A,\ x' \in X,\ \sum_{x'' \in [x']_R} t(x_1)(a)(x'') = \sum_{x'' \in [x']_R} t(x_2)(a)(x'')$.

$[x]_R$: equivalence class of x with respect to R.

The definition only makes sense if the automaton has finite branching.

Example



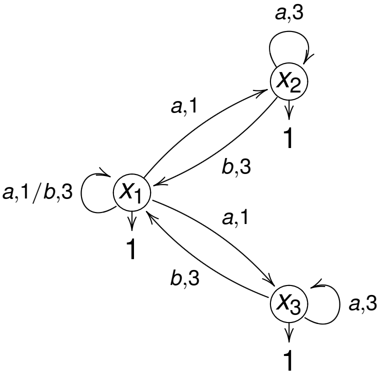
$R = \{\langle x_i, x_j \rangle \mid x_i, x_j \in X\}$

$\langle x_i, x_j \rangle \in R$

1. $o(x_i) = o(x_j) = 1$
2. $[x_i]_R = [x_j]_R = X$

We have to check for all states that the sum of the weights of all the transitions with the same label is the same.

Example



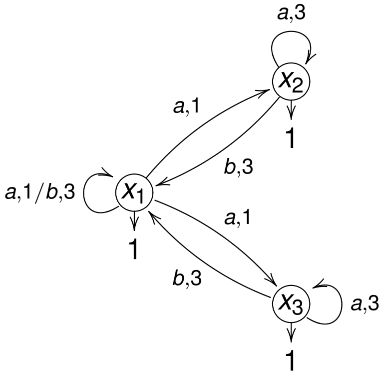
$$R = \{\langle x_i, x_j \rangle \mid x_i, x_j \in X\}$$

$\langle x_i, x_j \rangle \in R$

1. $o(x_i) = o(x_j) = 1$
2. $[x_i]_R = [x_j]_R = X$

We have to check for all states that the sum of the weights of all the transitions with the same label is the same.

Example



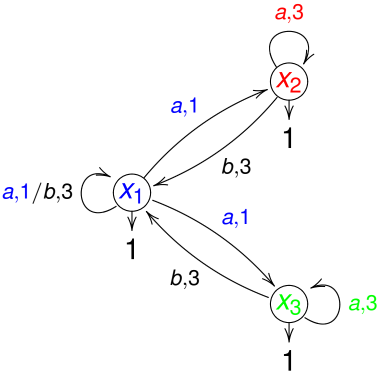
$$R = \{\langle x_i, x_j \rangle \mid x_i, x_j \in X\}$$

$\langle x_i, x_j \rangle \in R$

1. $o(x_i) = o(x_j) = 1$
2. $[x_i]_R = [x_j]_R = X$

   We have to check for all states that the sum of the weights of all the transitions with the same label is the same.

Example



$$R = \{\langle x_i, x_j\rangle \mid x_i, x_j \in X\}$$

$\langle x_i, x_j\rangle \in R$

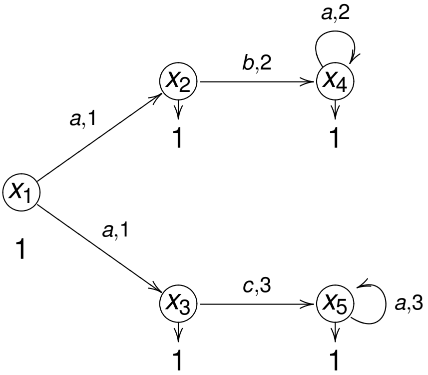
1. $o(x_i) = o(x_j) = 1$
2. $[x_i]_R = [x_j]_R = X$

   We have to check for all states that the sum of the weights of all the transitions with the same label is the same.

