Deciding the value 1 problem for probabilistic leaktight automata

Séminaire Automates

Nathanaël Fijalkow,
joint work with Hugo Gimbert and Youssouf Oualhadj

LIAFA, CNRS & Université Denis Diderot - Paris 7, France
nath@liafa.jussieu.fr

November 25th, 2011

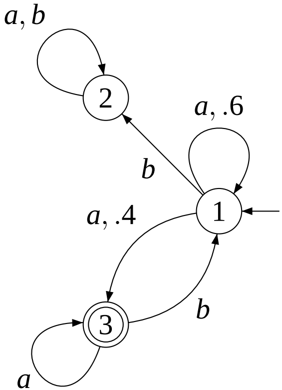
Outline

1. The value 1 problem for probabilistic automata
   - Definitions
   - Deciding the isolation problem
2. An algebraic solution to the limitedness problem for distance automata
   - Taking a step back: weighted automata
   - Leung’s algorithm
3. Towards an algebraic treatment of probabilistic automata
   - First tries
   - Leaks
   - The good semiring
   - The completeness proof using Simon’s theorem

Outline

Outline

Probabilistic automata (Rabin, 1963)



$$\mathbb{P}_{\mathcal{A}} : A^* \to [0, 1]$$

Cutpoint and value

Fix $0 < \lambda \leq 1$, define:

$$L_\lambda = \{w \mid \mathbb{P}_\mathcal{A}(w) \geq \lambda\}.$$

Cutpoint and value

Fix $0 < \lambda \leq 1$, define:

$$L_\lambda = \{w \mid \mathbb{P}_{\mathcal{A}}(w) \geq \lambda\}.$$

λ is isolated if there exists $\delta > 0$ such that for all $w \in A^*$, we have

$$|\mathbb{P}_{\mathcal{A}}(w) - \lambda| \geq \delta$$

Cutpoint and value

Fix $0 < \lambda \leq 1$, define:

$$L_\lambda = \{w \mid \mathbb{P}_\mathcal{A}(w) \geq \lambda\}.$$

λ is isolated if there exists $\delta > 0$ such that for all $w \in A^*$, we have

$$|\mathbb{P}_\mathcal{A}(w) - \lambda| \geq \delta$$

Theorem (Rabin, 1963)

If λ is isolated, then L_λ is a regular language.

Outline

1. The value 1 problem for probabilistic automata
   - Definitions
   - Deciding the isolation problem
2. An algebraic solution to the limitedness problem for distance automata
   - Taking a step back: weighted automata
   - Leung's algorithm
3. Towards an algebraic treatment of probabilistic automata
   - First tries
   - Leaks
   - The good semiring
   - The completeness proof using Simon's theorem

The isolation problem

Fix $0 \leq \lambda \leq 1$, the isolation problem is:

Instance: a probabilistic automaton $\mathcal{A}$

Question: is λ isolated in $\mathcal{A}$?

The isolation problem

Fix $0 \leq \lambda \leq 1$, the isolation problem is:

Instance: a probabilistic automaton $\mathcal{A}$

Question: is λ isolated in $\mathcal{A}$?

For $0 < \lambda < 1$, Bertoni showed that this is undecidable (in 1974)!

The value 1 problem

For $\lambda = 1$ the isolation problem can be formulated as: “are there words accepted by $\mathcal{A}$ with probability arbitrarily close to 1”.

The value 1 problem

For $\lambda = 1$ the isolation problem can be formulated as: “are there words accepted by $\mathcal{A}$ with probability arbitrarily close to 1”. Equivalently, define $\mathrm{val}(\mathcal{A}) = \sup_w \mathbb{P}_{\mathcal{A}}(w)$, then the problem is:

$$\text{“}\mathrm{val}(\mathcal{A}) \stackrel{?}{=} 1\text{”}.$$

The value 1 problem

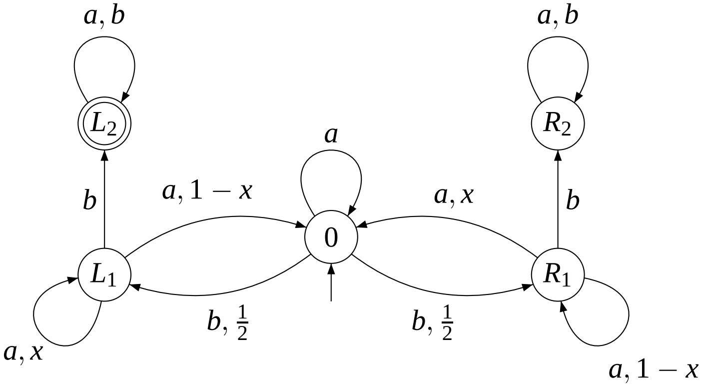
For $\lambda = 1$ the isolation problem can be formulated as: "are there words accepted by $\mathcal{A}$ with probability arbitrarily close to 1".
Equivalently, define $\mathrm{val}(\mathcal{A}) = \sup_w \mathbb{P}_{\mathcal{A}}(w)$, then the problem is:

$$\text{“}\mathrm{val}(\mathcal{A}) \stackrel{?}{=} 1\text{”.}$$

Theorem (Gimbert, Oualhadj, 2010)

The value 1 *problem is undecidable.*

An intuition



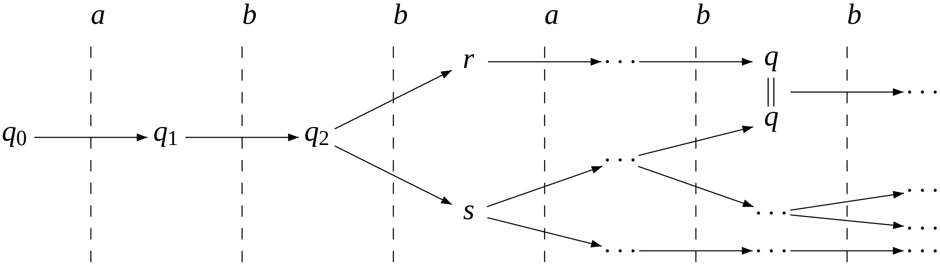
has value 1 if and only if $x > \frac{1}{2}$.

