Quantitative Evaluation of Systems via Weighted Logics and Weighted Automata

Benjamin Monmege
LIF, Aix-Marseille Université

Journées SDA² - 05/07/2017

Based on joint works with Paul Gastin, Benedikt Bollig and Marc Zeitoun

Software Verification

Software Verification

Critical Software

- communication systems
- e-commerce
- health databases
- energy production

Software Verification

Software Verification

Property to be verified

Critical Software

- communication systems
- e-commerce
- health databases
- energy production

TO BE VERIFIED

Software Verification

Property to be verified

Is the property verified
or not by the software?

Critical Software
- communication systems
- e-commerce
- health databases
- energy production

TO BE VERIFIED

Software Verification

Property to be verified

- May an *error* state be reached?
- Is there a book written by X, rented by Y?
- Does this leader election protocol permit to elect the leader?

erified
ware?

are
systems

- e-commerce
- health databases
- energy production

TO BE VERIFIED

Software Verification

Property to be verified

- May an *error* state be reached?
- Is there a book written by X, rented by Y?
- Does this leader election protocol permit to elect the leader?

From **Boolean** to **Quantitative** Verification

- What is the probability for an *error* state to be reached?
- How many books, written by X, have been rented by Y?
- What is the maximal delay ensuring that this leader election protocol permits the election?

erified

tware?

are

systems

- e-commerce
- health databases
- energy production

TO BE VERIFIED

Formal Verification

Property to be verified

Is the property verified or not by the software?

Critical Software

- communication systems
- e-commerce
- health databases
- energy production

TO BE VERIFIED

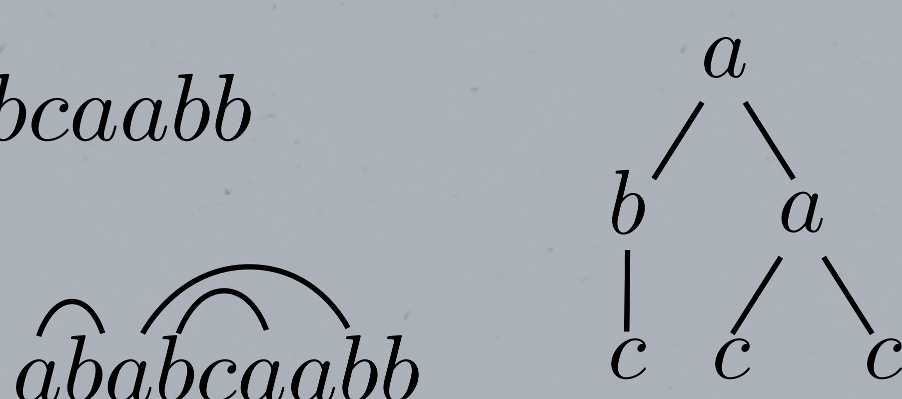
Formal Verification

Property to be verified

Is the property verified or not by the model?

Formal Model

$ababcaabb$



Critical Software

- communication systems
- e-commerce
- health databases
- energy production

TO BE VERIFIED

Formal Verification

Property to be verified
Formal Specification

$$(a+b)^\star c(ac)^+$$

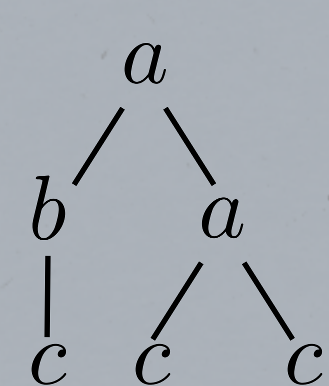
$$\forall x \forall y\ (x < y \Rightarrow \exists z\ (x < z < y))$$

$$\mathsf{F}\,\mathsf{G}\,(p\,\mathsf{U}\,q)$$

Is the property verified
or not by the model?

Formal Model

$ababcaabb$



Critical Software
- communication systems
- e-commerce
- health databases
- energy production

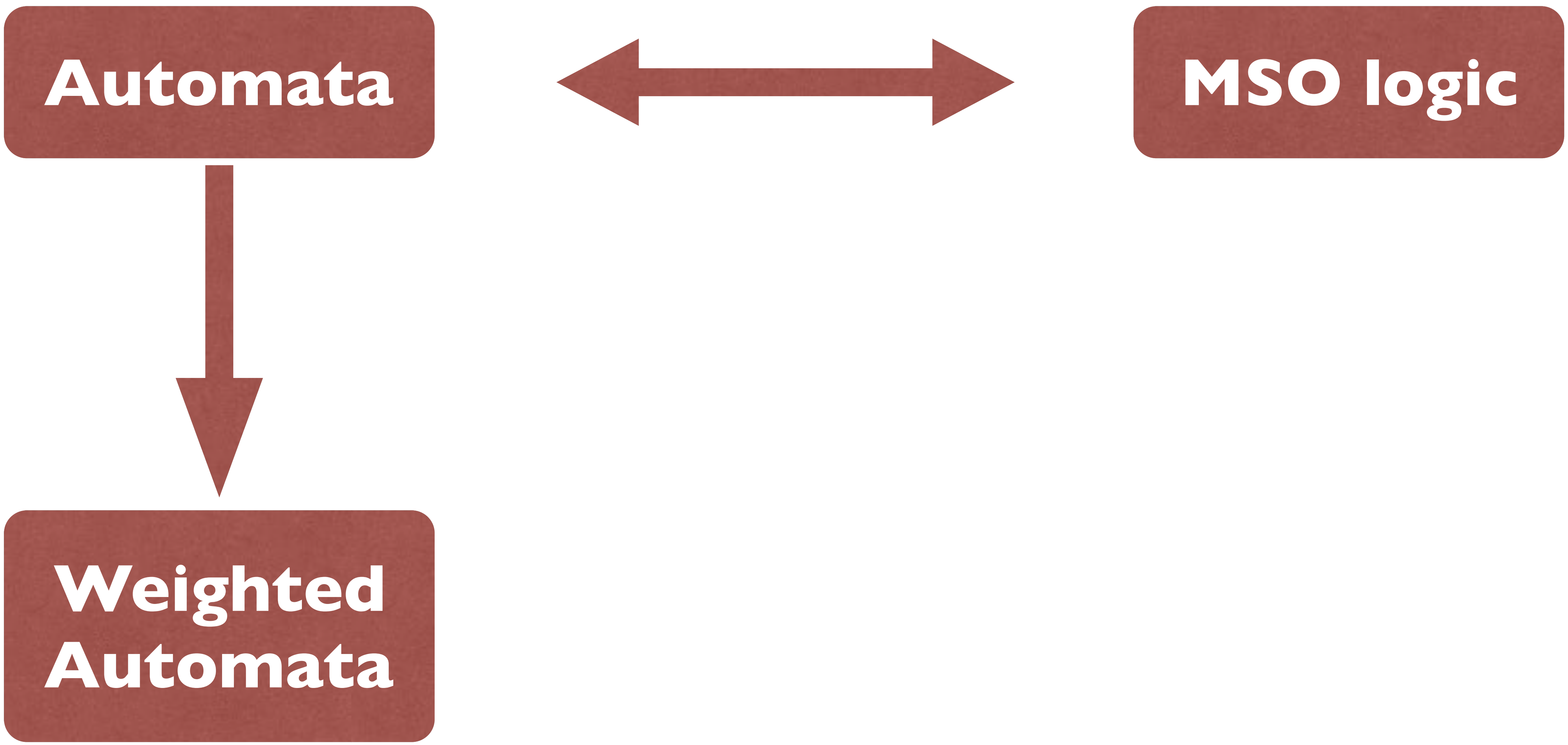
TO BE VERIFIED

Qualitative/Quantitative

- Qualitative, Boolean: [Büchi'60], [Elgot'61], [Trakhtenbrot'61]

Qualitative/Quantitative

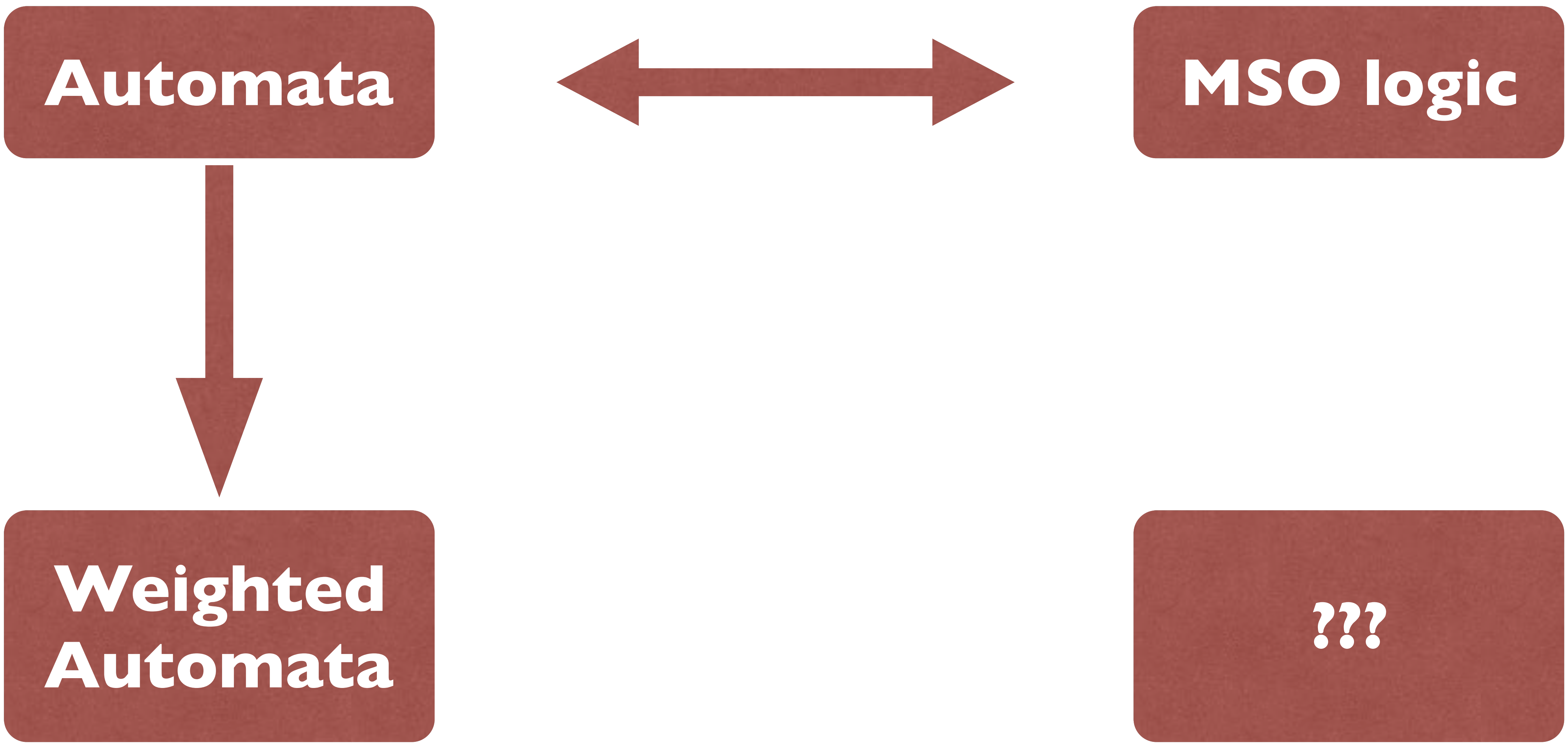
- Qualitative, Boolean: [Büchi’60], [Elgot’61], [Trakhtenbrot’61]



- Quantitative, weights

Qualitative/Quantitative

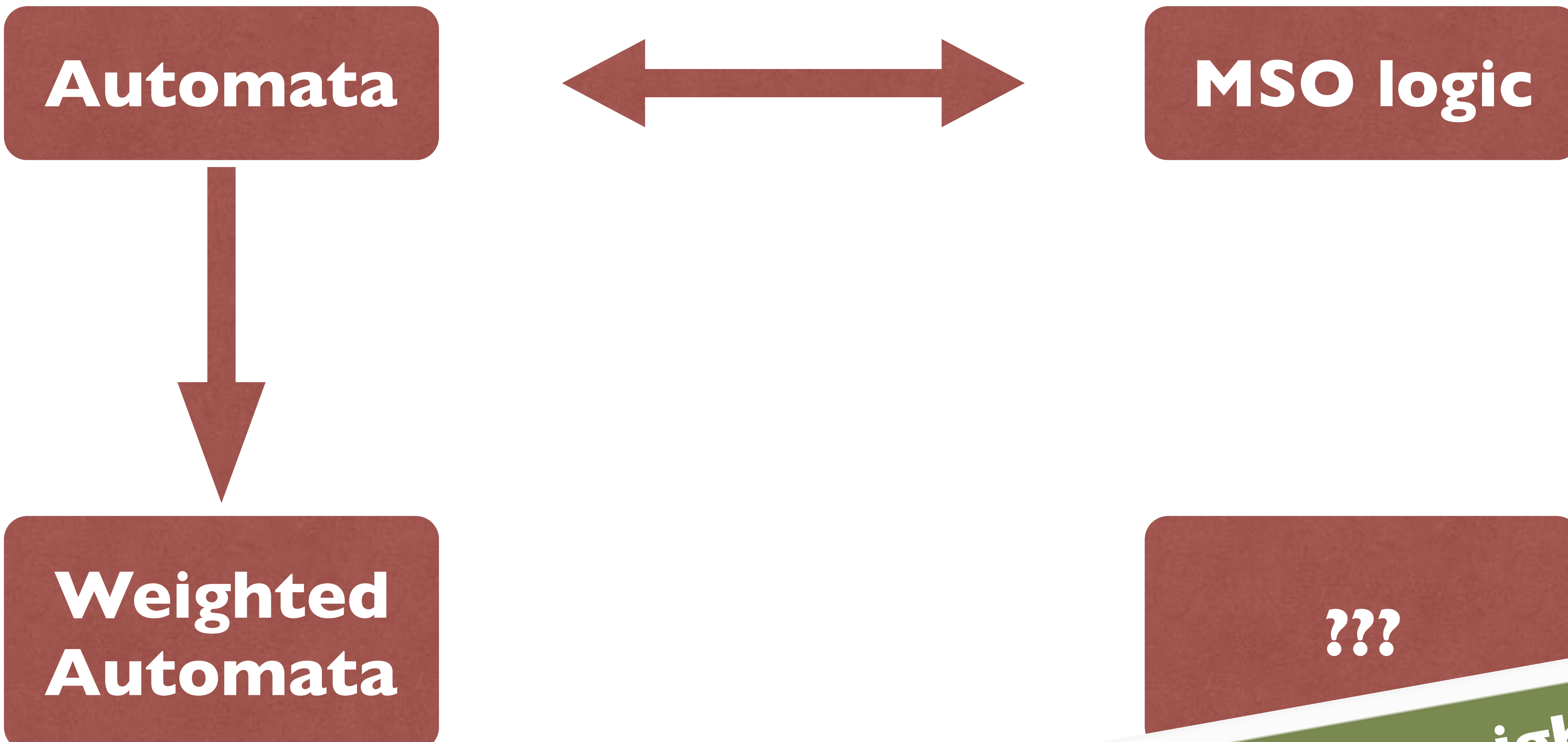
- Qualitative, Boolean: [Büchi'60], [Elgot'61], [Trakhtenbrot'61]



- Quantitative, weights

Qualitative/Quantitative

- Qualitative, Boolean: [Büchi'60], [Elgot'61], [Trakhtenbrot'61]



- Quantitative, weights

Find suitable weighted MSO logic

Qualitative/Quantitative

- Qualitative, Boolean: [Büchi'60], [Elgot'61], [Trakhtenbrot'61]

Automata ⟷ **MSO logic**

Focus on definability / qualitative

Weighted Automata

???

Find suitable weighted MSO logic

- Quantitative, weights

Qualitative/Quantitative

- Qualitative, Boolean: [Büchi'60], [Elgot'61], [Trakhtenbrot'61]

Automata ⟷ **MSO logic**

Focus on definability / qualitative

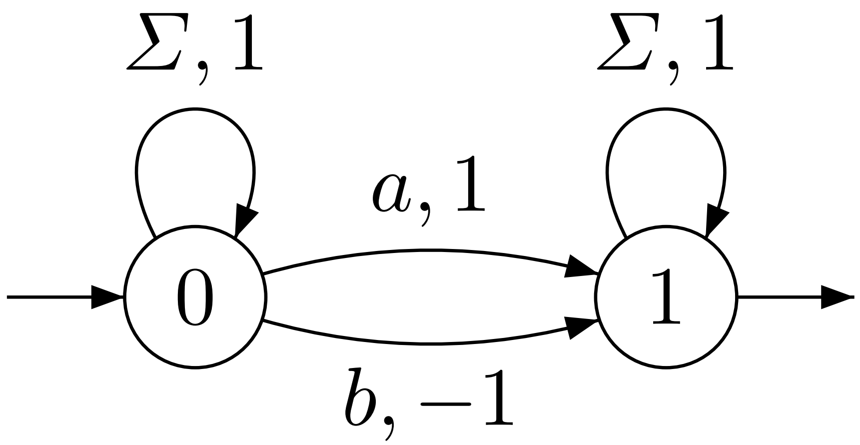
Focus on computation / quantitative

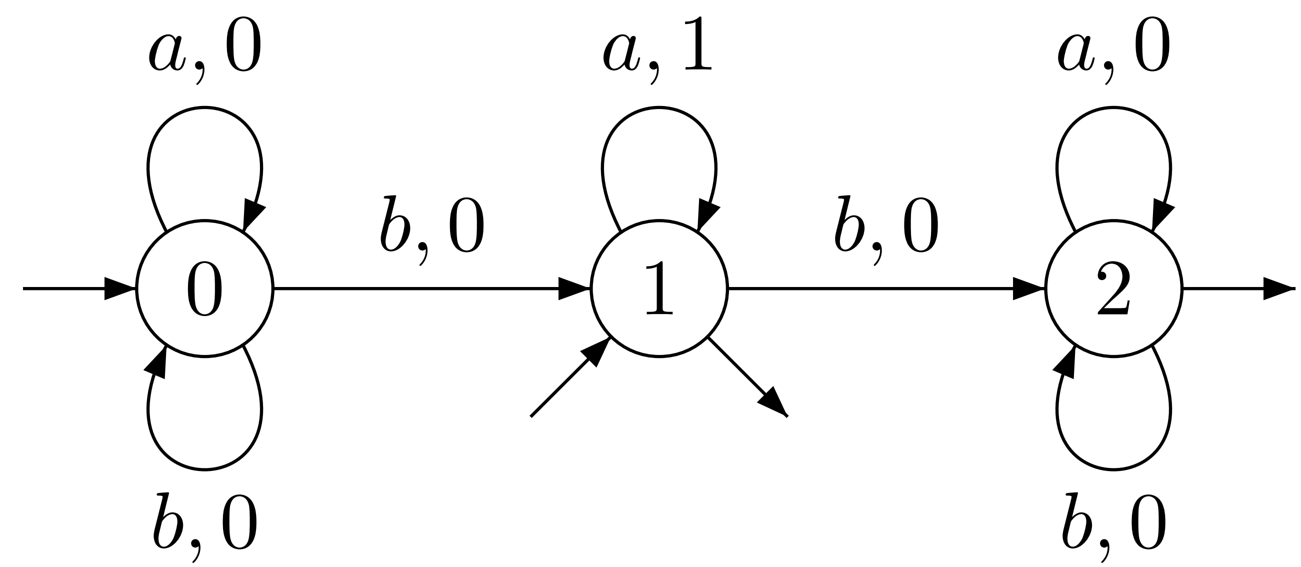
Weighted Automata — **???**

Find suitable weighted MSO logic

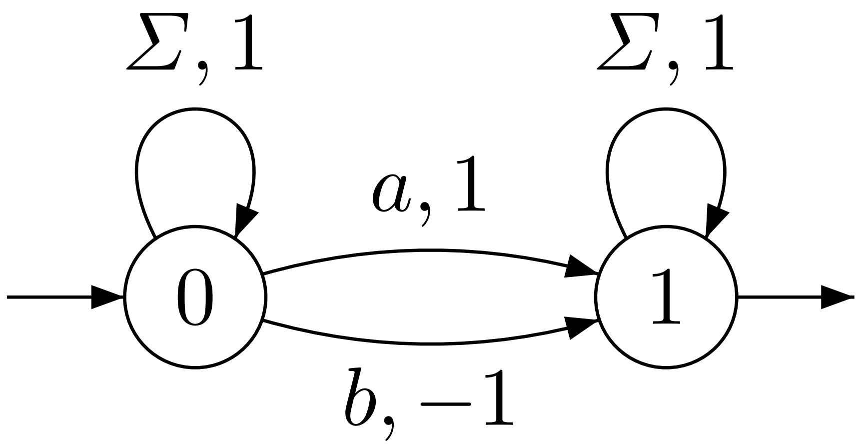
- Quantitative, weights

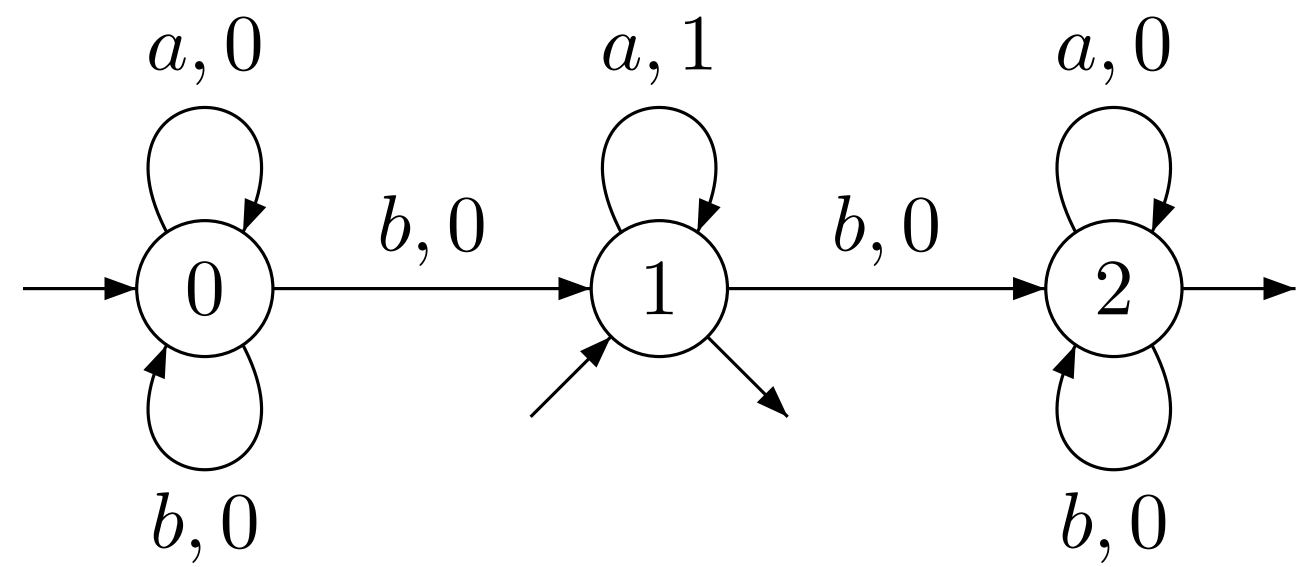
Weighted Automata





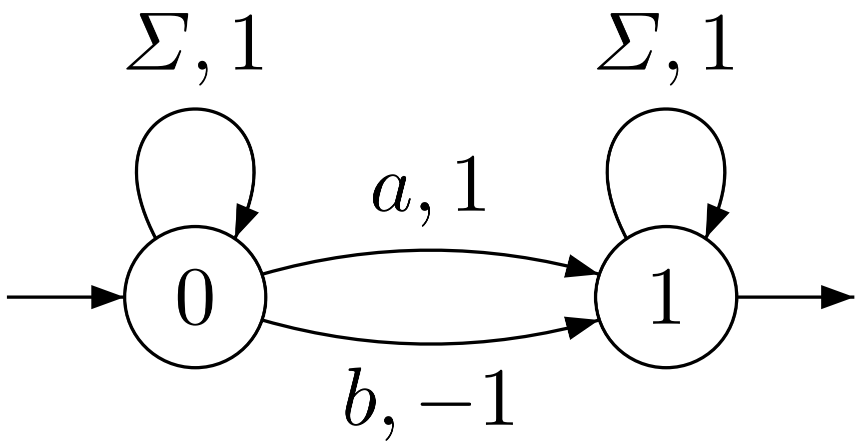
Weighted Automata

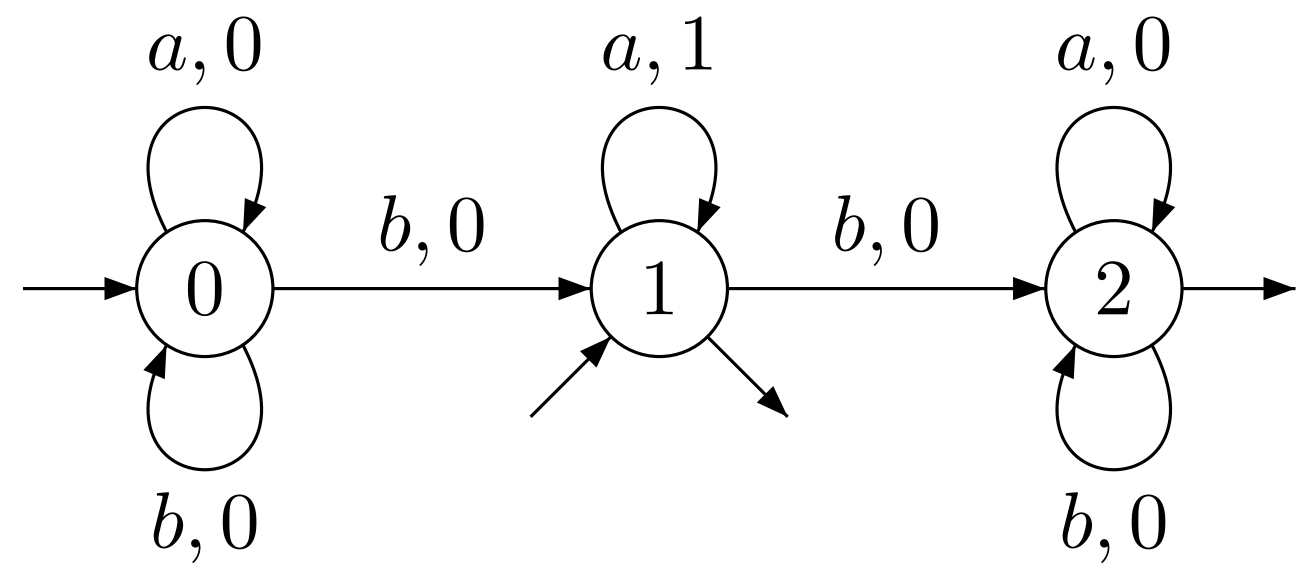




$(\mathbf{Z}, +, \times, 0, 1)$

Weighted Automata



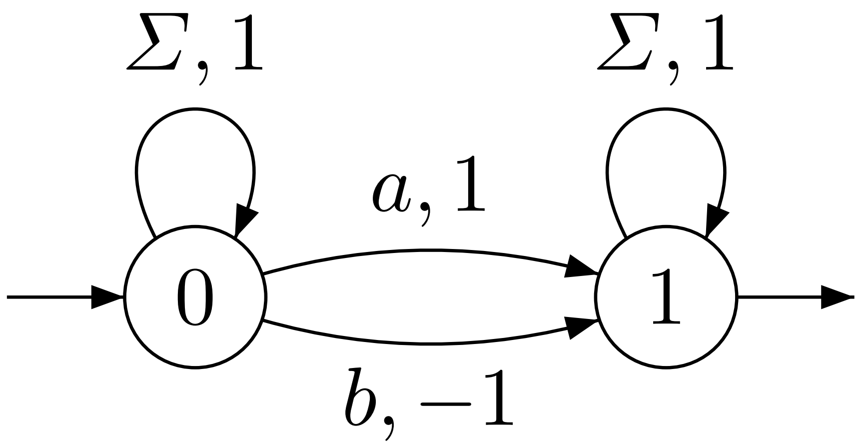


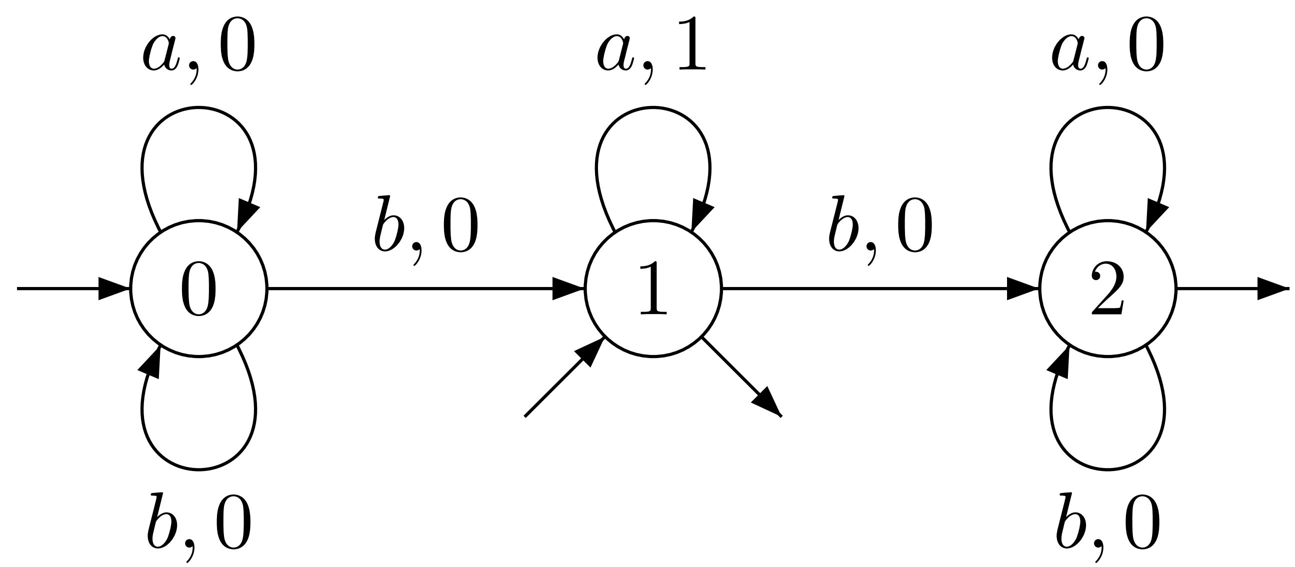
$$(\mathbf{Z}, +, \times, 0, 1)$$

$$0 \xrightarrow{a,1} 1 \xrightarrow{b,1} 1 \xrightarrow{a,1} 1$$

1

Weighted Automata





$$(\mathbf{Z}, +, \times, 0, 1)$$

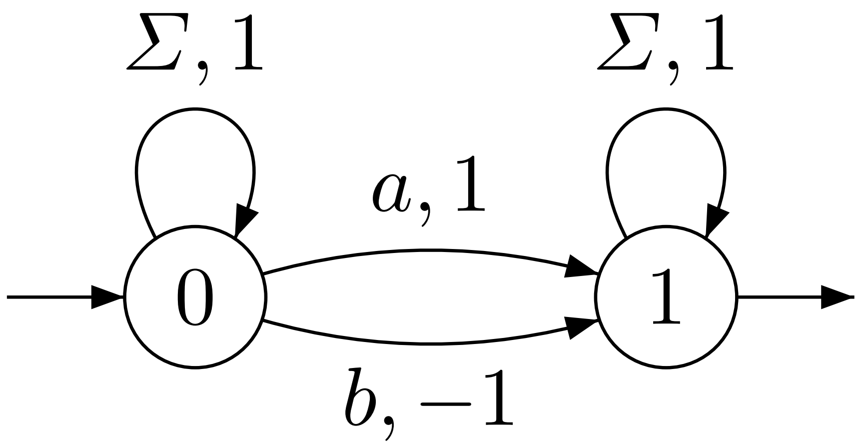
$$0 \xrightarrow{a,1} 1 \xrightarrow{b,1} 1 \xrightarrow{a,1} 1$$