Equivalence II : weighted language equivalence

Given a word $w \in A^*$, we defined its weight in a state x of a weighted automaton $(X, \langle o, t \rangle)$ inductively:

1. $L(x)(\epsilon) = o(x)$
2. $L(x)(aw) = \sum_{x' \in X} t(x)(a)(x') \times L(w)(x')$

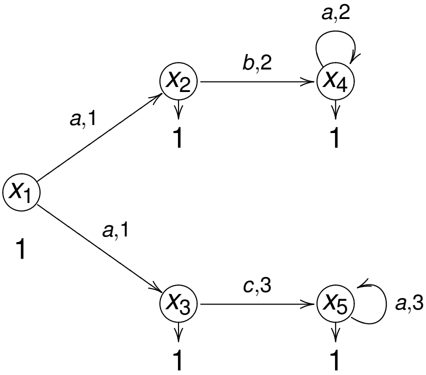


$L(x_1)(aba^n) = 2 \times 2^n$
$L(x_1)(aca^n) = 3 \times 3^n$
$L(x_1)(-) = 0$

Equivalence II : weighted language equivalence

Given a word $w \in A^*$, we defined its weight in a state x of a weighted automaton $(X, \langle o, t \rangle)$ inductively:

1. $L(x)(\epsilon) = o(x)$
2. $L(x)(aw) = \sum_{x' \in X} t(x)(a)(x') \times L(w)(x')$



$L(x_1)(aba^n) = 2 \times 2^n$
$L(x_1)(aca^n) = 3 \times 3^n$
$L(x_1)(-) = 0$

Next. . .

- Model weighted automata as coalgebras;
- Show how to derive the equivalences above canonically.

A coalgebra primer

Definition (Coalgebra)

Given a functor $\mathcal{G}\colon C \to C$ on a category C, a $\mathcal{G}$-*coalgebra* is an object X in C together with an arrow $f\colon X \to \mathcal{G}X$.

For many categories and functors, such pair (X, f) represents a transition system, the type of which is determined by the functor $\mathcal{G}$.

- Deterministic automata $(X, \langle o, t\rangle\colon X \to 2 \times X^A)$ $\quad \mathcal{G}(X) = 2 \times X^A$
- Labeled transition systems $(X, t\colon X \to (\mathcal{P}_\omega X)^A)$ $\quad \mathcal{G}(X) = \mathcal{P}_\omega(X)^A$
- $\vdots$

A coalgebra primer

Definition (Coalgebra)

Given a functor $\mathcal{G}\colon C \to C$ on a category C, a $\mathcal{G}$-*coalgebra* is an object X in C together with an arrow $f\colon X \to \mathcal{G}X$.

For many categories and functors, such pair (X, f) represents a transition system, the type of which is determined by the functor $\mathcal{G}$.

- Deterministic automata $(X, \langle o, t\rangle\colon X \to 2 \times X^A)$ $\quad \mathcal{G}(X) = 2 \times X^A$
- Labeled transition systems $(X, t\colon X \to (\mathcal{P}_\omega X)^A)$ $\quad \mathcal{G}(X) = \mathcal{P}_\omega(X)^A$
- $\vdots$

A coalgebra primer

Definition (Coalgebra)

Given a functor $\mathcal{G}\colon C \to C$ on a category C, a $\mathcal{G}$-*coalgebra* is an object X in C together with an arrow $f\colon X \to \mathcal{G}X$.

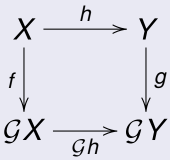
For many categories and functors, such pair (X, f) represents a transition system, the type of which is determined by the functor $\mathcal{G}$.

- Deterministic automata $(X, \langle o, t\rangle\colon X \to 2 \times X^A)$ $\quad \mathcal{G}(X) = 2 \times X^A$
- Labeled transition systems $(X, t\colon X \to (\mathcal{P}_\omega X)^A)$ $\quad \mathcal{G}(X) = \mathcal{P}_\omega(X)^A$
- $\vdots$

A coalgebra primer (cont'd)

Definition (Coalgebra homomorphism)

A $\mathcal{G}$*-homomorphism* from a $\mathcal{G}$-coalgebra (X, f) to a $\mathcal{G}$-coalgebra (Y, g) is an arrow $h : X \to Y$ preserving the transition structure, *i.e.*, such that the following diagram commutes.