A very restricted case

Theorem (Fijalkow, Gimbert, Oualhadj, 2011)

The isolation problem is (still) undecidable if we randomise only on ***one*** *transition.*

Sketch of proof (1)



Sketch of proof (2)

Given $\mathcal{A}$ reading words from A^*, we construct $\mathcal{B}$ over a new alphabet B, with one probabilistic transition, and a morphism $\widehat{\_} : A^* \to B^*$ such that:

$$\forall w \in A^*, \mathbb{P}_{\mathcal{A}}(w) = \mathbb{P}_{\mathcal{B}}(\widehat{w}).$$

Sketch of proof (2)

Given $\mathcal{A}$ reading words from A^*, we construct $\mathcal{B}$ over a new alphabet B, with one probabilistic transition, and a morphism $\widehat{\_} : A^* \to B^*$ such that:

$$\forall w \in A^*, \mathbb{P}_{\mathcal{A}}(w) = \mathbb{P}_{\mathcal{B}}(\widehat{w}).$$

$$\widehat{a} = \mathrm{check}(a, q_0) \cdot * \cdot \mathrm{apply}(a, q_0) \ldots \mathrm{check}(a, q_{n-1}) \cdot * \cdot \mathrm{apply}(a, q_{n-1}) \cdot \mathrm{merge}.$$

Sketch of proof (2)

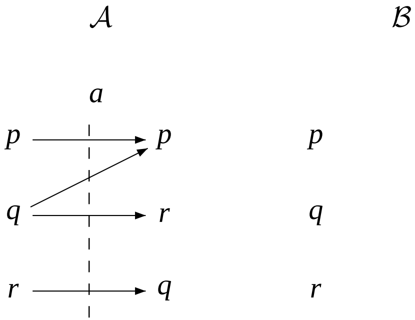
Given $\mathcal{A}$ reading words from A^*, we construct $\mathcal{B}$ over a new alphabet B, with one probabilistic transition, and a morphism $\widehat{\_} : A^* \to B^*$ such that:

$$\forall w \in A^*, \mathbb{P}_{\mathcal{A}}(w) = \mathbb{P}_{\mathcal{B}}(\widehat{w}).$$

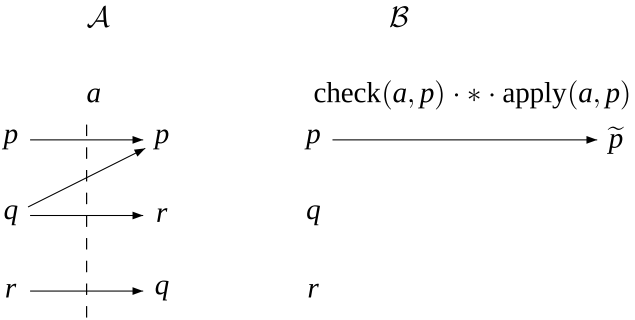
$$\widehat{a} = \text{check}(a, q_0) \cdot * \cdot \text{apply}(a, q_0) \ldots \text{check}(a, q_{n-1}) \cdot * \cdot \text{apply}(a, q_{n-1}) \cdot \text{merge}.$$

$$\begin{tikzcd}
 & & s_0 \arrow[r, "\text{apply}(a,q)"'] & \widetilde{r_0} \arrow[r, "\text{merge}"'] & r_0 \\
q \arrow[r, "\text{check}(a,q)"] & s_* \arrow[ur, "*"] \arrow[dr, "*"'] & & & \\
 & & s_1 \arrow[r, "\text{apply}(a,q)"] & \widetilde{r_1} \arrow[r, "\text{merge}"] & r_1
\end{tikzcd}$$

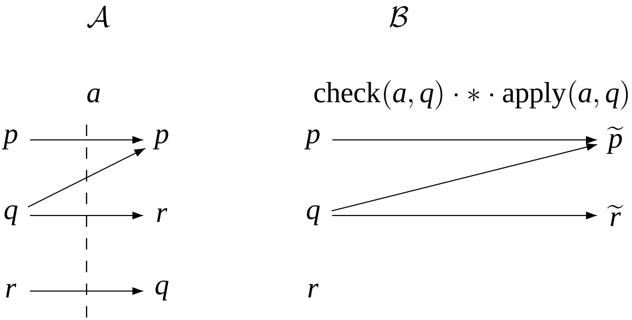
Sketch of proof (3)



Sketch of proof (3)



Sketch of proof (3)

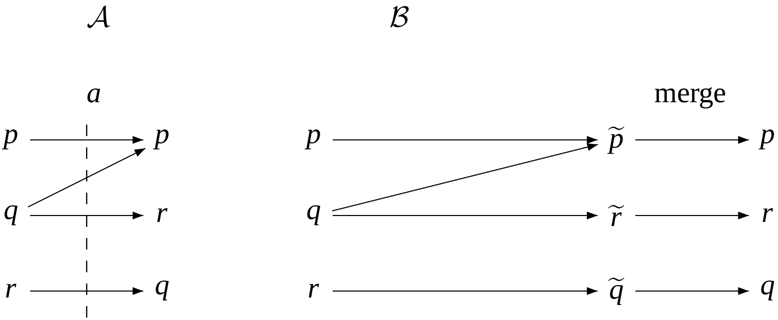


Sketch of proof (3)

$\mathcal{A}$ $\qquad\qquad\qquad\qquad$ $\mathcal{B}$

$$\begin{tikzcd}[row sep=large] & a & \\ p \arrow[rr] & & p \\ q \arrow[urr] \arrow[rr] & & r \\ r \arrow[rr] & & q \end{tikzcd} \qquad \begin{tikzcd}[row sep=large] & \text{check}(a,r) \cdot * \cdot \text{apply}(a,r) & \\ p \arrow[rr] & & \widetilde{p} \\ q \arrow[urr] \arrow[rr] & & \widetilde{r} \\ r \arrow[rr] & & \widetilde{q} \end{tikzcd}$$

