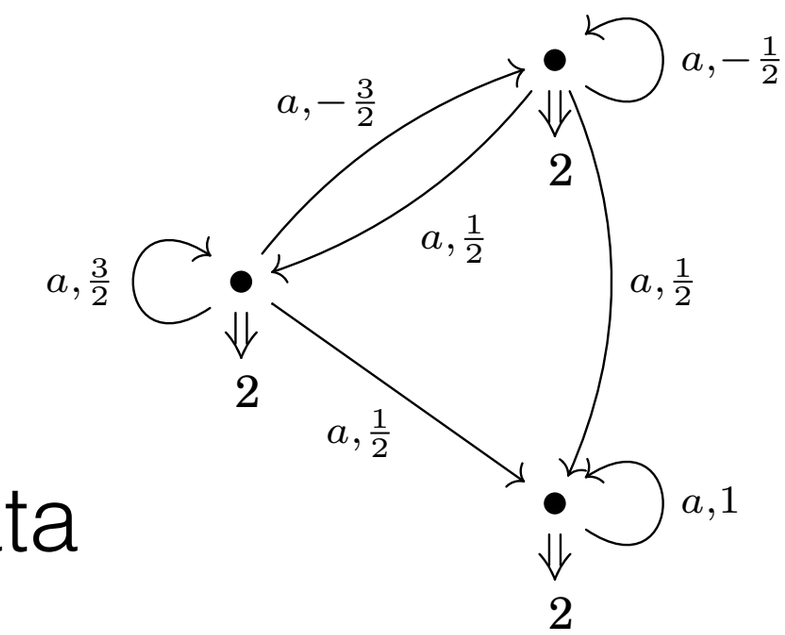
(Co)-Algebraic and Analytic Aspects of Weighted Automata Minimization and Equivalence, Part I.

Alexandra Silva

University College London

UCL ENGINEERING
Change the world

Weighted automata



- Generalization of classical automata
- Interesting applications: speech processing, image recognition, information theory, machine learning, …
- Fruitful research area since the 60's

SCHUTZENBERGER, M. P. 1961. On the definition of a family of automata. *Inform. and Control 4,* 2–3, 275–270.

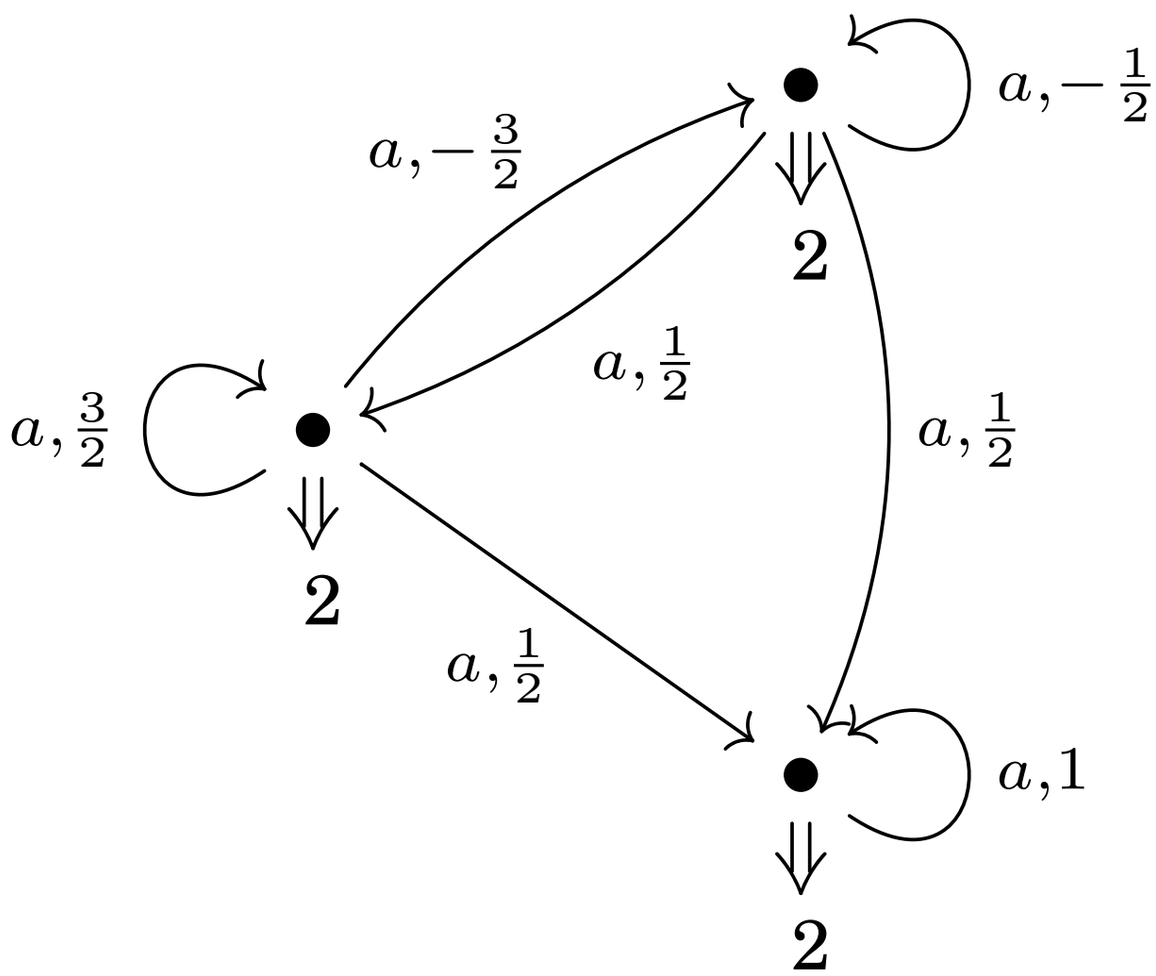
Recent (co)algebraic research

- Minimization [Boreale, Bonchi, Bonsangue & Silva 2011; Adamek, Bonchi, Huelbusch, Koenig, Milius, Silva 2012; Koenig&Kupper 2014; …]
- Equivalence — proof methods [Silva 2010, Milius 2010, Bonsangue, Milius, Silva 2013; Urabe & Hasuo 2014a,b; …]
- Decidability [Bonsangue, Milius, Silva 2013; Milius 2015; …]

This tutorial

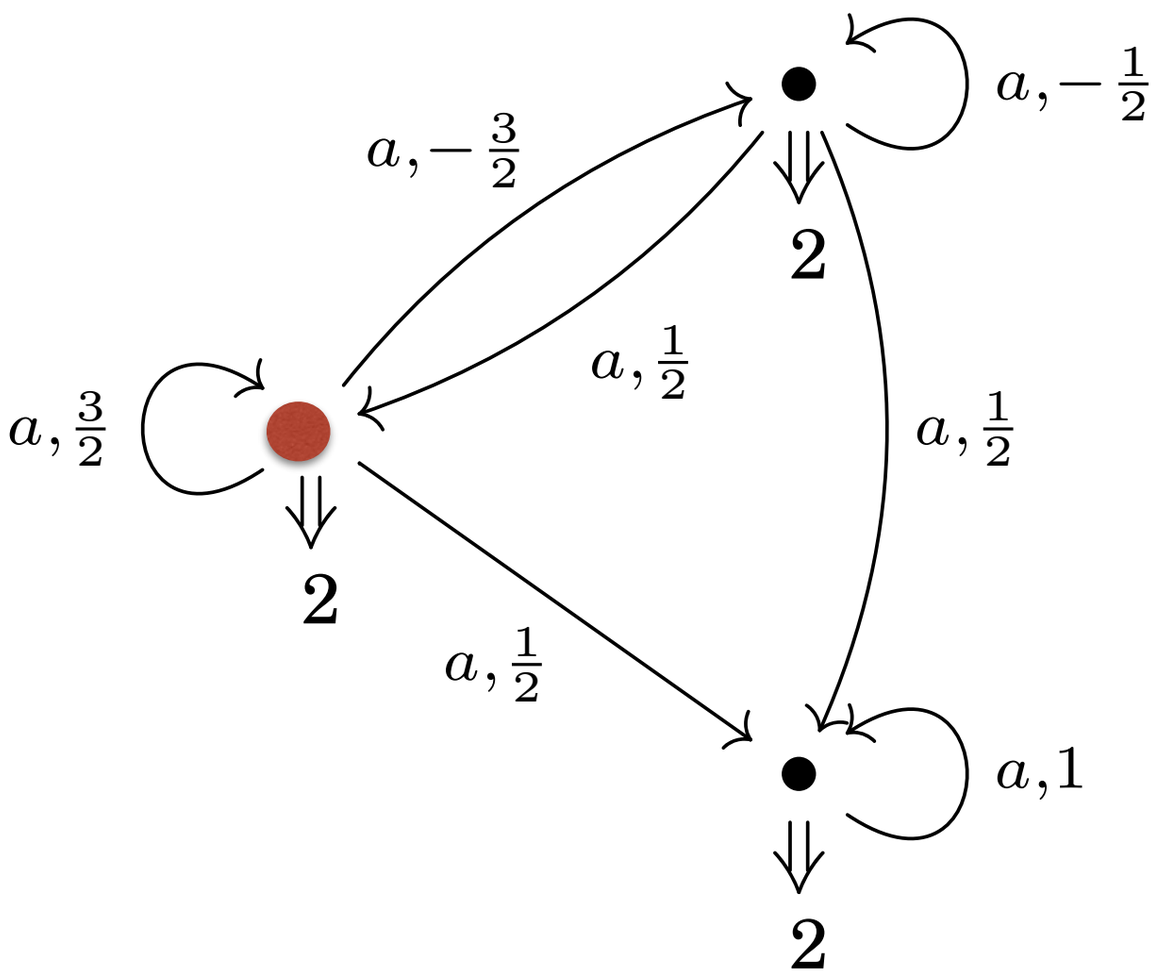
- Weighted automata as coalgebras
- Weighted bisimulation
- Language equivalence

Weighted automata intuitively



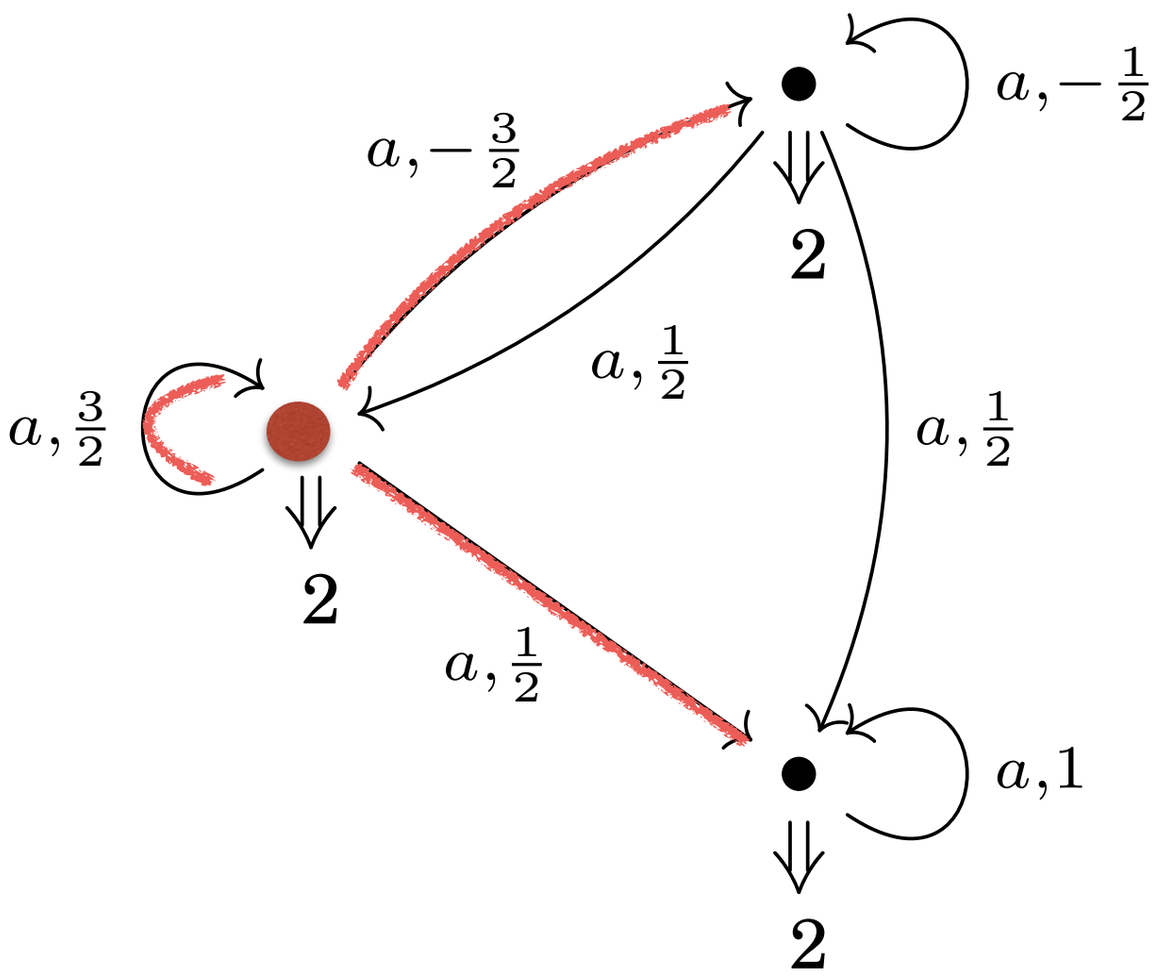
Intuition: takes a word, runs it through the automaton and computes a weight