Definition (Final coalgebra)

A $\mathcal{G}$-coalgebra (Ω, ω) is said to be *final* if for any $\mathcal{G}$-coalgebra (X, f) there exists a unique $\mathcal{G}$-homomorphism $[\![-]\!]_X^{\mathcal{G}} : X \to \Omega$.

A coalgebra primer (cont'd)

Definition (Coalgebra homomorphism)

A $\mathcal{G}$*-homomorphism* from a $\mathcal{G}$-coalgebra (X, f) to a $\mathcal{G}$-coalgebra (Y, g) is an arrow $h : X \to Y$ preserving the transition structure, *i.e.*, such that the following diagram commutes.

$$\begin{array}{ccc} X & \xrightarrow{h} & Y \\ {\scriptstyle f}\downarrow & & \downarrow{\scriptstyle g} \\ \mathcal{G}X & \xrightarrow[\mathcal{G}h]{} & \mathcal{G}Y \end{array}$$

Definition (Final coalgebra)

A $\mathcal{G}$-coalgebra (Ω, ω) is said to be *final* if for any $\mathcal{G}$-coalgebra (X, f) there exists a unique $\mathcal{G}$-homomorphism $[\![-]\!]_X^{\mathcal{G}} \colon X \to \Omega$.

A coalgebra primer (cont'd)

Final coalgebras can be viewed as the universe of all possible $\mathcal{G}$-*behaviours*: the unique homomorphism $[\![-]\!]_X^{\mathcal{G}}\colon X \to \Omega$ maps every state of a coalgebra X to a canonical representative of its behaviour.

Definition (Behavioural equivalence)

Two states $x_1, x_2 \in X$ are $\mathcal{G}$-*behaviourally equivalent* ($x_1 \approx_{\mathcal{G}} x_2$) iff $[\![x_1]\!]_X^{\mathcal{G}} = [\![x_2]\!]_X^{\mathcal{G}}$.

A coalgebra primer (cont'd)

Final coalgebras can be viewed as the universe of all possible $\mathcal{G}$-*behaviours*: the unique homomorphism $[\![-]\!]_X^{\mathcal{G}} \colon X \to \Omega$ maps every state of a coalgebra X to a canonical representative of its behaviour.

Definition (Behavioural equivalence)

Two states $x_1, x_2 \in X$ are $\mathcal{G}$-*behaviourally equivalent* ($x_1 \approx_{\mathcal{G}} x_2$) iff $[\![x_1]\!]_X^{\mathcal{G}} = [\![x_2]\!]_X^{\mathcal{G}}$.

Weighted automata as *Set*-coalgebras

Goal: Show a functor $\mathcal{W}\colon Set \to Set$ such that $\approx_{\mathcal{W}}$ coincides with weighted bisimilarity.

Definition (Field valuation Functor)

Let $\mathbb{K}$ be a field. The field valuation functor $\mathbb{K}_\omega^-\colon Set \to Set$ is defined as follows. For each set X, $\mathbb{K}_\omega^X$ is the set of functions from X to $\mathbb{K}$ with finite support. For each function $h\colon X \to Y$, $\mathbb{K}_\omega^h\colon \mathbb{K}_\omega^X \to \mathbb{K}_\omega^Y$ is the function mapping each $\varphi \in \mathbb{K}_\omega^X$ into $\varphi^h \in \mathbb{K}_\omega^Y$ defined, for all $y \in Y$, by

$$\varphi^h(y) = \sum_{x' \in h^{-1}(y)} \varphi(x')$$

Weighted automata as *Set*-coalgebras

Goal: Show a functor $\mathcal{W}\colon Set \to Set$ such that $\approx_{\mathcal{W}}$ coincides with weighted bisimilarity.