Sketch of proof (3)

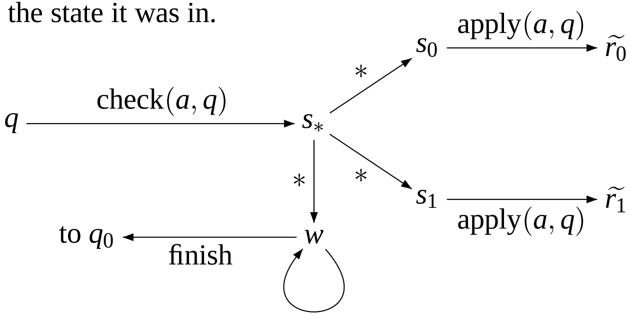


Sketch of proof (4)

$\mathcal{B}$ is unable to check that a letter $\text{check}(a, q)$ is actually followed by the corresponding $\text{apply}(a, q)$: inbetween, it will go through s_* and "forget" the state it was in.

Sketch of proof (4)

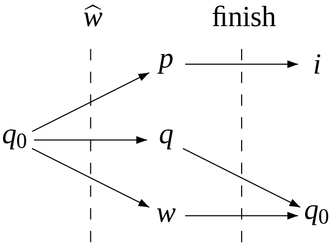
$\mathcal{B}$ is unable to check that a letter $\mathrm{check}(a, q)$ is actually followed by the corresponding $\mathrm{apply}(a, q)$: inbetween, it will go through s_* and “forget” the state it was in.



$$\sup_n \mathbb{P}_{\mathcal{B}}((\widehat{w} \cdot \mathrm{finish})^n) = \mathbb{P}_{\mathcal{A}}(w)$$

Sketch of proof (5)

Assume $p \in F$, $q \notin F$ and i is the initial state of a (deterministic) automaton recognizing $(\widehat{A^*} \cdot \text{finish})^*$.



Our objective

Define a **large** and **interesting** subclass of probabilistic automata for which the value 1 problem is decidable.

Outline

Outline

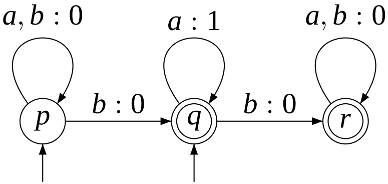
Probabilistic automata VS distance automata

Consider a semiring $(\mathcal{K}, +, \cdot)$. An automaton computes in the semiring $\mathcal{K}$ if $\text{val}(w) = \sum\{\Pi(\rho) \mid \rho \text{ is a run over } w\}$.

Probabilistic automata VS distance automata

Consider a semiring $(\mathcal{K}, +, \cdot)$. An automaton computes in the semiring $\mathcal{K}$ if $\text{val}(w) = \sum\{\Pi(\rho) \mid \rho \text{ is a run over } w\}$.

- Classical automata compute in the boolean semiring.
- Probabilistic automata compute in $(\mathbb{R}, +, \cdot)$ (there is a catch here).
- Distance automata compute in the tropical semiring $(\mathbb{N} \cup \{\infty\}, \min, +)$. Here is an example:



The value 1 problem VS the limitedness problem

The value 1 problem for probabilistic automata is:

"are there words accepted with probability arbitrarily close to 1?".

The **un**limitedness problem for distance automata is:

"are there words with arbitrarily high value?".

The value 1 problem VS the limitedness problem

The value 1 problem for probabilistic automata is:

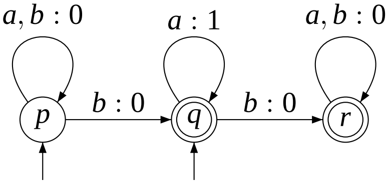
"are there words accepted with probability arbitrarily close to 1?".
undecidable

The **un**limitedness problem for distance automata is:

"are there words with arbitrarily high value?".
decidable (Hashiguchi, 1988)

Outline

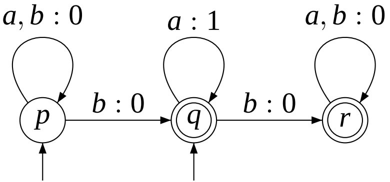
Weighted automata using algebra (Schützenberger)



$$\langle a \rangle = \begin{pmatrix} 0 & \infty & \infty \\ \infty & 1 & \infty \\ \infty & \infty & 0 \end{pmatrix} \qquad \langle b \rangle = \begin{pmatrix} 0 & 0 & \infty \\ \infty & \infty & 0 \\ \infty & \infty & 0 \end{pmatrix}$$

$$I = \begin{pmatrix} 0 & 0 & \infty \end{pmatrix} \qquad F = \begin{pmatrix} \infty \\ 0 \\ 0 \end{pmatrix}$$

Weighted automata using algebra (Schützenberger)

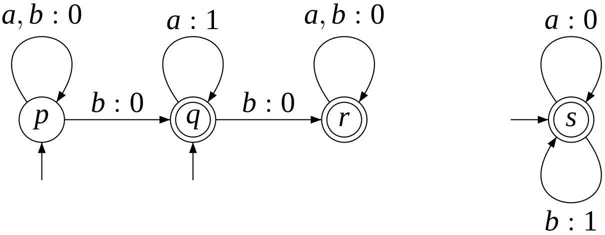


$$\langle a \rangle = \begin{pmatrix} 0 & \infty & \infty \\ \infty & 1 & \infty \\ \infty & \infty & 0 \end{pmatrix} \qquad \langle b \rangle = \begin{pmatrix} 0 & 0 & \infty \\ \infty & \infty & 0 \\ \infty & \infty & 0 \end{pmatrix}$$

$$I \cdot \langle aaabaa \rangle \cdot F = \begin{pmatrix} 0 & 0 & \infty \end{pmatrix} \cdot \begin{pmatrix} 0 & 2 & 2 \\ \infty & \infty & 3 \\ \infty & \infty & 0 \end{pmatrix} \cdot \begin{pmatrix} \infty \\ 0 \\ 0 \end{pmatrix} = 2$$