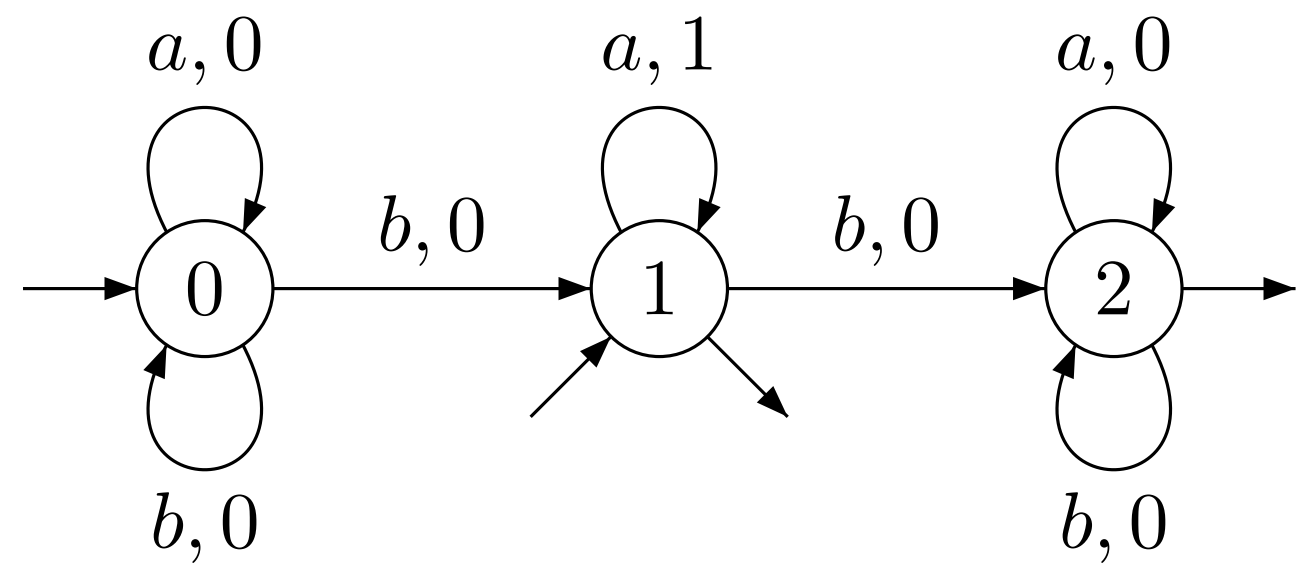
1

$$0 \xrightarrow{a,1} 0 \xrightarrow{b,-1} 1 \xrightarrow{a,1} 1$$

-1

Weighted Automata





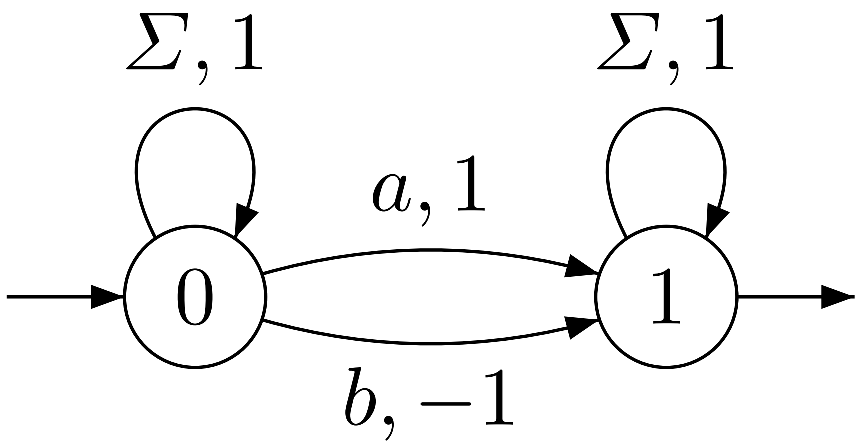
$$(\mathbf{Z}, +, \times, 0, 1)$$

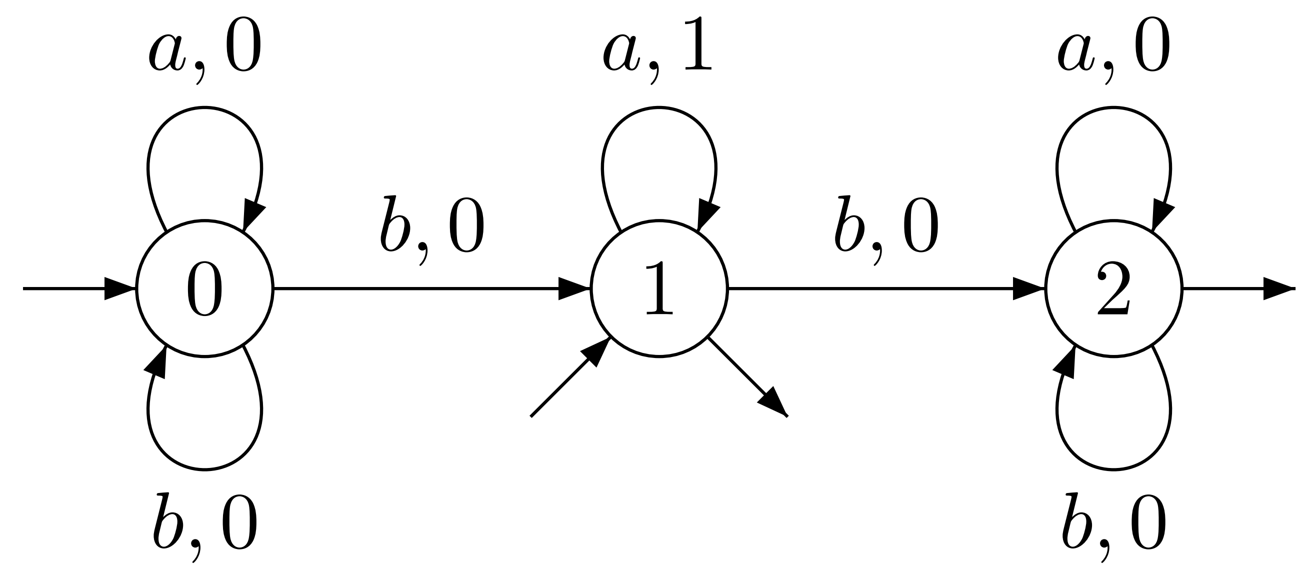
$0 \xrightarrow{a,1} 1 \xrightarrow{b,1} 1 \xrightarrow{a,1} 1$ — 1

$0 \xrightarrow{a,1} 0 \xrightarrow{b,-1} 1 \xrightarrow{a,1} 1$ — -1

$0 \xrightarrow{a,1} 0 \xrightarrow{b,1} 0 \xrightarrow{a,1} 1$ — 1

Weighted Automata





$$(\mathbf{Z}, +, \times, 0, 1)$$

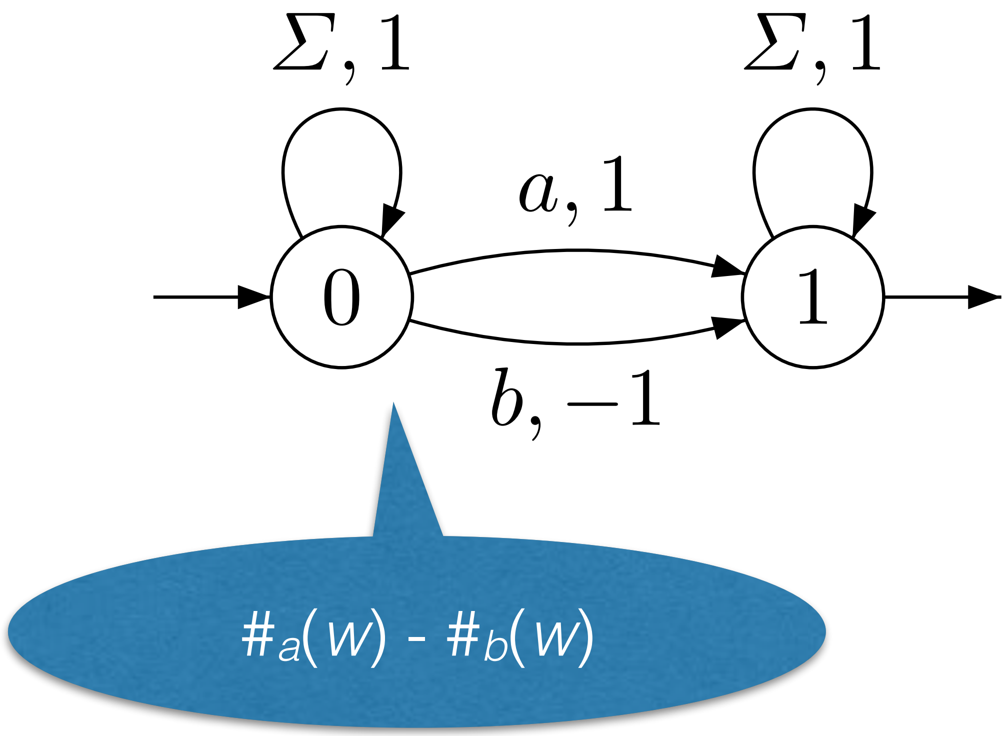
$$0 \xrightarrow{a,1} 1 \xrightarrow{b,1} 1 \xrightarrow{a,1} 1$$ 1

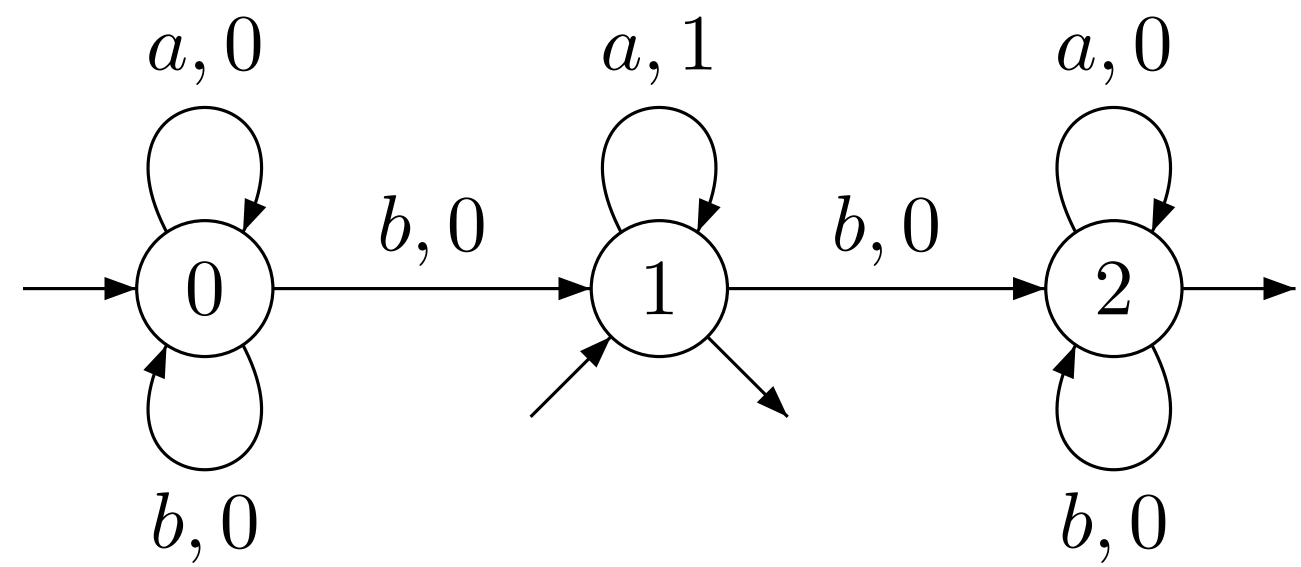
$$0 \xrightarrow{a,1} 0 \xrightarrow{b,-1} 1 \xrightarrow{a,1} 1$$ -1

$$0 \xrightarrow{a,1} 0 \xrightarrow{b,1} 0 \xrightarrow{a,1} 1$$ 1

Semantics of *aba*: 1+(-1)+1 = 1

Weighted Automata





$$(\mathbf{Z}, +, \times, 0, 1)$$

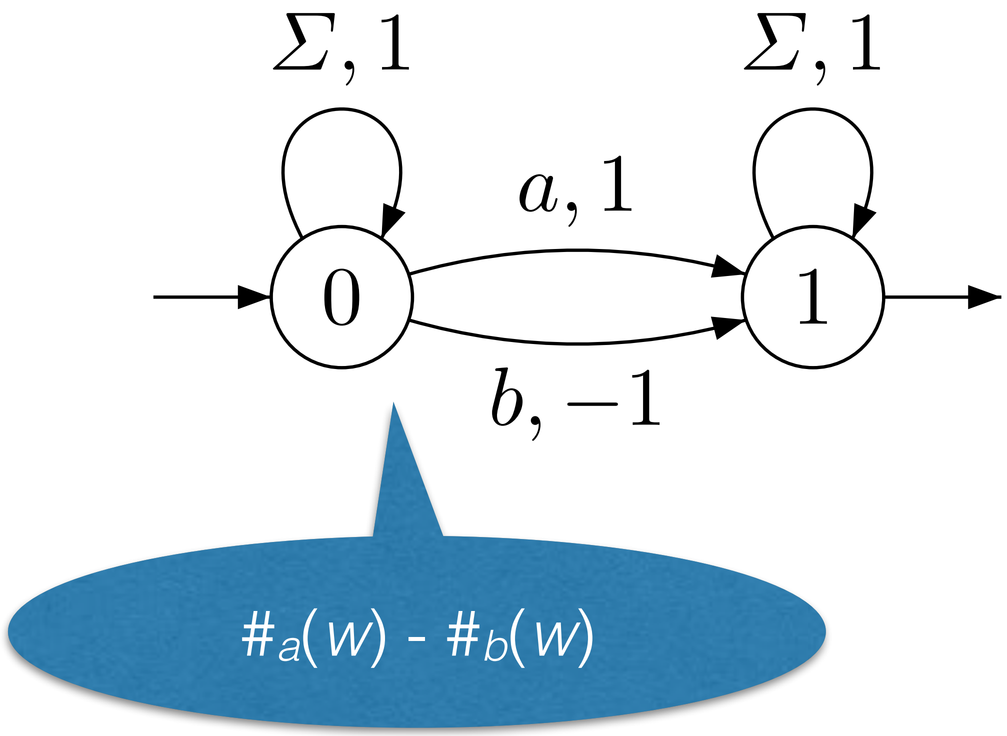
$$0 \xrightarrow{a,1} 1 \xrightarrow{b,1} 1 \xrightarrow{a,1} 1$$ 1

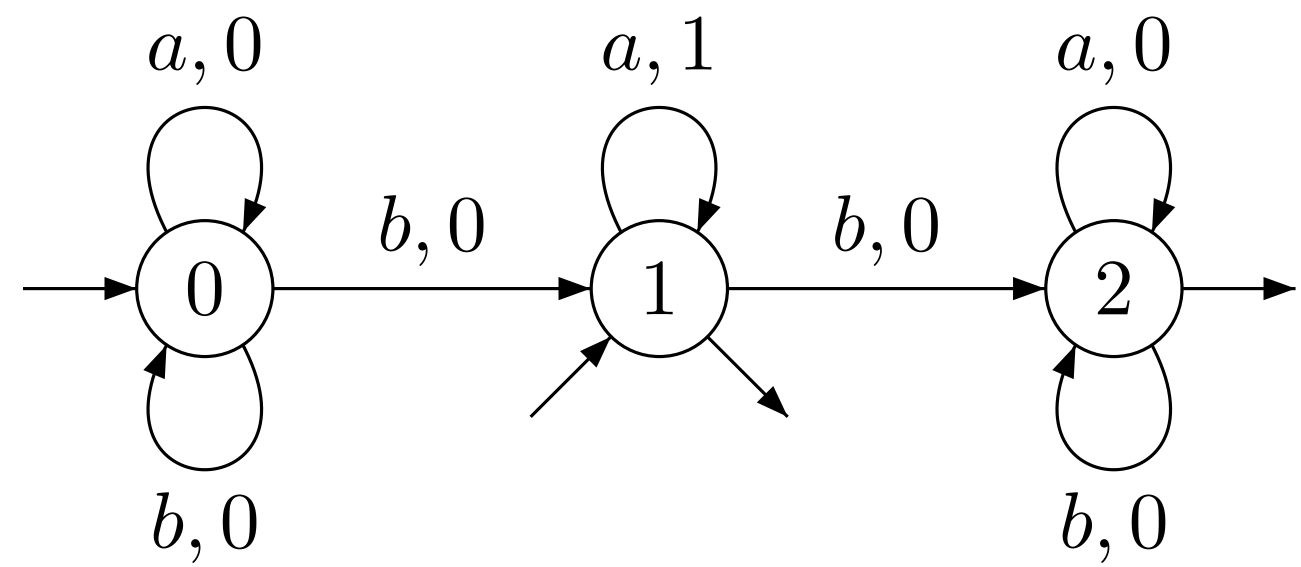
$$0 \xrightarrow{a,1} 0 \xrightarrow{b,-1} 1 \xrightarrow{a,1} 1$$ -1

$$0 \xrightarrow{a,1} 0 \xrightarrow{b,1} 0 \xrightarrow{a,1} 1$$ 1

Semantics of *aba*: 1+(-1)+1 = 1

Weighted Automata





$(\mathbf{Z}, +, \times, 0, 1)$

$(\mathbf{Z} \cup \{-\infty\}, \max, +, -\infty, 0)$

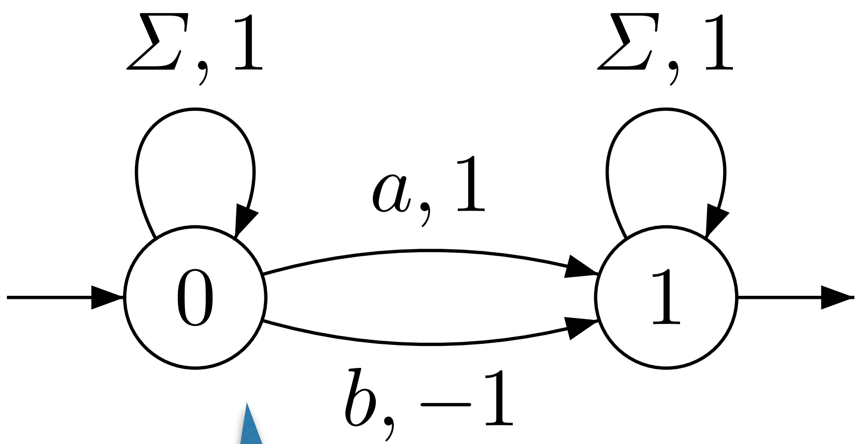
$0 \xrightarrow{a,1} 1 \xrightarrow{b,1} 1 \xrightarrow{a,1} 1$ — 1

$0 \xrightarrow{a,1} 0 \xrightarrow{b,-1} 1 \xrightarrow{a,1} 1$ — -1

$0 \xrightarrow{a,1} 0 \xrightarrow{b,1} 0 \xrightarrow{a,1} 1$ — 1

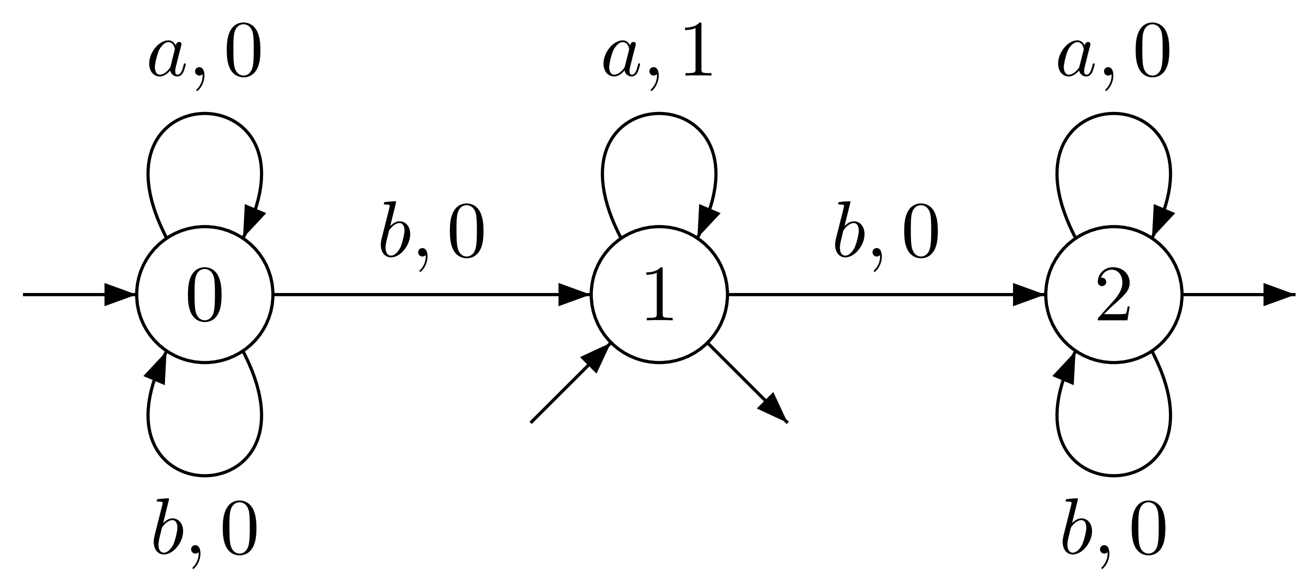
Semantics of *aba*: 1+(-1)+1 = 1

Weighted Automata



$\#_a(w)$ - $\#_b(w)$

$(\mathbf{Z}, +, \times, 0, 1)$



$(\mathbf{Z} \cup \{-\infty\}, \max, +, -\infty, 0)$

$0 \xrightarrow{a,1} 1 \xrightarrow{b,1} 1 \xrightarrow{a,1} 1$ — 1

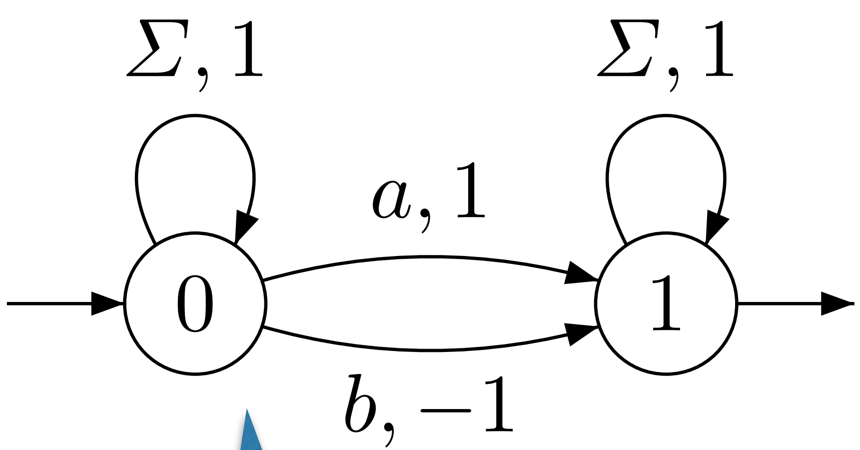
$0 \xrightarrow{a,1} 0 \xrightarrow{b,-1} 1 \xrightarrow{a,1} 1$ — -1

$0 \xrightarrow{a,1} 0 \xrightarrow{b,1} 0 \xrightarrow{a,1} 1$ — 1

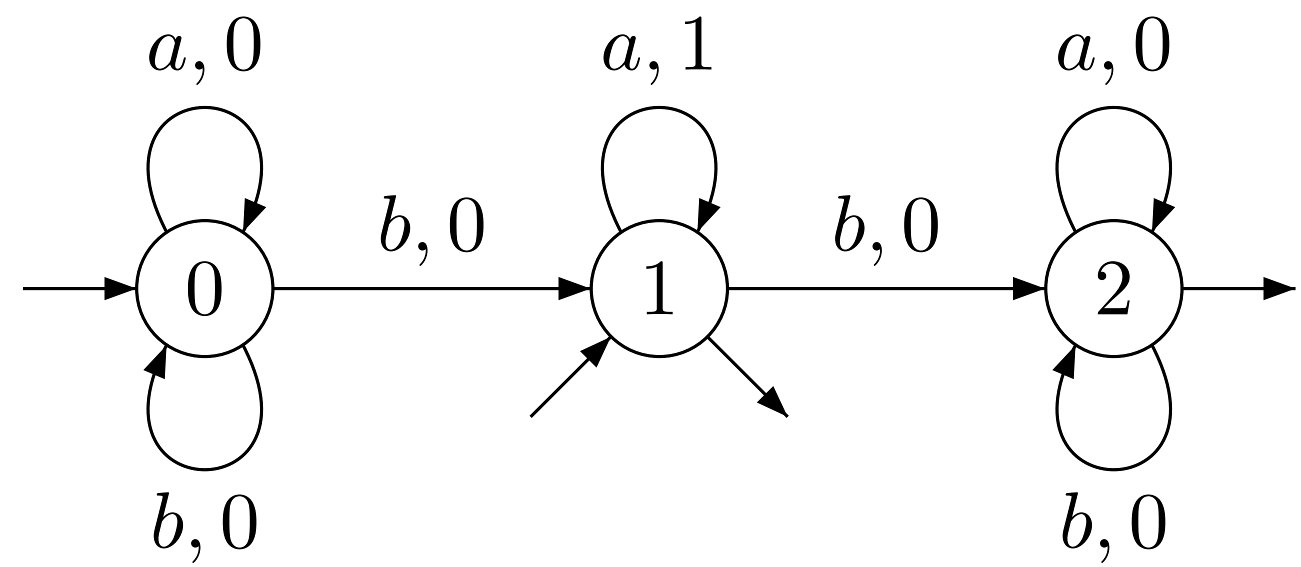
Semantics of *aba*: 1+(-1)+1 = 1

$1 \xrightarrow{a,1} 1 \xrightarrow{a,1} 1 \xrightarrow{b,0} 2 \xrightarrow{a,0} 2$ — 2

Weighted Automata



$\#_a(w)$ - $\#_b(w)$



$(\mathbf{Z}, +, \times, 0, 1)$

$(\mathbf{Z} \cup \{-\infty\}, \max, +, -\infty, 0)$

$0 \xrightarrow{a,1} 1 \xrightarrow{b,1} 1 \xrightarrow{a,1} 1$ — 1

$0 \xrightarrow{a,1} 0 \xrightarrow{b,-1} 1 \xrightarrow{a,1} 1$ — -1

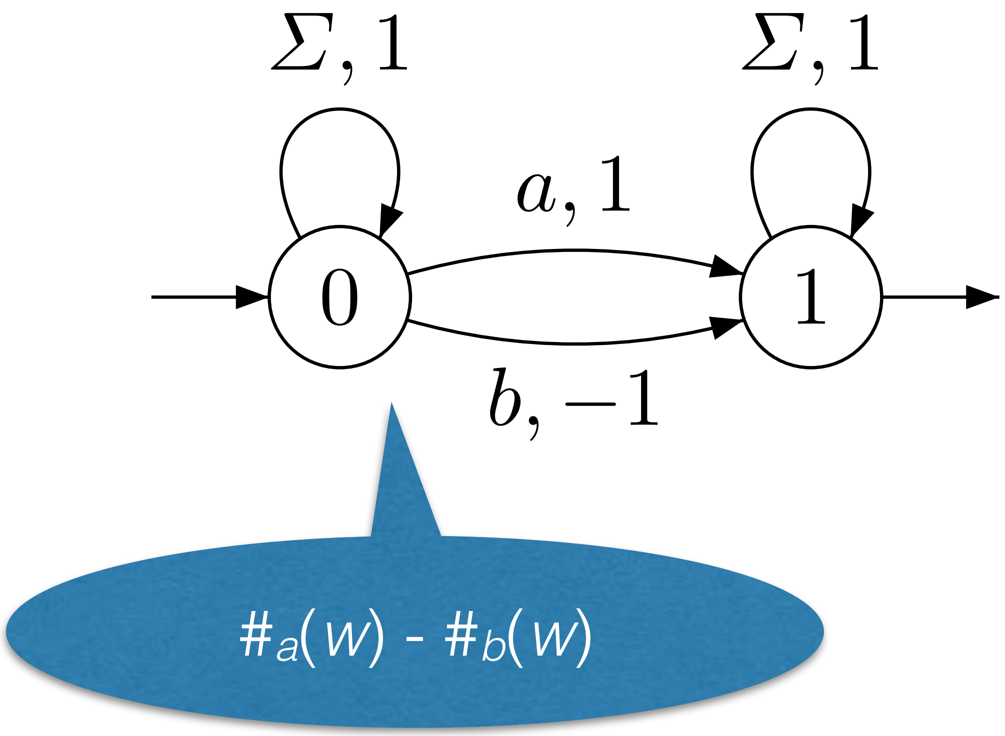
$0 \xrightarrow{a,1} 0 \xrightarrow{b,1} 0 \xrightarrow{a,1} 1$ — 1

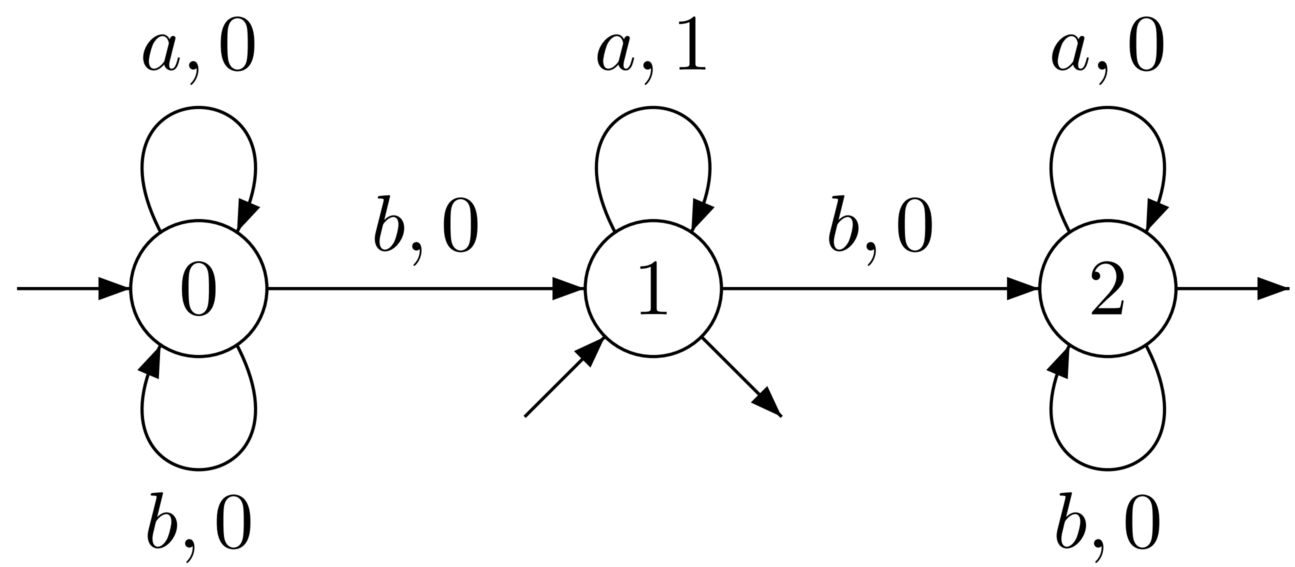
$1 \xrightarrow{a,1} 1 \xrightarrow{a,1} 1 \xrightarrow{b,0} 2 \xrightarrow{a,0} 2$ — 2