Definition (Field valuation Functor)

Let $\mathbb{K}$ be a field. The field valuation functor $\mathbb{K}_\omega^{-}\colon Set \to Set$ is defined as follows. For each set X, $\mathbb{K}_\omega^X$ is the set of functions from X to $\mathbb{K}$ with finite support. For each function $h\colon X \to Y$, $\mathbb{K}_\omega^h\colon \mathbb{K}_\omega^X \to \mathbb{K}_\omega^Y$ is the function mapping each $\varphi \in \mathbb{K}_\omega^X$ into $\varphi^h \in \mathbb{K}_\omega^Y$ defined, for all $y \in Y$, by

$$\varphi^h(y) = \sum_{x' \in h^{-1}(y)} \varphi(x')$$

Weighted automata as *Set*-coalgebras

Weighted automata are coalgebras for the functor $\mathcal{W} = \mathbb{K} \times (\mathbb{K}^{-}_{\omega})^{A} \colon \mathit{Set} \to \mathit{Set}$.

Every function $f \colon X \to \mathcal{W}(X)$ consists of a pair of functions $\langle o, t \rangle$ with $o \colon X \to \mathbb{K}$ and $t \colon X \to (\mathbb{K}^{X}_{\omega})^{A}$.

Canonical equivalence

Proposition

The functor $\mathcal{W}$ has a final coalgebra.

Theorem

Let $(X, \langle o, t \rangle)$ be a weighted automaton and let x_1, x_2 be two states in X. Then, x_1 and x_2 are weighted bisimilar iff $x_1 \approx_{\mathcal{W}} x_2$, i.e., $[\![x_1]\!]_X^{\mathcal{W}} = [\![x_2]\!]_X^{\mathcal{W}}$.

Canonical equivalence

Proposition

The functor $\mathcal{W}$ has a final coalgebra.

Theorem

Let $(X, \langle o, t \rangle)$ be a weighted automaton and let x_1, x_2 be two states in X. Then, x_1 and x_2 are weighted bisimilar iff $x_1 \approx_{\mathcal{W}} x_2$, i.e., $[\![x_1]\!]_X^{\mathcal{W}} = [\![x_2]\!]_X^{\mathcal{W}}$.

Weighted automata as *Vect*-coalgebras

Goal: Coalgebraic characterization of weighted language equivalence.

- Introduce linear weighted automata (as coalgebras);
- Show the canonical equivalence is weighted language equivalence;
- Show a construction from weighted automata to linear weighted automata.

Weighted automata as *Vect*-coalgebras

Goal: Coalgebraic characterization of weighted language equivalence.

- Introduce linear weighted automata (as coalgebras);
- Show the canonical equivalence is weighted language equivalence;
- Show a construction from weighted automata to linear weighted automata.

Linear weighted automata

Definition (LWA)

A *linear weighted automaton* (LWA, for short) with input alphabet A over the field $\mathbb{K}$ is a coalgebra for the functor $\mathcal{L} = \mathbb{K} \times id^A\colon \mathit{Vect} \to \mathit{Vect}$.

A LWA is a pair $(V, \langle o, t \rangle)$, where:

1. V is a vector space;
2. $o\colon V \to \mathbb{K}$ is a linear map associating to each state its output weight;
3. $t\colon V \to V^A$ is a linear map that for each input $a \in A$ associates a next state (i.e., a vector) in V.

Linear weighted automata

Definition (LWA)

A *linear weighted automaton* (LWA, for short) with input alphabet A over the field $\mathbb{K}$ is a coalgebra for the functor $\mathcal{L} = \mathbb{K} \times id^A \colon Vect \to Vect$.

A LWA is a pair $(V, \langle o, t \rangle)$, where:

1. V is a vector space;
2. $o \colon V \to \mathbb{K}$ is a linear map associating to each state its output weight;
3. $t \colon V \to V^A$ is a linear map that for each input $a \in A$ associates a next state (i.e., a vector) in V.