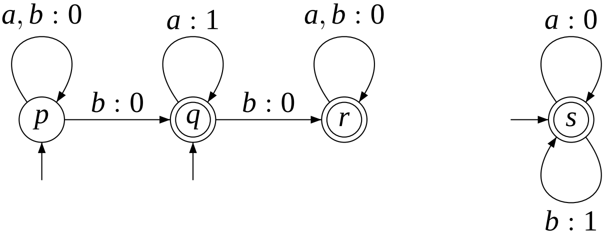
$$\begin{cases} k \in \mathbb{N} & \text{best run has value } k \\ \infty & \text{no run} \end{cases}$$

Towards Leung's algorithm: ♯-expressions



$$\mathrm{val}((a^n \cdot b)^n \cdot a^n) = n$$

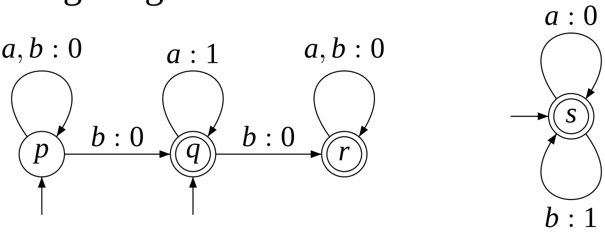
Towards Leung's algorithm: ♯-expressions



$$\mathrm{val}((a^n \cdot b)^n \cdot a^n) = n$$

An unlimitedness witness is $(a^\sharp \cdot b)^\sharp \cdot a^\sharp$.

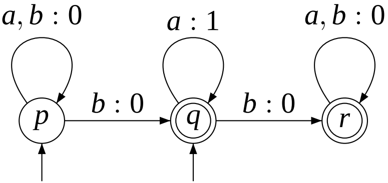
Towards Leung's algorithm: stabilization

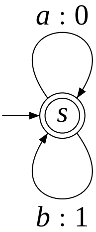


$$\langle a \rangle = \begin{pmatrix} 0 & \infty & \infty & \infty \\ \infty & \color{red}{1} & \infty & \infty \\ \infty & \infty & 0 & \infty \\ \infty & \infty & \infty & 0 \end{pmatrix} \qquad \langle a^n \rangle = \begin{pmatrix} 0 & \infty & \infty & \infty \\ \infty & \color{red}{n} & \infty & \infty \\ \infty & \infty & 0 & \infty \\ \infty & \infty & \infty & 0 \end{pmatrix}$$

$$\langle a^\sharp \rangle = \begin{pmatrix} 0 & \infty & \infty & \infty \\ \infty & \color{red}{\infty} & \infty & \infty \\ \infty & \infty & 0 & \infty \\ \infty & \infty & \infty & 0 \end{pmatrix}$$

Towards Leung's algorithm: stabilization





$$\langle a \rangle = \begin{pmatrix} 0 & \infty & \infty & \infty \\ \infty & {\color{red}1} & \infty & \infty \\ \infty & \infty & 0 & \infty \\ \infty & \infty & \infty & 0 \end{pmatrix} \qquad \langle a^n \rangle = \begin{pmatrix} 0 & \infty & \infty & \infty \\ \infty & {\color{red}n} & \infty & \infty \\ \infty & \infty & 0 & \infty \\ \infty & \infty & \infty & 0 \end{pmatrix}$$

$$\langle a^\sharp \rangle = \begin{pmatrix} 0 & \infty & \infty & \infty \\ \infty & {\color{red}\infty} & \infty & \infty \\ \infty & \infty & 0 & \infty \\ \infty & \infty & \infty & 0 \end{pmatrix} \qquad \left\{ \begin{array}{cc} k \in \mathbb{N} & \text{best run has value } k \\ \infty & \text{arbitrarily high value} \\ \infty & \text{no run} \end{array} \right.$$

Leung's algorithm

To ensure termination we project the tropical semiring $(\mathbb{N} \cup \infty, \min, +)$ into the **finite** semiring $(\{0, 1, \infty\}, \min, +)$.

Leung's algorithm

To ensure termination we project the tropical semiring $(\mathbb{N} \cup \infty, \min, +)$ into the **finite** semiring $(\{0, 1, \infty\}, \min, +)$.

Compute a monoid inside the monoid $\mathcal{M}_{Q \times Q}(\{0, 1, \infty\}, \min, +)$.

- Compute $\langle a \rangle$ for $a \in A$.
- Close under product and stabilization.
- If there exists a matrix M such that $I \cdot M \cdot F = \infty$ then "unlimited", otherwise "limited".

Leung's algorithm: termination and correction

Termination: the monoid $\mathcal{M}_{Q\times Q}(\{0,1,\infty\},\min,+)$ is finite.

Leung's algorithm: termination and correction

Termination: the monoid $\mathcal{M}_{Q\times Q}(\{0,1,\infty\},\min,+)$ is finite.

Correction: the proof is complicated, and relies on Simon's theorem.

Outline

Our objective (again)

Decide the value 1 problem for a *subclass* of probabilistic automata, by **algebraic** and **non-numerical** means.

Our objective (again)

Decide the value 1 problem for a *subclass* of probabilistic automata, by **algebraic** and **non-numerical** means.

- **algebraic:** focus on the automaton structure,
- **non-numerical:** abstract away the values.

Our objective (again)

Decide the value 1 problem for a *subclass* of probabilistic automata, by **algebraic** and **non-numerical** means.

- **algebraic:** focus on the automaton structure,
- **non-numerical:** abstract away the values.

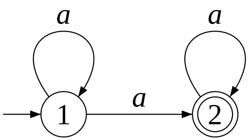
Hence we consider non-deterministic automata: we project $(\mathbb{R}, +, \cdot)$ into the boolean semiring $(\{0, 1\}, +, \cdot)$.