$0 \xrightarrow{a,0} 0 \xrightarrow{a,0} 0 \xrightarrow{b,0} 1 \xrightarrow{a,1} 1$ — 1

Semantics of *aba*: 1+(-1)+1 = 1

Weighted Automata





$(\mathbf{Z}, +, \times, 0, 1)$

$(\mathbf{Z} \cup \{-\infty\}, \max, +, -\infty, 0)$

$0 \xrightarrow{a,1} 1 \xrightarrow{b,1} 1 \xrightarrow{a,1} 1$ — 1

$0 \xrightarrow{a,1} 0 \xrightarrow{b,-1} 1 \xrightarrow{a,1} 1$ — -1

$0 \xrightarrow{a,1} 0 \xrightarrow{b,1} 0 \xrightarrow{a,1} 1$ — 1

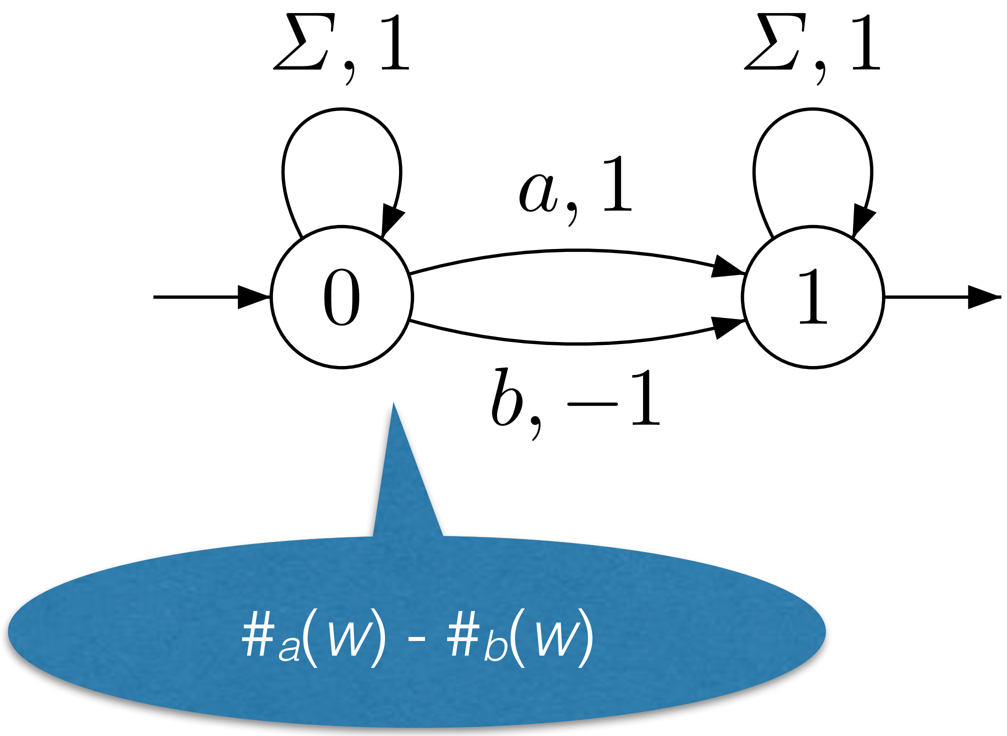
$1 \xrightarrow{a,1} 1 \xrightarrow{a,1} 1 \xrightarrow{b,0} 2 \xrightarrow{a,0} 2$ — 2

$0 \xrightarrow{a,0} 0 \xrightarrow{a,0} 0 \xrightarrow{b,0} 1 \xrightarrow{a,1} 1$ — 1

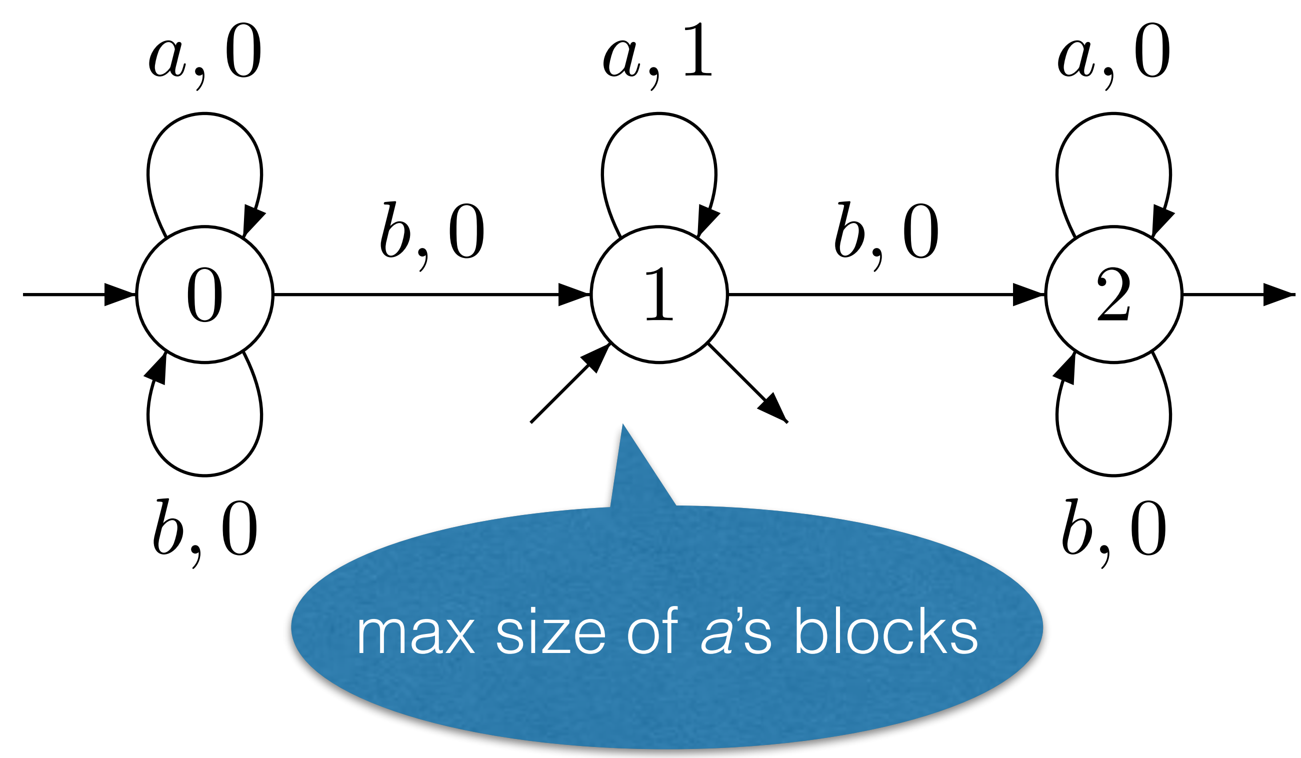
Semantics of *aba*: 1+(-1)+1 = 1

Semantics of *aaba*: max(2,1) = 2

Weighted Automata



$(\mathbf{Z}, +, \times, 0, 1)$



$(\mathbf{Z} \cup \{-\infty\}, \max, +, -\infty, 0)$

$0 \xrightarrow{a,1} 1 \xrightarrow{b,1} 1 \xrightarrow{a,1} 1$ 1

$0 \xrightarrow{a,1} 0 \xrightarrow{b,-1} 1 \xrightarrow{a,1} 1$ -1

$0 \xrightarrow{a,1} 0 \xrightarrow{b,1} 0 \xrightarrow{a,1} 1$ 1

$1 \xrightarrow{a,1} 1 \xrightarrow{a,1} 1 \xrightarrow{b,0} 2 \xrightarrow{a,0} 2$ 2

$0 \xrightarrow{a,0} 0 \xrightarrow{a,0} 0 \xrightarrow{b,0} 1 \xrightarrow{a,1} 1$ 1

Semantics of *aba*: 1+(-1)+1 = 1

Semantics of *aaba*: max(2,1) = 2

How to Specify Quantitative Properties?

How to Specify Quantitative Properties?

Weighted Monadic Second Order Logic [Droste&Gastin 05] generalized to trees [Droste&Vogler 06], infinite words [Droste&Rahonis 07], nested words [Mathissen 10] or pictures [Fichtner 11]

How to Specify Quantitative Properties?

Weighted Monadic Second Order Logic [Droste&Gastin 05] generalized to trees [Droste&Vogler 06], infinite words [Droste&Rahonis 07], nested words [Mathissen 10] or pictures [Fichtner 11]

Weighted Regular Expressions over finite words [Kleene 56, Schützenberger 61]

How to Specify Quantitative Properties?

Weighted Monadic Second Order Logic [Droste&Gastin 05] generalized to trees [Droste&Vogler 06], infinite words [Droste&Rahonis 07], nested words [Mathissen 10] or pictures [Fichtner 11]

Weighted Regular Expressions over finite words [Kleene 56, Schützenberger 61]

Weighted Temporal Logics:
PCTL [Hansson&Jonsson 94], WLTL [Mandrali 12]

How to Specify Quantitative Properties?

Weighted Monadic Second Order Logic [Droste&Gastin 05] generalized to trees [Droste&Vogler 06], infinite words [Droste&Rahonis 07], nested words [Mathissen 10] or pictures [Fichtner 11]

Weighted Regular Expressions over finite words [Kleene 56, Schützenberger 61]

Weighted Temporal Logics:
PCTL [Hansson&Jonsson 94], WLTL [Mandrali 12]

- Core weighted logic for weighted automata
- Enhancing the logic to handle more properties: FO vs pebbles
- A special case: the transducers

Monadic Second Order Logic (MSO)

$$\varphi ::= \top \mid P_a(x) \mid x \leq y \mid x \in X \mid \neg\varphi \mid \varphi \wedge \varphi \mid \forall x\ \varphi \mid \forall X\ \varphi$$

Monadic Second Order Logic (MSO)

$$\varphi ::= \top \mid P_a(x) \mid x \leq y \mid x \in X \mid \neg\varphi \mid \varphi \wedge \varphi \mid \forall x\ \varphi \mid \forall X\ \varphi$$

- Examples

$$\varphi_1 = \exists x\, P_a(x)$$

Monadic Second Order Logic (MSO)

$$\varphi ::= \top \mid P_a(x) \mid x \leq y \mid x \in X \mid \neg\varphi \mid \varphi \wedge \varphi \mid \forall x\ \varphi \mid \forall X\ \varphi$$

- Examples

$$\varphi_1 = \exists x\, P_a(x)$$

« There is a letter a
in the word »

Monadic Second Order Logic (MSO)

$$\varphi ::= \top \mid P_a(x) \mid x \leq y \mid x \in X \mid \neg\varphi \mid \varphi \wedge \varphi \mid \forall x\ \varphi \mid \forall X\ \varphi$$

- Examples

$$\varphi_1 = \exists x\, P_a(x)$$

« There is a letter a in the word »

$$\varphi_2 = \forall x\, \exists y\, (y \leqslant x \wedge P_a(y))$$

Monadic Second Order Logic (MSO)

$$\varphi ::= \top \mid P_a(x) \mid x \leq y \mid x \in X \mid \neg\varphi \mid \varphi \wedge \varphi \mid \forall x\ \varphi \mid \forall X\ \varphi$$

- Examples

$$\varphi_1 = \exists x\, P_a(x)$$

« There is a letter a in the word »

$$\varphi_2 = \forall x\, \exists y\, (y \leqslant x \wedge P_a(y))$$

« The first letter of the word is a. »

Monadic Second Order Logic (MSO)

$$\varphi ::= \top \mid P_a(x) \mid x \leq y \mid x \in X \mid \neg\varphi \mid \varphi \wedge \varphi \mid \forall x\ \varphi \mid \forall X\ \varphi$$

- Examples

$$\varphi_1 = \exists x\, P_a(x)$$

« There is a letter a in the word »

$$\varphi_2 = \forall x\, \exists y\, (y \leqslant x \wedge P_a(y))$$

« The first letter of the word is a. »

$$\varphi_3 = \exists X \forall x \forall y\ \ (\varphi_{first}(x) \Rightarrow x \in X) \wedge (y = x + 1 \Rightarrow (x \in X \Leftrightarrow y \notin X)) \wedge (\varphi_{last}(x) \Rightarrow x \notin X)$$

Monadic Second Order Logic (MSO)

$$\varphi ::= \top \mid P_a(x) \mid x \leq y \mid x \in X \mid \neg\varphi \mid \varphi \wedge \varphi \mid \forall x\ \varphi \mid \forall X\ \varphi$$

- **Examples**

$$\varphi_1 = \exists x\, P_a(x)$$

« There is a letter a in the word »

$$\varphi_2 = \forall x\, \exists y\, (y \leqslant x \wedge P_a(y))$$

« The first letter of the word is a. »

$$\varphi_3 = \exists X \forall x \forall y\ \ (\varphi_{first}(x) \Rightarrow x \in X) \wedge (y = x+1 \Rightarrow (x \in X \Leftrightarrow y \notin X)) \wedge (\varphi_{last}(x) \Rightarrow x \notin X)$$

« The word has even length. »

Weighted MSO

$$\varphi ::= s \mid P_a(x) \mid x \leqslant y \mid x \in X \mid \neg P_a(x) \mid \neg(x \leqslant y) \mid \neg(x \in X)$$
$$\mid \varphi \vee \varphi \mid \varphi \wedge \varphi \mid \exists x\, \varphi \mid \forall x\, \varphi \mid \exists X\, \varphi \mid \forall X\, \varphi$$

[Droste-Gastin, 05]

Weighted MSO

$$\varphi ::= s \mid P_a(x) \mid x \leqslant y \mid x \in X \mid \neg P_a(x) \mid \neg(x \leqslant y) \mid \neg(x \in X)$$
$$\mid \varphi \vee \varphi \mid \varphi \wedge \varphi \mid \exists x\, \varphi \mid \forall x\, \varphi \mid \exists X\, \varphi \mid \forall X\, \varphi$$

Negation restricted to atomic formulae

[Droste-Gastin, 05]

Weighted MSO

$$\varphi ::= s \mid P_a(x) \mid x \leqslant y \mid x \in X \mid \neg P_a(x) \mid \neg(x \leqslant y) \mid \neg(x \in X)$$
$$\mid \varphi \vee \varphi \mid \varphi \wedge \varphi \mid \exists x\, \varphi \mid \forall x\, \varphi \mid \exists X\, \varphi \mid \forall X\, \varphi$$

Negation restricted to atomic formulae

Arbitrary constants from a semiring

[Droste-Gastin, 05]

Weighted MSO

$$\varphi ::= s \mid P_a(x) \mid x \leqslant y \mid x \in X \mid \neg P_a(x) \mid \neg(x \leqslant y) \mid \neg(x \in X)$$
$$\mid \varphi \vee \varphi \mid \varphi \wedge \varphi \mid \exists x\, \varphi \mid \forall x\, \varphi \mid \exists X\, \varphi \mid \forall X\, \varphi$$

- Semantics in a semiring $\mathbb{S} = (S, +, \times, 0, 1)$
  - Atomic formulae: 0, 1
  - disjunction, existential quantifications: sum
  - conjunction, universal quantifications: product
- Inspired from the boolean semiring $\mathbb{B} = (\{0,1\}, \vee, \wedge, 0, 1)$

[Droste-Gastin, 05]

Weighted MSO

$$\varphi ::= s \mid P_a(x) \mid x \leqslant y \mid x \in X \mid \neg P_a(x) \mid \neg(x \leqslant y) \mid \neg(x \in X)$$
$$\mid \varphi \vee \varphi \mid \varphi \wedge \varphi \mid \exists x\, \varphi \mid \forall x\, \varphi \mid \exists X\, \varphi \mid \forall X\, \varphi$$

- Examples

$$\varphi_1 = \exists x\, P_a(x)$$

$$[\![\varphi_1]\!](w) = |w|_a$$

[Droste-Gastin, 05]

Weighted MSO

$$\varphi ::= s \mid P_a(x) \mid x \leqslant y \mid x \in X \mid \neg P_a(x) \mid \neg(x \leqslant y) \mid \neg(x \in X)$$
$$\mid \varphi \vee \varphi \mid \varphi \wedge \varphi \mid \exists x\, \varphi \mid \forall x\, \varphi \mid \exists X\, \varphi \mid \forall X\, \varphi$$

- Examples

$$\varphi_1 = \exists x\, P_a(x) \qquad \varphi_2 = \forall x\, \exists y\, (y \leqslant x \wedge P_a(y))$$

$$[\![\varphi_1]\!](w) = |w|_a \qquad [\![\varphi_2]\!](abaab) = 1 \times 1 \times 2 \times 3 \times 3$$

$$[\![\varphi_2]\!](a^n) = n!$$

[Droste-Gastin, 05]

Weighted MSO

$$\varphi ::= s \mid P_a(x) \mid x \leqslant y \mid x \in X \mid \neg P_a(x) \mid \neg(x \leqslant y) \mid \neg(x \in X)$$
$$\mid \varphi \vee \varphi \mid \varphi \wedge \varphi \mid \exists x\, \varphi \mid \forall x\, \varphi \mid \exists X\, \varphi \mid \forall X\, \varphi$$

- Examples

$$\varphi_1 = \exists x\, P_a(x) \qquad \varphi_2 = \forall x\, \exists y\, (y \leqslant x \wedge P_a(y))$$

$$[\![\varphi_1]\!](w) = |w|_a \qquad [\![\varphi_2]\!](abaab) = 1 \times 1 \times 2 \times 3 \times 3$$

$$[\![\varphi_2]\!](a^n) = n!$$

Too big to be computed by a weighted automaton

[Droste-Gastin, 05]

Weighted MSO

$$\varphi ::= s \mid P_a(x) \mid x \leqslant y \mid x \in X \mid \neg P_a(x) \mid \neg(\ldots x \in X)$$

$$\mid \varphi \vee \varphi \mid \varphi \wedge \varphi \mid \ldots$$

We need to restrict weighted MSO

- Examples

$$\varphi_1 = \exists x\, P_a(x) \qquad \varphi_2 = \forall x\, \exists y\, (y \leqslant x \wedge P_a(y))$$

$$[\![\varphi_1]\!](w) = |w|_a \qquad [\![\varphi_2]\!](abaab) = 1 \times 1 \times 2 \times 3 \times 3$$

$$[\![\varphi_2]\!](a^n) = n!$$

Too big to be computed by a weighted automaton

[Droste-Gastin, 05]

Weighted MSO

$$\varphi ::= s \mid P_a(x) \mid x \leqslant y \mid x \in X \mid \neg P_a(x) \mid \neg(x \leqslant y) \mid \neg(x \in X)$$
$$\mid \varphi \vee \varphi \mid \varphi \wedge \varphi \mid \exists x\, \varphi \mid \forall x\, \varphi \mid \exists X\, \varphi \mid \forall X\, \varphi$$

Theorem: weighted automata = restricted wMSO

[Droste-Gastin, 05]

Weighted MSO

$$\varphi ::= s \mid P_a(x) \mid x \leqslant y \mid x \in X \mid \neg P_a(x) \mid \neg(x \leqslant y) \mid \neg(x \in X)$$
$$\mid \varphi \vee \varphi \mid \varphi \wedge \varphi \mid \exists x\, \varphi \mid \forall x\, \varphi \mid \exists X\, \varphi \mid \forall X\, \varphi$$

Theorem: weighted automata = restricted wMSO

[Droste-Gastin, 05]

Weighted MSO

$$\varphi ::= s \mid P_a(x) \mid x \leqslant y \mid x \in X \mid \neg P_a(x) \mid \neg(x \leqslant y) \mid \neg(x \in X)$$
$$\mid \varphi \vee \varphi \mid \varphi \wedge \varphi \mid \exists x\, \varphi \mid \forall x\, \varphi \mid \exists X\, \varphi \mid \forall X\, \varphi$$

φ almost boolean

Theorem: weighted automata = restricted wMSO

[Droste-Gastin, 05]

Weighted MSO

$$\varphi ::= s \mid P_a(x) \mid x \leqslant y \mid x \in X \mid \neg P_a(x) \mid \neg(x \leqslant y) \mid \neg(x \in X)$$
$$\mid \varphi \vee \varphi \mid \varphi \wedge \varphi \mid \exists x\, \varphi \mid \forall x\, \varphi \mid \exists X\, \varphi \mid \forall X\, \varphi$$

commutativity

φ almost boolean

Theorem: weighted automata = restricted wMSO

[Droste-Gastin, 05]

Core weighted MSO logic

- Boolean fragment

$$\varphi ::= \top \mid P_a(x) \mid x \leqslant y \mid x \in X \mid \neg\varphi \mid \varphi \wedge \varphi \mid \forall x\, \varphi \mid \forall X\, \varphi$$

[Gastin-Monmege, 15]

Core weighted MSO logic

- Boolean fragment

$$\varphi ::= \top \mid P_a(x) \mid x \leqslant y \mid x \in X \mid \neg\varphi \mid \varphi \wedge \varphi \mid \forall x\, \varphi \mid \forall X\, \varphi$$

- Step formulae $\quad \Psi ::= s \mid \varphi\,?\,\Psi : \Psi$

[Gastin-Monmege, 15]

Core weighted MSO logic

- Boolean fragment

$$\varphi ::= \top \mid P_a(x) \mid x \leqslant y \mid x \in X \mid \neg\varphi \mid \varphi \wedge \varphi \mid \forall x\, \varphi \mid \forall X\, \varphi$$

- Step formulae $\Psi ::= s \mid \varphi \,?\, \Psi : \Psi$

if … then … else …

[Gastin-Monmege, 15]

Core weighted MSO logic

- Boolean fragment

$$\varphi ::= \top \mid P_a(x) \mid x \leqslant y \mid x \in X \mid \neg\varphi \mid \varphi \wedge \varphi \mid \forall x\, \varphi \mid \forall X\, \varphi$$

- Step formulae $\Psi ::= s \mid \varphi \,?\, \Psi : \Psi$

if … then … else …

$$P_a(x) \,?\, 1 : 0$$

[Gastin-Monmege, 15]

Core weighted MSO logic

- Boolean fragment

$$\varphi ::= \top \mid P_a(x) \mid x \leqslant y \mid x \in X \mid \neg\varphi \mid \varphi \wedge \varphi \mid \forall x\, \varphi \mid \forall X\, \varphi$$

- Step formulae $\Psi ::= s \mid \varphi \,?\, \Psi : \Psi$

if ... then ... else ...

$$P_a(x)\,?\,1 : 0$$

$$P_a(x)\,?\,1 : (P_b(x)\,?\,-1 : 0)$$

[Gastin-Monmege, 15]

Core weighted MSO logic

- Boolean fragment

$$\varphi ::= \top \mid P_a(x) \mid x \leqslant y \mid x \in X \mid \neg\varphi \mid \varphi \wedge \varphi \mid \forall x\, \varphi \mid \forall X\, \varphi$$

- Step formulae $\Psi ::= s \mid \varphi \,?\, \Psi : \Psi$

if … then … else …

$$P_a(x) \,?\, 1 : 0$$

$$P_a(x) \,?\, 1 : (P_b(x) \,?\, -1 : 0)$$

$$x \in X_1 \,?\, s_1 : (x \in X_2 \,?\, s_2 : \cdots (x \in X_{n-1} \,?\, s_{n-1} : s_n) \cdots)$$

[Gastin-Monmege, 15]

Core weighted MSO logic

- Boolean fragment

$$\varphi ::= \top \mid P_a(x) \mid x \leqslant y \mid x \in X \mid \neg\varphi \mid \varphi \wedge \varphi \mid \forall x\, \varphi \mid \forall X\, \varphi$$

- Step formulae $\Psi ::= s \mid \varphi \,?\, \Psi : \Psi$

if ... then ... else ...

$$P_a(x) \,?\, 1 : 0$$

$$P_a(x) \,?\, 1 : (P_b(x) \,?\, -1 : 0)$$

$$x \in X_1 \,?\, s_1 : (x \in X_2 \,?\, s_2 : \cdots (x \in X_{n-1} \,?\, s_{n-1} : s_n) \cdots)$$

$$[\![\Psi]\!](w, \sigma) = s$$

some value occurring in Ψ

[Gastin-Monmege, 15]

Core weighted MSO logic

- Boolean fragment

$$\varphi ::= \top \mid P_a(x) \mid x \leqslant y \mid x \in X \mid \neg\varphi \mid \varphi \wedge \varphi \mid \forall x\, \varphi \mid \forall X\, \varphi$$

- Step formulae $\quad \Psi ::= s \mid \varphi\,?\,\Psi : \Psi$

$$P_a(x)\,?\,1 : 0$$

P_a ...

$x \in$ ... $X_2\,?\,s_2 : \cdots (x \in X_{n-1}\,?\,s_{n-1} : s_n) \cdots)$

A step formula takes finitely many values
For each value, the pre-image is MSO-definable

$$[\![\Psi]\!](w, \sigma) = s$$

some value occurring in Ψ

[Gastin-Monmege, 15]

Core weighted MSO logic

- Boolean fragment

$$\varphi ::= \top \mid P_a(x) \mid x \leqslant y \mid x \in X \mid \neg\varphi \mid \varphi \wedge \varphi \mid \forall x\, \varphi \mid \forall X\, \varphi$$

- Step formulae $\quad \Psi ::= s \mid \varphi\,?\,\Psi : \Psi$

- core wMSO $\quad \Phi ::= \mathbf{0} \mid \varphi\,?\,\Phi : \Phi \mid \Phi + \Phi \mid \sum_x \Phi \mid \sum_X \Phi \mid \prod_x \Psi$

[Gastin-Monmege, 15]

Core weighted MSO logic

- Boolean fragment

$$\varphi ::= \top \mid P_a(x) \mid x \leqslant y \mid x \in X \mid \neg\varphi \mid \varphi \wedge \varphi \mid \forall x\, \varphi \mid \forall X\, \varphi$$

- Step formulae $\Psi ::= s \mid \varphi \,?\, \Psi : \Psi$

- core wMSO $\Phi ::= \mathbf{0} \mid \varphi \,?\, \Phi : \Phi \mid \Phi + \Phi \mid \sum_x \Phi \mid \sum_X \Phi \mid \prod_x \Psi$

no constants

[Gastin-Monmege, 15]

Core weighted MSO logic

- Boolean fragment

$$\varphi ::= \top \mid P_a(x) \mid x \leqslant y \mid x \in X \mid \neg\varphi \mid \varphi \wedge \varphi \mid \forall x\, \varphi \mid \forall X\, \varphi$$

- Step formulae $\Psi ::= s \mid \varphi\,?\,\Psi : \Psi$

- core wMSO $\Phi ::= \mathbf{0} \mid \varphi\,?\,\Phi : \Phi \mid \Phi + \Phi \mid \sum_x \Phi \mid \sum_X \Phi \mid \prod_x \Psi$

no constants

if … then … else …

[Gastin-Monmege, 15]

Core weighted MSO logic

- Boolean fragment

$$\varphi ::= \top \mid P_a(x) \mid x \leqslant y \mid x \in X \mid \neg\varphi \mid \varphi \wedge \varphi \mid \forall x\, \varphi \mid \forall X\, \varphi$$

- Step formulae $\Psi ::= s \mid \varphi \,?\, \Psi : \Psi$

- core wMSO $\Phi ::= \mathbf{0} \mid \varphi \,?\, \Phi : \Phi \mid \Phi + \Phi \mid \sum_x \Phi \mid \sum_X \Phi \mid \prod_x \Psi$

no constants

if ... then ... else ...