Equivalence

The final $\mathcal{L}$-coalgebra

- $\mathbb{K}^{A^*}$ carries a vector space structure: the sum of two languages $\sigma_1, \sigma_2 \in \mathbb{K}^{A^*}$ is the language $\sigma_1 + \sigma_2$ defined for each word $w \in A^*$ as $\sigma_1 + \sigma_2(w) = \sigma_1(w) + \sigma_2(w)$;

Equivalence

The final $\mathcal{L}$-coalgebra

- $\mathbb{K}^{A^*}$ carries a vector space structure: the sum of two languages $\sigma_1, \sigma_2 \in \mathbb{K}^{A^*}$ is the language $\sigma_1 + \sigma_2$ defined for each word $w \in A^*$ as $\sigma_1 + \sigma_2(w) = \sigma_1(w) + \sigma_2(w)$; the product of a language σ for a scalar $k \in \mathbb{K}$ is $k\sigma$ defined as $k\sigma(w) = k \cdot \sigma(w)$;

Equivalence

The final $\mathcal{L}$-coalgebra

- $\mathbb{K}^{A^*}$ carries a vector space structure: the sum of two languages $\sigma_1, \sigma_2 \in \mathbb{K}^{A^*}$ is the language $\sigma_1 + \sigma_2$ defined for each word $w \in A^*$ as $\sigma_1 + \sigma_2(w) = \sigma_1(w) + \sigma_2(w)$; the product of a language σ for a scalar $k \in \mathbb{K}$ is $k\sigma$ defined as $k\sigma(w) = k \cdot \sigma(w)$; the element 0 of $\mathbb{K}^{A^*}$ is the language mapping each word into the 0 of $\mathbb{K}$.

Equivalence

The final $\mathcal{L}$-coalgebra

- $\mathbb{K}^{A^*}$ carries a vector space structure: the sum of two languages $\sigma_1, \sigma_2 \in \mathbb{K}^{A^*}$ is the language $\sigma_1 + \sigma_2$ defined for each word $w \in A^*$ as $\sigma_1 + \sigma_2(w) = \sigma_1(w) + \sigma_2(w)$; the product of a language σ for a scalar $k \in \mathbb{K}$ is $k\sigma$ defined as $k\sigma(w) = k \cdot \sigma(w)$; the element 0 of $\mathbb{K}^{A^*}$ is the language mapping each word into the 0 of $\mathbb{K}$.
- The *empty function* $emp\colon \mathbb{K}^{A^*} \to \mathbb{K}$ and the *derivative function* $der\colon \mathbb{K}^{A^*} \to (\mathbb{K}^{A^*})^A$ are defined for all $\sigma \in \mathbb{K}^{A^*}$, $a \in A$ as

$$emp(\sigma) = \sigma(\epsilon) \quad der(\sigma)(a) = \sigma_a$$

  where $\sigma_a : A^* \to \mathbb{K}$ denotes the *a-derivative* of σ that is defined for all $w \in A^*$ as

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

Equivalence

The final $\mathcal{L}$-coalgebra

- $\mathbb{K}^{A^*}$ carries a vector space structure: the sum of two languages $\sigma_1, \sigma_2 \in \mathbb{K}^{A^*}$ is the language $\sigma_1 + \sigma_2$ defined for each word $w \in A^*$ as $\sigma_1 + \sigma_2(w) = \sigma_1(w) + \sigma_2(w)$; the product of a language σ for a scalar $k \in \mathbb{K}$ is $k\sigma$ defined as $k\sigma(w) = k \cdot \sigma(w)$; the element 0 of $\mathbb{K}^{A^*}$ is the language mapping each word into the 0 of $\mathbb{K}$.
- The *empty function* $emp\colon \mathbb{K}^{A^*} \to \mathbb{K}$ and the *derivative function* $der\colon \mathbb{K}^{A^*} \to (\mathbb{K}^{A^*})^A$ are defined for all $\sigma \in \mathbb{K}^{A^*}$, $a \in A$ as

$$emp(\sigma) = \sigma(\epsilon) \quad der(\sigma)(a) = \sigma_a$$

where $\sigma_a : A^* \to \mathbb{K}$ denotes the *a-derivative* of σ that is defined for all $w \in A^*$ as

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

Proposition

The maps $emp\colon \mathbb{K}^{A^*} \to \mathbb{K}$ and $der\colon \mathbb{K}^{A^*} \to (\mathbb{K}^{A^*})^A$ are linear.