Weighted automata intuitively



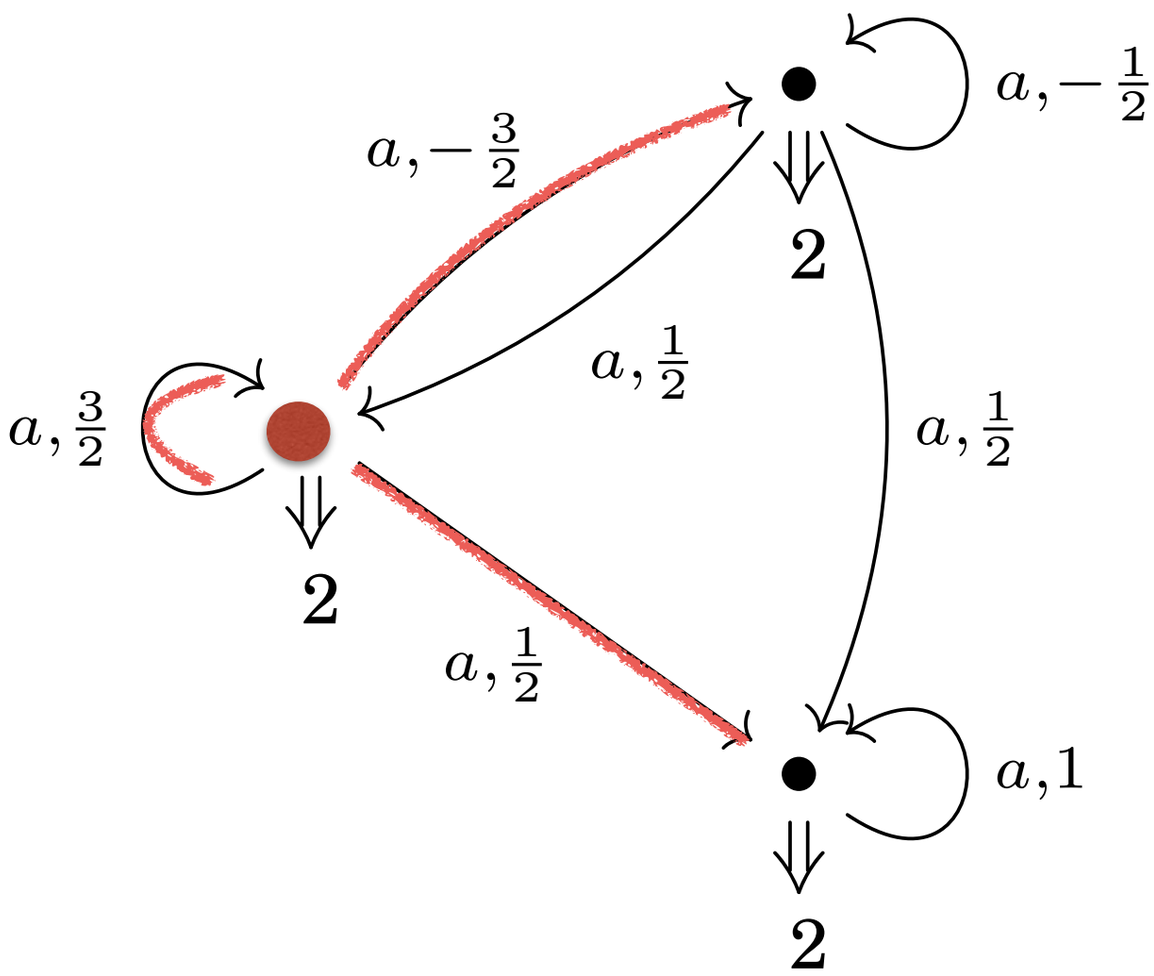
Intuition: takes a word, runs it through the automaton and computes a weight

Weighted automata intuitively



Intuition: takes a word, runs it through the automaton and computes a weight

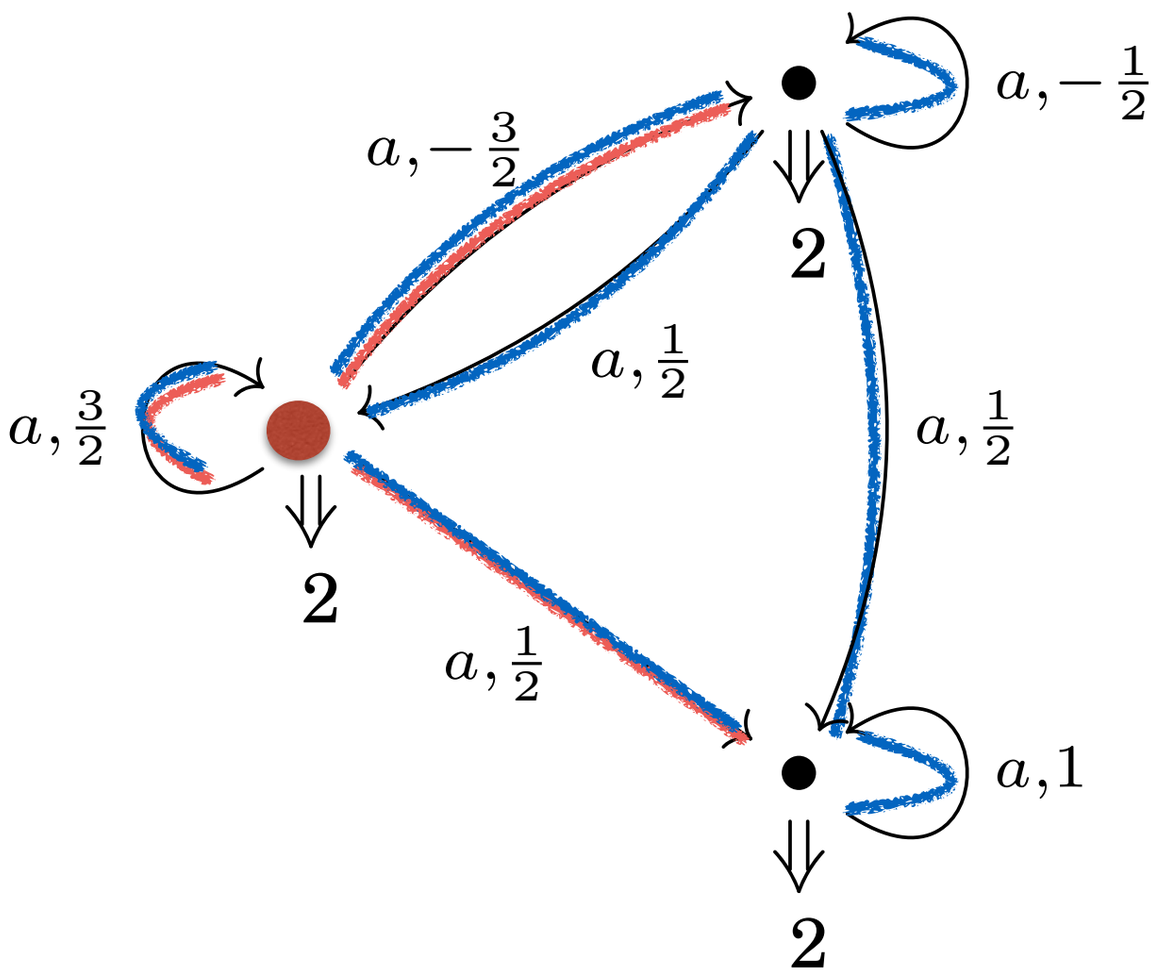
Weighted automata intuitively



$$a \mapsto \frac{3}{2} \times 2 - \frac{3}{2} \times 2 + \frac{1}{2} \times 2 = 1$$

Intuition: takes a word, runs it through the automaton and computes a weight

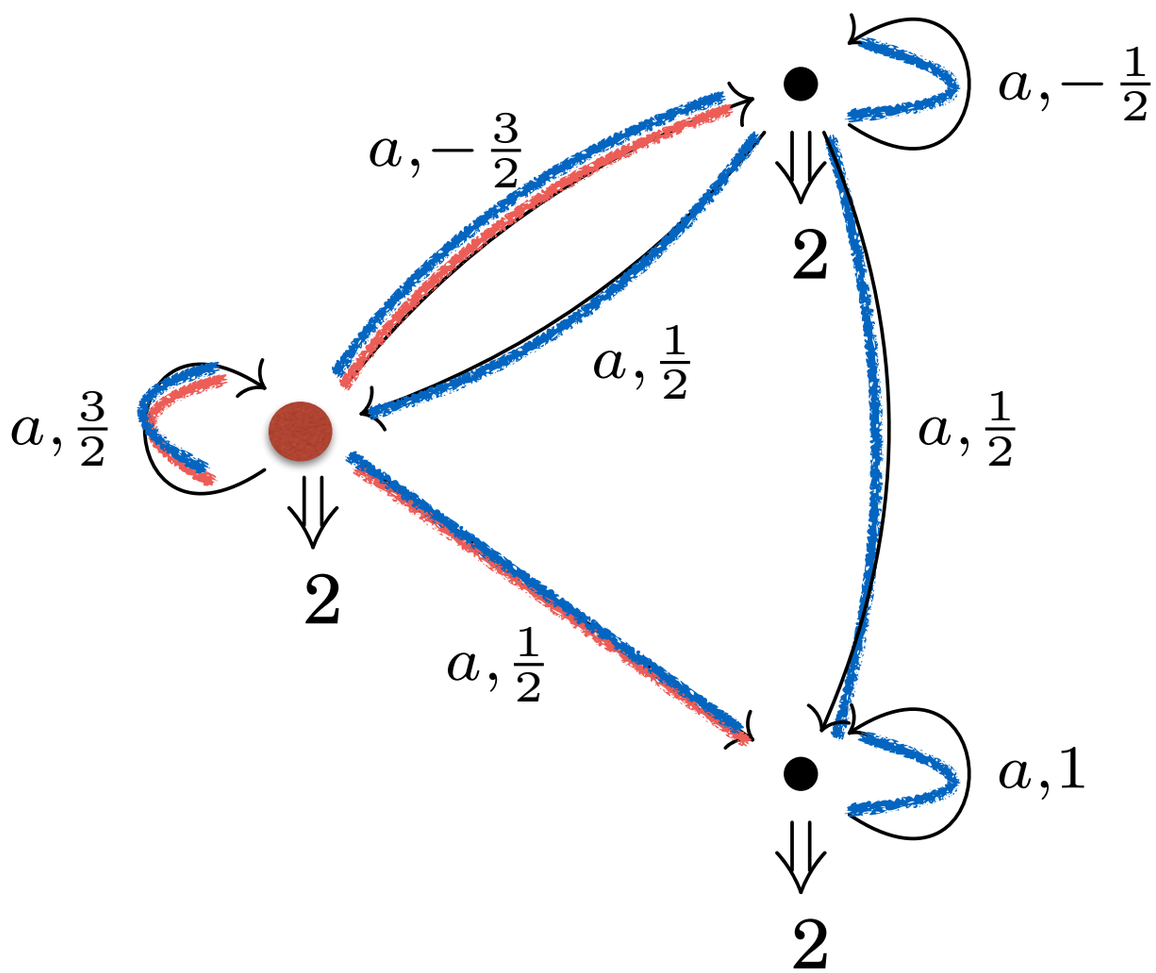
Weighted automata intuitively



$$a \mapsto \frac{3}{2} \times 2 - \frac{3}{2} \times 2 + \frac{1}{2} \times 2 = 1$$