Assigns a value from Ψ to each position

[Gastin-Monmege, 15]

Core weighted MSO logic

- Boolean fragment

$$\varphi ::= \top \mid P_a(x) \mid x \leqslant y \mid x \in X \mid \neg\varphi \mid \varphi \wedge \varphi \mid \forall x\, \varphi \mid \forall X\, \varphi$$

- Step formulae $\quad \Psi ::= s \mid \varphi \,?\, \Psi : \Psi$

- core wMSO $\quad \Phi ::= \mathbf{0} \mid \varphi \,?\, \Phi : \Phi \mid \Phi + \Phi \mid \sum_x \Phi \mid \sum_X \Phi \mid \prod_x \Psi$

no constants

if … then … else …

Assigns a value from Ψ to each position

$$\{\!|\prod_x \Psi|\!\}(w,\sigma) = \{\{(\llbracket\Psi\rrbracket(w,\sigma[x \mapsto i]))_i\}\} \in \mathbb{N}\langle R^\star\rangle$$

singleton multiset

[Gastin-Monmege, 15]

Core weighted MSO logic

- Boolean fragment

$$\varphi ::= \top \mid P_a(x) \mid x \leqslant y \mid x \in X \mid \neg\varphi \mid \varphi \wedge \varphi \mid \forall x\, \varphi \mid \forall X\, \varphi$$

- Step formulae $\Psi ::= s \mid \varphi \,?\, \Psi : \Psi$

- core wMSO $\Phi ::= \mathbf{0} \mid \varphi \,?\, \Phi : \Phi \mid \Phi + \Phi \mid \sum_x \Phi \mid \sum_X \Phi \mid \prod_x \Psi$

- Semantics
  - $\{\!|\, \mathbf{0} \,|\!\}(w,\sigma) = \varnothing$
  - sums over multisets $\{\!|\, \Phi_1 + \Phi_2 \,|\!\}(w,\sigma) = \{\!|\, \Phi_1 \,|\!\}(w,\sigma) \uplus \{\!|\, \Phi_2 \,|\!\}(w,\sigma)$

$$\{\!|\, \prod_x \Psi \,|\!\}(w,\sigma) = \{\{(\llbracket \Psi \rrbracket(w,\sigma[x \mapsto i]))_i\}\} \in \mathbb{N}\langle R^\star \rangle$$

[Gastin-Monmege, 15]

Core weighted MSO logic

$$\Phi = \sum_X \sum_y \varphi_{block}(X, y) ? \prod_x (x \in X ? 1 : 0) : \mathbf{0}$$

Core weighted MSO logic

$$\Phi = \sum_X \sum_y \varphi_{block}(X,y) ? \prod_x (x \in X ? 1 : 0) : \mathbf{0}$$

$$\varphi_{block}(X,y) = \left[X = \varnothing \wedge (\varphi_{first}(y) \vee \varphi_{last}(y)) \wedge P_b(y) \right]$$

Core weighted MSO logic

$$\Phi = \sum_X \sum_y \varphi_{block}(X,y) ? \prod_x (x \in X ? 1 : 0) : \mathbf{0}$$

$$\varphi_{block}(X,y) = \left[X = \varnothing \wedge (\varphi_{first}(y) \vee \varphi_{last}(y)) \wedge P_b(y) \right]$$

$$\vee \left[(\varphi_{last}(y) \vee P_b(y+1)) \wedge \right.$$

Core weighted MSO logic

$$\Phi = \sum_X \sum_y \varphi_{block}(X,y) ? \prod_x (x \in X ? 1 : 0) : \mathbf{0}$$

$$\begin{aligned}\varphi_{block}(X,y) = & \Big[X = \varnothing \wedge (\varphi_{first}(y) \vee \varphi_{last}(y)) \wedge P_b(y)\Big] \\ & \vee \Big[(\varphi_{last}(y) \vee P_b(y+1)) \wedge \\ & \quad \exists x \Big(x \leq y \wedge (\varphi_{first}(x) \vee P_b(x-1)) \wedge\end{aligned}$$

Core weighted MSO logic

$$\Phi = \sum_X \sum_y \varphi_{block}(X,y) ? \prod_x (x \in X ? 1 : 0) : \mathbf{0}$$

$$\begin{aligned}
\varphi_{block}(X,y) = & \left[X = \varnothing \wedge (\varphi_{first}(y) \vee \varphi_{last}(y)) \wedge P_b(y) \right] \\
& \vee \Big[(\varphi_{last}(y) \vee P_b(y+1)) \wedge \\
& \quad \exists x \Big(x \leq y \wedge (\varphi_{first}(x) \vee P_b(x-1)) \wedge \\
& \quad\quad \forall z ((x \leq z \leq y \Leftrightarrow z \in X) \wedge (z \in X \Rightarrow P_a(z))) \Big) \Big]
\end{aligned}$$

Core weighted MSO logic

$$\Phi = \sum_X \sum_y \varphi_{block}(X,y) ? \prod_x (x \in X ? 1 : 0) : \mathbf{0}$$

$$\begin{aligned}\varphi_{block}(X,y) = \Big[X = \varnothing \wedge (\varphi_{first}(y) \vee \varphi_{last}(y)) \wedge P_b(y)\Big] \\ \vee \Big[(\varphi_{last}(y) \vee P_b(y+1)) \wedge \\ \exists x \Big(x \leq y \wedge (\varphi_{first}(x) \vee P_b(x-1)) \wedge \\ \forall z((x \leq z \leq y \Leftrightarrow z \in X) \wedge (z \in X \Rightarrow P_a(z)))\Big)\Big]\end{aligned}$$

$$\{\!|\, \Phi \,|\!\}(baabab) = \{\{000000, 011000, 000010, 000000\}\}$$

Multisets of weight structures

- A run generates a sequence of weights $\mathrm{wgt}(\rho) = s_1 s_2 \cdots s_n$
- Abstract semantics $\{\!|\mathcal{A}|\!\}(w) = \{\{\mathrm{wgt}(\rho) \mid \rho \text{ run on } w\}\}$

multiset

[Gastin-Monmege, 15][Droste-Perevoshchikov, 14]

Multisets of weight structures

- A run generates a sequence of weights $\mathrm{wgt}(\rho) = s_1 s_2 \cdots s_n$
- Abstract semantics $\{\!|\mathcal{A}|\!\}(w) = \{\!\{\mathrm{wgt}(\rho) \mid \rho \text{ run on } w\}\!\}$

multiset

$$\{\!|\mathcal{A}|\!\} : \Sigma^\star \to \mathbb{N}\langle R^\star \rangle$$

weights of A

[Gastin-Monmege, 15][Droste-Perevoshchikov, 14]

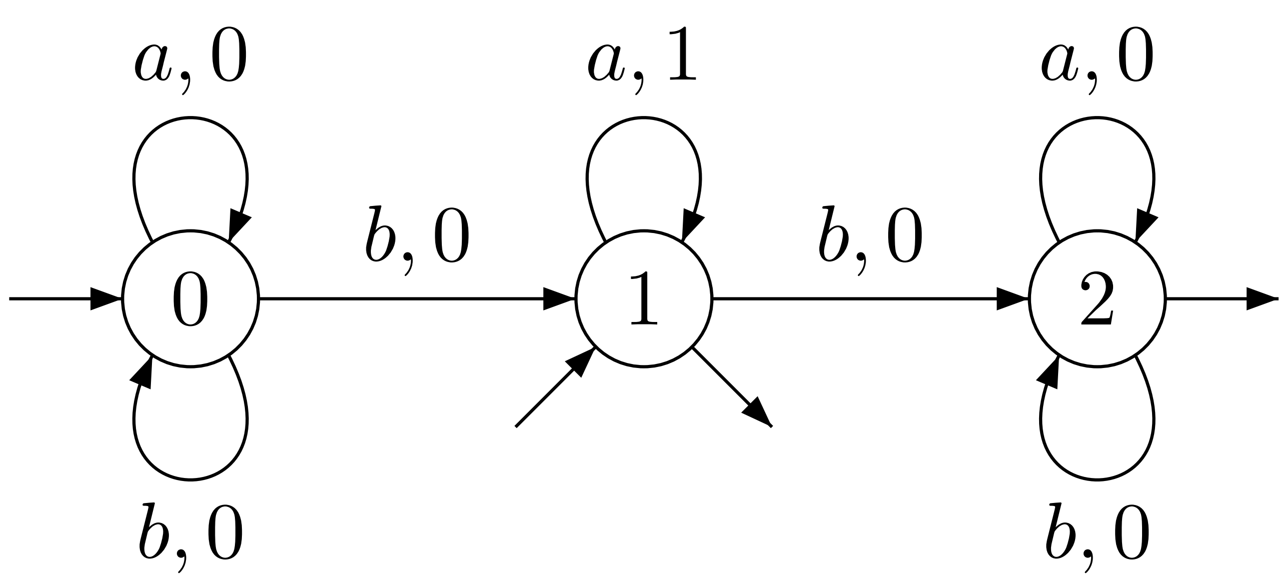
Multisets of weight structures

- A run generates a sequence of weights $\mathrm{wgt}(\rho) = s_1 s_2 \cdots s_n$
- Abstract semantics $\{\!|\mathcal{A}|\!\}(w) = \{\{\mathrm{wgt}(\rho) \mid \rho \text{ run on } w\}\}$

$$\{\!|\mathcal{A}|\!\} : \Sigma^\star \to \mathbb{N}\langle R^\star \rangle$$

multiset

weights of A



$$\{\!|\ \mathcal{A}\ |\!\}(baabab) = \{\{000000, 011000, 000010, 000000\}\}$$

[Gastin-Monmege, 15][Droste-Perevoshchikov, 14]

Multisets of weight structures

- A run generates a sequence of weights $\mathrm{wgt}(\rho) = s_1 s_2 \cdots s_n$
- Abstract semantics $\{\!|\mathcal{A}|\!\}(w) = \{\{\mathrm{wgt}(\rho) \mid \rho \text{ run on } w\}\}$

  $\{\!|\mathcal{A}|\!\} : \Sigma^\star \to \mathbb{N}\langle R^\star \rangle$

  multiset

  weights of A

- Aggregation $\mathrm{aggr} : \mathbb{N}\langle R^\star \rangle \to S$

[Gastin-Monmege, 15][Droste-Perevoshchikov, 14]

Multisets of weight structures

Semiring: sum-product

$$\mathsf{aggr}_{\mathrm{sp}}(A) = \sum \prod A = \sum_{r_1 \cdots r_n \in A} r_1 \times \cdots \times r_n$$

$$\{\!|\mathcal{A}|\!\} \colon \Sigma^\star \to \mathbb{N}\langle R^\star \rangle$$

multiset

weights of A

- Aggregation $\mathsf{aggr} \colon \mathbb{N}\langle R^\star \rangle \to S$

[Gastin-Monmege, 15][Droste-Perevoshchikov, 14]

Multisets of weight structures

Semiring: sum-product

$$\mathsf{aggr}_{\mathtt{sp}}(A) = \sum \prod A = \sum_{r_1 \cdots r_n \in A} r_1 \times \cdots \times r_n$$

Valuation monoid: sum-valuation

$$\mathsf{aggr}_{\mathtt{sv}}(A) = \sum \mathsf{Val}(A) = \sum_{r_1 \cdots r_n \in A} \mathsf{Val}(r_1 \cdots r_n)$$

weights of A

- Aggregation $\mathsf{aggr} \colon \mathbb{N}\langle R^\star \rangle \to S$

[Gastin-Monmege, 15][Droste-Perevoshchikov, 14]

Multisets of weight structures

Semiring: sum-product

$$\mathsf{aggr}_{\mathtt{sp}}(A) = \sum \prod A = \sum_{r_1 \cdots r_n \in A} r_1 \times \cdots \times r_n$$

Valuation monoid: sum-valuation

$$\mathsf{aggr}_{\mathtt{sv}}(A) = \sum \mathsf{Val}(A) = \sum_{r_1 \cdots r_n \in A} \mathsf{Val}(r_1 \cdots r_n)$$

weights of A

Average value
Discounted value…

- Aggregation $\to S$

[Gastin-Monmege, 15][Droste-Perevoshchikov, 14]

Multisets of weight structures

Semiring: sum-product

$$\mathsf{aggr}_{\mathtt{sp}}(A) = \sum \prod A = \sum_{r_1 \cdots r_n \in A} r_1 \times \cdots \times r_n$$

Valuation monoid: sum-valuation

$$\mathsf{aggr}_{\mathtt{sv}}(A) = \sum \mathsf{Val}(A) = \sum_{r_1 \cdots r_n \in A} \mathsf{Val}(r_1 \cdots r_n)$$

Valuation structure: evaluator-valuation

$$\mathsf{aggr}_{\mathsf{ef}}(A) = F(\mathrm{Val}(A)) \text{ with } F \colon \mathbb{N}\langle U \rangle \to S \text{ and } \mathrm{Val} \colon U^\star \to U$$

[Gastin-Monmege, 15][Droste-Perevoshchikov, 14]

Multisets of weight structures

- A run generates a sequence of weights $\mathsf{wgt}(\rho) = s_1 s_2 \cdots s_n$
- Abstract semantics $\{|\mathcal{A}|\}(w) = \{\{\mathsf{wgt}(\rho) \mid \rho \text{ run on } w\}\}$

$$\{|\mathcal{A}|\} \colon \Sigma^\star \to \mathbb{N}\langle R^\star \rangle$$

multiset

weights of A

- Aggregation $\mathsf{aggr} \colon \mathbb{N}\langle R^\star \rangle \to S$
- Concrete semantics $[\![\mathcal{A}]\!] = \mathsf{aggr} \circ \{|\mathcal{A}|\} \colon \Sigma^\star \to S$

[Gastin-Monmege, 15][Droste-Perevoshchikov, 14]

Core weighted MSO logic

$$\varphi ::= \top \mid P_a(x) \mid x \leqslant y \mid x \in X \mid \neg\varphi \mid \varphi \wedge \varphi \mid \forall x\, \varphi \mid \forall X\, \varphi$$

$$\Psi ::= s \mid \varphi\,?\,\Psi : \Psi$$

$$\Phi ::= \mathbf{0} \mid \varphi\,?\,\Phi : \Phi \mid \Phi + \Phi \mid \textstyle\sum_x \Phi \mid \sum_X \Phi \mid \prod_x \Psi$$

Theorem: weighted automata = core wMSO

[Gastin-Monmege, 15]

Core weighted MSO logic

$$\varphi ::= \top \mid P_a(x) \mid x \leqslant y \mid x \in X \mid \neg\varphi \mid \varphi \wedge \varphi \mid \forall x\, \varphi \mid \forall X\, \varphi$$

$$\Psi ::= s \mid \varphi\, ?\, \Psi : \Psi$$

$$\Phi ::= \mathbf{0} \mid \varphi\, ?\, \Phi : \Phi \mid \Phi + \Phi \mid \sum_x \Phi \mid \sum_X \Phi \mid \prod_x \Psi$$

Theorem: weighted automata = core wMSO

- Abstract semantics $\{\!|-|\!\}: \Sigma^\star \to \mathbb{N}\langle R^\star \rangle$

[Gastin-Monmege, 15]

Core weighted MSO logic

$$\varphi ::= \top \mid P_a(x) \mid x \leqslant y \mid x \in X \mid \neg\varphi \mid \varphi \wedge \varphi \mid \forall x\, \varphi \mid \forall X\, \varphi$$

$$\Psi ::= s \mid \varphi\, ?\, \Psi : \Psi$$

$$\Phi ::= \mathbf{0} \mid \varphi\, ?\, \Phi : \Phi \mid \Phi + \Phi \mid \textstyle\sum_x \Phi \mid \sum_X \Phi \mid \prod_x \Psi$$

Theorem: weighted automata = core wMSO

- Abstract semantics $\{\!|\, - \,|\!\}\colon \Sigma^\star \to \mathbb{N}\langle R^\star \rangle$
- Concrete semantics $[\![-]\!] = \mathsf{aggr} \circ \{\!|\, - \,|\!\}\colon \Sigma^\star \to S$

[Gastin-Monmege, 15]

Core weighted MSO logic

$$\varphi ::= \top \mid P_a(x) \mid x \leqslant y \mid x \in X \mid \neg\varphi \mid \varphi \wedge \varphi \mid \forall x\, \varphi \mid \forall X\, \varphi$$

~~$\Psi ::= s \mid \varphi ? \Psi : \Psi$~~ $\Psi ::= s \mid x \in X \,?\, \Psi : \Psi$

$$\Phi ::= \mathbf{0} \mid \varphi ? \Phi : \Phi \mid \Phi + \Phi \mid \textstyle\sum_x \Phi \mid \sum_X \Phi \mid \prod_x \Psi$$

Theorem: weighted automata = core wMSO

- Abstract semantics $\{\!| - |\!\}\colon \Sigma^\star \to \mathbb{N}\langle R^\star \rangle$
- Concrete semantics $[\![-]\!] = \mathsf{aggr} \circ \{\!| - |\!\}\colon \Sigma^\star \to S$

[Gastin-Monmege, 15]

Core weighted MSO logic

$$\varphi ::= \top \mid P_a(x) \mid x \leqslant y \mid x \in X \mid \neg\varphi \mid \varphi \wedge \varphi \mid \forall x\, \varphi \mid \forall X\, \varphi$$

$$\sout{\Psi ::= s \mid \varphi\,?\,\Psi : \Psi} \qquad \Psi ::= s \mid x \in X\,?\,\Psi : \Psi$$

$$\Phi ::= \mathbf{0} \mid \varphi\,?\,\Phi : \Phi \mid \Phi + \Phi \mid \textstyle\sum_x \Phi \mid \sum_X \Phi \mid \prod_x \Psi$$

Theorem: weighted automata = core wMSO

- Abstract semanti...
- Concrete ... $\to S$

Easy constructive proofs
preservation of the constants
no restriction on core wMSO
no hypotheses on weights

[Gastin-Monmege, 15]

Extensions

More general models than words:
trees, nested words…

More powerful logics:
deciding if a wMSO formula is expressible in core wMSO?

More powerful automata: finding equivalent fragments of wMSO

Weighted FO logic

$$\varphi ::= \top \mid P_a(x) \mid x \leqslant y \mid x \in X \mid \neg\varphi \mid \varphi \wedge \varphi \mid \forall x\, \varphi \mid \forall X\, \varphi$$
$$\Psi ::= s \mid \varphi\, ?\, \Psi : \Psi$$
$$\Phi ::= \mathbf{0} \mid \varphi\, ?\, \Phi : \Phi \mid \Phi + \Phi \mid \textstyle\sum_x \Phi \mid \sum_X \Phi \mid \prod_x \Psi$$

Weighted FO logic

We can keep Boolean MSO or restrict to FO…

$$\varphi ::= \top \mid P_a(x) \mid x \leqslant y \mid x \in X \mid \neg\varphi \mid \varphi \wedge \varphi \mid \forall x\, \varphi \mid \forall X\, \varphi$$

$$\Phi ::= s \mid \varphi ? \Phi : \Phi \mid \Phi + \Phi \mid \Phi \times \Phi \mid \sum_x \Phi \mid \prod_x \Phi$$

Reintroduction of the product

Unconditional product quantification

Weighted FO logic

We can keep Boolean MSO or restrict to FO…

$$\varphi ::= \top \mid P_a(x) \mid x \leqslant y \mid x \in X \mid \neg\varphi \mid \varphi \wedge \varphi \mid \forall x\, \varphi \mid \forall X\, \varphi$$

$$\Phi ::= s \mid \varphi ? \Phi : \Phi \mid \Phi + \Phi \mid \Phi \times \Phi \mid \sum_x \Phi \mid \prod_x \Phi$$

Reintroduction of the product

Unconditional product quantification

$$\varphi_2 = \prod_x \sum_y (x \leq y \wedge P_a(x)) ? 1 : 0 \qquad [\![\varphi_2]\!](a^n) = n!$$

Weighted FO logic

We can keep Boolean MSO or restrict to FO…

$$\varphi ::= \top \mid P_a(x) \mid x \leqslant y \mid x \in X \mid \neg\varphi \mid \varphi \wedge \varphi \mid \forall x\, \varphi \mid \forall X\, \varphi$$

$$\Phi ::= s \mid \varphi ? \Phi : \Phi \mid \Phi + \Phi \mid \Phi \times \Phi \mid \sum_x \Phi \mid \prod_x \Phi$$

Reintroduction of the product

Unconditional product quantification

$$\left[\!\!\left[\prod_x \prod_y 2 \right]\!\!\right](w) = 2^{|w|^2}$$

Pebble weighted automata

$I \in \mathbb{S}^Q$

$T \in \mathbb{S}^Q$

$$\mathcal{A} = (Q, A, I, \delta, T)$$

$\delta : Q \times \mathrm{Test} \times \mathrm{Move} \times Q \to \mathbb{S}$

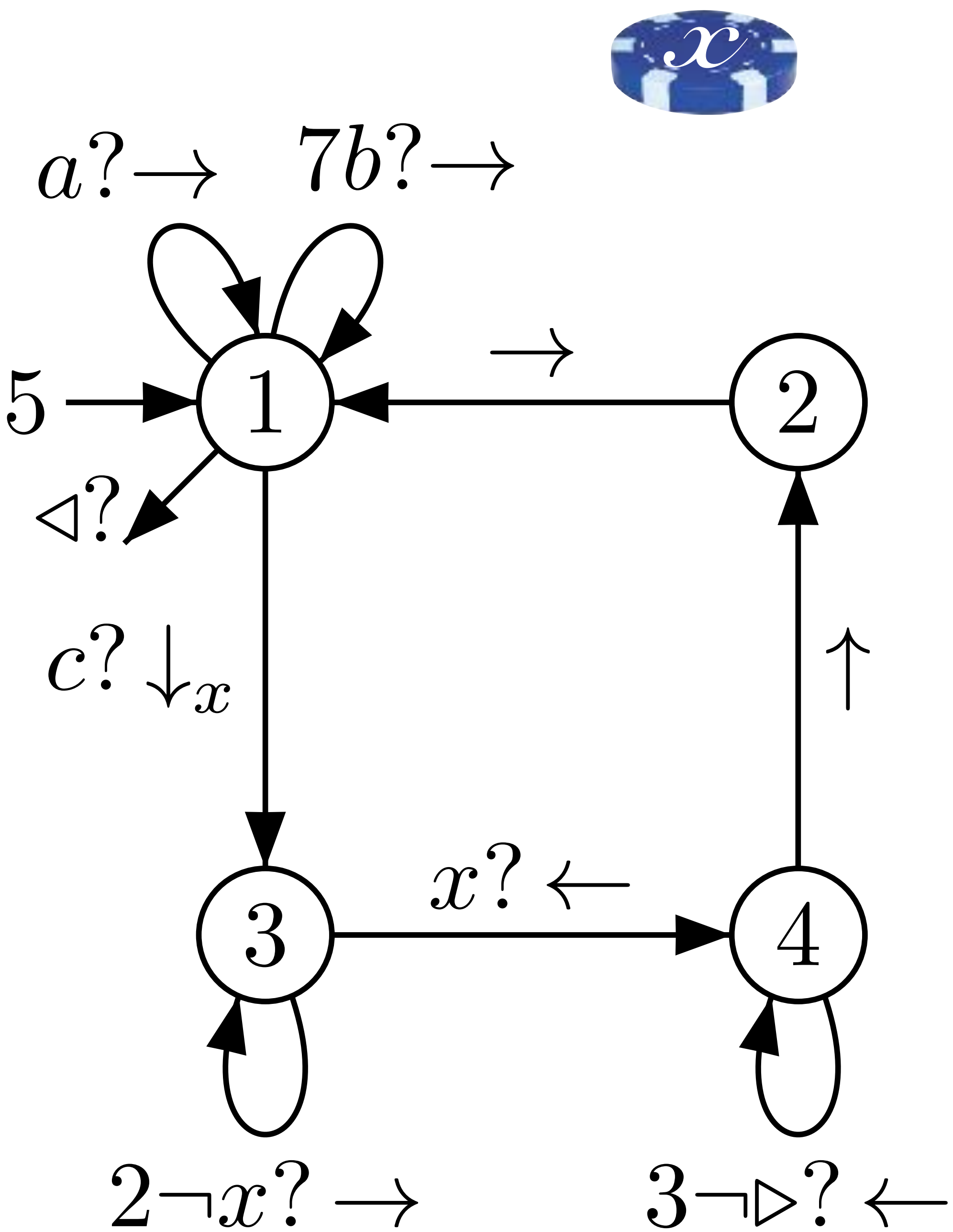
$\mathrm{Move} = \{\rightarrow, \leftarrow, \uparrow\} \cup \{\downarrow_x \mid x \in \mathrm{Peb}\}$

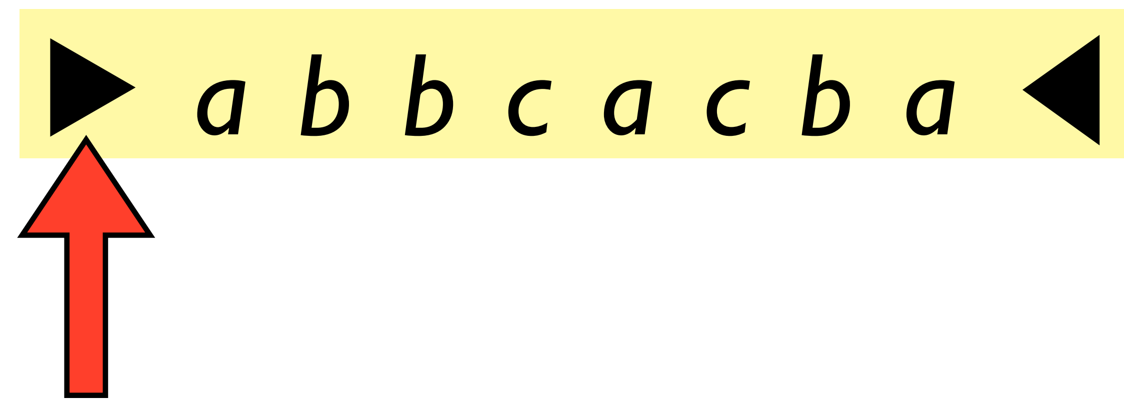
Run as a finite sequence of configurations (W, σ, q, i, π)

with free pebbles $\sigma : \mathrm{Peb} \to \mathrm{pos}(W)$

and a stack of currently dropped pebbles $\pi \in (\mathrm{Peb} \times \mathrm{pos}(W))^*$

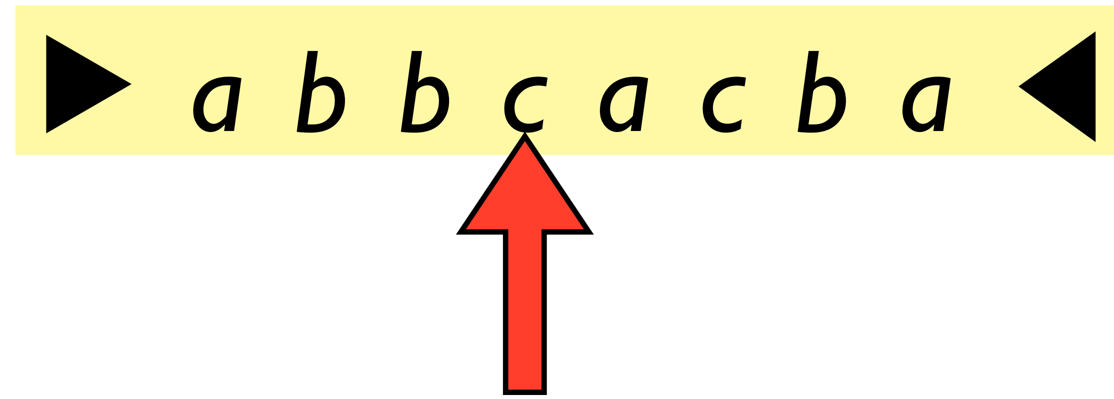
Pebble weighted automata

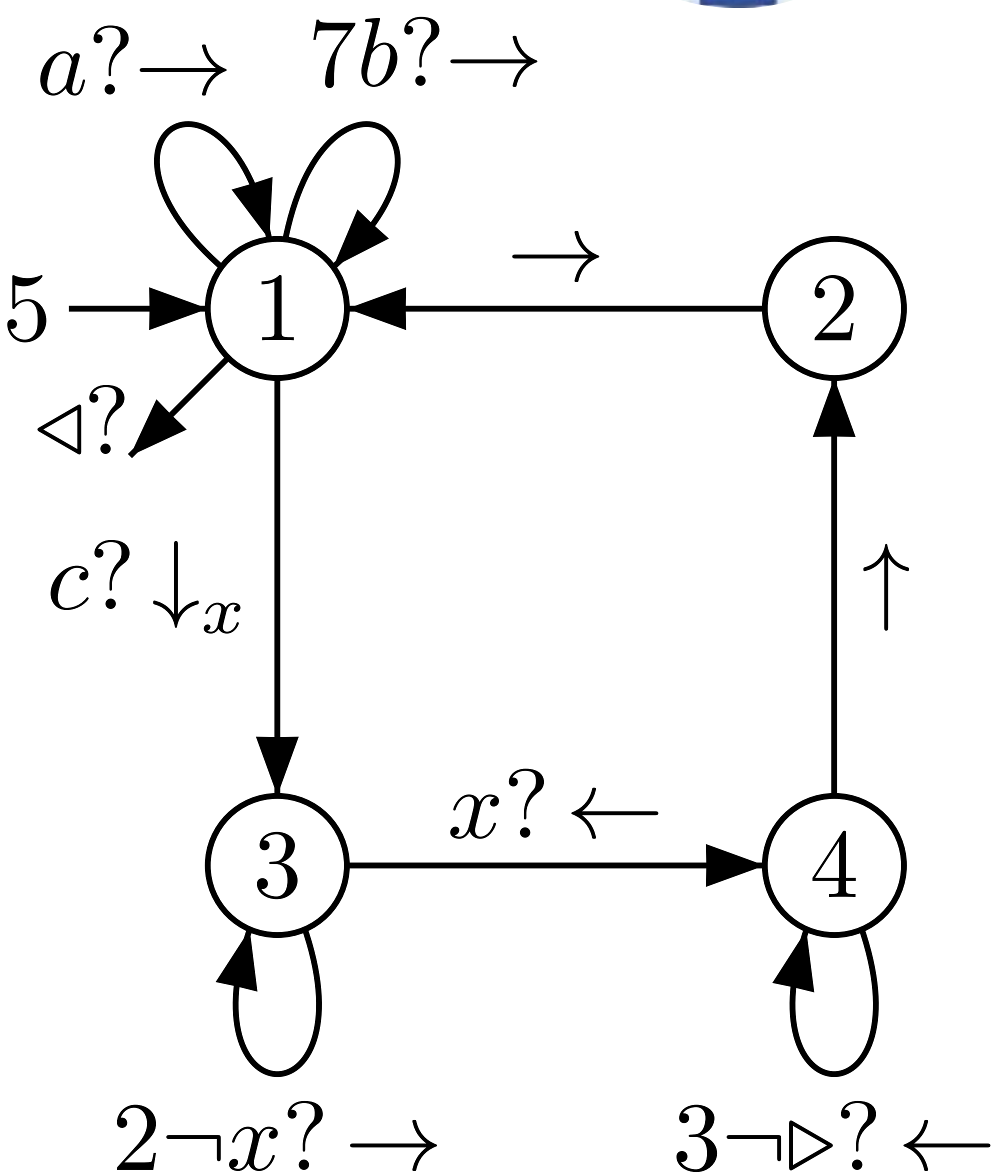




[Globerman and Harel, 96][Engelfriet and Hoogeboom,99]

Pebble weighted automata

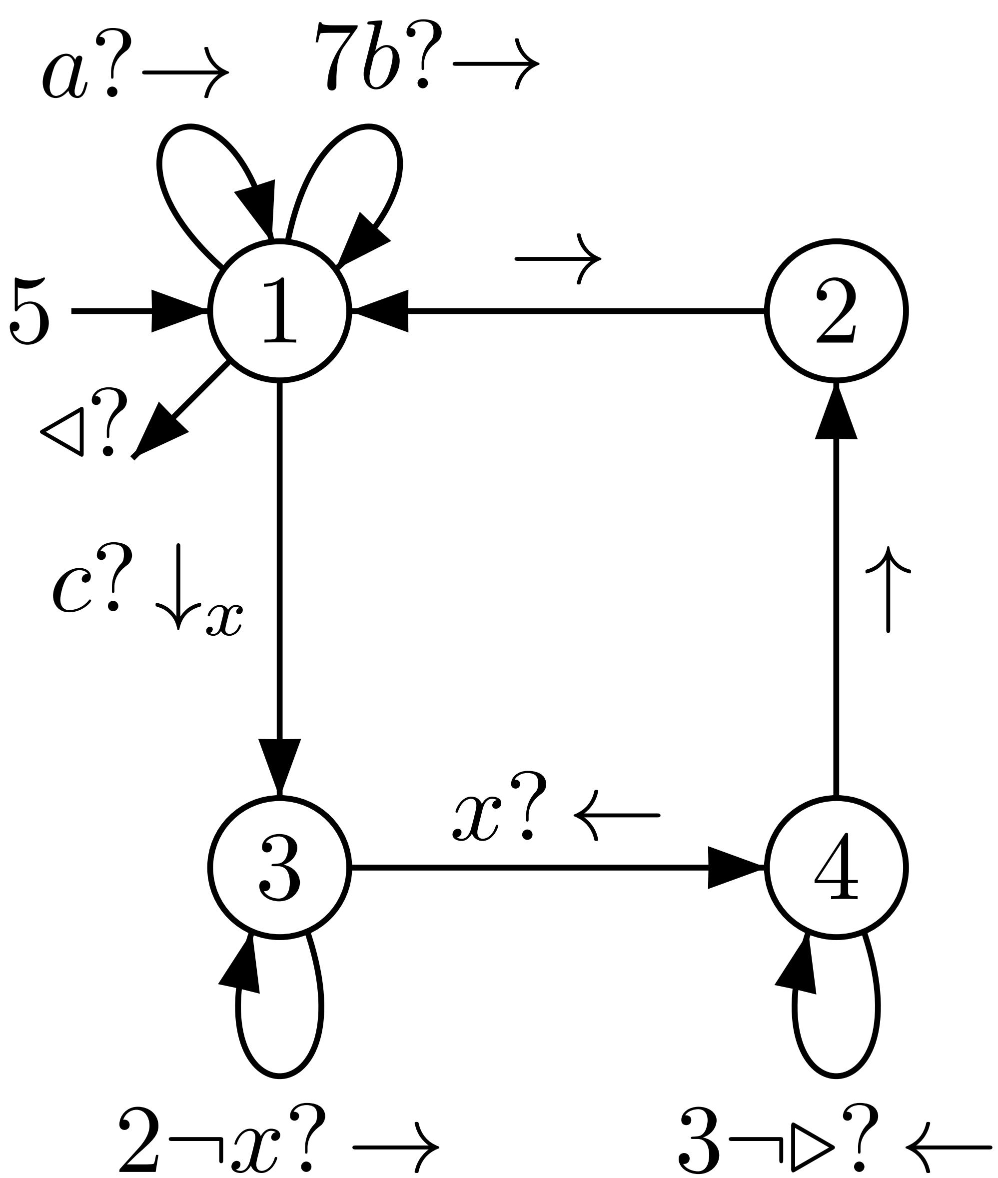


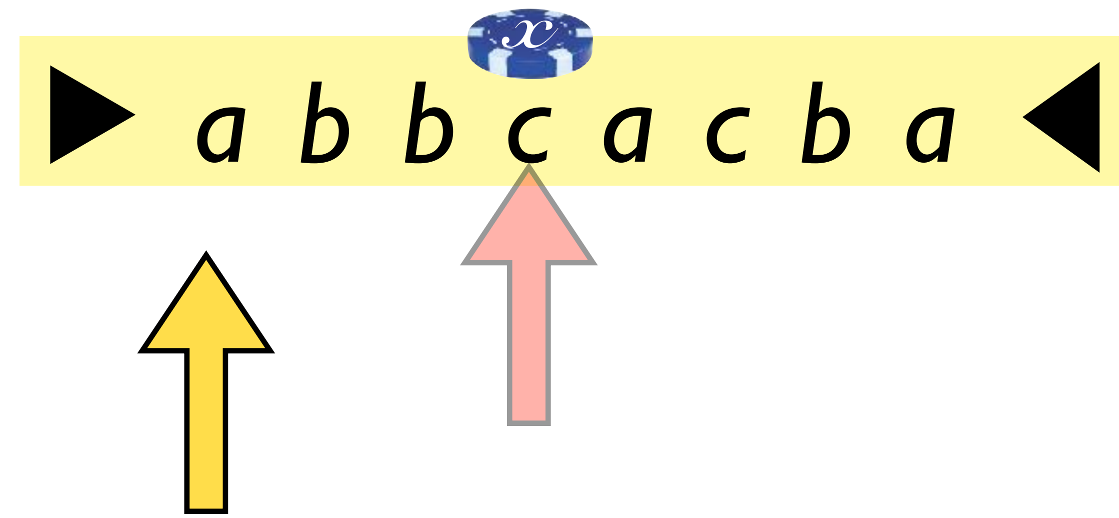


5 x1x7x7

[Globerman and Harel, 96][Engelfriet and Hoogeboom,99]