Equivalence (cont'd)

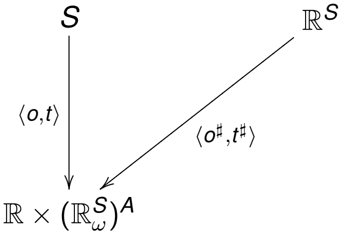
$(\mathbb{K}^{A^*}, \langle emp, der \rangle)$ is an $\mathcal{L}$-coalgebra.

Theorem (Finality)

From every $\mathcal{L}$-coalgebra $(V, \langle o, t \rangle)$ there exists a unique $\mathcal{L}$-homomorphism into $(\mathbb{K}^{A^*}, \langle emp, der \rangle)$.

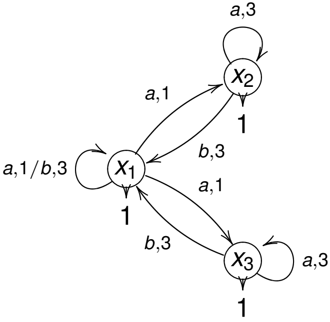
$$\begin{array}{ccc} V & \xrightarrow{[\![-]\!]_V^{\mathcal{L}}} & \mathbb{K}^{A^*} \\ {\scriptstyle \langle o,t \rangle}\downarrow & & \downarrow{\scriptstyle \langle emp,der \rangle} \\ \mathcal{L}(V) & \xrightarrow[\mathcal{L}([\![-]\!]_V^{\mathcal{L}})]{} & \mathcal{L}(\mathbb{K}^{A}) \end{array}$$

From weighted automata to linear weighted automata



$$o^\sharp\left(\begin{pmatrix} v_1 \\ \vdots \\ v_n \end{pmatrix}\right) = \sum v_i \times o(s_i) \qquad t^\sharp\left(\begin{pmatrix} v_1 \\ \vdots \\ v_n \end{pmatrix}\right)(a)(s_j) = \sum v_i \times t(s_i)(a)(s_j)$$

Example

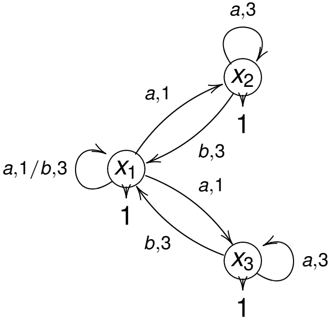


$$O_X = \begin{pmatrix} 1 & 1 & 1 \end{pmatrix} \quad T_{X_a} = \begin{pmatrix} 1 & 0 & 0 \\ 1 & 3 & 0 \\ 1 & 0 & 3 \end{pmatrix} \quad T_{X_b} = \begin{pmatrix} 3 & 3 & 3 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}$$

Example

$$V = \{k_1 x_1 + k_2 x_2 + k_3 x_3 \mid k_1, k_2, k_3 \in \mathbb{K}\}$$

$$o^\sharp(v) = o^\sharp(k_1 x_1 + k_2 x_2 + k_3 x_3) = k_1 + k_2 + k_3$$

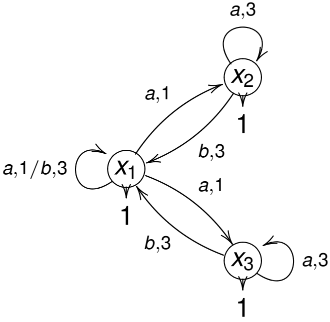


$$O_X = \begin{pmatrix} 1 & 1 & 1 \end{pmatrix} \quad T_{X_a} = \begin{pmatrix} 1 & 0 & 0 \\ 1 & 3 & 0 \\ 1 & 0 & 3 \end{pmatrix} \quad T_{X_b} = \begin{pmatrix} 3 & 3 & 3 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}$$