Intuition: takes a word, runs it through the automaton and computes a weight

Weighted automata intuitively

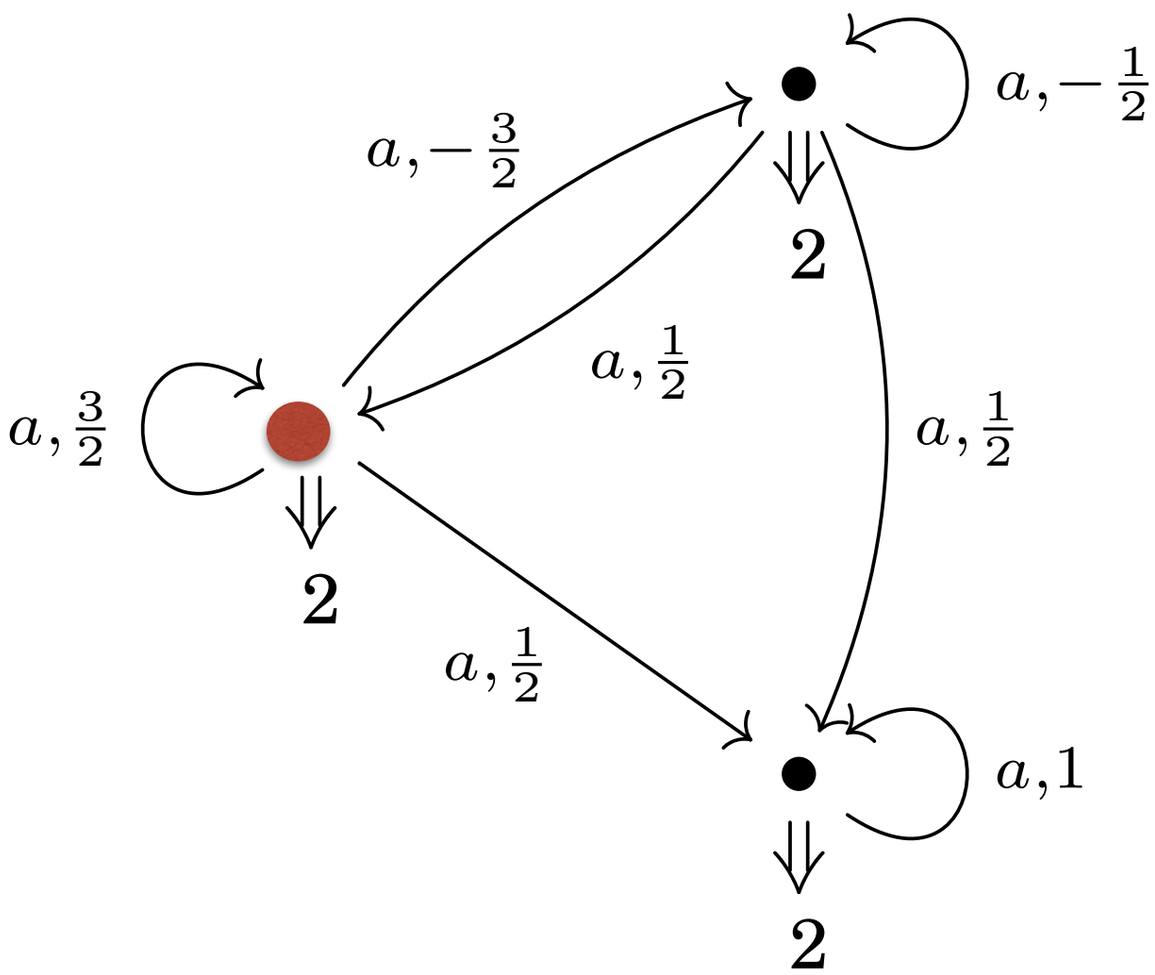


$$a \mapsto \frac{3}{2} \times 2 - \frac{3}{2} \times 2 + \frac{1}{2} \times 2 = 1$$

$$aa \mapsto \frac{3}{2}\left(\frac{3}{2} \times 2 - \frac{3}{2} \times 2 + \frac{1}{2} \times 2\right)$$
$$-\frac{3}{2} \times \left(-\frac{1}{2} \times 2 + \frac{1}{2} \times 2 + \frac{1}{2} \times 2\right)$$
$$+\frac{1}{2} \times (1 \times 2) = 1$$

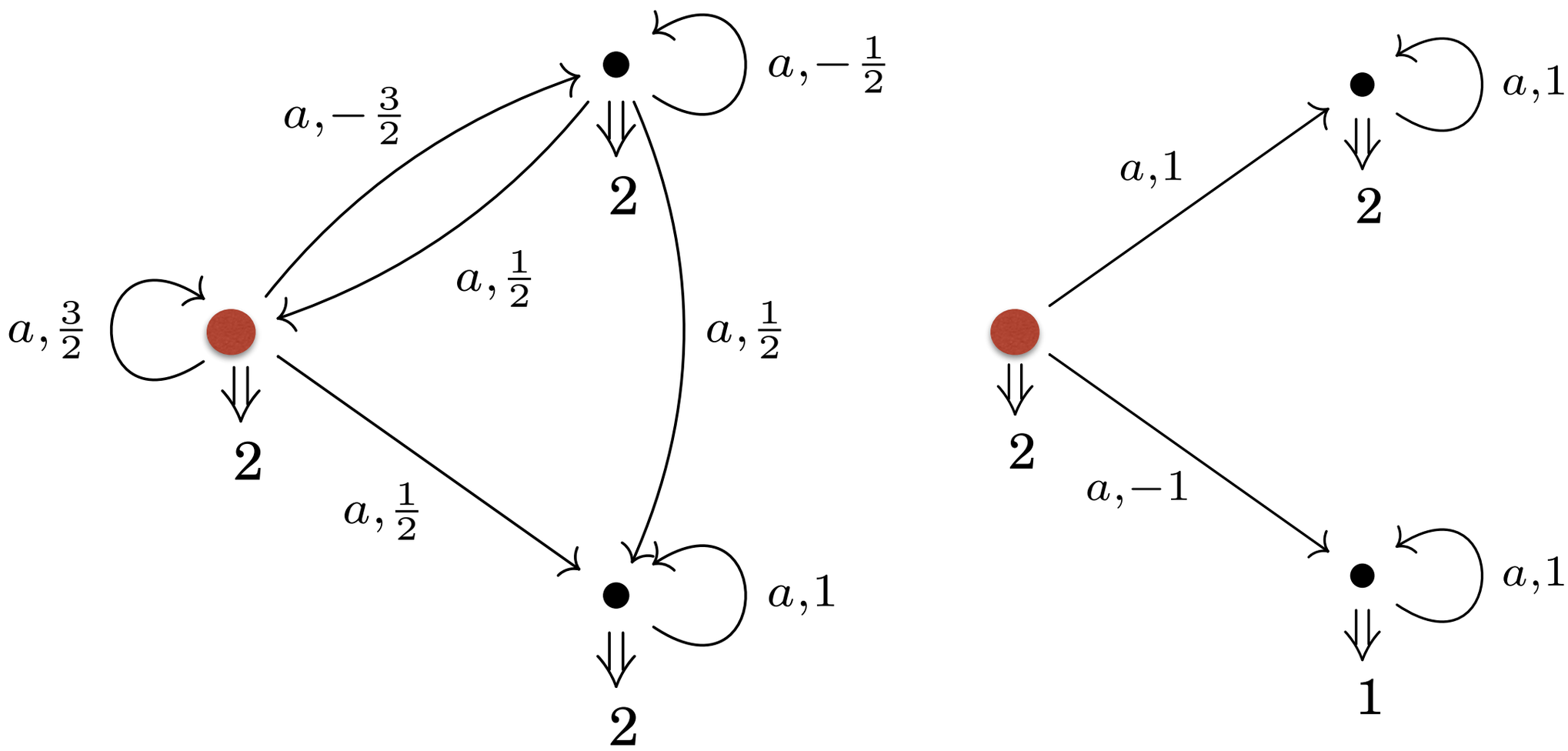
Intuition: takes a word, runs it through the automaton and computes a weight

Weighted automata intuitively



Conjecture: assigns 1 to every word a^n

Weighted automata intuitively



Conjecture: ● assigns 1 to every word a^n

Weighted automata - formally

Weights: semiring $\boxed{(\mathbb{S}, +, \cdot, 0, 1)}$

- Reals with usual addition and multiplication
- Natural Numbers with usual addition and multiplication
- Tropical semiring $(\mathbb{N} \cup \{\infty\}, \min, +, \infty, 0)$

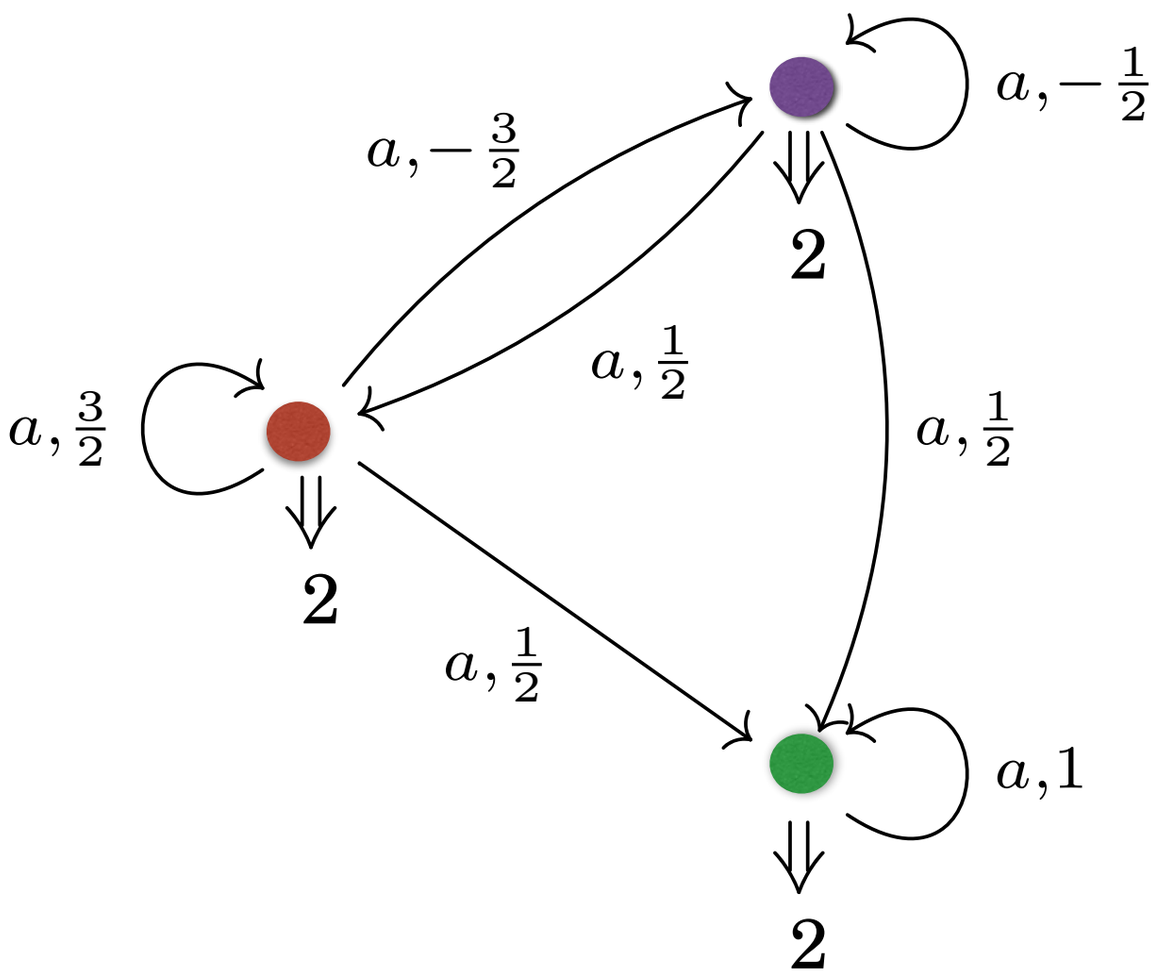
Weighted automata - formally

A weighted automaton over alphabet A is a tuple

$$(S, \alpha, (T)_{a \in A}, \beta)$$

- (finite) set of states
- input vector — initial *state*
- transition matrices
- output vector — final *state*

Example



$$\alpha = \begin{pmatrix} 1 & 0 & 0 \end{pmatrix}$$

$$T_a = \begin{array}{c} \\ \\ \\ \end{array}\begin{array}{c} \begin{array}{ccc} \quad & \quad & \quad \end{array} \\ \begin{pmatrix} 3/2 & 1/2 & 0 \\ -3/2 & -1/2 & 0 \\ 1/2 & 1/2 & 1 \end{pmatrix} \end{array}$$

$$\beta = \begin{pmatrix} 2 & 2 & 2 \end{pmatrix}$$

Language semantics

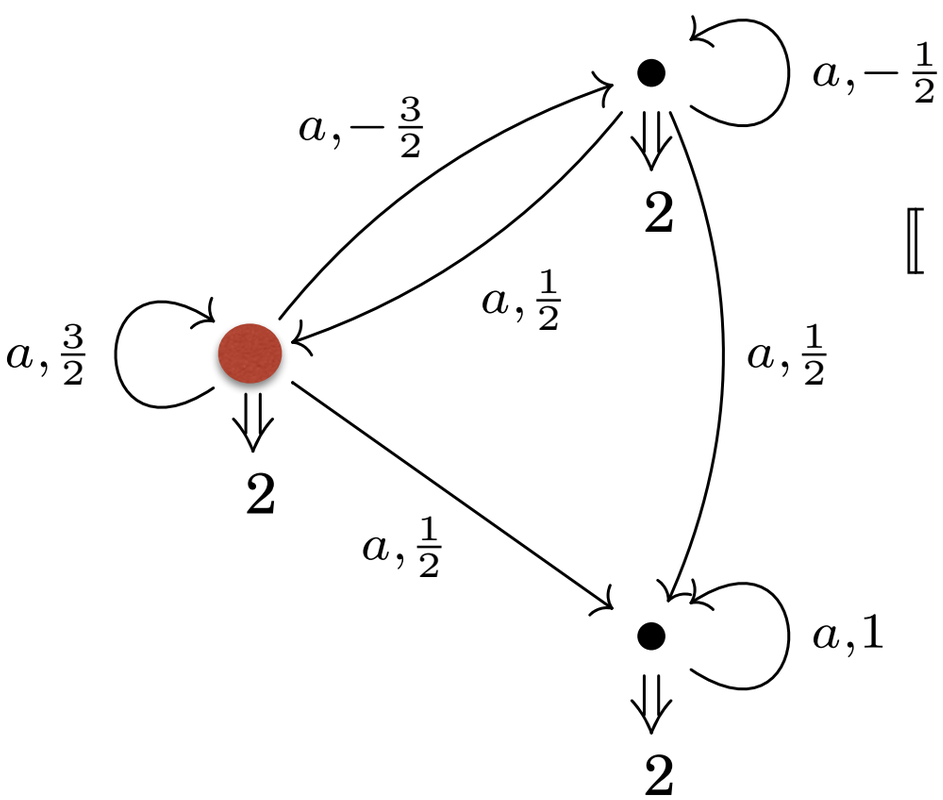
$$[\![-]\!] \colon S \to (A^* \to \mathbb{S})$$

$$[\![s]\!](a_1 \cdots a_k) = \beta \times T_{a_1} \times \cdots \times T_{a_k} \times \alpha^T$$