Our objective (again)

Decide the value 1 problem for a *subclass* of probabilistic automata, by **algebraic** and **non-numerical** means.

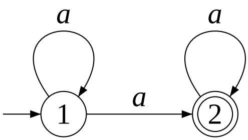
- **algebraic:** focus on the automaton structure,
- **non-numerical:** abstract away the values.

Hence we consider non-deterministic automata: we project $(\mathbb{R}, +, \cdot)$ into the boolean semiring $(\{0, 1\}, +, \cdot)$.



Outline

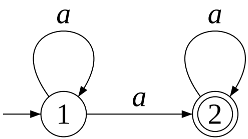
Defining stabilization



$$\langle a \rangle = \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix}$$

In $\langle a \rangle$, the state 1 is transient and the state 2 is recurrent.

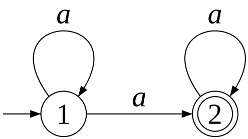
Defining stabilization



$$\langle a \rangle = \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix} \qquad \langle a^\sharp \rangle = \begin{pmatrix} 0 & 1 \\ 0 & 1 \end{pmatrix}$$

In $\langle a \rangle$, the state 1 is transient and the state 2 is recurrent.

Defining stabilization



$$\langle a \rangle = \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix} \qquad \langle a^\sharp \rangle = \begin{pmatrix} 0 & 1 \\ 0 & 1 \end{pmatrix}$$

In $\langle a \rangle$, the state 1 is transient and the state 2 is recurrent.

$$M^\sharp(s,t) = \begin{cases} 1 & \text{if } M(s,t) = 1 \text{ and } t \text{ recurrent in } M, \\ 0 & \text{otherwise.} \end{cases}$$

(This definition gives an asymmetric monoid, this is unusual.)

A first algorithm

Compute a monoid inside the **finite** monoid $\mathcal{M}_{Q\times Q}(\{0,1\},+,\cdot)$.

- Compute $\langle a\rangle$ for $a \in A$:

$$\langle a\rangle(s,t) = \begin{cases} 1 & \text{if } \mathbb{P}_{\mathcal{A}}(s \xrightarrow{a} t) > 0, \\ 0 & \text{otherwise.} \end{cases}$$

- Close under product and stabilization.

A first algorithm

Compute a monoid inside the **finite** monoid $\mathcal{M}_{Q\times Q}(\{0,1\},+,\cdot)$.

- Compute $\langle a\rangle$ for $a\in A$:

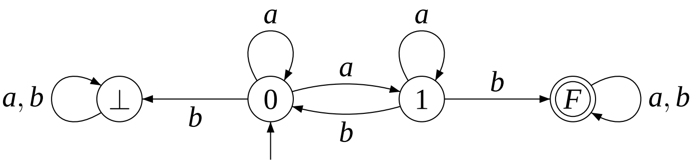
$$\langle a\rangle(s,t)=\begin{cases}1 & \text{if } \mathbb{P}_{\mathcal{A}}(s\xrightarrow{a}t)>0,\\ 0 & \text{otherwise.}\end{cases}$$

- Close under product and stabilization.
- If there exists a matrix M such that

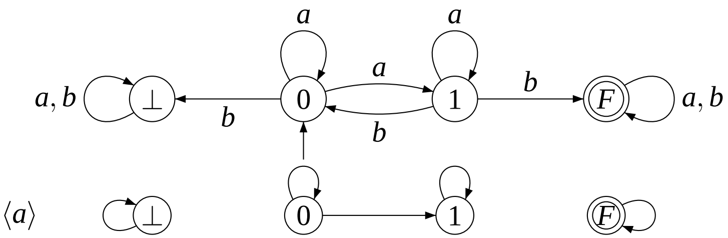
$$\forall t\in Q,\quad M(s_0,t)=1\Rightarrow t\in F$$

then "$\mathcal{A}$ has value 1", otherwise "$\mathcal{A}$ does not have value 1".