Pebble weighted automata

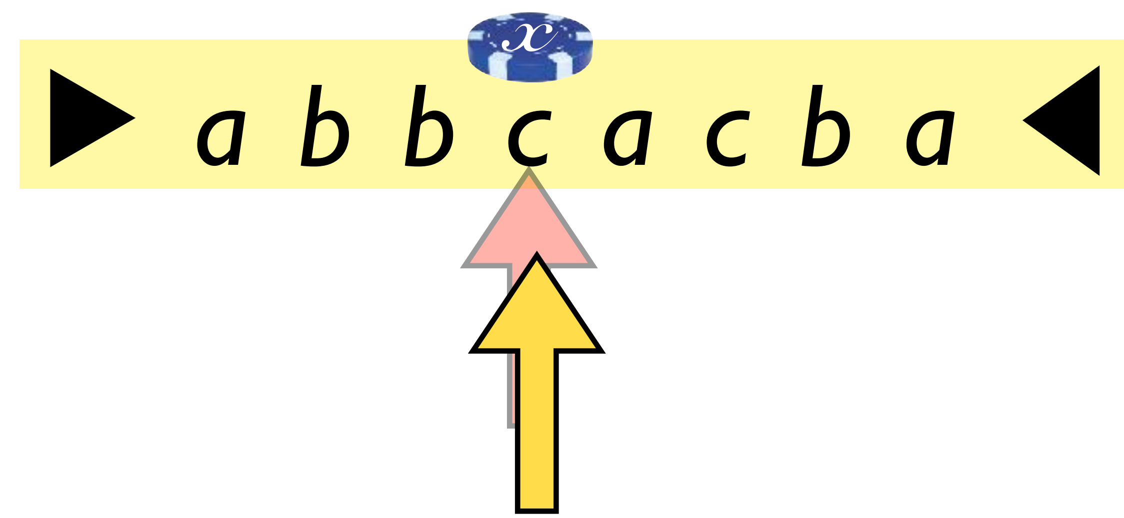


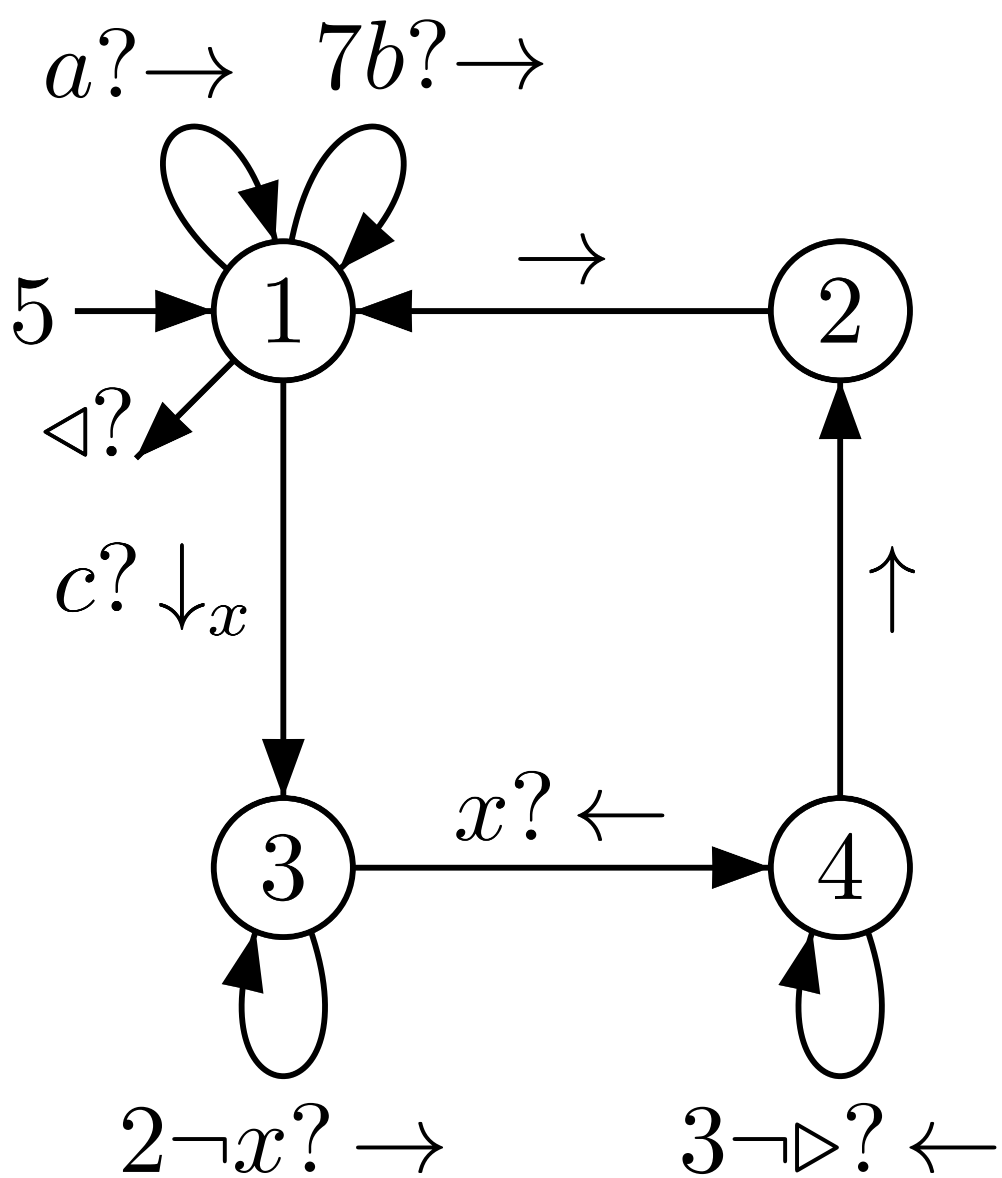


5 x1x7x7x1

[Globerman and Harel, 96][Engelfriet and Hoogeboom,99]

Pebble weighted automata

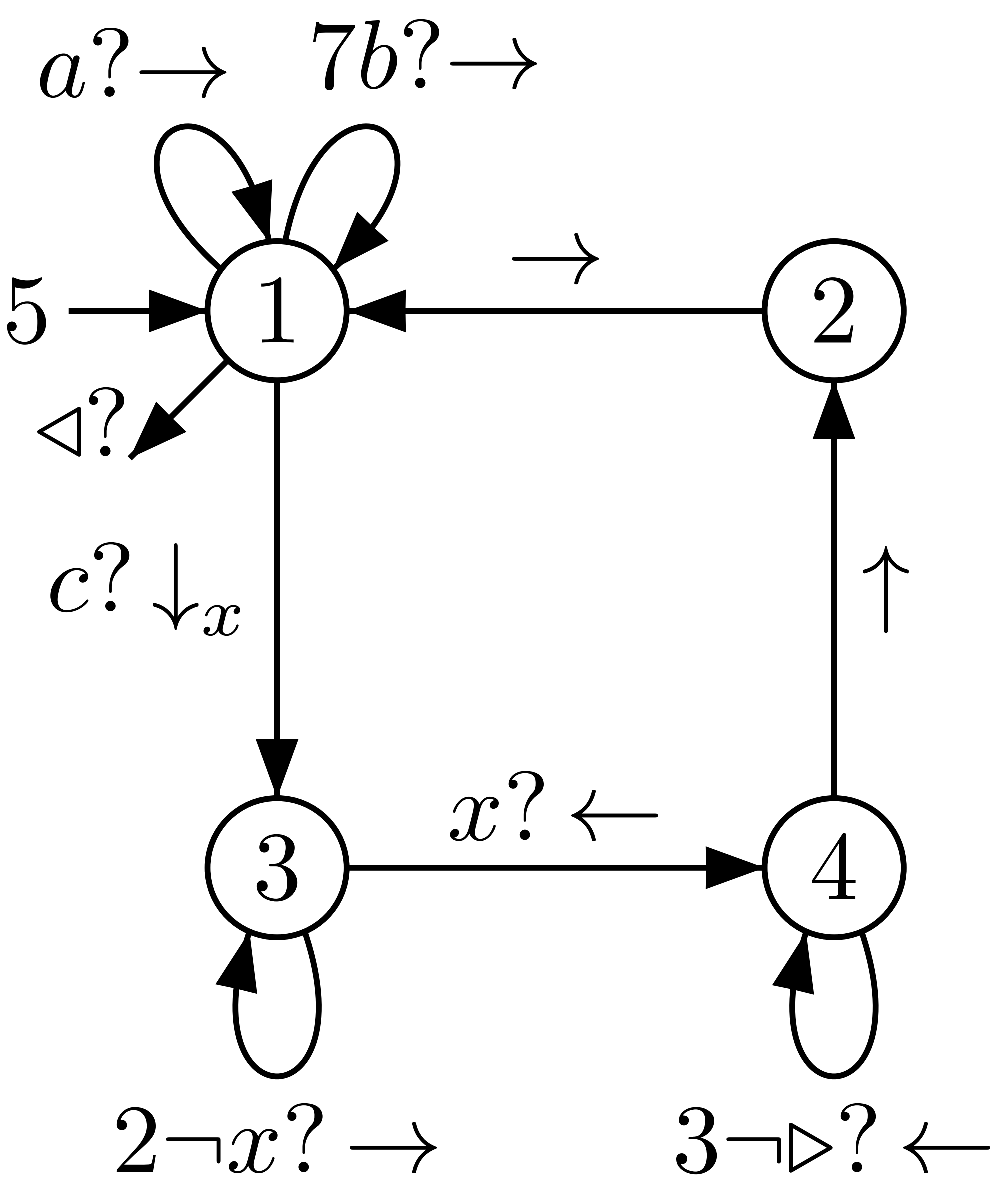


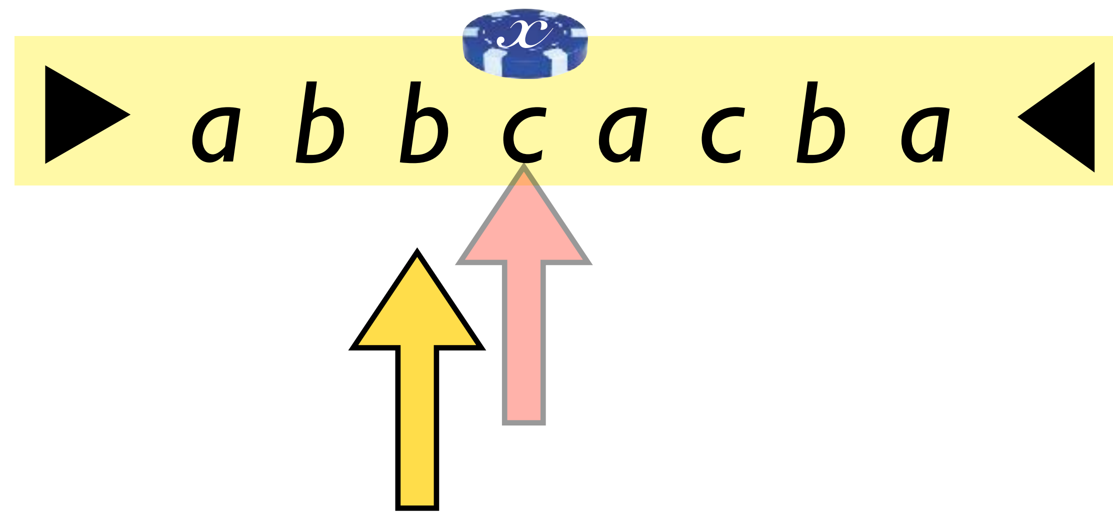


5 x1x7x7x1
x2x2x2

[Globerman and Harel, 96][Engelfriet and Hoogeboom,99]

Pebble weighted automata



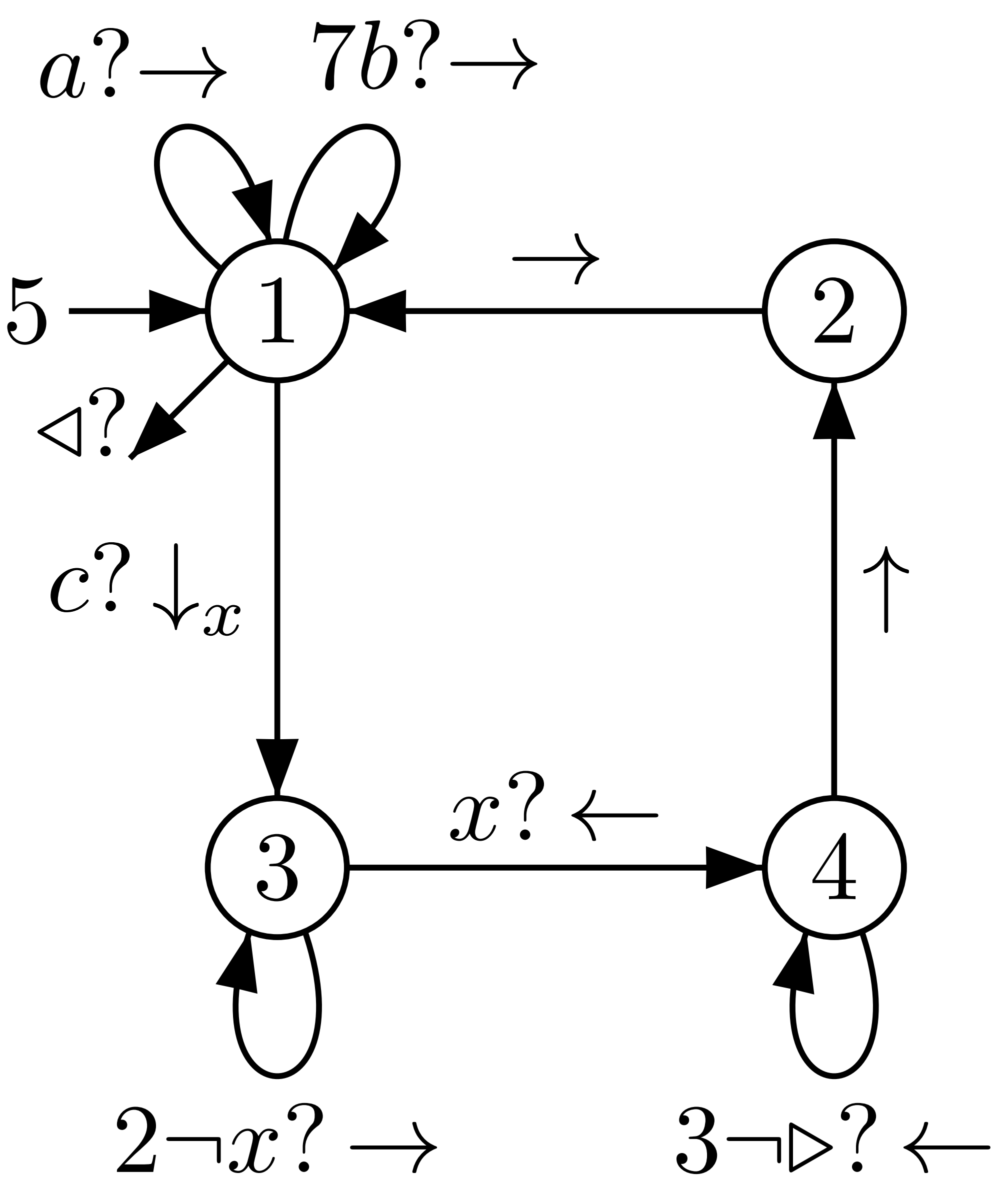


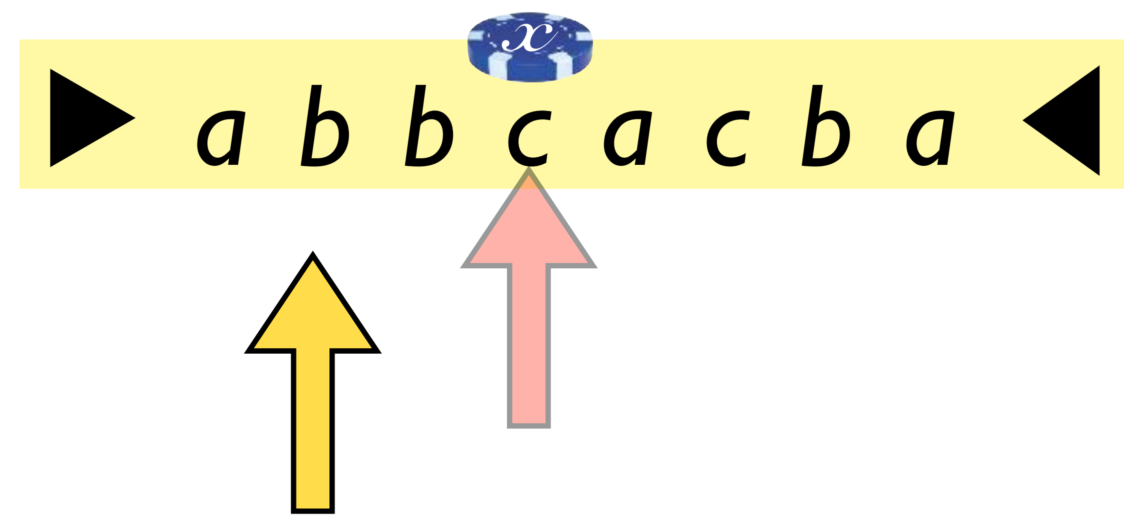
5 x1x7x7x1

x2x2x2x1

[Globerman and Harel, 96][Engelfriet and Hoogeboom,99]

Pebble weighted automata

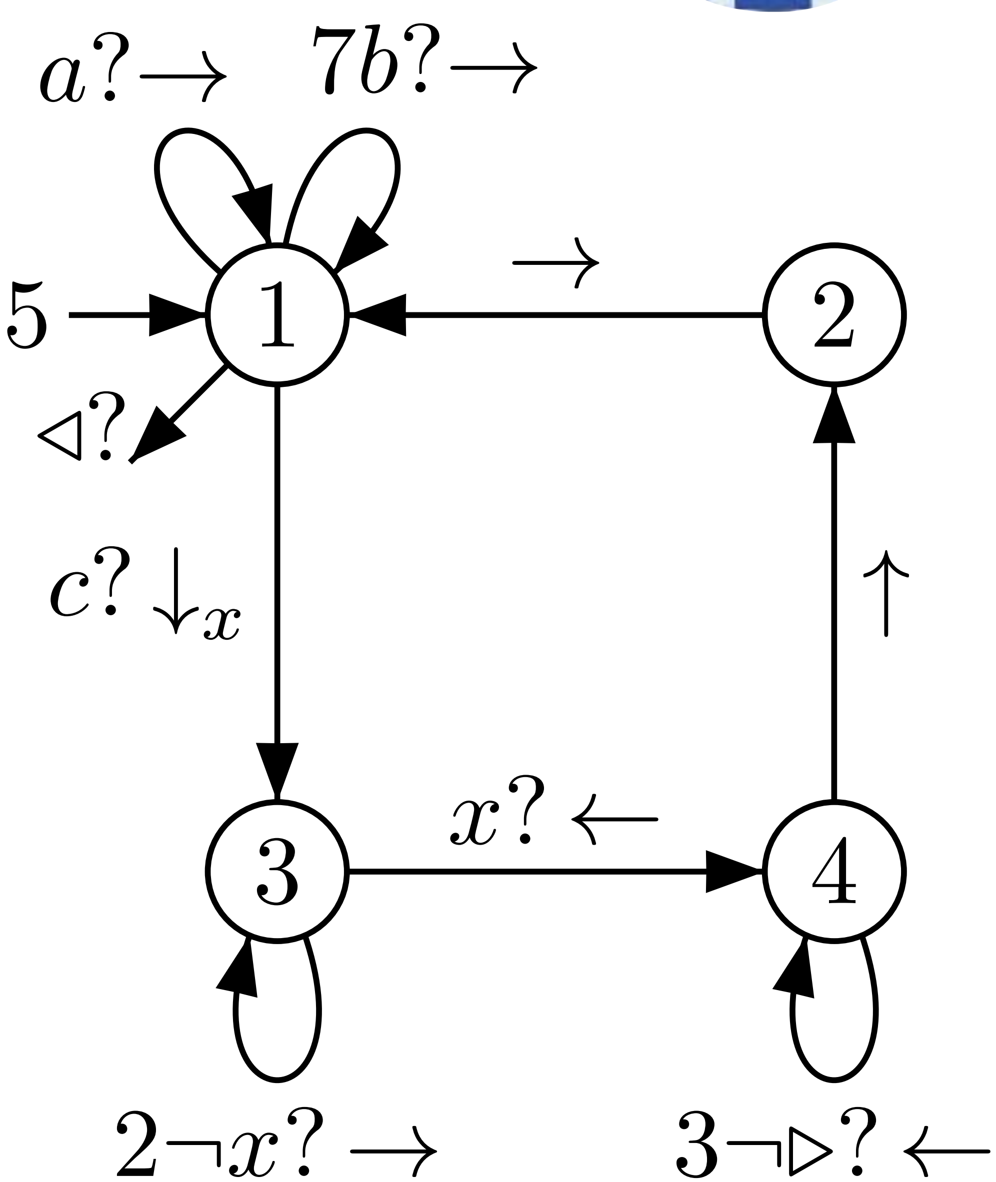


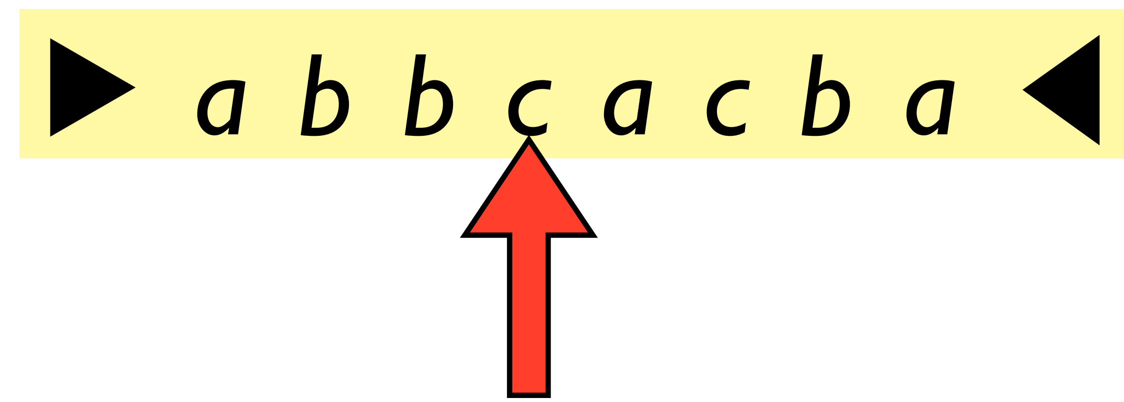


5 x1x7x7x1
x2x2x2x1x3

[Globerman and Harel, 96][Engelfriet and Hoogeboom,99]

Pebble weighted automata

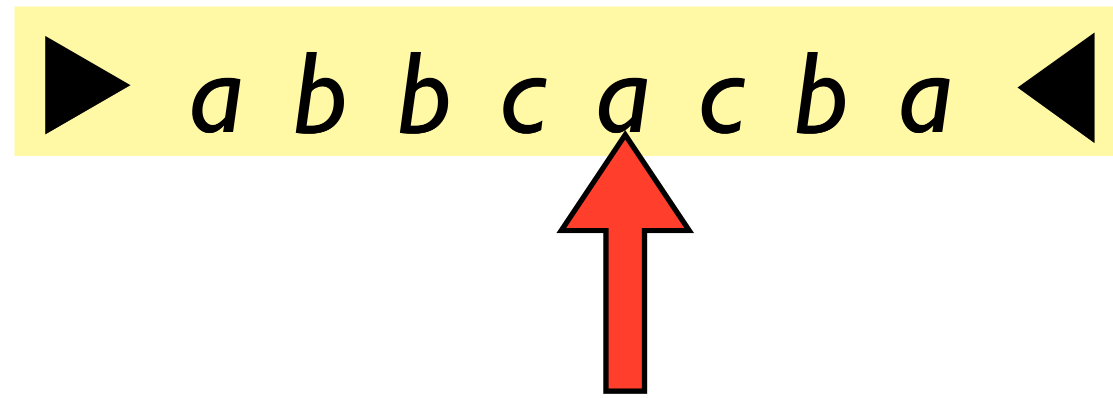


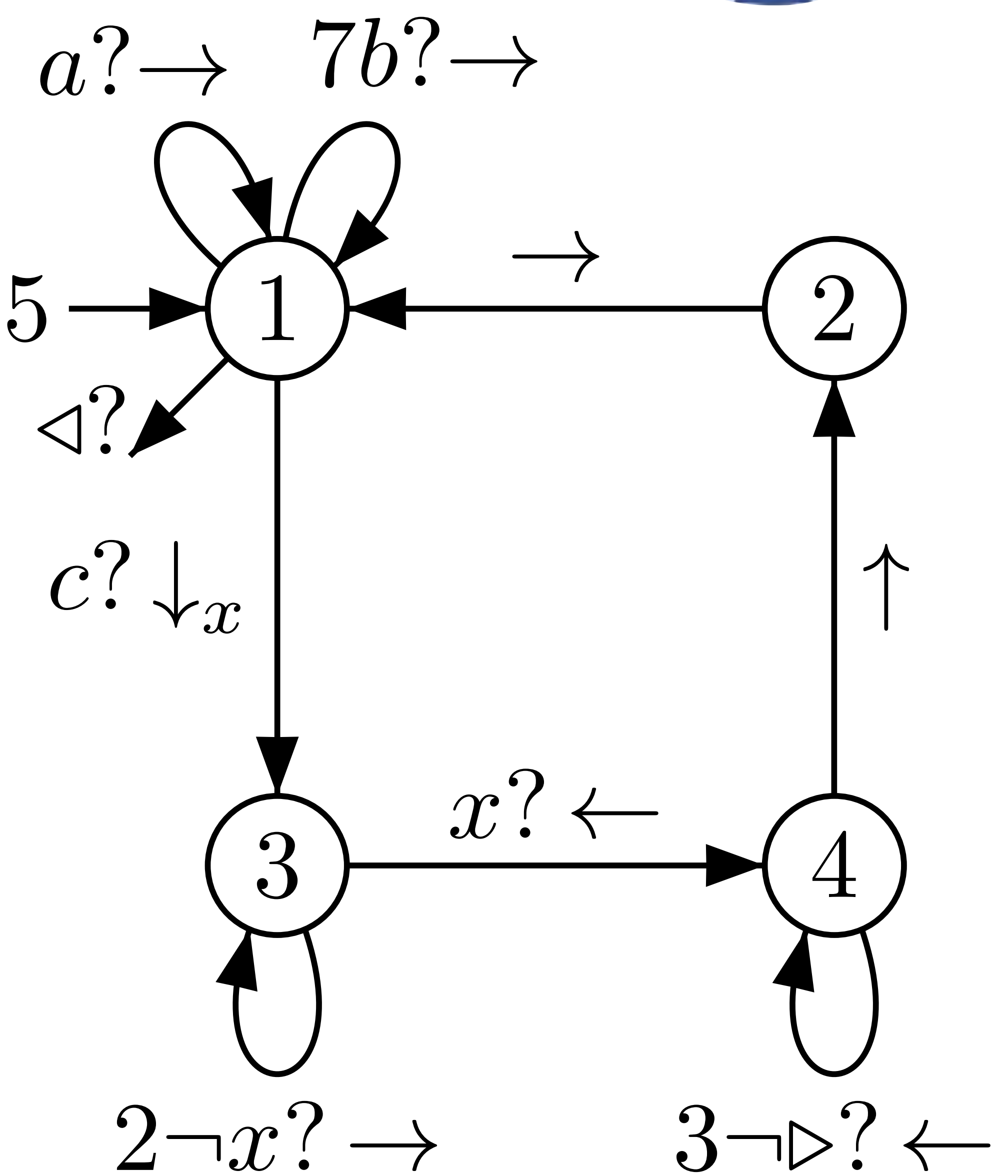


5 x1x7x7x1
x2x2x2x1x3x1

[Globerman and Harel, 96][Engelfriet and Hoogeboom,99]