Language semantics

$$[\![-]\!] \colon S \to (A^* \to \mathbb{S})$$

$$[\![s]\!](a_1 \cdots a_k) = \beta \times T_{a_1} \times \cdots \times T_{a_k} \times \alpha^T$$



$$[\![\bullet]\!](aa) = \begin{pmatrix} 2 & 2 & 2 \end{pmatrix} \times \begin{pmatrix} 3/2 & 1/2 & 0 \\ -3/2 & -1/2 & 0 \\ 1/2 & 1/2 & 1 \end{pmatrix}^2 \times \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}$$

$$= 1$$

Language semantics

$$\begin{pmatrix} 3/2 & 1/2 & 0 \\ -3/2 & -1/2 & 0 \\ 1/2 & 1/2 & 1 \end{pmatrix}^n = \begin{pmatrix} 3/2 & 1/2 & 0 \\ -3/2 & -1/2 & 0 \\ 1/2 & 1/2 & 1 \end{pmatrix}$$

Language semantics

$$\begin{pmatrix} 3/2 & 1/2 & 0 \\ -3/2 & -1/2 & 0 \\ 1/2 & 1/2 & 1 \end{pmatrix}^n = \begin{pmatrix} 3/2 & 1/2 & 0 \\ -3/2 & -1/2 & 0 \\ 1/2 & 1/2 & 1 \end{pmatrix}$$

$$[\![\bullet]\!](a^n) = \begin{pmatrix} 2 & 2 & 2 \end{pmatrix} \times \begin{pmatrix} 3/2 & 1/2 & 0 \\ -3/2 & -1/2 & 0 \\ 1/2 & 1/2 & 1 \end{pmatrix}^n \times \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix} = 1$$

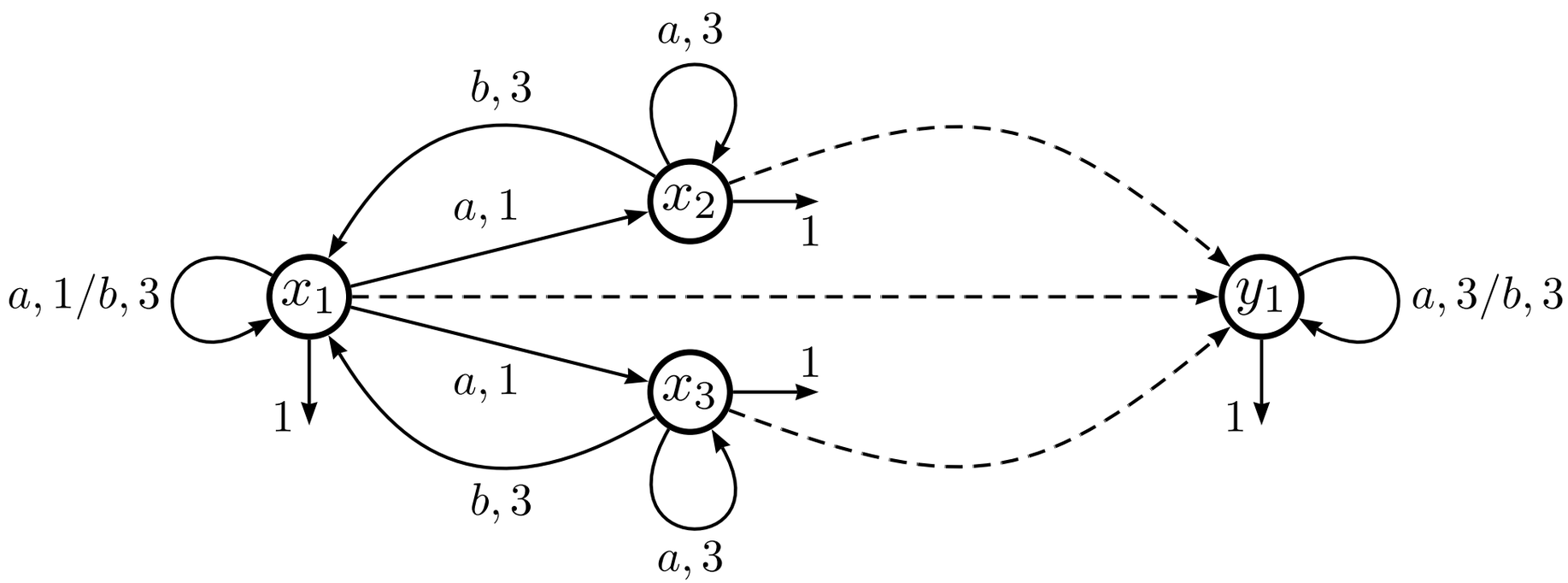
Another type of semantics

An equivalence relation $R \subseteq X \times X$ is a *weighted bisimulation* if for all $(x_1, x_2) \in R$, it holds that:

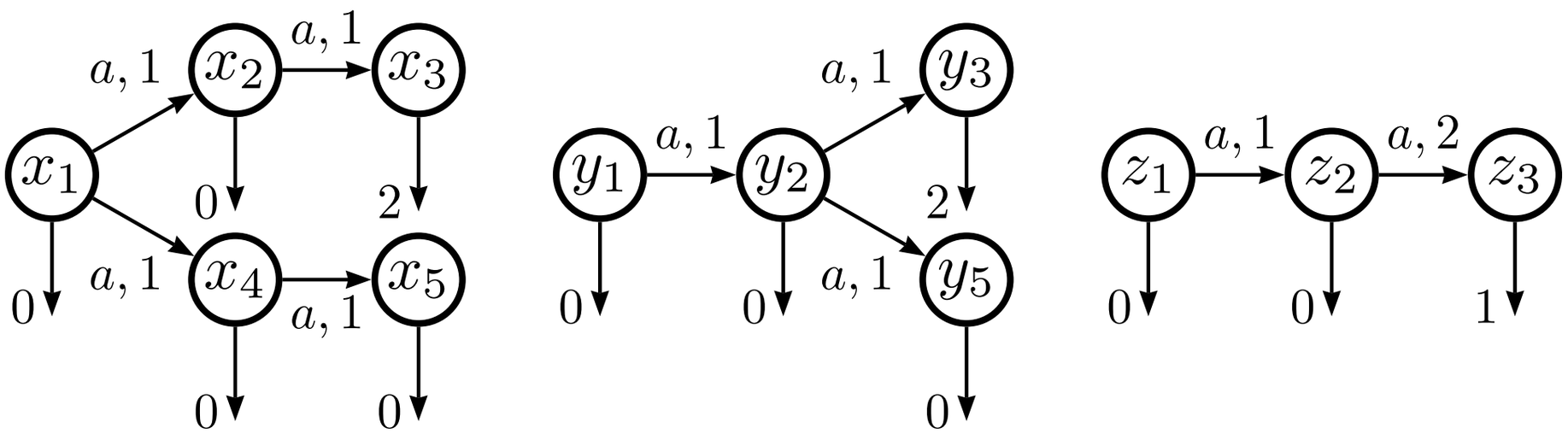
1. $\beta(x_1) = \beta(x_2)$,
2. $\forall a \in A,\ x' \in X,\ \sum_{x'' \in [x']_R} T_a(x_1, x'') = \sum_{x'' \in [x']_R} T_a(x_2, x'')$.

Peter Buchholz, Bisimulation relations for weighted automata, Theoret. Comput. Sci. 393 (1–3) (2008) 109–123.

Bisimilarity — example



Bisimilarity vs language



Weighted automata, coalgebraically I

$$\mathbb{S}^X = \{\varphi \mid \varphi \colon X \to \mathbb{S} \text{ with finite support}\}$$

$$h \colon X \to Y$$

$$\mathbb{S}^h(\varphi)(y) = \sum_{h(x)=y} \varphi(x)$$

Weighted automata, coalgebraically I

A weighted automaton is a coalgebra for the functor

$$\mathcal{W}(X) = \mathbb{S} \times (\mathbb{S}^X)^A$$

Weighted automata, coalgebraically I

A weighted automaton is a coalgebra for the functor

$$\mathcal{W}(X) = \mathbb{S} \times (\mathbb{S}^X)^A$$

$$X \to \mathbb{S} \times (\mathbb{S}^X)^A \quad \begin{cases} \beta \colon X \to \mathbb{S} \\ T_a \colon X \to \mathbb{S}^X, \quad \forall a \in A \end{cases}$$

Weighted automata, coalgebraically I

A weighted automaton is a coalgebra for the functor

$$\mathcal{W}(X) = \mathbb{S} \times (\mathbb{S}^X)^A$$

Coalgebraic modelling does not include the initial vector

$$X \to \mathbb{S} \times (\mathbb{S}^X)^A$$

$$T_a \colon X \to \mathbb{S}^X, \quad \forall a \in A$$

Some facts about $\mathcal{W}$

$\mathcal{W}$ is a bounded functor (Gumm'01)

There exists a final $\mathcal{W}$ coalgebra

$\mathcal{W}$ does not preserve weak pullbacks

Behavioral equivalence

$$\begin{tikzcd} X \arrow[r, dotted, "!"] \arrow[d, "<\beta,T>"'] & \Omega \arrow[d, "\omega"] \\ \mathcal{W}(X) \arrow[r, dotted, "\mathcal{W}(!)"] & \mathcal{W}(\Omega) \end{tikzcd}$$

$$x \approx y \iff !(x) = !(y)$$

Theorem

x is (Bucholz)*weighted bisimilar* to y iff $x \approx y$

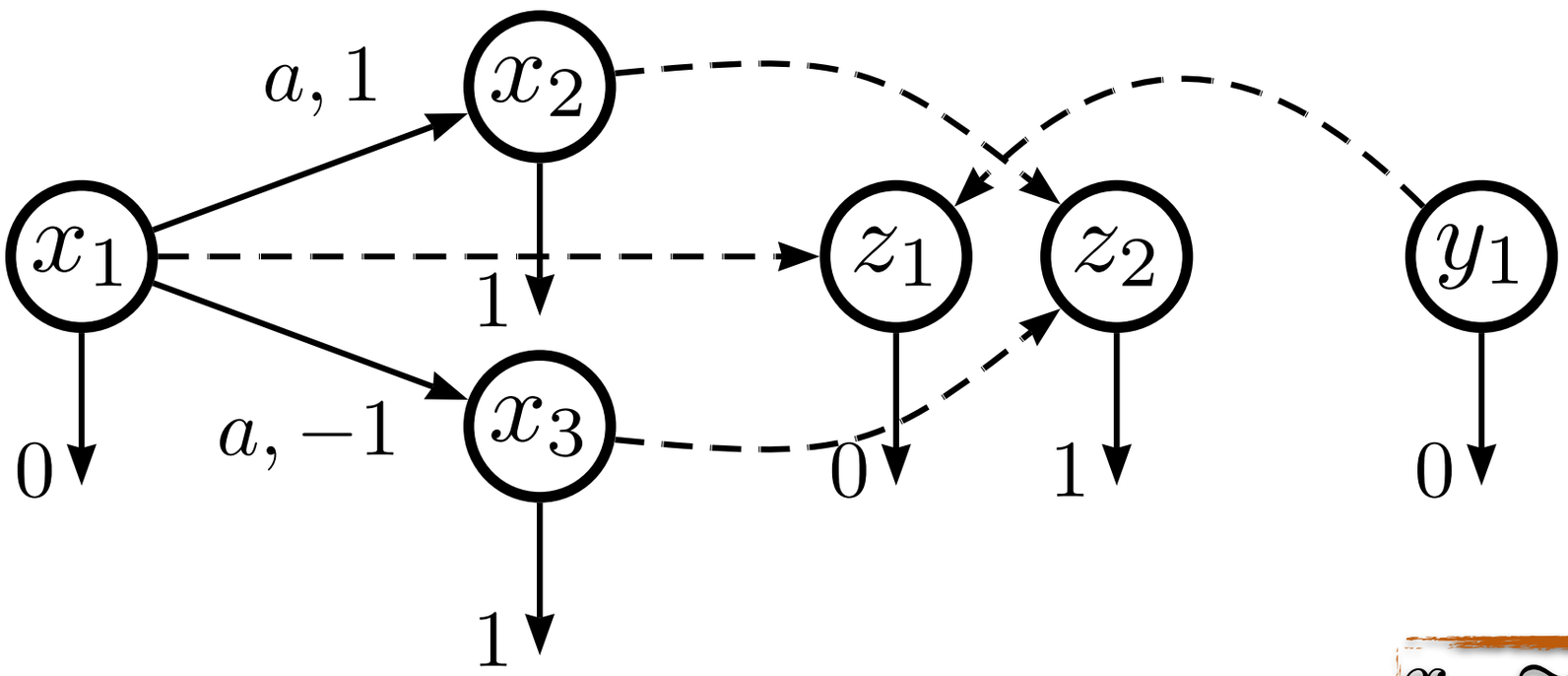
How about Aczel-Mendler bisimilarity?