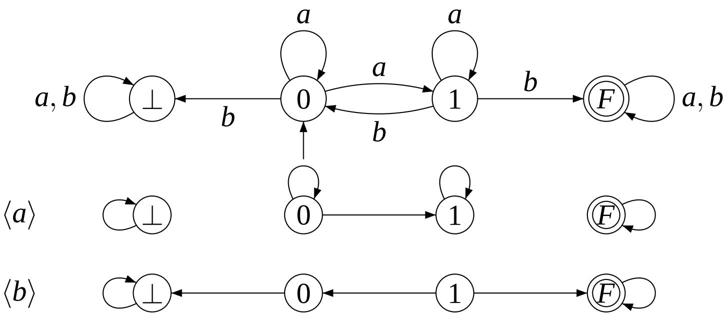
An example



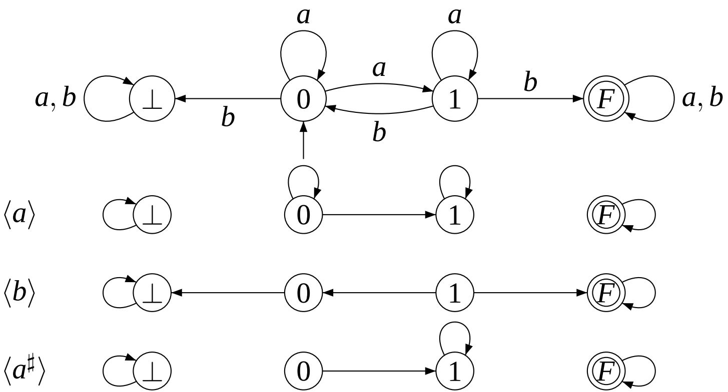
An example



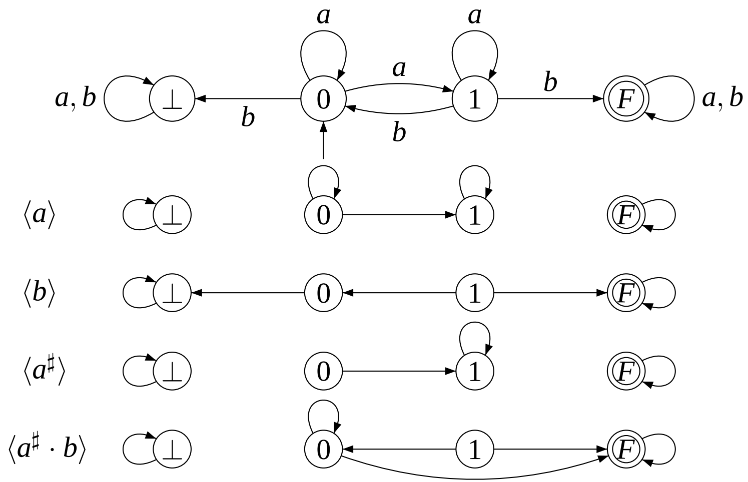
An example



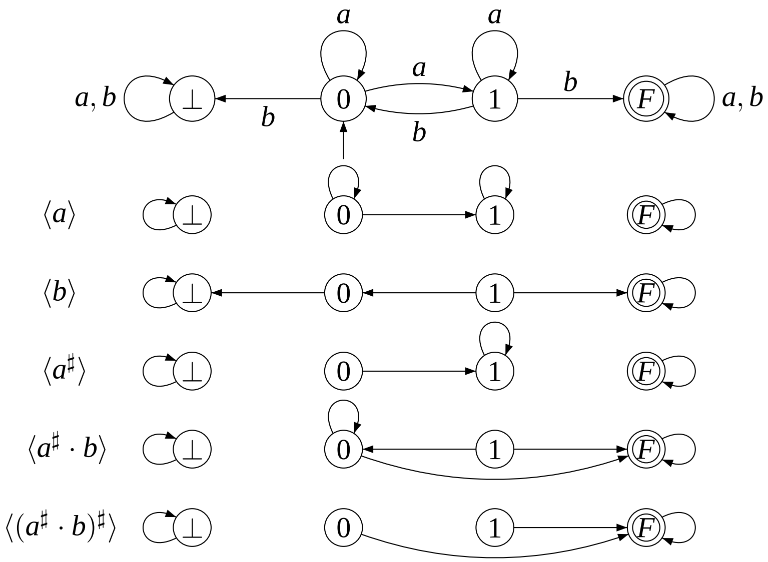
An example



An example



An example



a
a
a
a, b
⊥
0
1
F
a, b
b
b
b
$\langle a \rangle$
$\langle b \rangle$
$\langle a^\sharp \rangle$
$\langle a^\sharp \cdot b \rangle$
$\langle (a^\sharp \cdot b)^\sharp \rangle$

Correctness

Theorem

If there exists a matrix M such that

$$\forall t \in Q, \quad M(s_0, t) = 1 \Rightarrow t \in F$$

then $\mathcal{A}$ has value 1.

Correctness

Theorem

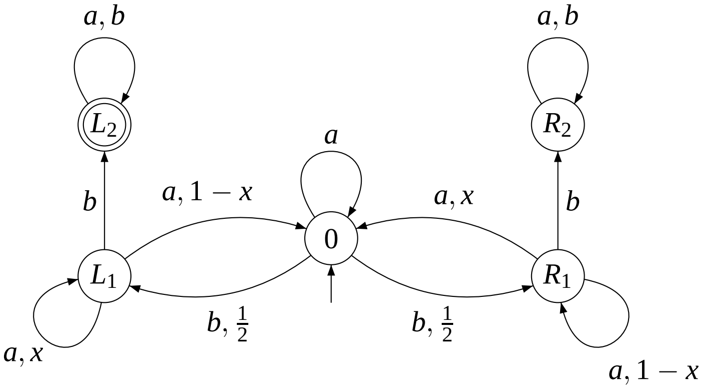
If there exists a matrix M such that

$$\forall t \in Q, \quad M(s_0, t) = 1 \Rightarrow t \in F$$

then $\mathcal{A}$ has value 1.

But the value 1 problem is undecidable, so…

No completeness

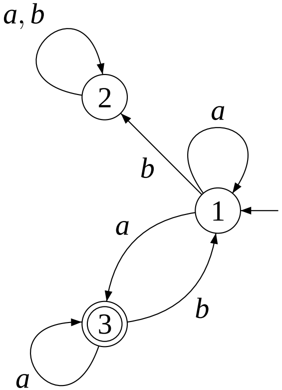


Left and right parts are symmetric, so for all M:

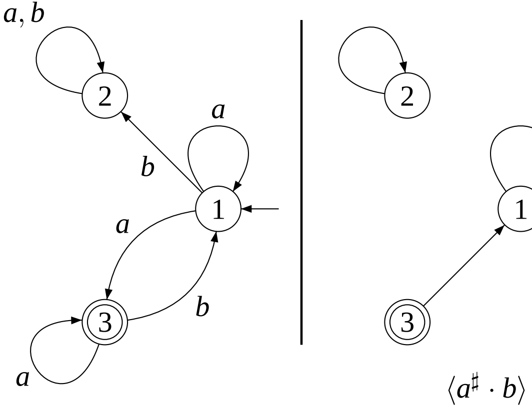
$$M(0, L_2) = 1 \Longleftrightarrow M(0, R_2) = 1.$$

Outline

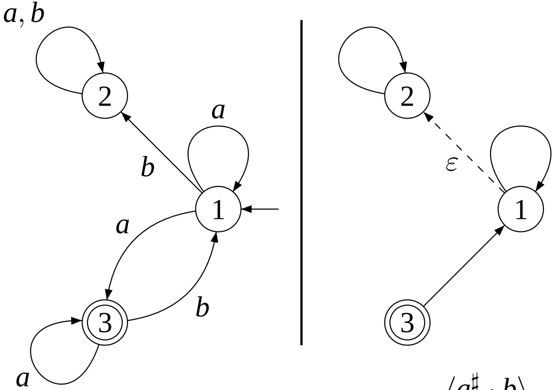
An example



An example



An example



$\langle a^\sharp \cdot b \rangle$

Outline

A three-valued semiring

Instead of $(\{0,1\},+,\cdot)$ we compute in $(\{0,\varepsilon,1\},+,\cdot)$, where $0<\varepsilon<1$.