Pebble weighted automata





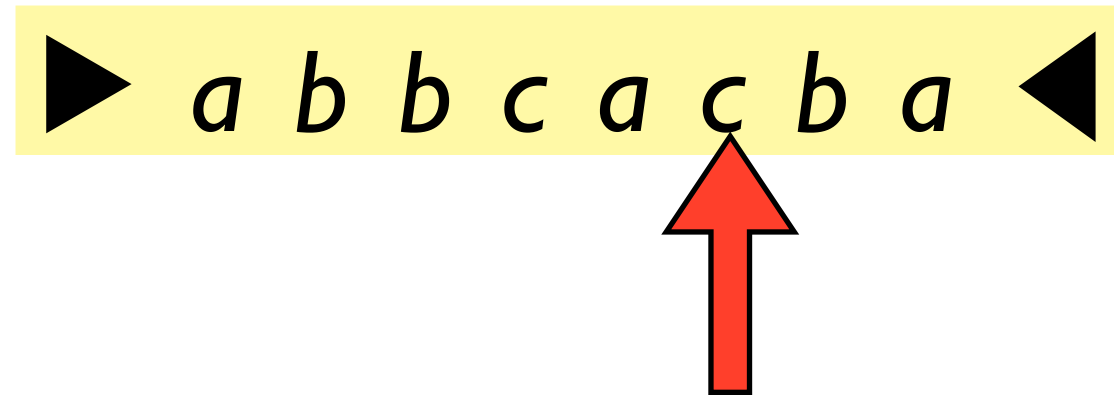
5 x1x7x7x1

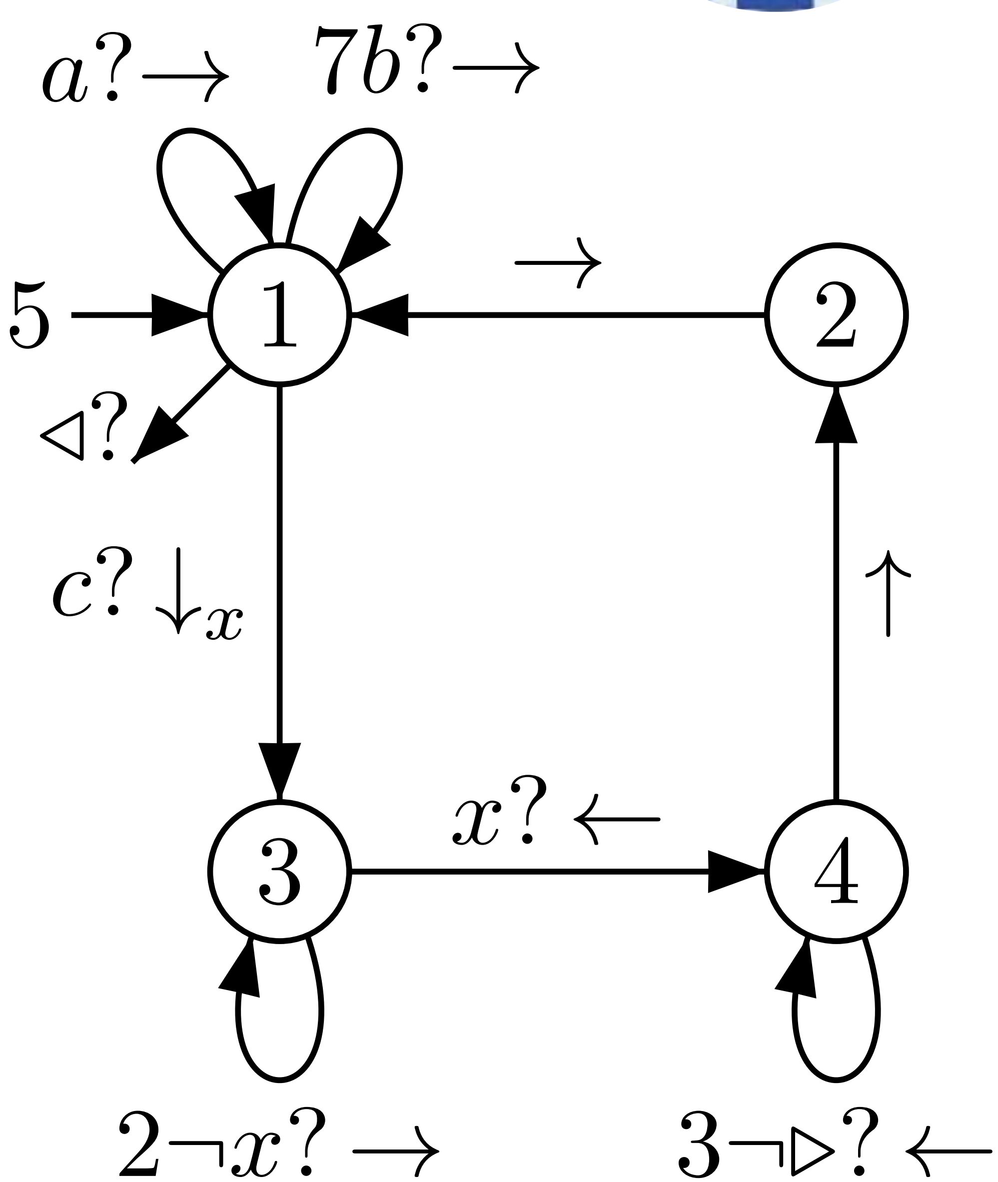
x2x2x2x1x3x1

x1

[Globerman and Harel, 96][Engelfriet and Hoogeboom,99]

Pebble weighted automata





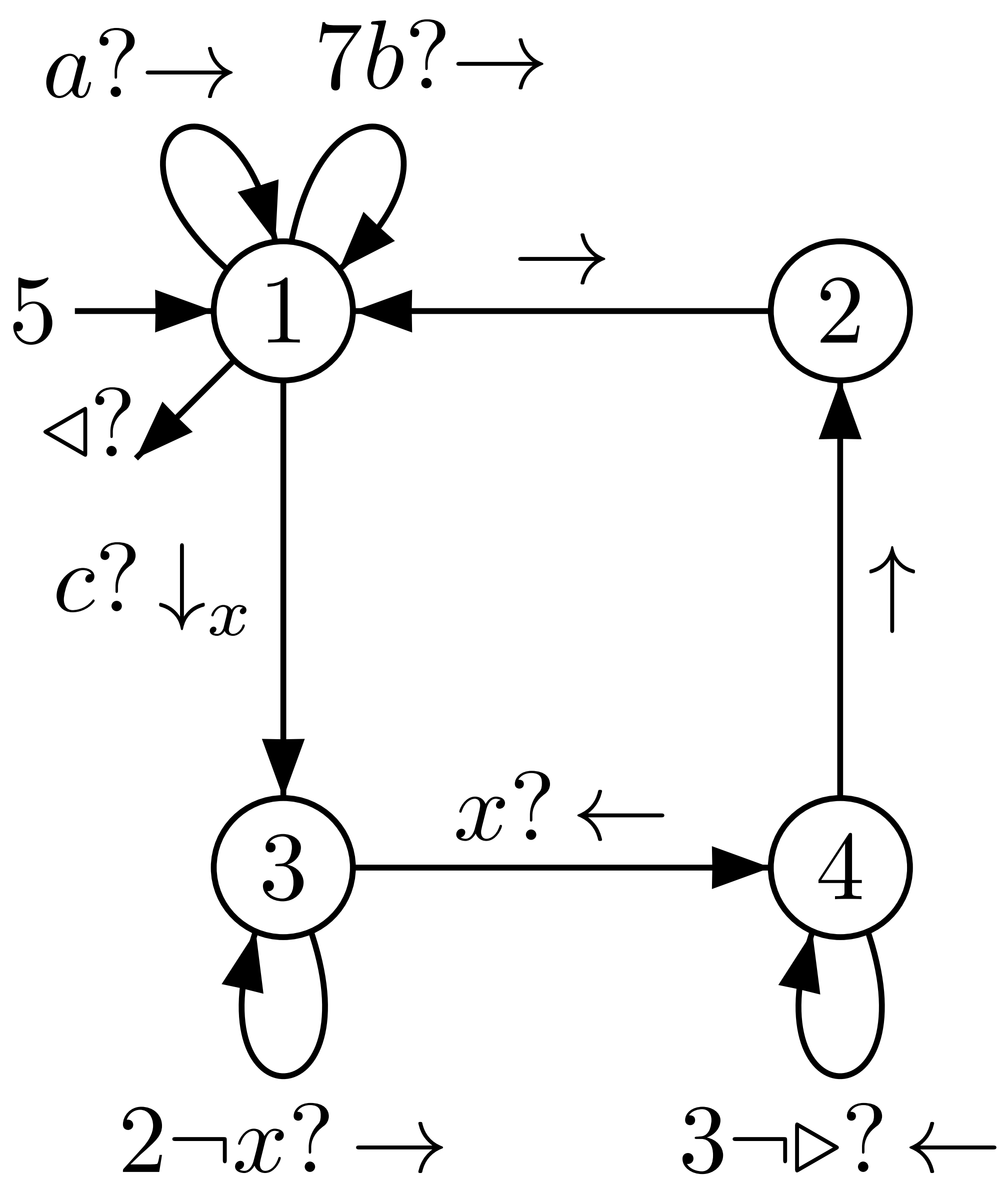
5 x1x7x7x1

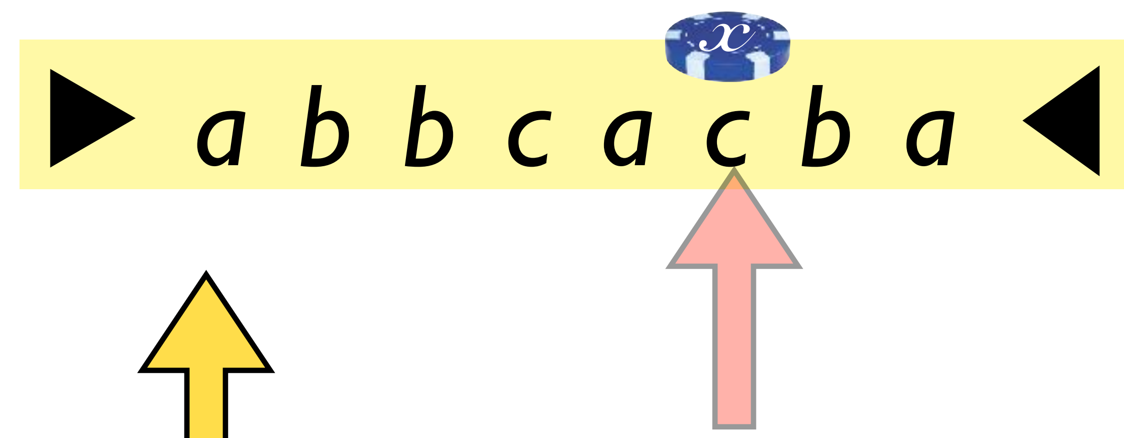
x2x2x2x1x3x1

x1x1

[Globerman and Harel, 96][Engelfriet and Hoogeboom,99]

Pebble weighted automata





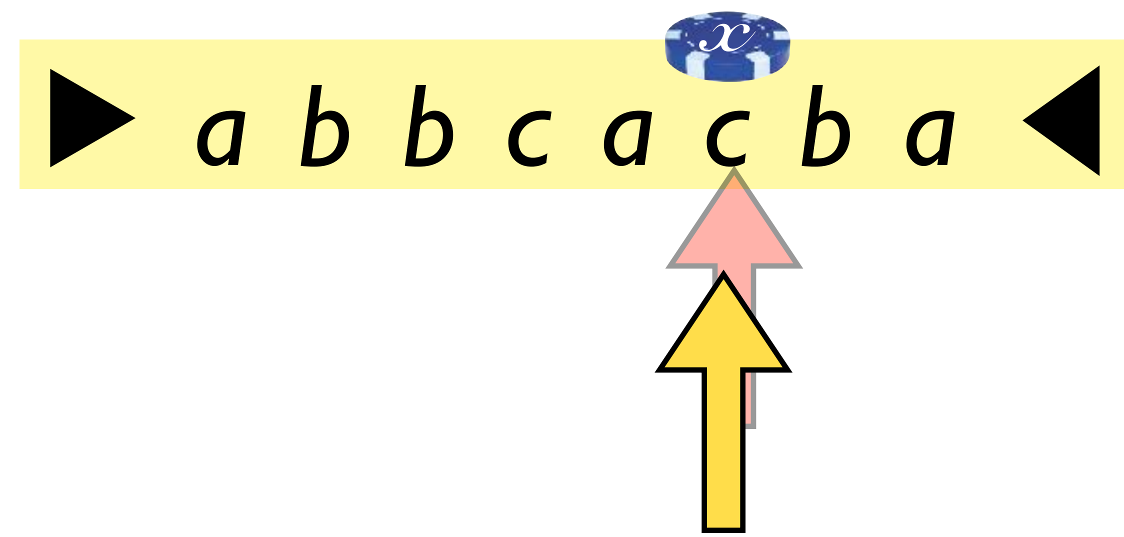
5 x1x7x7x1

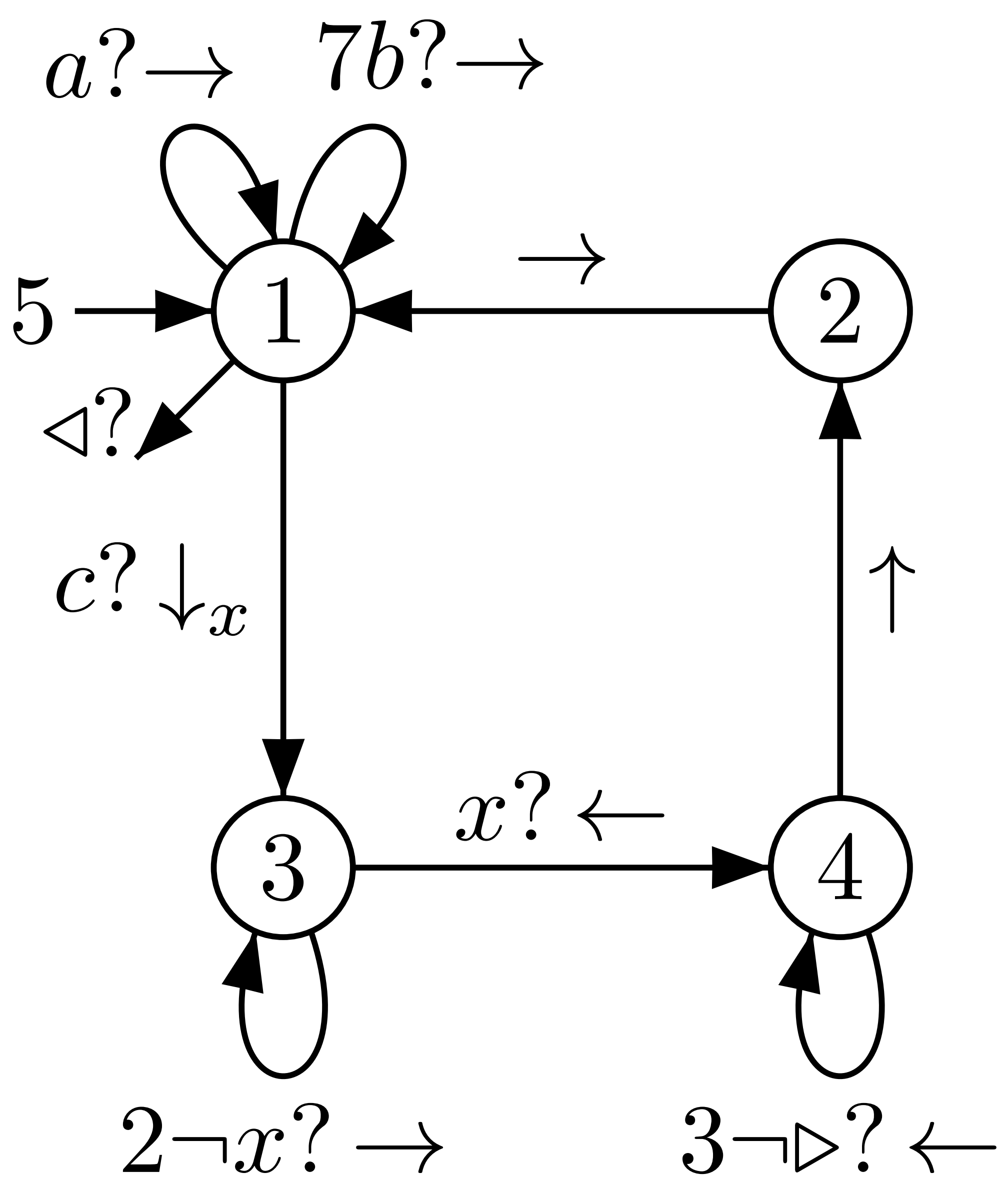
x2x2x2x1x3x1

x1x1x1

[Globerman and Harel, 96][Engelfriet and Hoogeboom,99]

Pebble weighted automata





5 x1x7x7x1

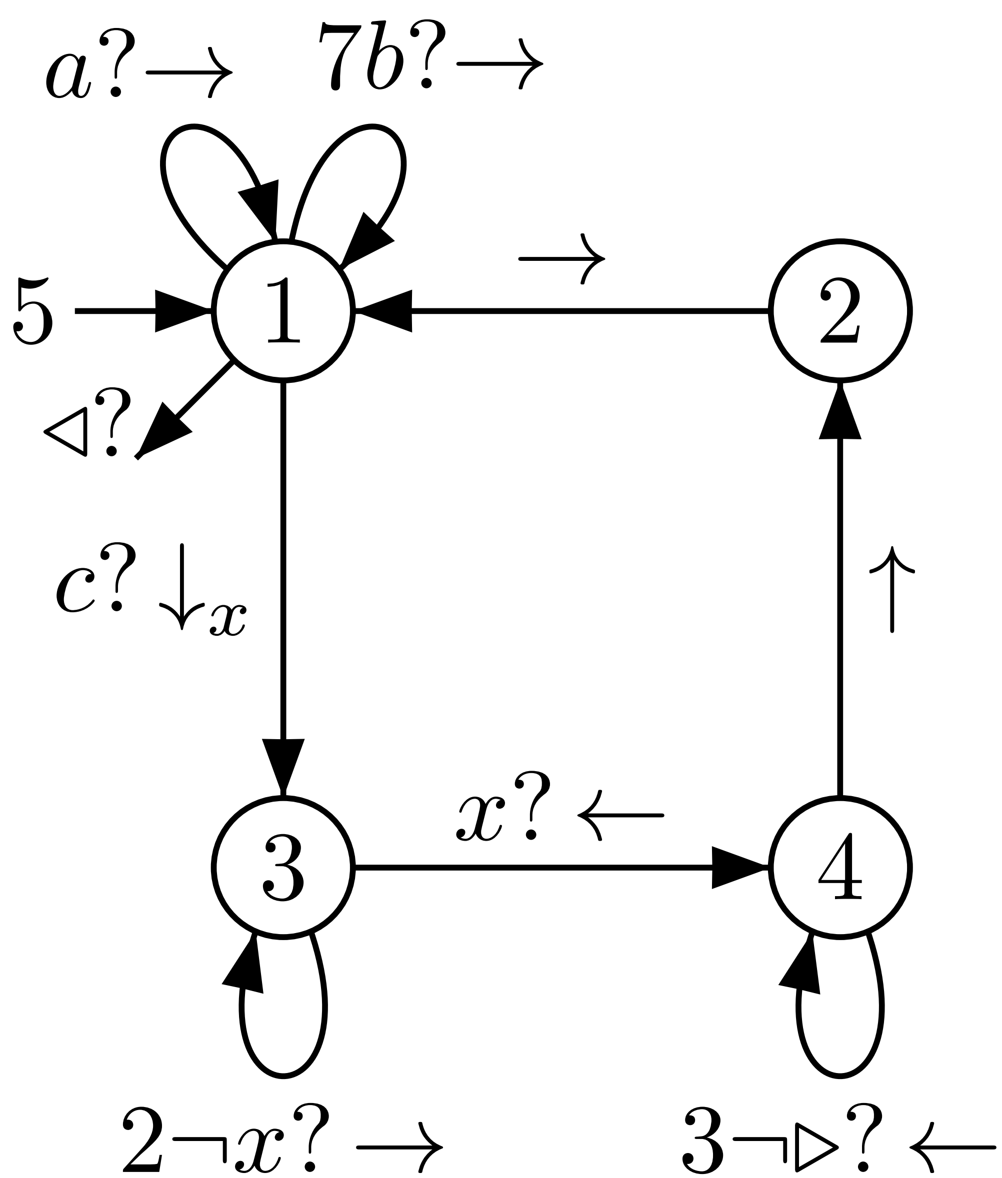
x2x2x2x1x3x1

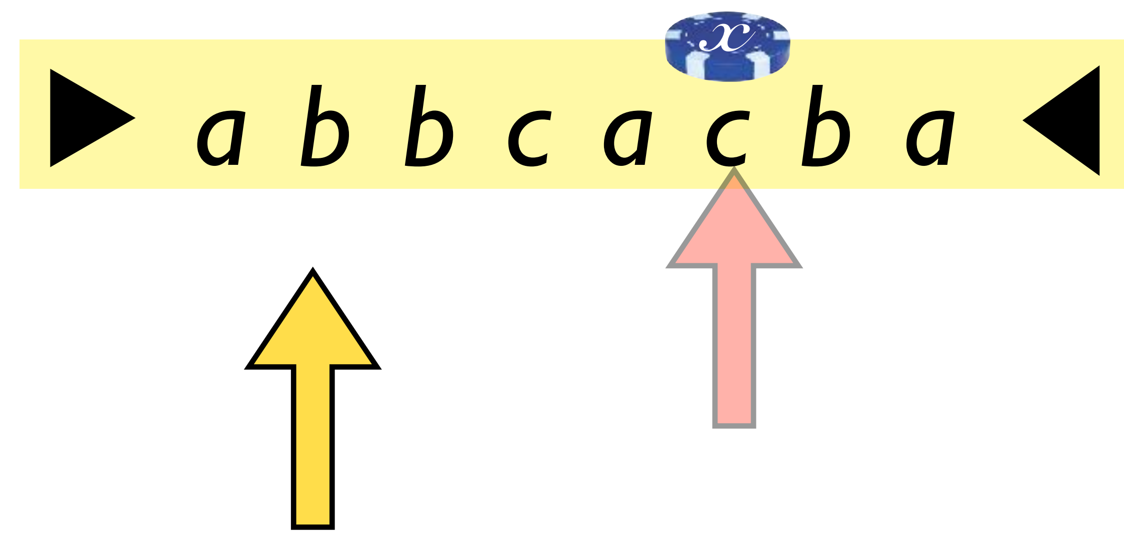
x1x1x1

x2x2x2x2x2

[Globerman and Harel, 96][Engelfriet and Hoogeboom,99]

Pebble weighted automata





5 x1x7x7x1

x2x2x2x1x3x1

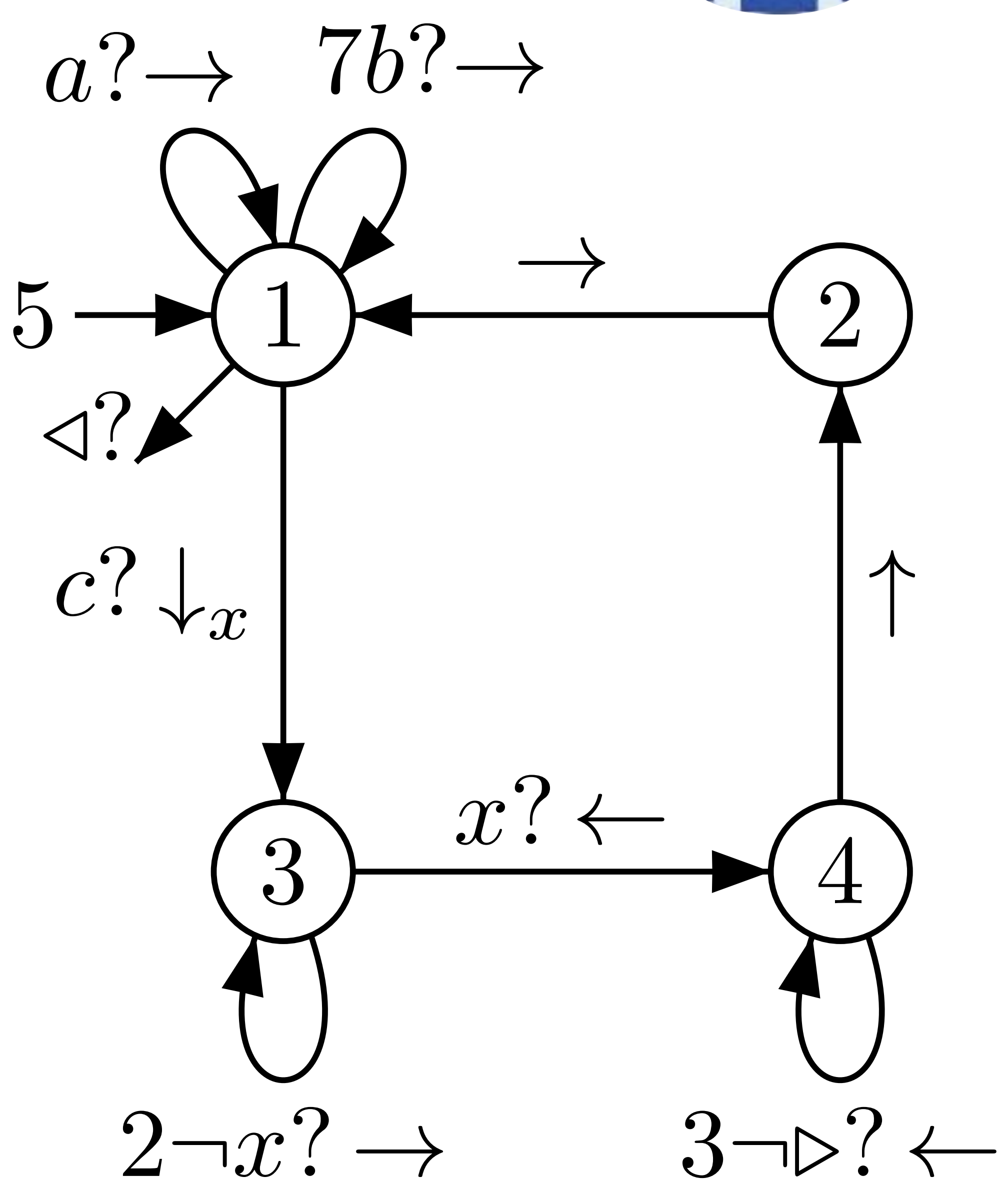
x1x1x1

x2x2x2x2x2x1x3x3x3

[Globerman and Harel, 96][Engelfriet and Hoogeboom,99]

Pebble weighted automata

▶ *a b b c a c b a* ◀



5 x1x7x7x1

x2x2x2x1x3x1

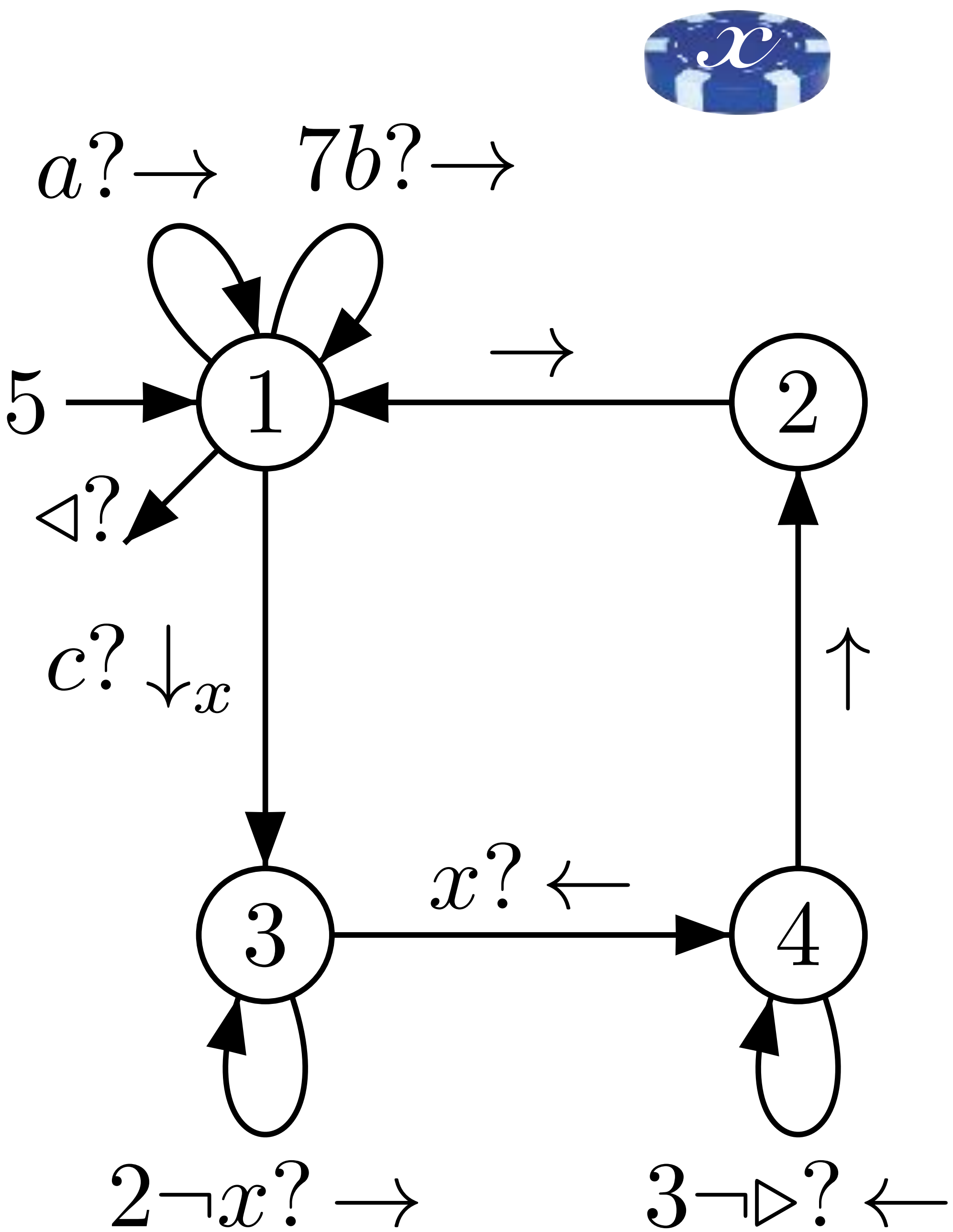
x1x1x1

x2x2x2x2x2x1x3x3x3x1

x1x7x1

[Globerman and Harel, 96][Engelfriet and Hoogeboom,99]

Pebble weighted automata



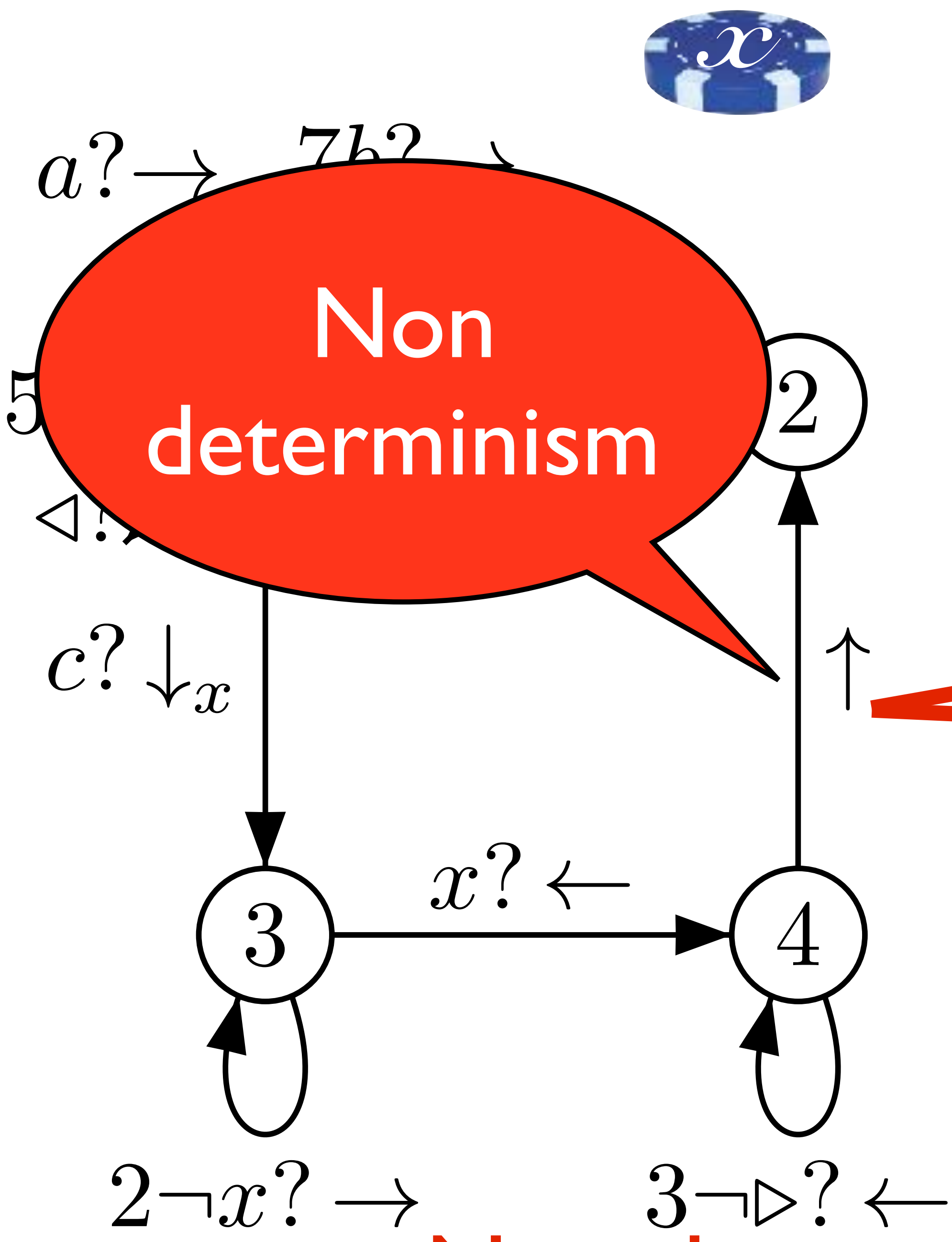
▶ *a b b c a c b a* ◀

Weight of the run: 35 562 240

5 x1x7x7x1
x2x2x2x1x3x1
x1x1x1
x2x2x2x2x2x1x3x3x3x1
x1x7x1

[Globerman and Harel, 96][Engelfriet and Hoogeboom,99]