$$
\begin{tikzcd}
X \arrow[d, "{<\beta_X, T_X>}"'] & R \arrow[l, "\pi_1"'] \arrow[r, "\pi_2"] \arrow[d, "{<\beta_R, T_R>}"'] & Y \arrow[d, "{<o_Y, t_Y>}"'] \\
\mathcal{W}(X) & \mathcal{W}(R) \arrow[l, "\mathcal{W}(\pi_1)"] \arrow[r, "\mathcal{W}(\pi_2)"'] & \mathcal{W}(Y)
\end{tikzcd}
$$

Notation: x and y are AM-bisimilar iff there exists *R* as above;

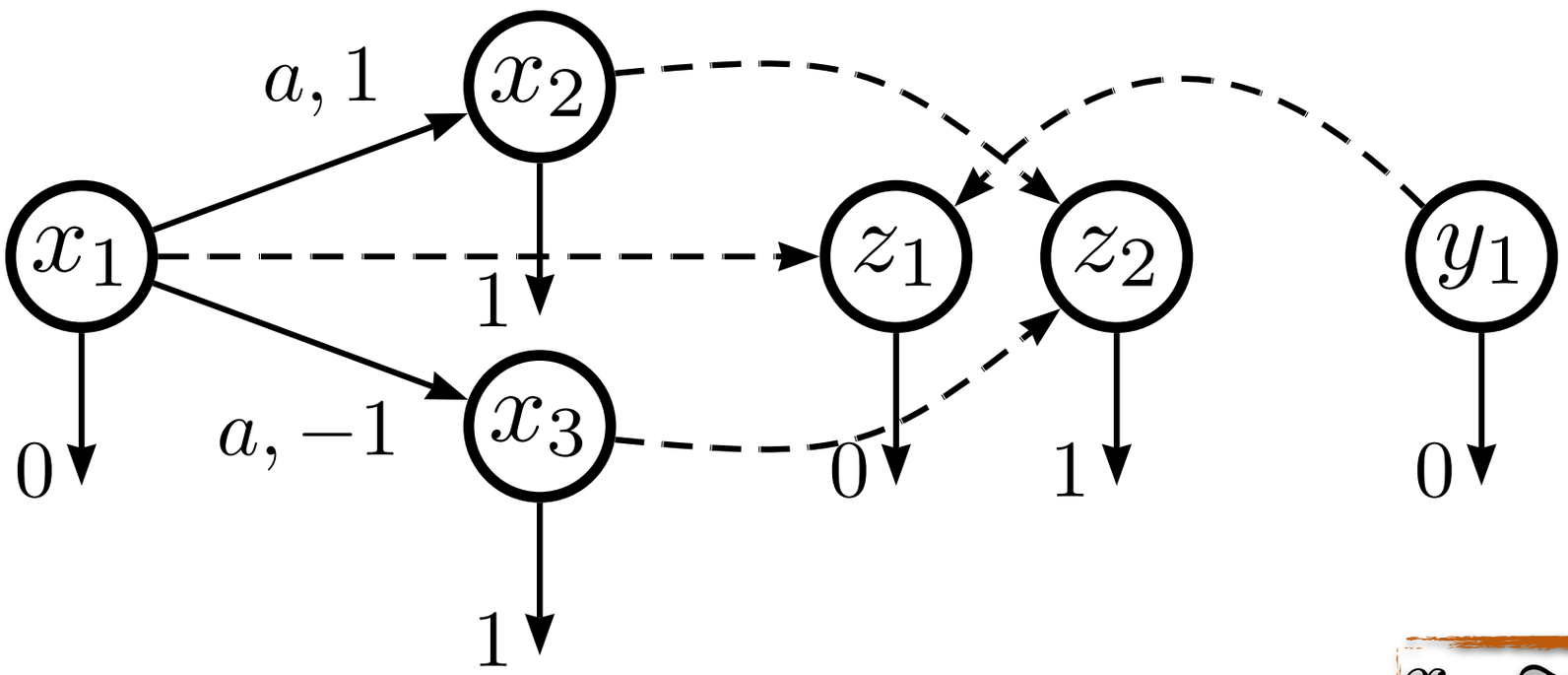
$$x \sim y$$

Bisimilarity vs Beh. equiv



$x_1 \approx x_2$ but $x_1 \not\sim y_1$

Bisimilarity vs Beh. equiv



$x_1 \approx x_2$ but $x_1 \not\sim y_1$

- Transitions with negative numbers play a key role
- Similar examples can be constructed for commutative monoids that are not zero-sum free (Gumm' 01)

Weighted automata, coalgebraically II

For simplicity let us take $\mathbb{S} = \mathbb{F}$ (field).

$\mathbf{Vect}$ — finite dimensional vector spaces and linear maps

Linear weighted automata: coalgebra in $\mathbf{Vect}$ for the functor

$$\mathcal{L}(V) = \mathbb{F} \times V^A$$

Semantics of LWA

$$\gamma : \mathbb{F}^{A^*} \to \mathbb{F} \qquad \gamma(f) = f(\varepsilon)$$

$$\mathcal{T}_a : \mathbb{F}^{A^*} \to \mathbb{F}^{A^*} \qquad T_a(f)(w) = f(aw)$$

$$< \gamma, \mathcal{T} >: \mathbb{F}^{A^*} \to \mathcal{L}(\mathbb{F}^{A^*})$$

Semantics of LWA

$< \gamma, \mathcal{T} >: \mathbb{F}^{A^*} \to \mathcal{L}(\mathbb{F}^{A^*})$ is the final coalgebra in $\mathbf{Vect}$

$$\begin{tikzcd}
V \arrow[r, dotted, "\mathbf{1}"] \arrow[d, "<\beta,T>"'] & \mathbb{F}^{A^*} \arrow[d, "<\gamma,\mathcal{T}>"] \\
\mathcal{L}(V) \arrow[r, dotted] & \mathcal{L}(\mathbb{F}^{A^*})
\end{tikzcd}$$

Semantics of LWA

$< \gamma, \mathcal{T} >: \mathbb{F}^{A^*} \to \mathcal{L}(\mathbb{F}^{A^*})$ is the final coalgebra in $\mathbf{Vect}$

$$\begin{tikzcd}
V \arrow[r, dotted, "\mathbf{l}"] \arrow[d, "<\beta,T>"'] & \mathbb{F}^{A^*} \arrow[d, "<\gamma,\mathcal{T}>"] \\
\mathcal{L}(V) \arrow[r, dotted] & \mathcal{L}(\mathbb{F}^{A^*})
\end{tikzcd}$$

From the commutativity one can compute for $\alpha \in V$

$$\mathbf{l}(\alpha)(a_1 \cdots a_k) = \beta \times T_{a_1} \times \cdots \times T_{a_k} \times \alpha^T$$

Semantics of LWA

$< \gamma, \mathcal{T} >: \mathbb{F}^{A^*} \to \mathcal{L}(\mathbb{F}^{A^*})$ is the final coalgebra in $\mathbf{Vect}$

$$\begin{tikzcd}
V \arrow[r, dotted, "\mathbf{l}"] \arrow[d, "<\beta,T>"'] & \mathbb{F}^{A^*} \arrow[d, "<\gamma,\mathcal{T}>"] \\
\mathcal{L}(V) \arrow[r, dotted] & \mathcal{L}(\mathbb{F}^{A^*})
\end{tikzcd}$$

From the commutativity one can compute for $\alpha \in V$

$$\mathbf{l}(\alpha)(a_1 \cdots a_k) = \beta \times T_{a_1} \times \cdots \times T_{a_k} \times \alpha^T$$

Language semantics

Mismatch

Weighted automata	**Linear weighted automata**
	$V \to \mathbb{S} \times V^A$
weighted bisimulation	language semantics
states	vectors

Mismatch

Weighted automata	**Linear weighted automata**
$X \to \mathbb{S} \times (\mathbb{S}^X)^A$	$V \to \mathbb{S} \times V^A$
weighted bisimulation	language semantics
states	vectors

Mismatch

Weighted automata	**Linear weighted automata**
	$V \to \mathbb{S} \times V^A$
weighted bisimulation	**language semantics**
states	vectors

Mismatch

Weighted automata	**Linear weighted automata**
$X \to \mathbb{S} \times (\mathbb{S}^X)^A$	$V \to \mathbb{S} \times V^A$
weighted bisimulation	**language semantics**
states	vectors

Combining both worlds

$$
\begin{tikzcd}
X \arrow[d, "<\beta,T>"'] \\
\mathbb{F} \times (\mathbb{F}^X)^A
\end{tikzcd}
$$

Combining both worlds

$$\begin{tikzcd} X \arrow[r, "e"] \arrow[d, "<\beta,T>"'] & \mathbb{F}^X \\ \mathbb{F} \times (\mathbb{F}^X)^A & \end{tikzcd}$$

Combining both worlds

$\mathbb{F}^X$ free vector space generated by X

$$\begin{tikzcd} X \arrow[r, "e"] \arrow[d, "<\beta,T>"'] & \mathbb{F}^X \\ \mathbb{F} \times (\mathbb{F}^X)^A & \end{tikzcd}$$

$$\begin{array}{c} x_1 \\ \vdots \\ x_n \end{array} \begin{pmatrix} r_1 \\ \vdots \\ r_n \end{pmatrix}$$

Combining both worlds

$\mathbb{F}^X$ free vector space generated by X

$$\begin{tikzcd}
X \arrow[r, "e"] \arrow[d, "<\beta,T>"'] & \mathbb{F}^X \arrow[dl, dotted, "<\hat{\beta},\hat{T}>"] \\
\mathbb{F} \times (\mathbb{F}^X)^A &
\end{tikzcd}$$

$$\begin{array}{c} x_1 \\ \vdots \\ x_n \end{array} \begin{pmatrix} r_1 \\ \vdots \\ r_n \end{pmatrix}$$

Combining both worlds

$\mathbb{F}^X$ free vector space generated by X

$$\begin{array}{c} x_1 \\ \vdots \\ x_n \end{array} \begin{pmatrix} r_1 \\ \vdots \\ r_n \end{pmatrix}$$

$$\begin{tikzcd} X \arrow[r, "e"] \arrow[d, "<\beta,T>"'] & \mathbb{F}^X \arrow[dl, dotted, "<\hat{\beta},\hat{T}>"] \\ \mathbb{F} \times (\mathbb{F}^X)^A & \end{tikzcd}$$

$<\hat{\beta},\hat{T}>$ is the unique linear extension of $<\beta,T>$

Combining both worlds

$\mathbb{F}^X$ free vector space generated by X

$$\begin{tikzcd}
X \arrow[r, "e"] \arrow[d, "<\beta,T>"'] & \mathbb{F}^X \arrow[dl, dotted, "<\hat{\beta},\hat{T}>"] \\
\mathbb{F} \times (\mathbb{F}^X)^A &
\end{tikzcd}$$

$<\hat{\beta},\hat{T}>$ is the unique linear extension of $<\beta,T>$

Combining both worlds

$\mathbb{F}^X$ free vector space generated by X

$$\begin{tikzcd}
X \arrow[r, "e"] \arrow[d, "<\beta,T>"'] & \mathbb{F}^X \arrow[dl, dotted, "<\hat{\beta},\hat{T}>"] \\
\mathbb{F} \times (\mathbb{F}^X)^A &
\end{tikzcd}$$

$$\hat{\beta}\begin{pmatrix} r_1 \\ \vdots \\ r_n \end{pmatrix} = \beta \times \begin{pmatrix} r_1 \\ \vdots \\ r_n \end{pmatrix}^T$$

$<\hat{\beta},\hat{T}>$ is the unique linear extension of $<\beta,T>$

Combining both worlds

$\mathbb{F}^X$ free vector space generated by X

$$\begin{tikzcd}
X \arrow[r, "e"] \arrow[d, "<\beta,T>"'] & \mathbb{F}^X \arrow[dl, dotted, "<\hat{\beta},\hat{T}>"] \\
\mathbb{F} \times (\mathbb{F}^X)^A &
\end{tikzcd}$$

$$\hat{T}_a \begin{pmatrix} r_1 \\ \vdots \\ r_n \end{pmatrix} = T_a \times \begin{pmatrix} r_1 \\ \vdots \\ r_n \end{pmatrix}^T$$

$<\hat{\beta},\hat{T}>$ is the unique linear extension of $<\beta,T>$

Semantics

$$
\begin{tikzcd}
X \arrow[r, "e"] \arrow[d, "<\beta,T>"'] & \mathbb{F}^X \arrow[r, dotted, "\mathbf{l}"] \arrow[dl, "<\hat{\beta},\hat{T}>"] & \mathbb{F}^{A^*} \arrow[d, "<\gamma,\mathcal{T}>"] \\
\mathbb{F} \times (\mathbb{F}^X)^A \arrow[rr, dotted] & & \mathcal{L}(\mathbb{F}^{A^*})
\end{tikzcd}
$$

Semantics

$$
\begin{tikzcd}
X \arrow[r, "e"] \arrow[d, "<\beta,T>"'] & \mathbb{F}^X \arrow[r, dotted, "\mathbf{l}"] \arrow[dl, "<\hat{\beta},\hat{T}>"] & \mathbb{F}^{A^*} \arrow[d, "<\gamma,\mathcal{T}>"] \\
\mathbb{F} \times (\mathbb{F}^X)^A \arrow[rr, dotted] & & \mathcal{L}(\mathbb{F}^{A^*})
\end{tikzcd}
$$

ker(**l**) is exactly weighted language equivalence

$$x_1 \approx x_2 \iff \mathbf{l}(e(x_1)) = \mathbf{l}(e(x_2))$$

Computing the kernel of $\mathbf{I}$

Three methods [Boreale, Bonchi, Bonsangue, Rutten & Silva' I&C 11]

- linear partition refinement — forward
- linear partition refinement — backward
- system of behavioural differential equations (rational streams)

Computing the kernel of $\mathbf{l}$

$$x_1 \approx x_2 \iff \mathbf{l}(e(x_1)) = \mathbf{l}(e(x_2))$$

Three methods [Boreale, Bonchi, Bonsangue, Rutten & Silva' I&C 11]

- linear partition refinement — forward
- linear partition refinement — backward
- system of behavioural differential equations (rational streams)

Linear partition refinement —forward

Consider the sequence $(R_i)_{i \geq 0}$ of subspaces of V defined inductively by

$$R_0 \;=\; ker(\beta) \qquad R_{i+1} \;= R_i \cap \bigcap_{a \in A} T_a(R_i)^{-1}$$

$T_a(R_i)^{-1}$ is the space $\{v \in V \mid T_a(v) \in R_i\}$.