A three-valued semiring

Instead of $(\{0,1\},+,\cdot)$ we compute in $(\{0,\varepsilon,1\},+,\cdot)$, where $0<\varepsilon<1$.

$+$	0	ε	1
0	0	ε	1
ε	ε	ε	1
1	1	1	1

$\cdot$	0	ε	1
0	0	0	0
ε	0	ε	ε
1	0	ε	1

The algorithm

- Compute $\langle a \rangle$ for $a \in A$:

$$\langle a \rangle(s,t) = \begin{cases} 1 & \text{if } \mathbb{P}_{\mathcal{A}}(s \xrightarrow{a} t) > 0, \\ 0 & \text{otherwise.} \end{cases}$$

- Close under product and stabilization:

$$M^{\sharp}(s,t) = \begin{cases} 1 & \text{if } M(s,t) = 1 \text{ and } t \text{ recurrent in } M, \\ \varepsilon & \text{if } M(s,t) = 1 \text{ and } t \text{ transient in } M, \\ \varepsilon & \text{if } M(s,t) = \varepsilon, \\ 0 & \text{otherwise.} \end{cases}$$

The algorithm

- Compute $\langle a \rangle$ for $a \in A$:

$$\langle a \rangle(s,t) = \begin{cases} 1 & \text{if } \mathbb{P}_{\mathcal{A}}(s \xrightarrow{a} t) > 0, \\ 0 & \text{otherwise.} \end{cases}$$

- Close under product and stabilization:

$$M^{\sharp}(s,t) = \begin{cases} 1 & \text{if } M(s,t) = 1 \text{ and } t \text{ recurrent in } M, \\ \varepsilon & \text{if } M(s,t) = 1 \text{ and } t \text{ transient in } M, \\ \varepsilon & \text{if } M(s,t) = \varepsilon, \\ 0 & \text{otherwise.} \end{cases}$$

- If there exists a matrix M such that

$$\forall t \in Q, \quad M(s_0, t) = 1 \Rightarrow t \in F$$

then "$\mathcal{A}$ has value 1", otherwise "$\mathcal{A}$ does not have value 1".

The control lemma

We say that a word w reify M in $\mathcal{M}_{\mathcal{A}}$ if:

- $M = \langle a \rangle$ and $w = a$;
- $M = M_1 \cdot M_2$ and there exists w_1 and w_2 reifying M_1 and M_2, respectively, such that $w = w_1 \cdot w_2$;
- $M = N^\sharp$ and there exists $x_1, \ldots, x_n$ each reifying N, such that $w = x_1 \ldots x_n$ for some $n \geq 1$.

The control lemma

We say that a word w reify M in $\mathcal{M}_{\mathcal{A}}$ if:

- $M = \langle a \rangle$ and $w = a$;
- $M = M_1 \cdot M_2$ and there exists w_1 and w_2 reifying M_1 and M_2, respectively, such that $w = w_1 \cdot w_2$;
- $M = N^\sharp$ and there exists $x_1, \ldots, x_n$ each reifying N, such that $w = x_1 \ldots x_n$ for some $n \geq 1$.

Lemma (The control lemma)

For all M in $\mathcal{M}_{\mathcal{A}}$, for all words w reifying M, for all states s, t in Q, we have:

$$M(s,t) \neq 0 \iff \mathbb{P}_{\mathcal{A}}(s \xrightarrow{w} t) > 0.$$

Leaktight automata

Definition

An automaton $\mathcal{A}$ is leaktight if for all M, we have

$$M(s,t) = \varepsilon \Longrightarrow (s \text{ is transient}) \text{ or } (M(t,s) = 1).$$

Leaktight automata

Definition

An automaton $\mathcal{A}$ is leaktight if for all M, we have

$$M(s,t) = \varepsilon \Longrightarrow (s \text{ is transient}) \text{ or } (M(t,s) = 1).$$

Theorem (Fijalkow,Gimbert,Oualhadj)

The value 1 *problem is decidable for leaktight automata.*

Outline

Decomposition trees

Fact

The set $\mathcal{M}_\mathcal{A}$ computed by the algorithm is a stabilization monoid.

Definition

A *decomposition tree* of a word $w \in A^+$ is a finite unranked ordered tree, whose nodes have labels in $(A^+, \mathcal{M}_\mathcal{A})$ and such that:

- the root is labeled by (w, u), for some $u \in \mathcal{M}_\mathcal{A}$,
- every leaf is labeled by $(a, \langle a \rangle)$ where a is a letter,
- every internal node with two children labeled by (w_1, u_1) and (w_2, u_2) is labeled by $(w_1 \cdot w_2, u_1 \cdot u_2)$,
- for every internal node with three or more children, there exists $e \in E(M)$ such that the node is labeled by $(w_1 \dots w_n, e^\sharp)$ and its children are labeled by $(w_1, e), \dots, (w_n, e)$.

Bounding the height of a decomposition tree

In a decomposition tree, an iteration node is said discontinuous if $M^\sharp \neq M$. The span of a decomposition tree is the maximal length of a path that contains no discontinuous path.

Bounding the height of a decomposition tree

In a decomposition tree, an iteration node is said discontinuous if $M^\sharp \neq M$. The span of a decomposition tree is the maximal length of a path that contains no discontinuous path.

Theorem (Simon, 1990)

Every word $w \in A^+$ has a decomposition tree whose span is less than $3 \cdot |\mathcal{M}_\mathcal{A}|$.