Example

$$V = \{k_1 x_1 + k_2 x_2 + k_3 x_3 \mid k_1, k_2, k_3 \in \mathbb{K}\}$$

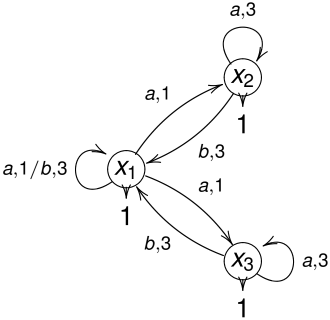
$$o^\sharp(v) = o^\sharp(k_1 x_1 + k_2 x_2 + k_3 x_3) = k_1 + k_2 + k_3 = O_X \times \begin{pmatrix} k_1 \\ k_2 \\ k_3 \end{pmatrix}$$



$$O_X = \begin{pmatrix} 1 & 1 & 1 \end{pmatrix} \quad T_{X_a} = \begin{pmatrix} 1 & 0 & 0 \\ 1 & 3 & 0 \\ 1 & 0 & 3 \end{pmatrix} \quad T_{X_b} = \begin{pmatrix} 3 & 3 & 3 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}$$

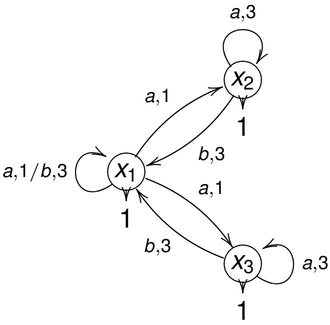
Example

$$
\begin{aligned}
V \quad &= \{k_1x_1 + k_2x_2 + k_3x_3 \mid k_1, k_2, k_3 \in \mathbb{K}\} \\
o^\sharp(v) \quad &= o^\sharp(k_1x_1 + k_2x_2 + k_3x_3) = k_1 + k_2 + k_3 \\
&= O_X \times \begin{pmatrix} k_1 \\ k_2 \\ k_3 \end{pmatrix} \\
t^\sharp(v)(a) &= k_1 t_X(x_1)(a) + k_2 t_X(x_2)(a) + k_3 t_X(x_3)(a) \\
&= k_1(x_1 + x_2 + x_3) + k_2 3x_2 + k_3 3x_3
\end{aligned}
$$



$$
O_X = \begin{pmatrix} 1 & 1 & 1 \end{pmatrix} \quad T_{X_a} = \begin{pmatrix} 1 & 0 & 0 \\ 1 & 3 & 0 \\ 1 & 0 & 3 \end{pmatrix} \quad T_{X_b} = \begin{pmatrix} 3 & 3 & 3 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}
$$

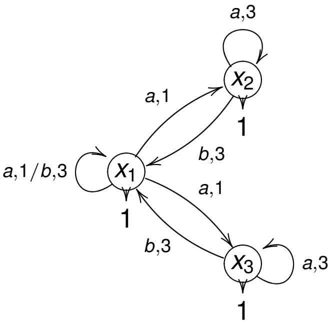
Example



$$
\begin{aligned}
V &= \{k_1x_1 + k_2x_2 + k_3x_3 \mid k_1, k_2, k_3 \in \mathbb{K}\} \\
o^\sharp(v) &= o^\sharp(k_1x_1 + k_2x_2 + k_3x_3) = k_1 + k_2 + k_3 \\
&= O_X \times \begin{pmatrix} k_1 \\ k_2 \\ k_3 \end{pmatrix} \\
t^\sharp(v)(a) &= k_1 t_X(x_1)(a) + k_2 t_X(x_2)(a) + k_3 t_X(x_3)(a) \\
&= k_1(x_1 + x_2 + x_3) + k_2 3x_2 + k_3 3x_3 \\
&= T_{X_a} \times \begin{pmatrix} k_1 \\ k_2 \\ k_3 \end{pmatrix}
\end{aligned}
$$

$$
O_X = \begin{pmatrix} 1 & 1 & 1 \end{pmatrix} \quad T_{X_a} = \begin{pmatrix} 1 & 0 & 0 \\ 1 & 3 & 0 \\ 1 & 0 & 3 \end{pmatrix} \quad T_{X_b} = \begin{pmatrix} 3 & 3 & 3 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}
$$