Linear partition refinement —forward

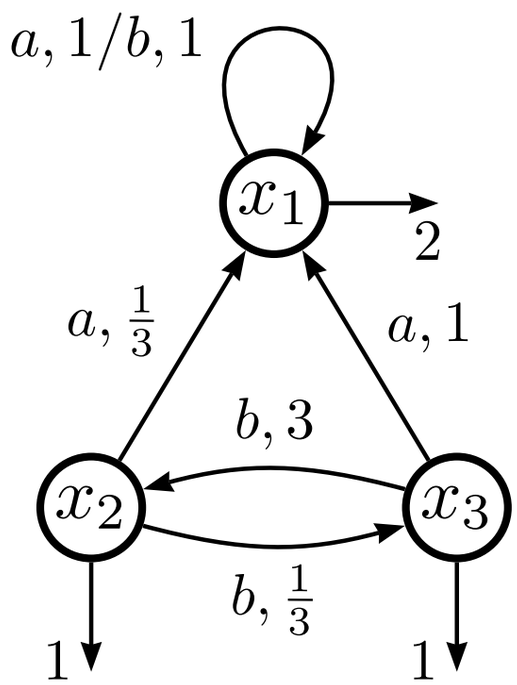
Consider the sequence $(R_i)_{i \geq 0}$ of subspaces of V defined inductively by

$$R_0 \;=\; ker(\beta) \qquad R_{i+1} \;= R_i \cap \bigcap_{a \in A} T_a(R_i)^{-1}$$

$T_a(R_i)^{-1}$ is the space $\{v \in V \mid T_a(v) \in R_i\}$.

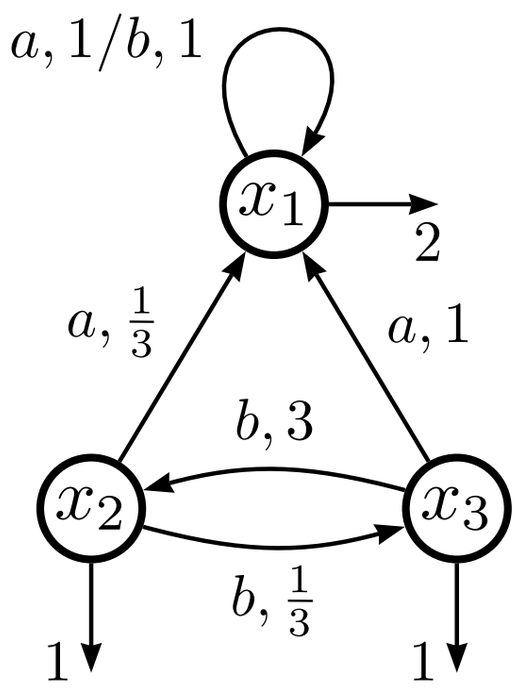
There is $j \leq \dim(V)$ such that $R_{j+1} = R_j$. The largest linear weighted bisimulation is $\approx_{\mathcal{L}} = R_j$.

Linear partition refinement example



$$\beta = \begin{pmatrix} 2 & 1 & 1 \end{pmatrix} \; T_a = \begin{pmatrix} 1 & \frac{1}{3} & 1 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} \; T_b = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 3 \\ 0 & \frac{1}{3} & 0 \end{pmatrix}$$

Linear partition refinement example

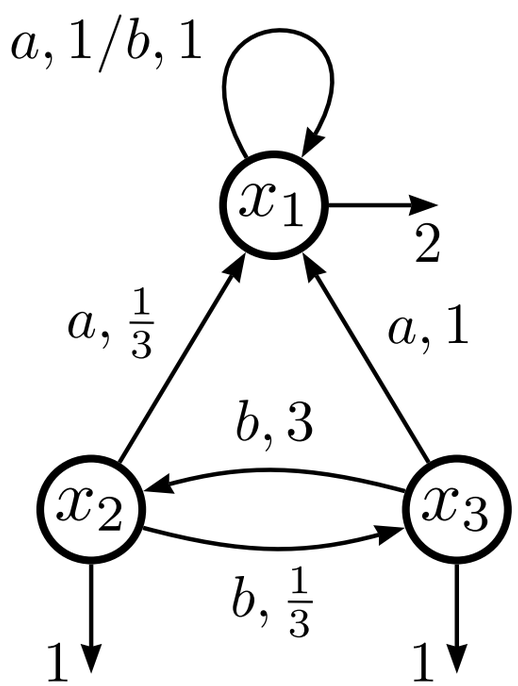


$$B_0 = \left\{ \begin{pmatrix} -\frac{1}{2} \\ 1 \\ 0 \end{pmatrix}, \begin{pmatrix} -\frac{1}{2} \\ 0 \\ 1 \end{pmatrix} \right\}.$$

Basis for ker(β)

$$\beta = \begin{pmatrix} 2 & 1 & 1 \end{pmatrix} \; T_a = \begin{pmatrix} 1 & \frac{1}{3} & 1 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} \; T_b = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 3 \\ 0 & \frac{1}{3} & 0 \end{pmatrix}$$

Linear partition refinement example



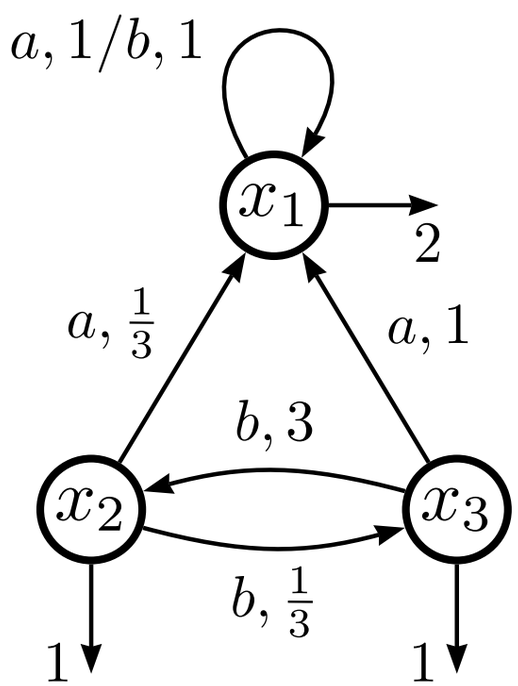
$$B_0 = \left\{ \begin{pmatrix} -\frac{1}{2} \\ 1 \\ 0 \end{pmatrix}, \begin{pmatrix} -\frac{1}{2} \\ 0 \\ 1 \end{pmatrix} \right\}.$$

Basis for ker(β)

$$B_1^a = \left\{ \begin{pmatrix} -\frac{1}{3} \\ 1 \\ 0 \end{pmatrix}, \begin{pmatrix} -1 \\ 0 \\ 1 \end{pmatrix} \right\} \quad \text{and} \quad B_1^b = \left\{ \begin{pmatrix} -\frac{1}{6} \\ 1 \\ 0 \end{pmatrix}, \begin{pmatrix} -\frac{3}{2} \\ 0 \\ 1 \end{pmatrix} \right\}.$$

$$\beta = \begin{pmatrix} 2 & 1 & 1 \end{pmatrix} \; T_a = \begin{pmatrix} 1 & \frac{1}{3} & 1 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} \; T_b = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 3 \\ 0 & \frac{1}{3} & 0 \end{pmatrix}$$

Linear partition refinement example



$$B_0 = \left\{ \begin{pmatrix} -\frac{1}{2} \\ 1 \\ 0 \end{pmatrix}, \begin{pmatrix} -\frac{1}{2} \\ 0 \\ 1 \end{pmatrix} \right\}.$$

Basis for ker(β)

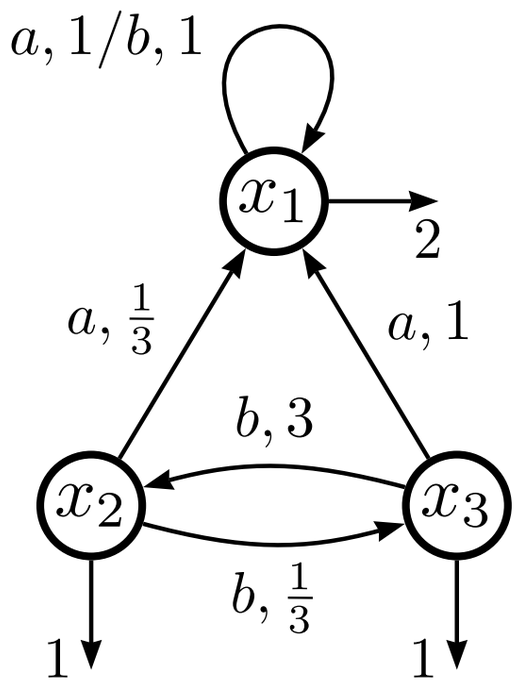
$$B_1^a = \left\{ \begin{pmatrix} -\frac{1}{3} \\ 1 \\ 0 \end{pmatrix}, \begin{pmatrix} -1 \\ 0 \\ 1 \end{pmatrix} \right\} \quad \text{and} \quad B_1^b = \left\{ \begin{pmatrix} -\frac{1}{6} \\ 1 \\ 0 \end{pmatrix}, \begin{pmatrix} -\frac{3}{2} \\ 0 \\ 1 \end{pmatrix} \right\}.$$

$$B_1 = \left\{ \begin{pmatrix} -2 \\ 3 \\ 1 \end{pmatrix} \right\}.$$

$$R_0 \cap t(R_0)(a)^{-1} \cap t(R_0)(b)^{-1}$$

$$\beta = \begin{pmatrix} 2 & 1 & 1 \end{pmatrix} \; T_a = \begin{pmatrix} 1 & \frac{1}{3} & 1 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} \; T_b = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 3 \\ 0 & \frac{1}{3} & 0 \end{pmatrix}$$

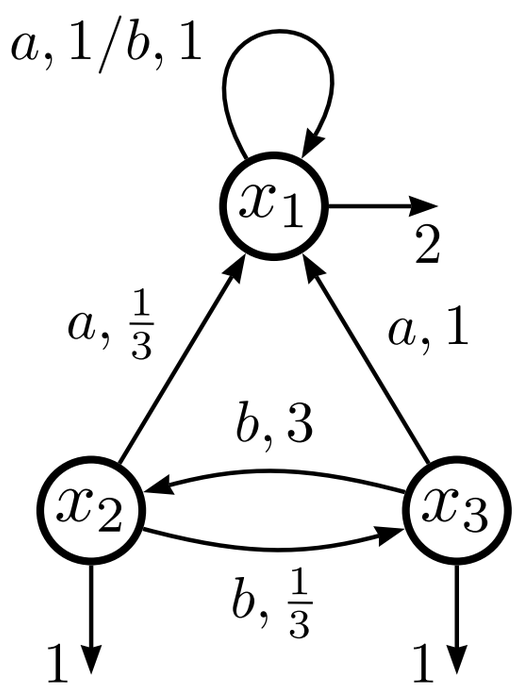
Linear partition refinement example



$$B_1 = \left\{ \begin{pmatrix} -2 \\ 3 \\ 1 \end{pmatrix} \right\}. \qquad R_0 \cap t(R_0)(a)^{-1} \cap t(R_0)(b)^{-1}$$

$$\beta = \begin{pmatrix} 2 & 1 & 1 \end{pmatrix} \; T_a = \begin{pmatrix} 1 & \frac{1}{3} & 1 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} \; T_b = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 3 \\ 0 & \frac{1}{3} & 0 \end{pmatrix}$$