Bounding the height of a decomposition tree

In a decomposition tree, an iteration node is said discontinuous if $M^\sharp \neq M$. The span of a decomposition tree is the maximal length of a path that contains no discontinuous path.

Theorem (Simon, 1990)

Every word $w \in A^+$ has a decomposition tree whose span is less than $3 \cdot |\mathcal{M}_\mathcal{A}|$.

Lemma (Simon, 1990)

Let $M \in E(\mathcal{M}_\mathcal{A})$, if $M^\sharp \neq M$, then $M^\sharp <_\mathcal{J} M$.

Bounding the height of a decomposition tree

In a decomposition tree, an iteration node is said discontinuous if $M^\sharp \neq M$. The span of a decomposition tree is the maximal length of a path that contains no discontinuous path.

Theorem (Simon, 1990)

Every word $w \in A^+$ has a decomposition tree whose span is less than $3 \cdot |\mathcal{M}_\mathcal{A}|$.

Lemma (Simon, 1990)

Let $M \in E(\mathcal{M}_\mathcal{A})$, if $M^\sharp \neq M$, then $M^\sharp <_\mathcal{J} M$.

Corollary

Every word $w \in A^+$ has a decomposition tree whose height is less than $3 \cdot |\mathcal{M}_\mathcal{A}| \cdot J(\mathcal{A})$.

Bounding the acceptance probability from below

Lemma

There exists a positive rational number η *which depends only on* $\mathcal{A}$ *such that: for all words* $w \in A^+$*, there exists* M *in* $\mathcal{M}_{\mathcal{A}}$ *satisfying for all states* $s, t \in Q$*,*

$$M(s,t) = 1 \Rightarrow \mathbb{P}_{\mathcal{A}}(s \xrightarrow{w} t) \geq \eta.$$

Bounding the acceptance probability from below

Lemma

There exists a positive rational number η which depends only on $\mathcal{A}$ such that: for all words $w \in A^+$, there exists M in $\mathcal{M}_\mathcal{A}$ satisfying for all states $s, t \in Q$,

$$M(s,t) = 1 \Rightarrow \mathbb{P}_\mathcal{A}(s \xrightarrow{w} t) \geq \eta.$$

Proof idea: given w, consider a decomposition tree of bounded height, and prove by induction that the lower bound 2^{-h+1} holds at depth h, going from leaves to the root.

The case of an iteration node (1)

The node is labelled by $(w_1 \dots w_n, \langle u^\sharp \rangle)$ and its children are labelled by $(w_1, \langle u \rangle), \dots, (w_n, \langle u \rangle)$, where $\langle u \rangle$ is idempotent, and η a lower bound shared by the $n \geq 3$ children.

The case of an iteration node (1)

The node is labelled by $(w_1 \dots w_n, \langle u^\sharp \rangle)$ and its children are labelled by $(w_1, \langle u \rangle), \dots, (w_n, \langle u \rangle)$, where $\langle u \rangle$ is idempotent, and η a lower bound shared by the $n \geq 3$ children.
Let s, t such that $\langle u^\sharp \rangle(s, t) = 1$, then:

$$\mathbb{P}_{\mathcal{A}}(s \xrightarrow{w_1 \dots w_n} t) \geq \underbrace{\mathbb{P}_{\mathcal{A}}(s \xrightarrow{w_1} t)}_{\geq \eta} \cdot \underbrace{\mathbb{P}_{\mathcal{A}}(t \xrightarrow{w_2 \dots w_n} t)}_{\geq \eta} \geq \eta^2 .$$

The case of an iteration node (1)

The node is labelled by $(w_1 \dots w_n, \langle u^\sharp \rangle)$ and its children are labelled by $(w_1, \langle u \rangle), \dots, (w_n, \langle u \rangle)$, where $\langle u \rangle$ is idempotent, and η a lower bound shared by the $n \geq 3$ children.
Let s, t such that $\langle u^\sharp \rangle(s,t) = 1$, then:

$$\mathbb{P}_{\mathcal{A}}(s \xrightarrow{w_1 \dots w_n} t) \geq \underbrace{\mathbb{P}_{\mathcal{A}}(s \xrightarrow{w_1} t)}_{\geq \eta} \cdot \underbrace{\mathbb{P}_{\mathcal{A}}(t \xrightarrow{w_2 \dots w_n} t)}_{\geq \eta} \geq \eta^2 .$$

The left inequality follows from induction hypothesis, since $\langle u \rangle(s,t) = 1$.

The case of an iteration node (2)

Consider the right inequality: $\mathbb{P}_{\mathcal{A}}(t \xrightarrow{w_2 \ldots w_n} t) \geq \eta$
Let $C = \{q \mid \langle u \rangle(t,q) \neq 0\}$, we have:

$$\begin{aligned}
\mathbb{P}_{\mathcal{A}}(t \xrightarrow{w_2 \ldots w_n} t) &= \textstyle\sum_{q \in C} \mathbb{P}_{\mathcal{A}}(t \xrightarrow{w_2 \ldots w_{n-1}} q) \cdot \underbrace{\mathbb{P}_{\mathcal{A}}(q \xrightarrow{w_n} t)}_{\geq \eta} \\
&\geq \eta \cdot \underbrace{\sum_{q \in C} \mathbb{P}_{\mathcal{A}}(t \xrightarrow{w_2 \ldots w_{n-1}} q)}_{=1} = \eta
\end{aligned}$$

Indeed, since t is recurrent and thanks to the leaktight assumption, we have $C \subseteq \{q \mid \langle u \rangle(q,t) = 1\}$, so the inequality follows from induction hypothesis, and the equality from the “control lemma”.

What I didn't (and won't) say

- One can decide whether an automaton is leaktight in PSPACE,
- The value 1 problem for probabilistic leaktight automata is PSPACE-complete,
- The class of leaktight automata subsumes all subclasses of probabilistic automata whose value 1 problem is known to be decidable,
- The class of leaktight automata is closed under parallel composition and synchronized product.

The end.

Thanks for your attention!