Example

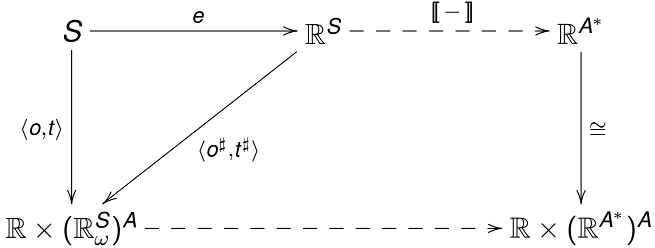


$$
\begin{aligned}
V &= \{k_1 x_1 + k_2 x_2 + k_3 x_3 \mid k_1, k_2, k_3 \in \mathbb{K}\} \\
o^\sharp(v) &= o^\sharp(k_1 x_1 + k_2 x_2 + k_3 x_3) = k_1 + k_2 + k_3 \\
&= O_X \times \begin{pmatrix} k_1 \\ k_2 \\ k_3 \end{pmatrix} \\
t^\sharp(v)(a) &= k_1 t_X(x_1)(a) + k_2 t_X(x_2)(a) + k_3 t_X(x_3)(a) \\
&= k_1(x_1 + x_2 + x_3) + k_2 3 x_2 + k_3 3 x_3 \\
&= T_{X_a} \times \begin{pmatrix} k_1 \\ k_2 \\ k_3 \end{pmatrix}
\end{aligned}
$$

$$
t_X^\sharp(v)(b) = k_1 3 x_1 + k_2 3 x_1 + k_3 3 x_1 = T_{X_a} \times \begin{pmatrix} k_1 \\ k_2 \\ k_3 \end{pmatrix}
$$

$$
O_X = \begin{pmatrix} 1 & 1 & 1 \end{pmatrix} \quad T_{X_a} = \begin{pmatrix} 1 & 0 & 0 \\ 1 & 3 & 0 \\ 1 & 0 & 3 \end{pmatrix} \quad T_{X_b} = \begin{pmatrix} 3 & 3 & 3 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}
$$

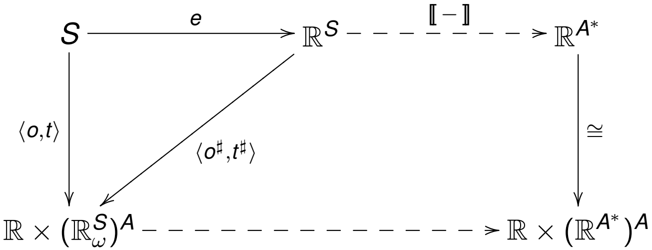
In a nutshell…



Lemma

Let $(X, \langle o, t \rangle)$ be a weighted automaton and $(\mathbb{K}^X_\omega, \langle o^\sharp, t^\sharp \rangle)$ be the corresponding linear weighted automaton. Then for all $x \in X$, $L(x) = [\![x]\!]^{\mathcal{L}}_{\mathbb{K}^X_\omega}$.

In a nutshell…



Lemma

Let $(X, \langle o, t\rangle)$ be a weighted automaton and $(\mathbb{K}^X_\omega, \langle o^\sharp, t^\sharp\rangle)$ be the corresponding linear weighted automaton. Then for all $x \in X$, $L(x) = [\![x]\!]^{\mathcal{L}}_{\mathbb{K}^X_\omega}$.

And now the connection with the previous talk. . .

- Solving linear systems of behavioural differential equations;
- Characterizing the final homomorphism by rational streams;
- Minimal automaton.

Conclusions

- Weighted automata characterized as coalgebras;
- Two canonical notions of equivalence;
- An algorithm for minimization.

In the paper:

- A different algorithm for minimization;
- Stream-like calculus for automata with input alphabet $|A| > 1$

Conclusions

- Weighted automata characterized as coalgebras;
- Two canonical notions of equivalence;
- An algorithm for minimization.

In the paper:

- A different algorithm for minimization;
- Stream-like calculus for automata with input alphabet $|A| > 1$