Linear partition refinement example

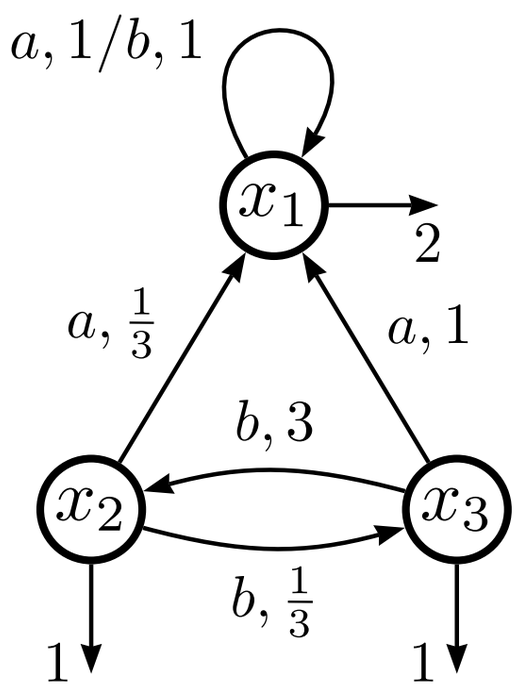


$$B_2^a = \left\{ \begin{pmatrix} -\frac{1}{3} \\ 1 \\ 0 \end{pmatrix}, \begin{pmatrix} -1 \\ 0 \\ 1 \end{pmatrix} \right\} \quad \text{and} \quad B_2^b = \left\{ \begin{pmatrix} -2 \\ 3 \\ 1 \end{pmatrix} \right\}.$$

$$B_1 = \left\{ \begin{pmatrix} -2 \\ 3 \\ 1 \end{pmatrix} \right\}. \qquad R_0 \cap t(R_0)(a)^{-1} \cap t(R_0)(b)^{-1}$$

$$\beta = \begin{pmatrix} 2 & 1 & 1 \end{pmatrix} \; T_a = \begin{pmatrix} 1 & \frac{1}{3} & 1 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} \; T_b = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 3 \\ 0 & \frac{1}{3} & 0 \end{pmatrix}$$

Linear partition refinement example



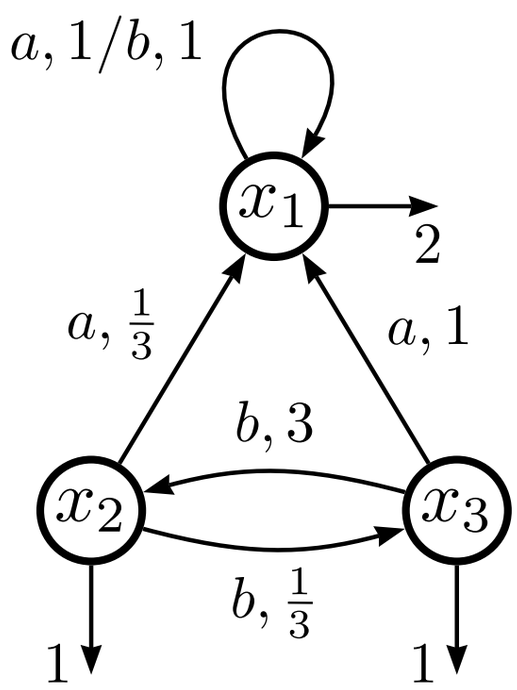
$$B_2^a = \left\{ \begin{pmatrix} -\frac{1}{3} \\ 1 \\ 0 \end{pmatrix}, \begin{pmatrix} -1 \\ 0 \\ 1 \end{pmatrix} \right\} \quad \text{and} \quad B_2^b = \left\{ \begin{pmatrix} -2 \\ 3 \\ 1 \end{pmatrix} \right\}.$$

$$B_2 = \left\{ \begin{pmatrix} -2 \\ 3 \\ 1 \end{pmatrix} \right\} \qquad R_1 \cap t(R_1)(a)^{-1} \cap t(R_0)(b)^{-1}$$

$$B_1 = \left\{ \begin{pmatrix} -2 \\ 3 \\ 1 \end{pmatrix} \right\}. \qquad R_0 \cap t(R_0)(a)^{-1} \cap t(R_0)(b)^{-1}$$

$$\beta = \begin{pmatrix} 2 & 1 & 1 \end{pmatrix} \; T_a = \begin{pmatrix} 1 & \frac{1}{3} & 1 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} \; T_b = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 3 \\ 0 & \frac{1}{3} & 0 \end{pmatrix}$$

Linear partition refinement example



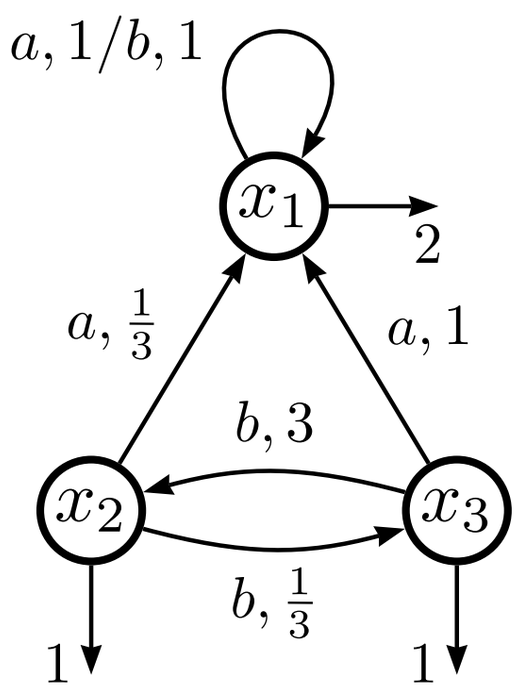
$$B_1 = \left\{ \begin{pmatrix} -2 \\ 3 \\ 1 \end{pmatrix} \right\}.$$

To check $v \approx w$ we need to check

$$v - w \in R_1$$

$$\beta = \begin{pmatrix} 2 & 1 & 1 \end{pmatrix} \; T_a = \begin{pmatrix} 1 & \frac{1}{3} & 1 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} \; T_b = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 3 \\ 0 & \frac{1}{3} & 0 \end{pmatrix}$$

Linear partition refinement example



$$B_1 = \left\{ \begin{pmatrix} -2 \\ 3 \\ 1 \end{pmatrix} \right\}.$$

To check $v \approx w$ we need to check

$$v - w \in R_1$$

e.g. $$x_1 \approx \frac{3}{2} x_1 + \frac{1}{2} x_3$$

$$\beta = \begin{pmatrix} 2 & 1 & 1 \end{pmatrix} \; T_a = \begin{pmatrix} 1 & \frac{1}{3} & 1 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} \; T_b = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 3 \\ 0 & \frac{1}{3} & 0 \end{pmatrix}$$

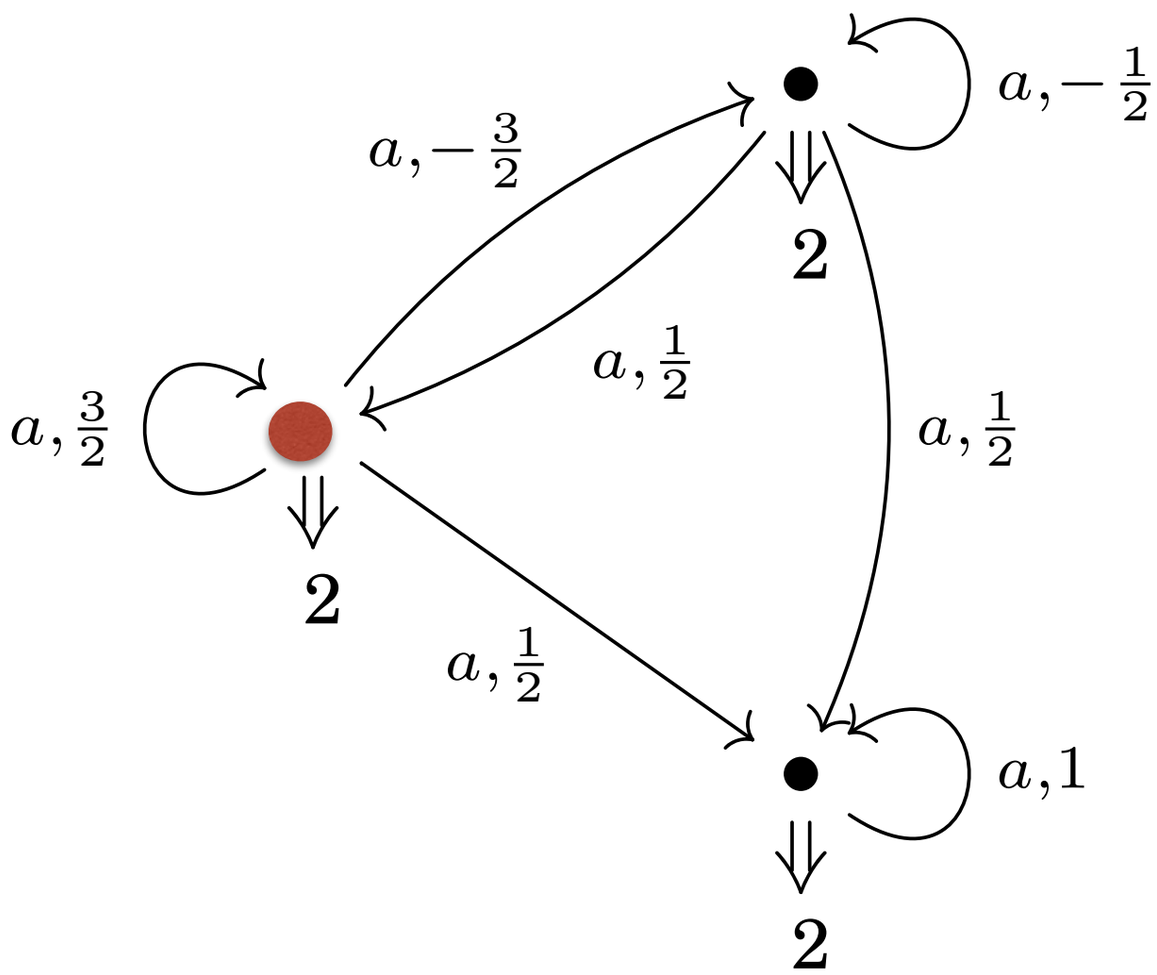
Some remarks

- There is a tight connection between the forward algorithm and the final sequence of the $\mathcal{L}$ functor
- The backward algorithm avoid the fact that ker( ) is a very large space (and hence computing a base might be tricky)
- The third algorithm in the paper is closely related to formal power series (which were a classical way to study weighted languages)
- What about semirings?

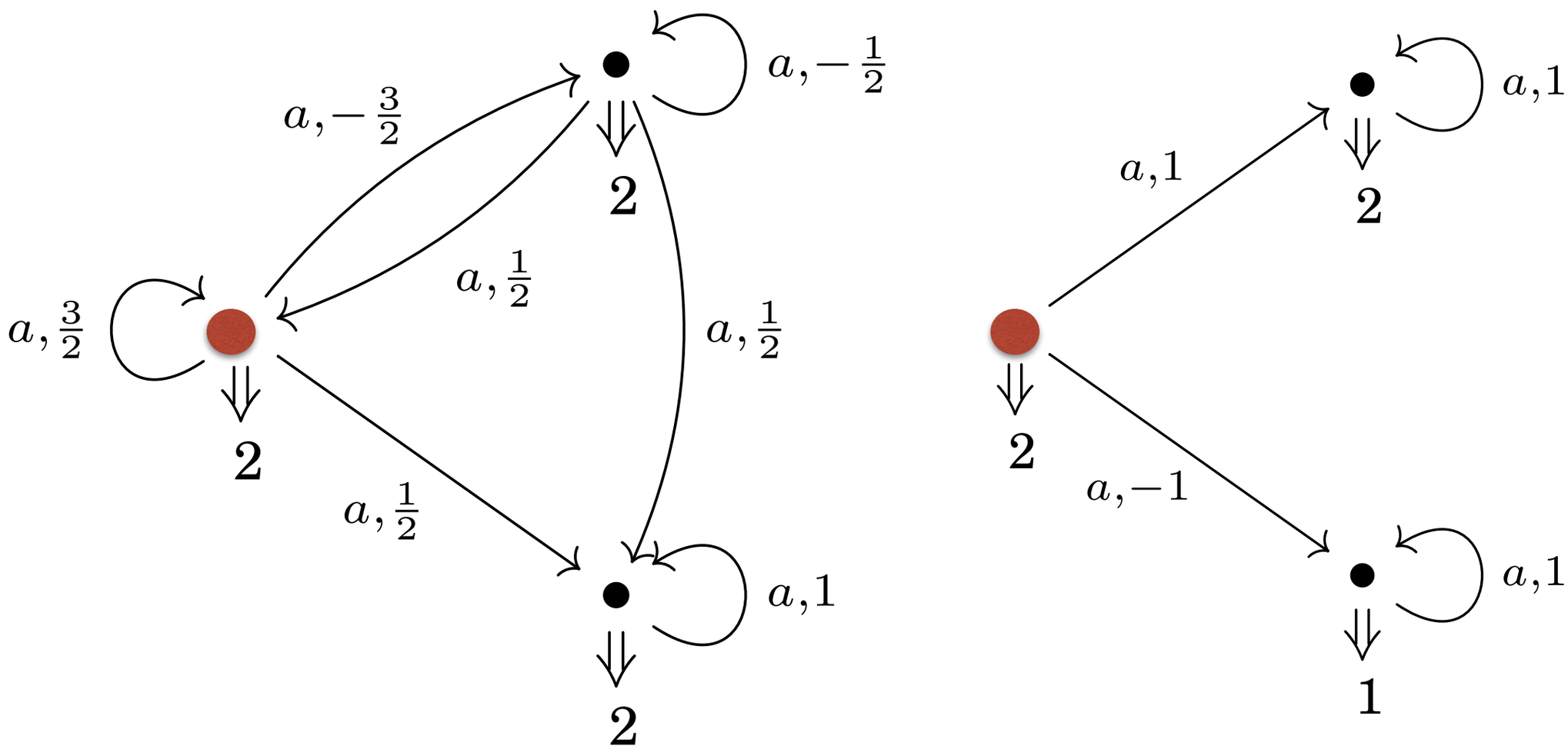
Some remarks

- There is a tight connection between the forward algorithm and the final sequence of the $\mathcal{L}$ functor
- The backward algorithm avoid the fact that ker( ) is a very large space (and hence computing a base might be tricky)
- The third algorithm in the paper is closely related to formal power series (which were a classical way to study weighted languages)
- What about semirings?