Pebble weighted automata



▶ *a b b c a c b a* ◀

Weight of the run: 35 562 240

5 x1x7x7x1
x2x2x2x1x3x1
x1x1x1

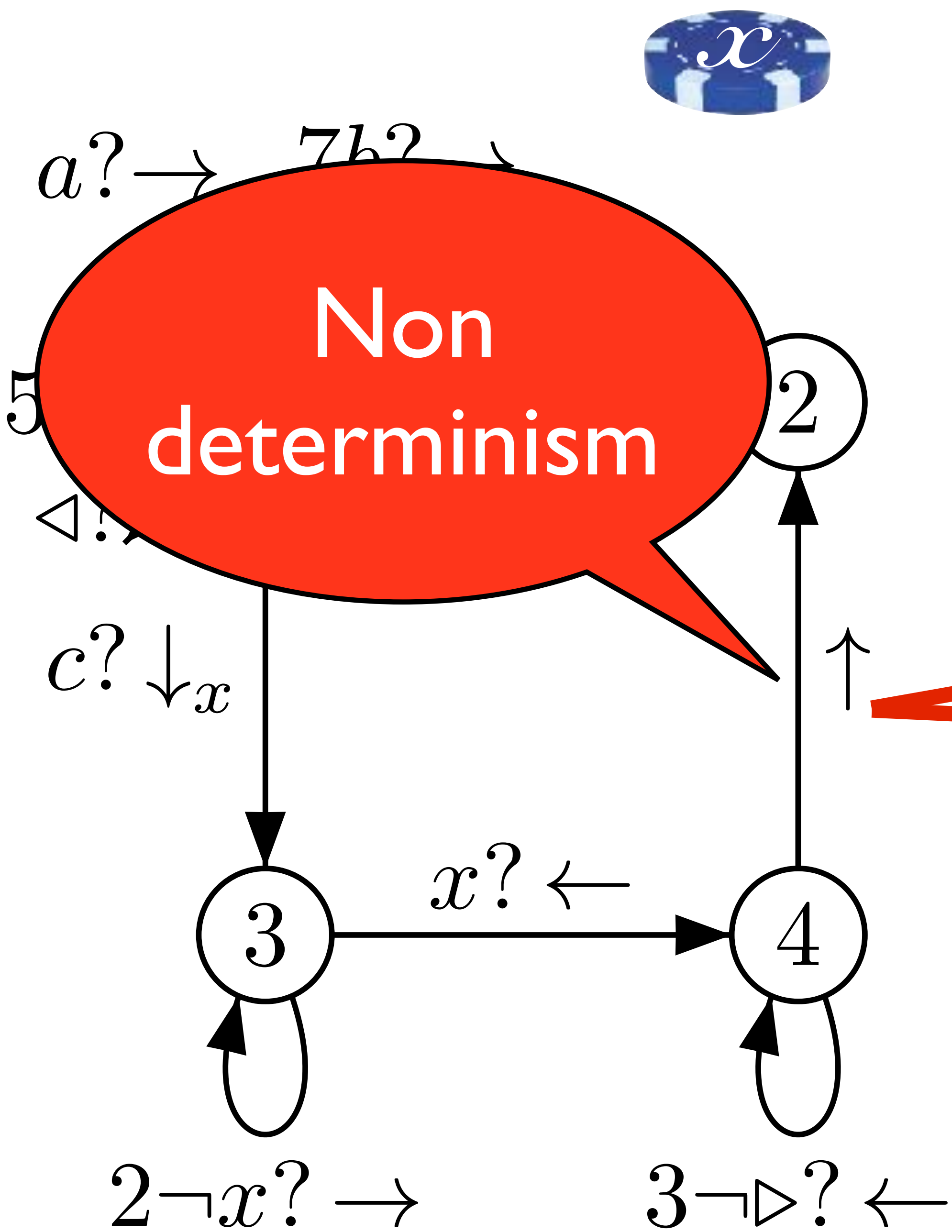
x2x2x2x2x2x1x3x3x3x1
x1x7x1

Non determinism resolved by sum

[Globerman and Harel, 96][Engelfriet and Hoogeboom,99]

Pebble weighted automata

▶ *a b b c a c b a* ◀



5 x1x7x7x1

x2x2x2x1x3x1

x3x1

Weight of the run: 35 562 240

Sum of the weights of the runs: $5.7^{|w|_b}.\prod_{w_i=c} 2^{i-2}(3^i-1)$

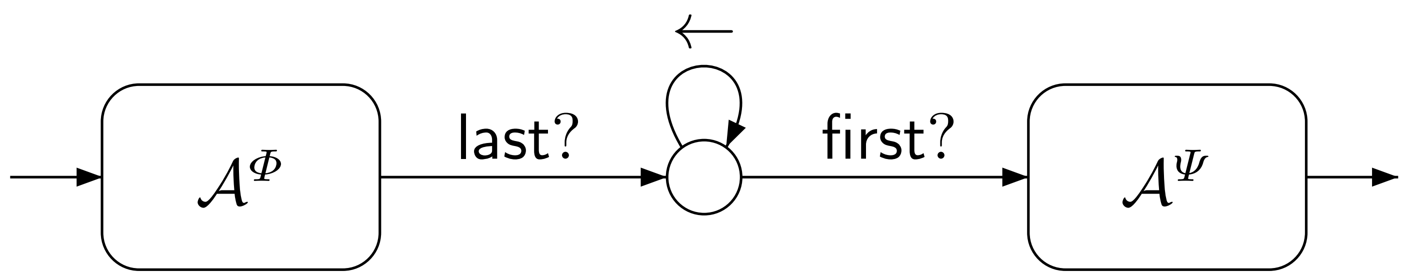
Non determinism resolved by sum

[Globerman and Harel, 96][Engelfriet and Hoogeboom,99]

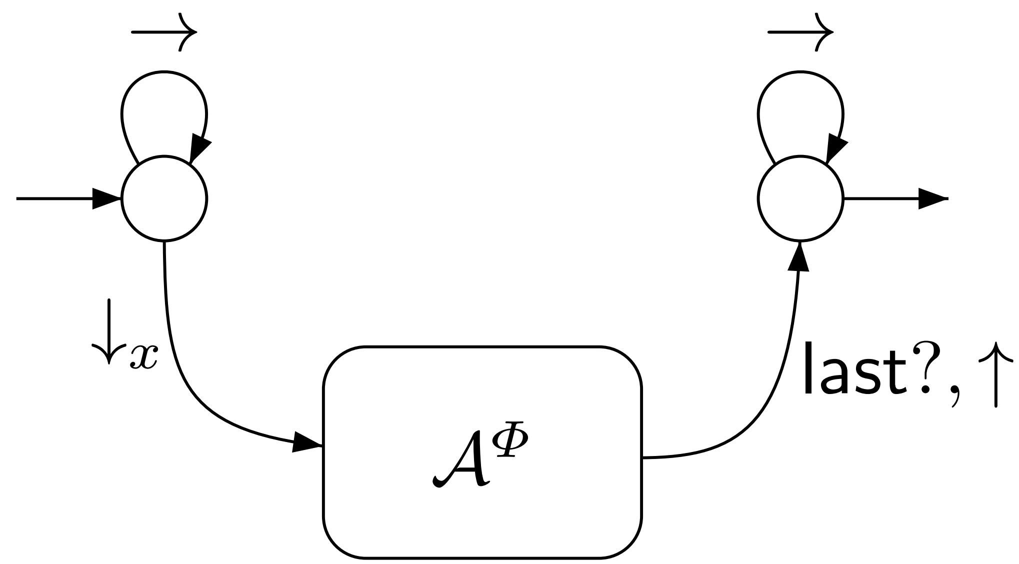
Translating a formula into an automaton

Sum by disjoint union of automata

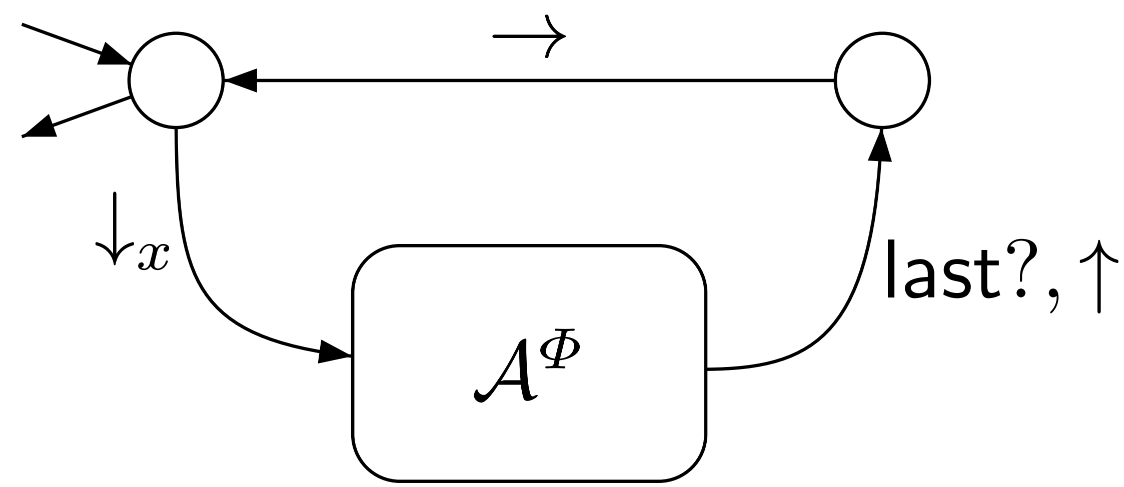
Product:



Sum quantification:



Product quantification:



Translating a formula into an automaton

Challenging for the *Boolean part*:

need unambiguous automata

Translating a formula into an automaton

Challenging for the *Boolean part*: need unambiguous automata

Use deterministic automata of size non-elementary...

Translating a formula into an automaton

Challenging for the *Boolean part*:
need unambiguous automata

Use deterministic automata
of size non-elementary...

Take advantage of the pebbles
to build a linear size automaton

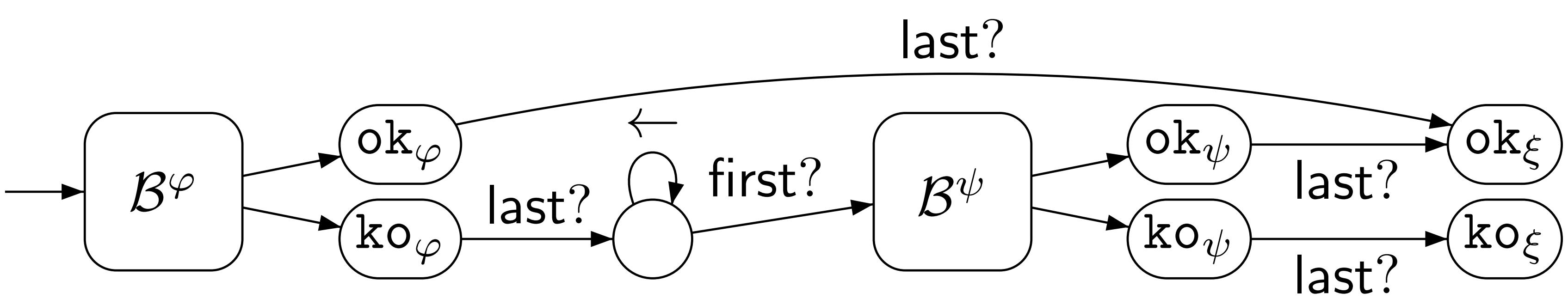
Translating a formula into an automaton

Challenging for the *Boolean part*:
need unambiguous automata

Use deterministic automata of size non-elementary...

Take advantage of the pebbles to build a linear size automaton

Disjunction/conjunction
$\xi = \varphi \vee \psi$



Translating a formula into an automaton

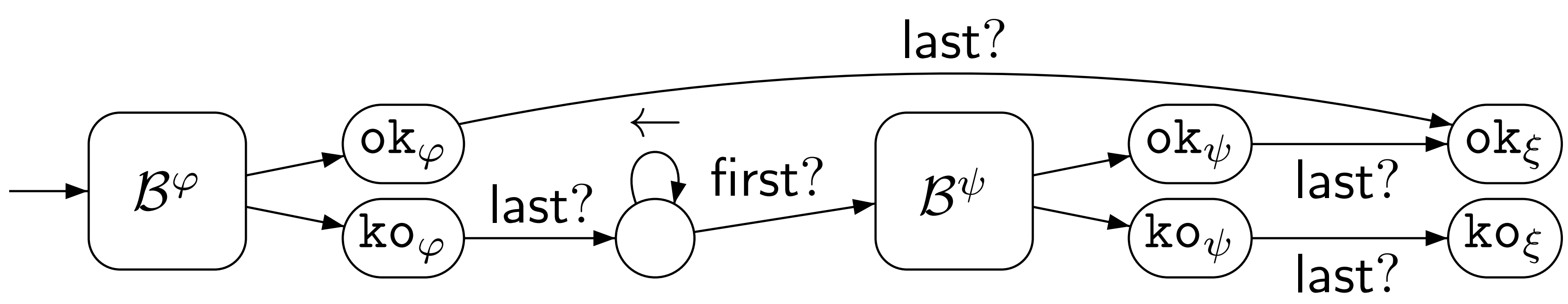
Challenging for the *Boolean part*: need unambiguous automata

Use deterministic automata of size non-elementary...

Take advantage of the pebbles to build a linear size automaton

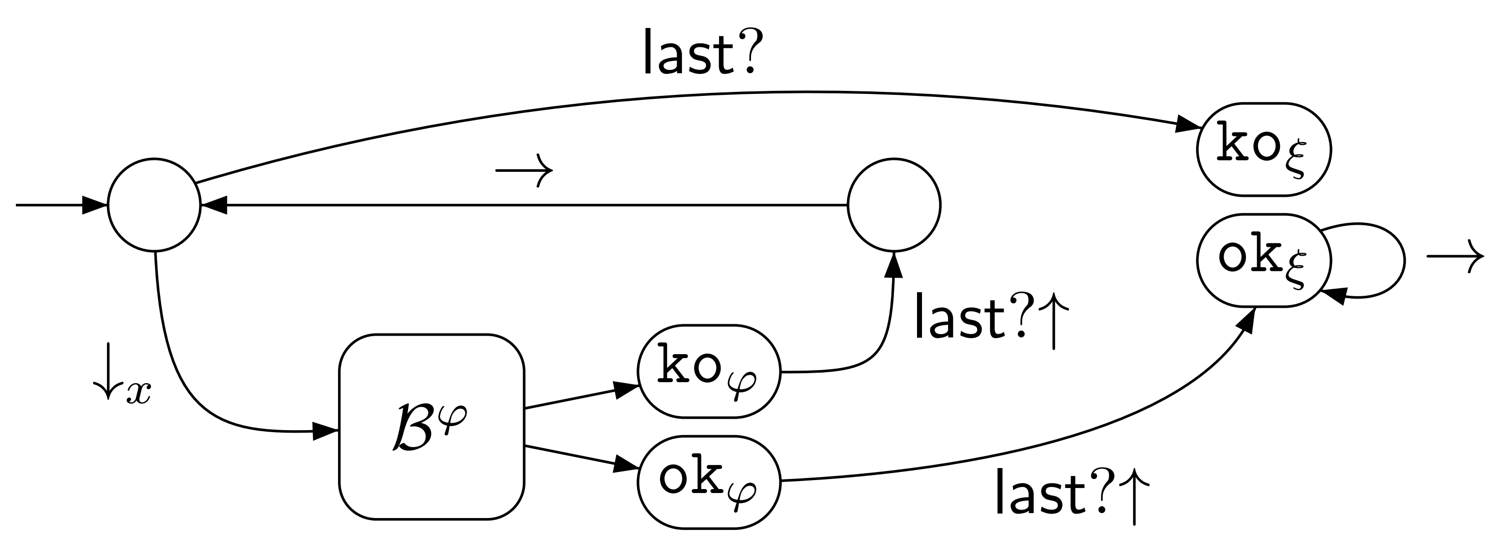
Disjunction/conjunction

$\xi = \varphi \vee \psi$



Existential/Universal quantifications

$\xi = \exists x\ \varphi$



Logic equivalent to PWA?

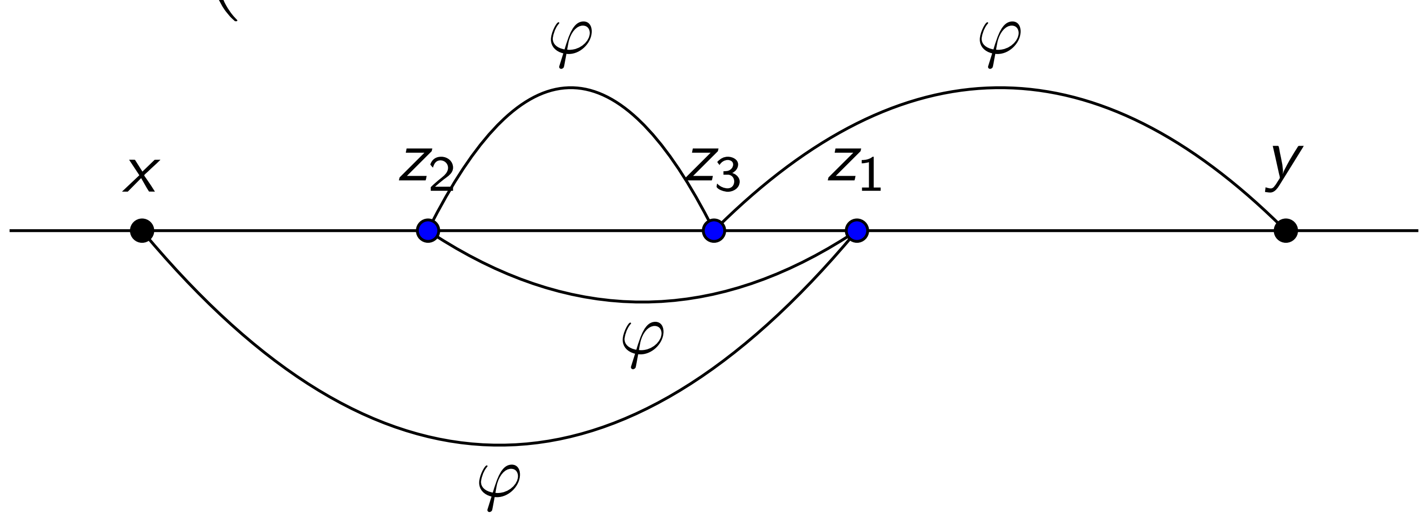
- Weighted FO misses a counting capability…
- Solution: weighted transitive closure operation

Logic equivalent to PWA?

- Weighted FO misses a counting capability…
- Solution: weighted transitive closure operation

$$\varphi^1(x, y) = \varphi(x, y)$$

$$\varphi^n(x, y) = \exists z_0 \cdots \exists z_n \Big(x = z_0 \wedge z_n = y \wedge \mathrm{diff}(z_0, \ldots, z_n) \wedge \big[\bigwedge_{1 \leq \ell \leq n} \varphi(z_{\ell-1}, z_\ell)\big] \Big)$$



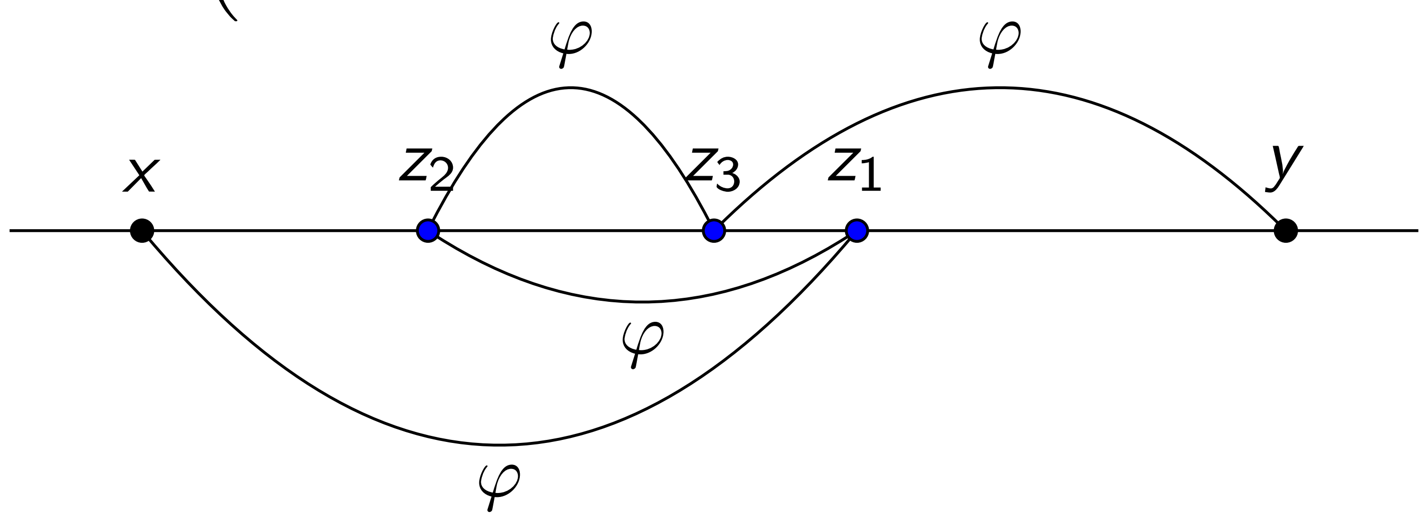
Logic equivalent to PWA?

- Weighted FO misses a counting capability…
- Solution: weighted transitive closure operation

$$\varphi^1(x,y) = \varphi(x,y)$$

$$\varphi^n(x,y) = \exists z_0 \cdots \exists z_n \Big(x = z_0 \wedge z_n = y \wedge \mathsf{diff}(z_0,\dots,z_n) \wedge \big[\bigwedge_{1\le \ell \le n} \varphi(z_{\ell-1}, z_\ell)\big]\Big)$$

$$\mathsf{TC}_{xy}\varphi = \bigvee_{n\geq 1} \varphi^n$$



Logic equivalent to PWA?

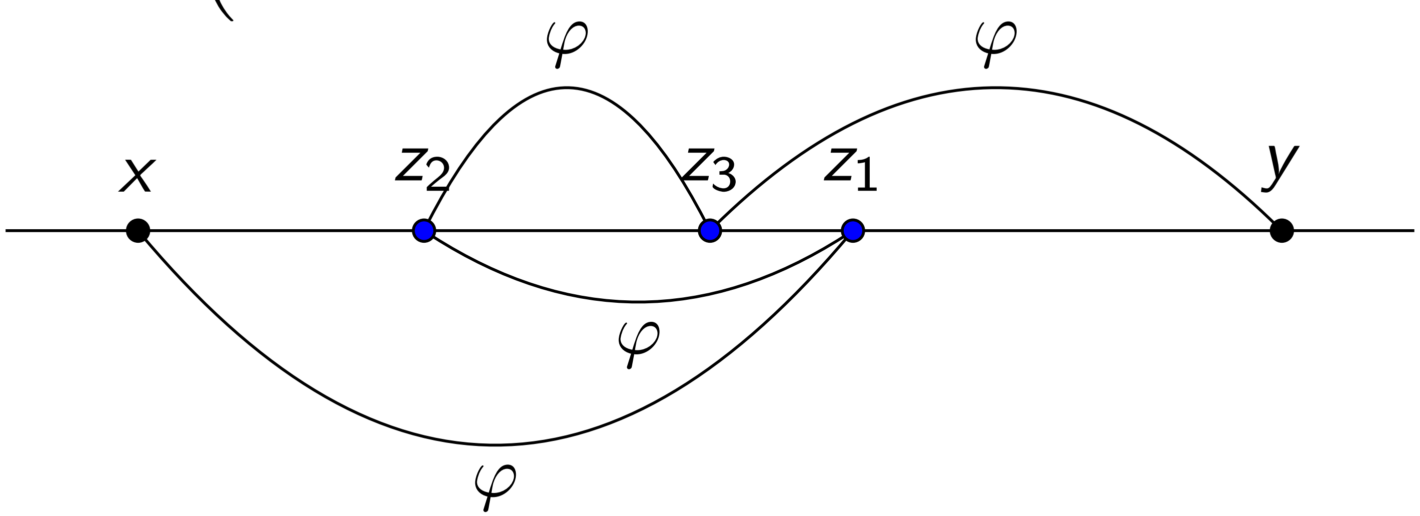
- Weighted FO misses a counting capability…

- Solution: weighted transitive closure operation

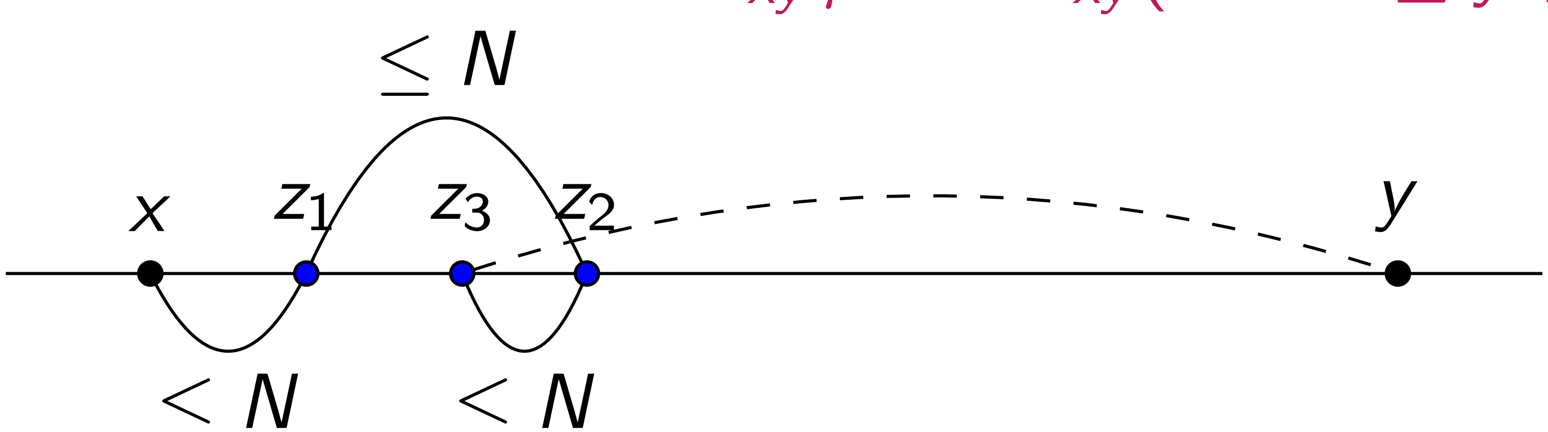
$$\varphi^1(x,y) = \varphi(x,y)$$

$$\varphi^n(x,y) = \exists z_0 \cdots \exists z_n \Big(x = z_0 \wedge z_n = y \wedge \mathrm{diff}(z_0,\dots,z_n) \wedge \big[\bigwedge_{1\le \ell \le n} \varphi(z_{\ell-1}, z_\ell) \big] \Big)$$

$$\mathrm{TC}_{xy}\varphi = \bigvee_{n\ge 1} \varphi^n$$



Bounded transitive closure : $N\text{-}\mathrm{TC}_{xy}\varphi = \mathrm{TC}_{xy}(x - N \le y \le x + N \wedge \varphi)$



Logic equivalent to PWA?

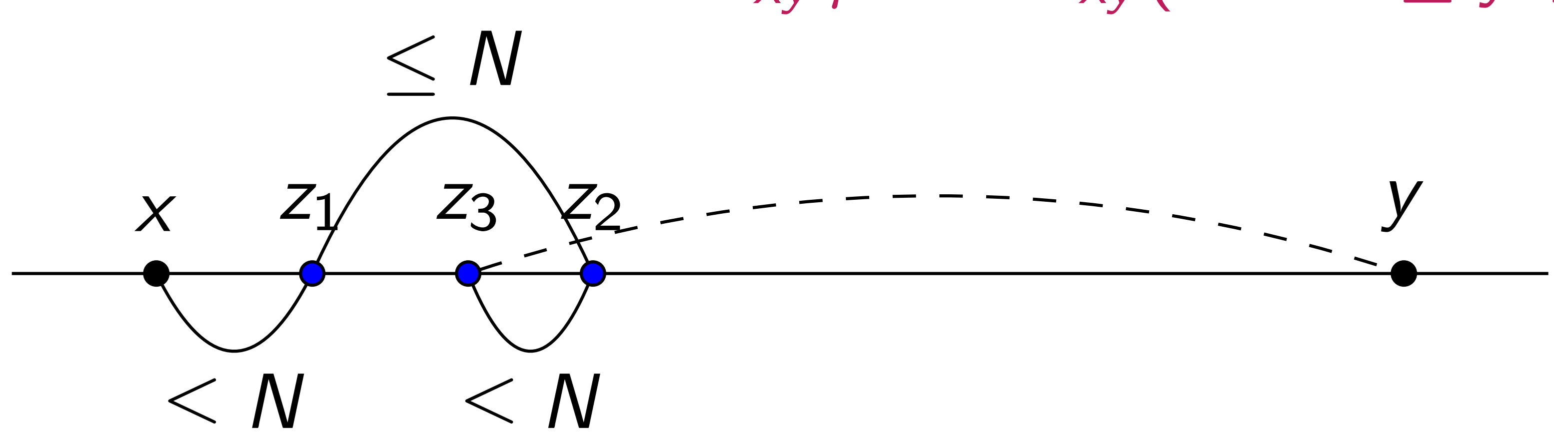
- Weighted FO misses a counting capability…
- Solution: weighted transitive closure operation

$$\varphi^1(x,y) = \varphi(x,y)$$

$$\varphi^n(x,y) = \exists z_0 \cdots \exists z_n \Big(x = z_0 \wedge z_n = y \wedge \mathrm{diff}(z_0,\dots,z_n) \wedge \big[\textstyle\bigwedge_{1\le \ell\le n} \varphi(z_{\ell-1},z_\ell)\big]\Big)$$

Theorem: PWA = wFO + bounded-TC

Bounded transitive closure : $N\text{-TC}_{xy}\varphi = \mathrm{TC}_{xy}(x - N \le y \le x + N \wedge \varphi)$



(Partial) conclusion

core wMSO = weighted automata

wFO + wTC = pebble navigating weighted automata

(Partial) conclusion

core wMSO = weighted automata

wFO + wTC = pebble navigating weighted automata

True for semirings, but other weight domains also!

average, discounted sums, ratio, energy…

True for finite words, but other input structures also!