Equivalence of weighted automata, algebraically



Equivalence of weighted automata, algebraically



Weighted expressions

$$E ::= x \mid \mathbf{0} \mid E \oplus E \mid \underline{r} \mid a.(r \bullet E) \mid \mu x.E$$

Weighted expressions

$$E ::= x \mid \mathbf{0} \mid E \oplus E \mid \underline{r} \mid a.(r \bullet E) \mid \mu x.E$$

$$E_1 \equiv E_2[E_1/x] \implies E_1 \equiv \mu x.E_2$$

$$(E_1 \oplus E_2) \oplus E_3 \equiv E \oplus (E_2 \oplus E_3)$$

$$a.(r \bullet E) \oplus a.(s \bullet E) \equiv a.((r+s) \bullet E)$$

$$a.(0 \bullet E) \equiv \mathbf{0}$$

$$E_1 \oplus E_2 \equiv E_2 \oplus E_1$$

$$\mu x.E \equiv E[\mu x.E/x]$$

$$\mathbf{0} \oplus E \equiv \mathbf{0}$$

$$\underline{r} \oplus \underline{s} \equiv \underline{r+s}$$

$$\underline{0} \equiv \mathbf{0}$$

Weighted expressions

$$E ::= x \mid \mathbf{0} \mid E \oplus E \mid \underline{r} \mid a.(r \bullet E) \mid \mu x.E$$

$$E_1 \equiv E_2[E_1/x] \implies E_1 \equiv \mu x.E_2$$

$$(E_1 \oplus E_2) \oplus E_3 \equiv E \oplus (E_2 \oplus E_3)$$

$$a.(r \bullet E) \oplus a.(s \bullet E) \equiv a.((r+s) \bullet E)$$

$$a.(0 \bullet E) \equiv \mathbf{0}$$

$$E_1 \oplus E_2 \equiv E_2 \oplus E_1$$

$$\mu x.E \equiv E[\mu x.E/x]$$

$$\mathbf{0} \oplus E \equiv \mathbf{0}$$

$$\underline{r} \oplus \underline{s} \equiv \underline{r+s}$$

$$\underline{0} \equiv \mathbf{0}$$

$$a.(r \bullet (E_1 \oplus E_2)) \equiv a.(r \bullet E_1) \oplus a.(r \bullet E_2)$$

$$a.(r \bullet b.(s \bullet E)) \equiv a.((rs) \bullet b.(1 \bullet E))$$

$$a.(r \bullet \underline{s}) \equiv a.(1 \bullet \underline{rs})$$

$$a.(r \bullet \mathbf{0}) \equiv \mathbf{0}.$$

Weighted expressions

$$E ::= x \mid \mathbf{0} \mid E \oplus E \mid \underline{r} \mid a.(r \bullet E) \mid \mu x.E$$

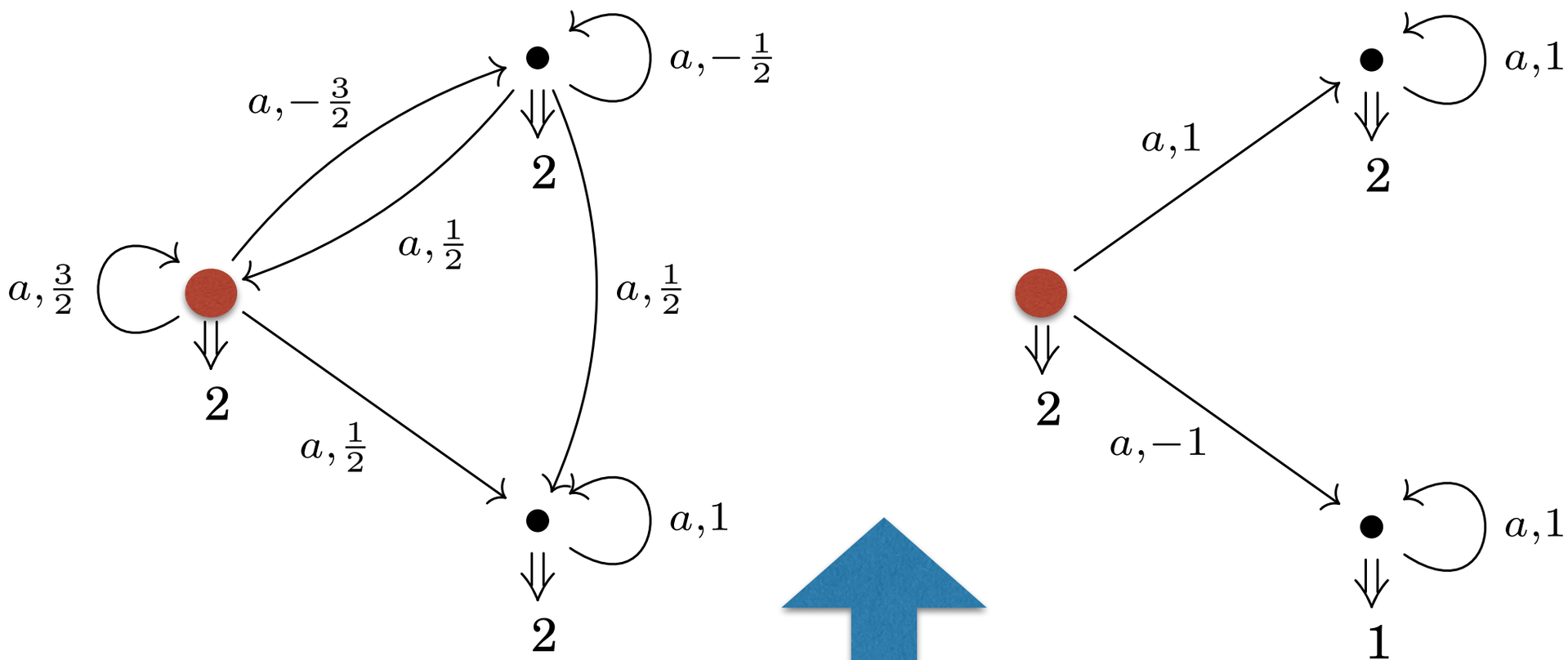
$$\begin{aligned}
E_1 \equiv E_2[E_1/x] &\implies E_1 \equiv \mu x.E_2 & a.(0 \bullet E) &\equiv \mathbf{0} & \mathbf{0} \oplus E &\equiv \mathbf{0} \\
(E_1 \oplus E_2) \oplus E_3 &\equiv E \oplus (E_2 \oplus E_3) & E_1 \oplus E_2 &\equiv E_2 \oplus E_1 & \underline{r} \oplus \underline{s} &\equiv \underline{r+s} \\
a.(r \bullet E) \oplus a.(s \bullet E) &\equiv a.((r+s) \bullet E) & \mu x.E &\equiv E[\mu x.E/x] & \underline{0} &\equiv \mathbf{0}
\end{aligned}$$

$$\begin{aligned}
a.(r \bullet (E_1 \oplus E_2)) &\equiv a.(r \bullet E_1) \oplus a.(r \bullet E_2) & a.(r \bullet \underline{s}) &\equiv a.(1 \bullet \underline{rs}) \\
a.(r \bullet b.(s \bullet E)) &\equiv a.((rs) \bullet b.(1 \bullet E)) & a.(r \bullet \mathbf{0}) &\equiv \mathbf{0}.
\end{aligned}$$

Sound and complete wrt weighted bisimilarity for noetherian semirings

[Bonsangue, Milius, Silva TOCL 2013]

Back to the example



Kleene theorem

$$E_1 = a.(1 \bullet E'') \oplus a.(-1 \bullet E) \oplus \underline{2}$$

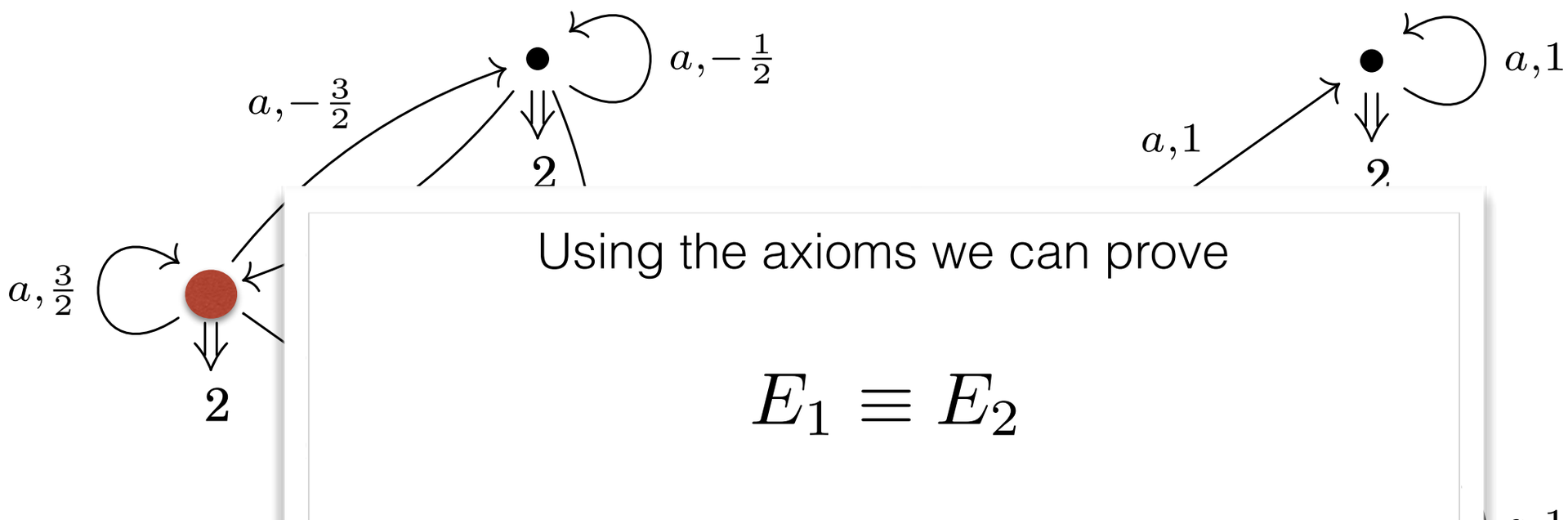
$$E_2 = \mu x.a.(\tfrac{3}{2} \bullet x) \oplus a.(-\tfrac{3}{2} \bullet E') \oplus a.(\tfrac{1}{2} \bullet E'') \oplus \underline{2}$$

$$E' = \mu z.a.(-\tfrac{1}{2} \bullet z) \oplus a.(\tfrac{1}{2} \bullet x) \oplus a.(\tfrac{1}{2} \bullet E'') \oplus \underline{2}$$

$$E = \mu y.a.(1 \bullet y) \oplus \underline{1}$$

$$E'' = \mu y.a.(1 \bullet y) \oplus \underline{2}$$

Back to the example



Using the axioms we can prove

$$E_1 \equiv E_2$$

We can also (even more easily) prove

$$E_1 \equiv \mu x.a.(1 \bullet x) + \underline{1}$$

semantically

$$a^n \mapsto 1$$

$E_1 = a.$ $\bullet\, y) \oplus \underline{1}$

$E_2 = \mu x.a.(\frac{3}{2} \bullet x) \oplus a.(-\frac{1}{2} \bullet E') \oplus a.(\frac{1}{2} \bullet E'') \oplus \underline{2}$ $E'' = \mu y.a.(1 \bullet y) \oplus \underline{2}$

$E' = \mu z.a.(-\frac{1}{2} \bullet z) \oplus a.(\frac{1}{2} \bullet x) \oplus a.(\frac{1}{2} \bullet E'') \oplus \underline{2}$

Some results, some problems

- Algorithms
- (Co)algebraic understanding of decidability
- Probabilistic automata

Conclusions

- Weighted automata are a very powerful generalisation of classical automata
- Algorithms for equivalence (and minimisation) vary depending on the semiring
- (Co)algebra — good abstract framework to study semantics, **algorithms, and decidability**

Conclusions

- Weighted automata are a very powerful generalisation of classical automata
- Algorithms for equivalence (and minimisation) vary depending on the semiring
- (Co)algebra — good abstract framework to study semantics, **algorithms, and decidability**

Thanks! Questions?