(un)ranked trees, nested words, grids,
arbitrary graphs, infinite structures…

(Partial) conclusion

core wMSO = weighted automata

wFO + wTC = pebble navigating weighted automata

True for semirings, but other weight domains also!

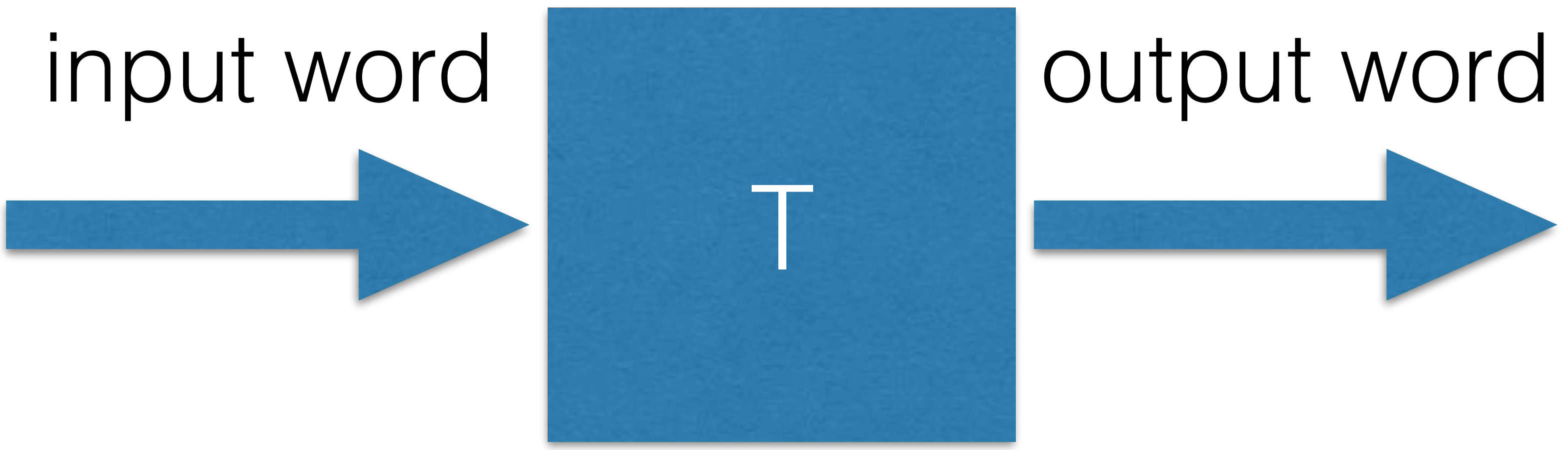
average, discounted sums, ratio, energy…

True for finite words, but other input structures also!

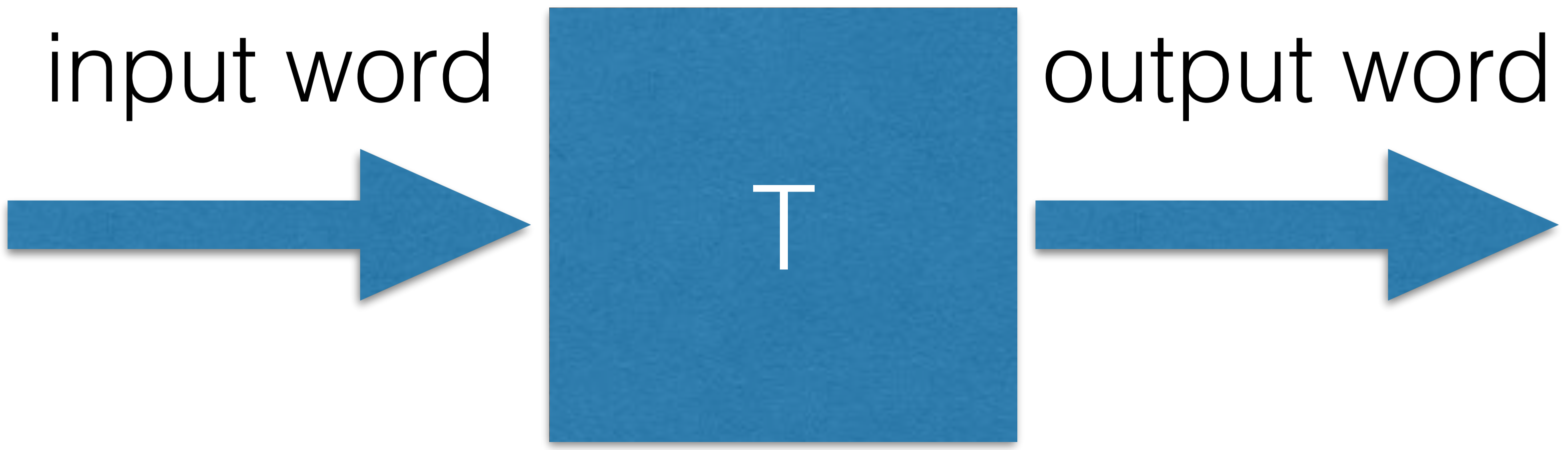
(un)ranked trees, nested words, grids,
arbitrary graphs, infinite structures…

Future works: more readable specification languages,
domain-specific applications

Application to transductions

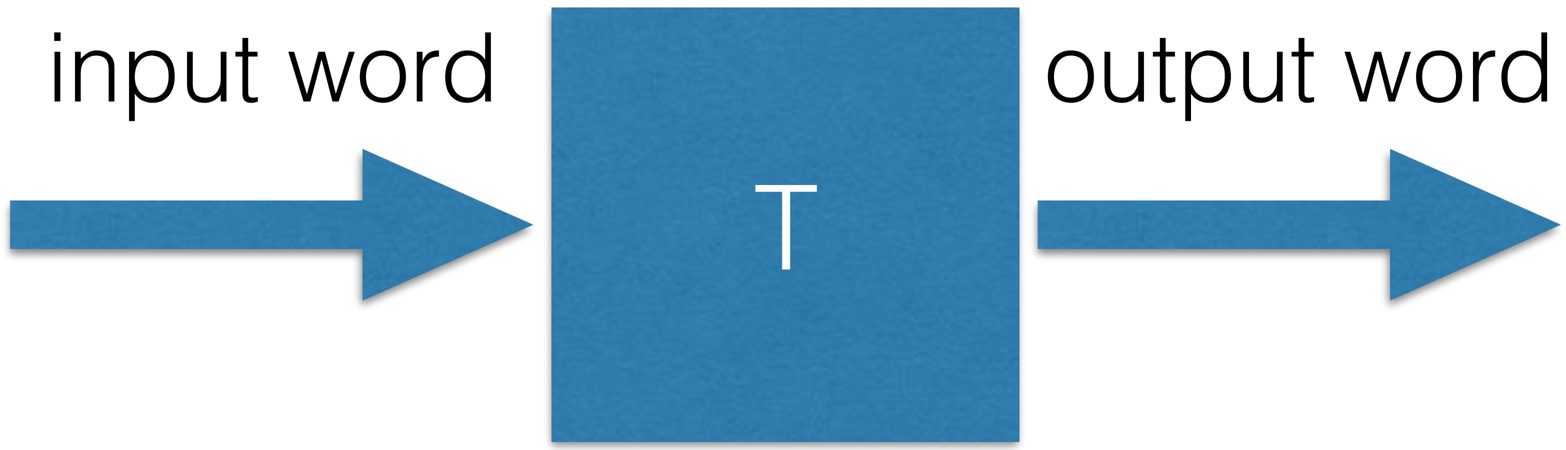


Application to transductions



Pattern
matching/replacement

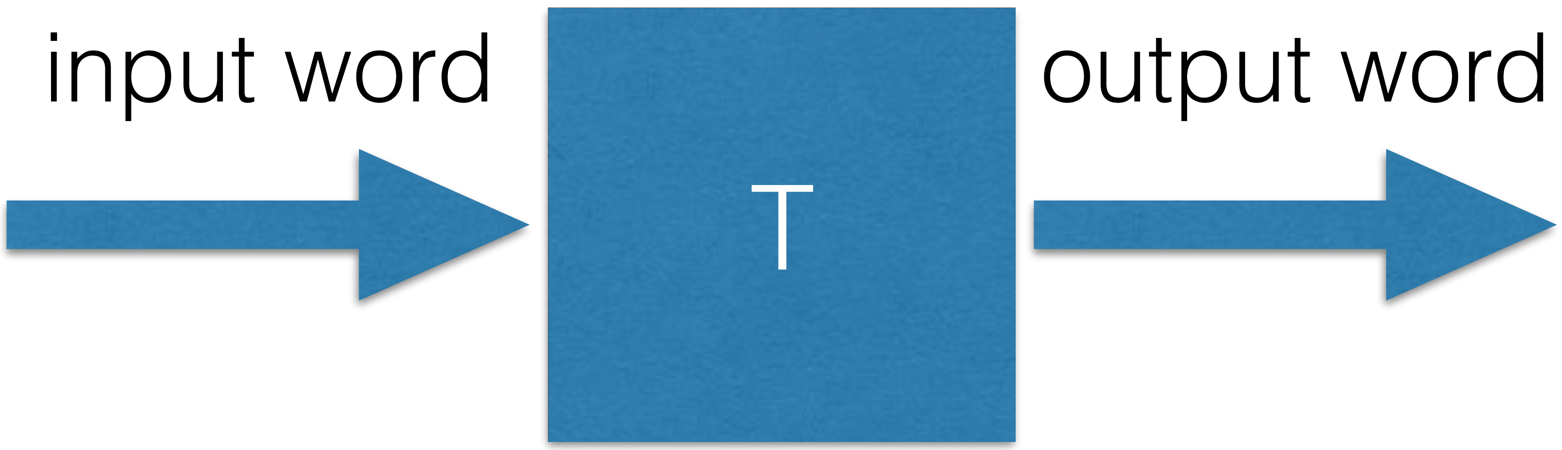
Application to transductions



Pattern matching/replacement

Tree/Graph rewriting

Application to transductions



Pattern matching/replacement

Tree/Graph rewriting

Update of XML databases

Existing models over words

Functions {

- Two-way Deterministic Finite-State Transducers
- Functional One-way Finite-State Transducers
- MSOT (à la Courcelle)
- Copyless Streaming String Transducers (Alur et al)

Existing models over words

Functions

- Two-way Deterministic Finite-State Transducers
- Functional One-way Finite-State Transducers
- MSOT (à la Courcelle)
- Copyless Streaming String Transducers (Alur et al)

Relations

- Two-way Non-Deterministic Finite-State Transducers
- Non-Deterministic Finite-State Transducers
- Non-deterministic Copyless Streaming String Transducers (Alur et Deshmukh)
- NMSOT (with free second-order variables)

Existing models over words

Functions
- Two-way Deterministic Finite-State Transducers
- Functional One-way Finite-State Transducers
- MSOT (à la Courcelle)
- Copyless Streaming String Transducers (Alur et al)

Relations
- Two-way Non-Deterministic Finite-State Transducers
- Non-Deterministic Finite-State Transducers
- Non-deterministic Copyless Streaming String Transducers (Alur et Deshmukh)
- NMSOT (with free second-order variables)

→ only finite valued relations…

Transduction as weights

- Desire: weight transitions with words… Difficult to equip A* with a semiring structure: how to combine several accepting runs?

- Works for deterministic or unambiguous automata: functional transducers

- For relations: semiring of languages

$$(2^{A^*}, \cup, \cdot, \varnothing, \{\varepsilon\})$$

Examples

$$\prod_x (P_x(a) ? \{aa\} : (P_x(b) ? \{bb\} : \varnothing))$$

Examples

$\prod_x (P_x(a)?\{aa\}:(P_x(b)?\{bb\}:\varnothing))$ *aba* -> *aabbaa*

Examples

$$\prod_x (P_x(a)?\{aa\} : (P_x(b)?\{bb\} : \varnothing))$$

aba -> *aabbaa*

$$\prod_x (P_x(\star)?\{insert\} : (P_x(a)?\{a\} : (P_x(b) : \{b\})))$$

Examples

$\prod_x (P_x(a)?\{aa\} : (P_x(b)?\{bb\} : \varnothing))$ *aba* -> *aabbaa*

$\prod_x (P_x(\star)?\{insert\} :(P_x(a)?\{a\} :(P_x(b) : \{b\})))$ *a*b* -> *ainsertb*

Examples

$$\prod_x (P_x(a)?\{aa\}:(P_x(b)?\{bb\}:\varnothing))$$ *aba* -> *aabbaa*

$$\prod_x (P_x(\star)?\{insert\}:(P_x(a)?\{a\}:(P_x(b):\{b\})))$$ *a*b* -> *ainsertb*

$$\sum_y P_y(\star)?\left[\prod_x (x=y)?\{insert\}:(P_x(\star)?\{\varepsilon\}:(P_x(a)?\{a\}:(P_x(b)?\{b\})))\right]$$

Examples

$$\prod_x (P_x(a)?\{aa\} : (P_x(b)?\{bb\} : \varnothing))$$

aba -> *aabbaa*

$$\prod_x (P_x(\star)?\{insert\} : (P_x(a)?\{a\} : (P_x(b) : \{b\})))$$

*a*b* -> *ainsertb*

$$\sum_y P_y(\star)?\left[\prod_x (x = y)?\{insert\} : (P_x(\star)?\{\varepsilon\} : (P_x(a)?\{a\} : (P_x(b)?\{b\})))\right]$$

*a*b*a* -> {*ainsertba,abinserta*}

Relation

Examples

$$\prod_x P_x(a)?\{a\}:(P_x(b)?\{\varepsilon\}) \times \prod_x P_x(a)?\{\varepsilon\}:(P_x(b):\{c\})$$

Examples

$$\prod_x P_x(a)?\{a\}:(P_x(b)?\{\varepsilon\})\times\prod_x P_x(a)?\{\varepsilon\}:(P_x(b):\{c\})$$

ababbaabb -> *aaaaccccc*

Examples

$$\prod_x P_x(a)?\{a\}:(P_x(b)?\{\varepsilon\})\times\prod_x P_x(a)?\{\varepsilon\}:($$

Not comp. by 1-way
Func Transducer

ababbaabb -> *aaaaccccc*

Examples

$$\prod_x P_x(a)?\{a\}:(P_x(b)?\{\varepsilon\})\times\prod_x P_x(a)?\{\varepsilon\}:($$

Not comp. by 1-way
Func Transducer

ababbaabb -> *aaaaccccc*

$$\prod_x P_x(a)?\{a\}:(P_x(b)?\{\varepsilon\})+\prod_x P_x(a)?\{\varepsilon\}:(P_x(b)\times\{c\})$$

Examples

$$\prod_x P_x(a)?\{a\}:(P_x(b)?\{\varepsilon\})\times\prod_x P_x(a)?\{\varepsilon\}:($$

Not comp. by 1-way
Func Transducer

ababbaabb -> *aaaaccccc*

$$\prod_x P_x(a)?\{a\}:(P_x(b)?\{\varepsilon\})+\prod_x P_x(a)?\{\varepsilon\}:(P_x(b)\times\{c\})$$

ababbaabb -> *{aaaa,ccccc}*

Examples

$$\prod_x P_x(a)?\{a\}:(P_x(b)?\{\varepsilon\})\times\prod_x P_x(a)?\{\varepsilon\}:($$

Not comp. by 1-way Func Transducer

ababbaabb -> *aaaaccccc*

$$\prod_x P_x(a)?\{a\}:(P_x(b)?\{\varepsilon\})+\prod_x P_x(a)?\{\varepsilon\}:(P_x(b)\times\{c\})$$

ababbaabb -> {*aaaa*,*ccccc*}

$$\prod_x P_x(a)?\{a,\varepsilon\}:(P_x(b)?\{b,\varepsilon\})$$

Examples

$$\prod_x P_x(a)?\{a\}:(P_x(b)?\{\varepsilon\})\times\prod_x P_x(a)?\{\varepsilon\}:($$

Not comp. by 1-way
Func Transducer

ababbaabb -> *aaaaccccc*

$$\prod_x P_x(a)?\{a\}:(P_x(b)?\{\varepsilon\})+\prod_x P_x(a)?\{\varepsilon\}:(P_x(b)\times\{c\})$$

ababbaabb -> {*aaaa*,*ccccc*}

$$\prod_x P_x(a)?\{a,\varepsilon\}:(P_x(b)?\{b,\varepsilon\})$$

aba -> {*ε*,*a*,*b*,*ab*,*ba*,*aa*,*aba*}

Examples

Not comp. by 1-way Func Transducer

$$\prod_x P_x(a)?\{a\}:(P_x(b)?\{\varepsilon\})\times\prod_x P_x(a)?\{\varepsilon\}:($$

ababbaabb -> *aaaaccccc*

$$\prod_x P_x(a)?\{a\}:(P_x(b)?\{\varepsilon\})+\prod_x P_x(a)?\{\varepsilon\}:(P_x(b)\times\{c\})$$

ababbaabb -> {*aaaa*,*ccccc*}

$$\prod_x P_x(a)?\{a,\varepsilon\}:(P_x(b)?\{b,\varepsilon\})$$

aba -> {*ε*,*a*,*b*,*ab*,*ba*,*aa*,*aba*}

$$\prod_x P_x(a)?A^*aA^*:(P_x(b)?A^*bA^*)$$

Examples

$$\prod_x P_x(a)?\{a\}:(P_x(b)?\{\varepsilon\})\times\prod_x P_x(a)?\{\varepsilon\}:($$

Not comp. by 1-way
Func Transducer

ababbaabb -> *aaaaccccc*

$$\prod_x P_x(a)?\{a\}:(P_x(b)?\{\varepsilon\})+\prod_x P_x(a)?\{\varepsilon\}:(P_x(b)\times\{c\})$$

ababbaabb -> {*aaaa*,*ccccc*}

$$\prod_x P_x(a)?\{a,\varepsilon\}:(P_x(b)?\{b,\varepsilon\})$$

aba

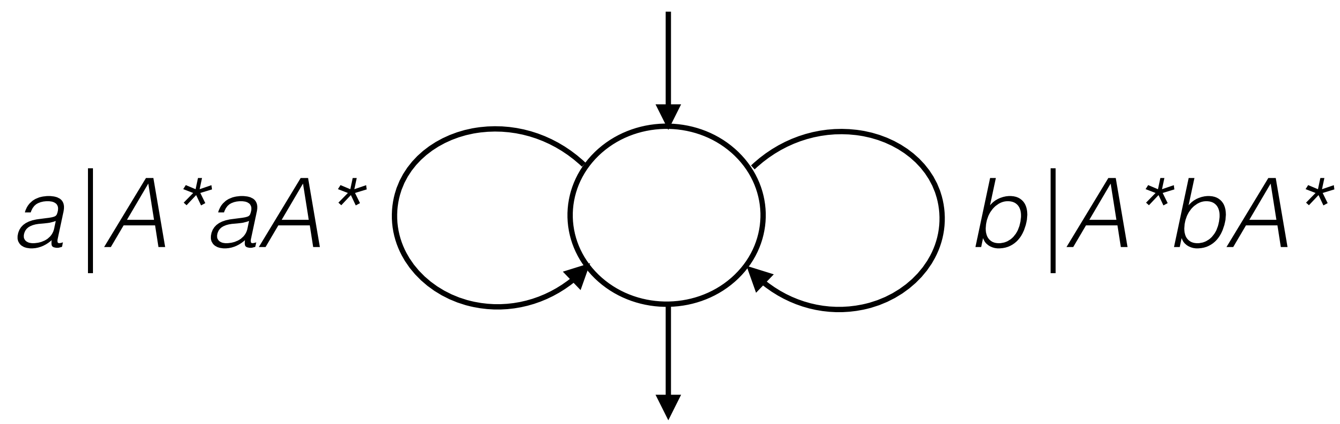
Infinitely-valued relation

$$\prod_x P_x(a)?A^*aA^*:(P_x(b)?A^*bA^*)$$

aba -> *A**aA*bA*aA**

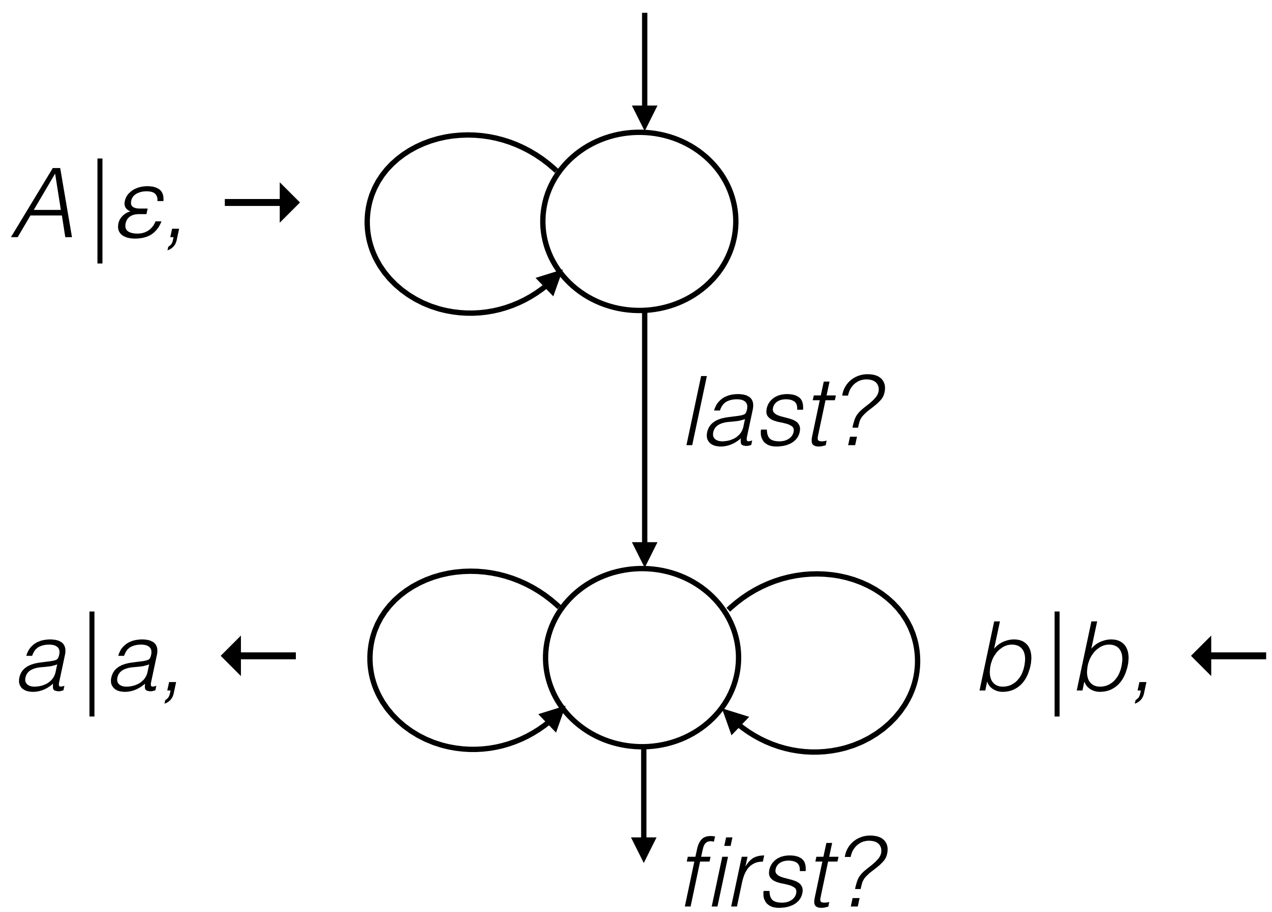
Transducers

$$\prod_x P_x(a)?\,A^* a A^* : (P_x(b)?\,A^* b A^*)$$

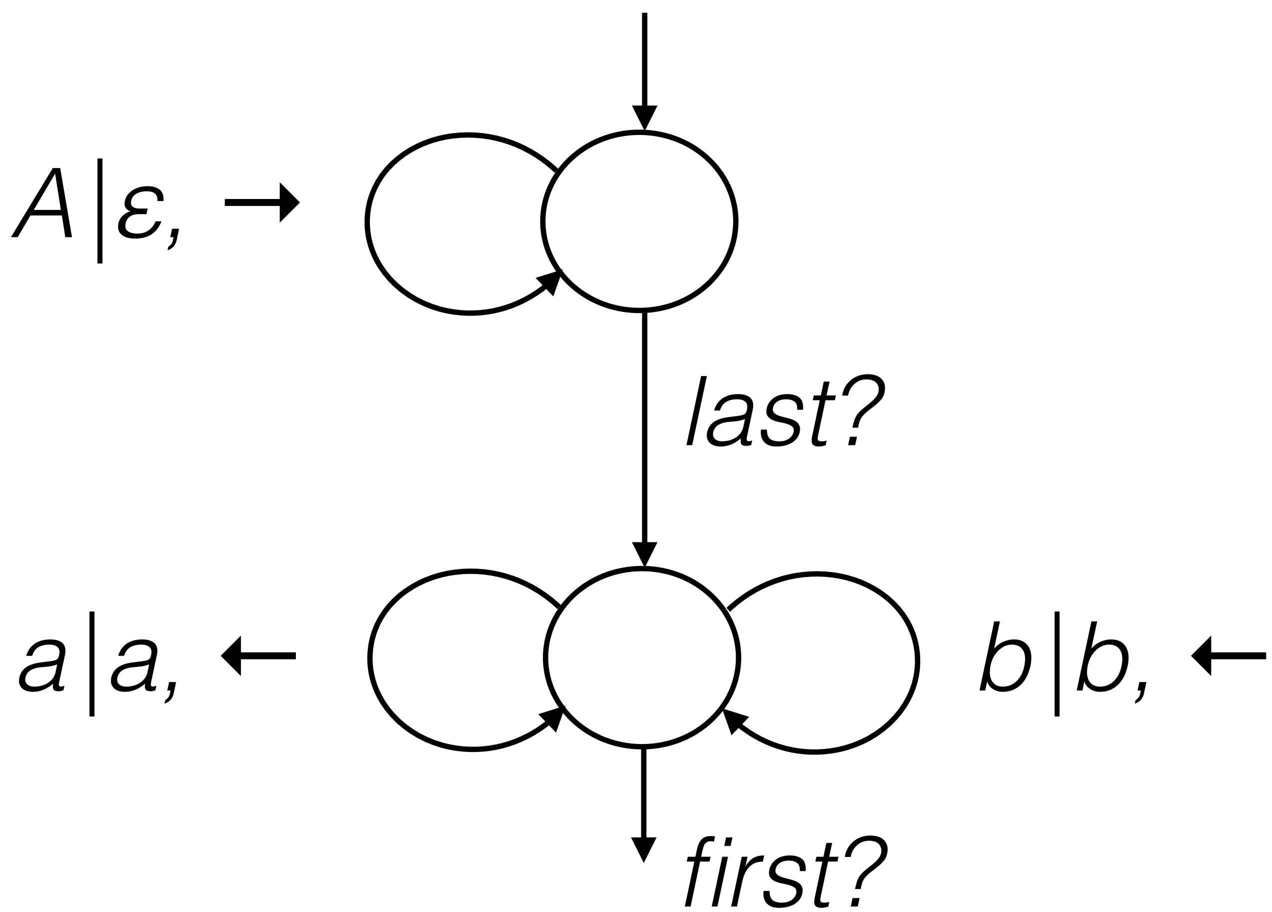


Infinite-valued, but deterministic

Reverse?

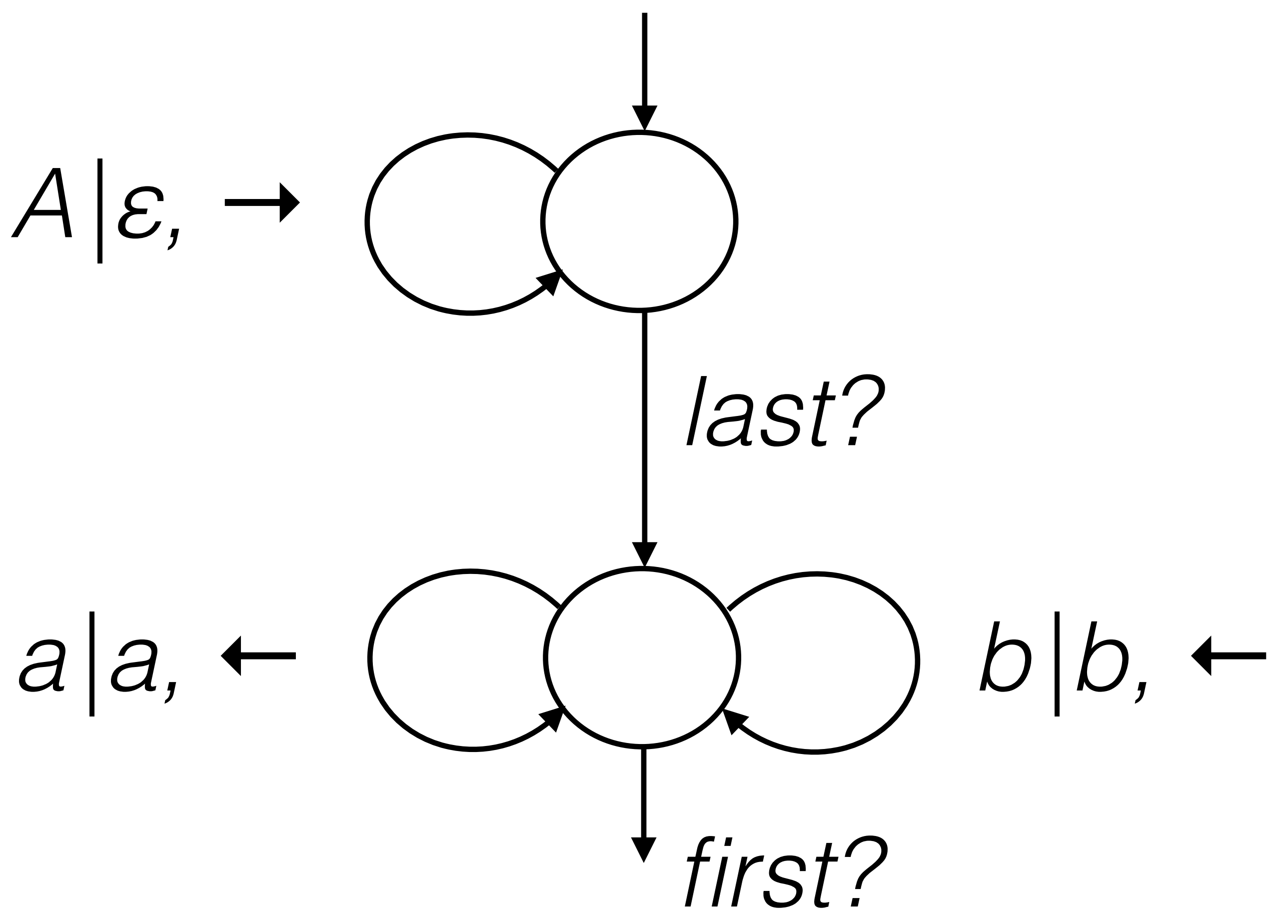


Reverse?



Impossible in FO…
… because of order of interpretation of product

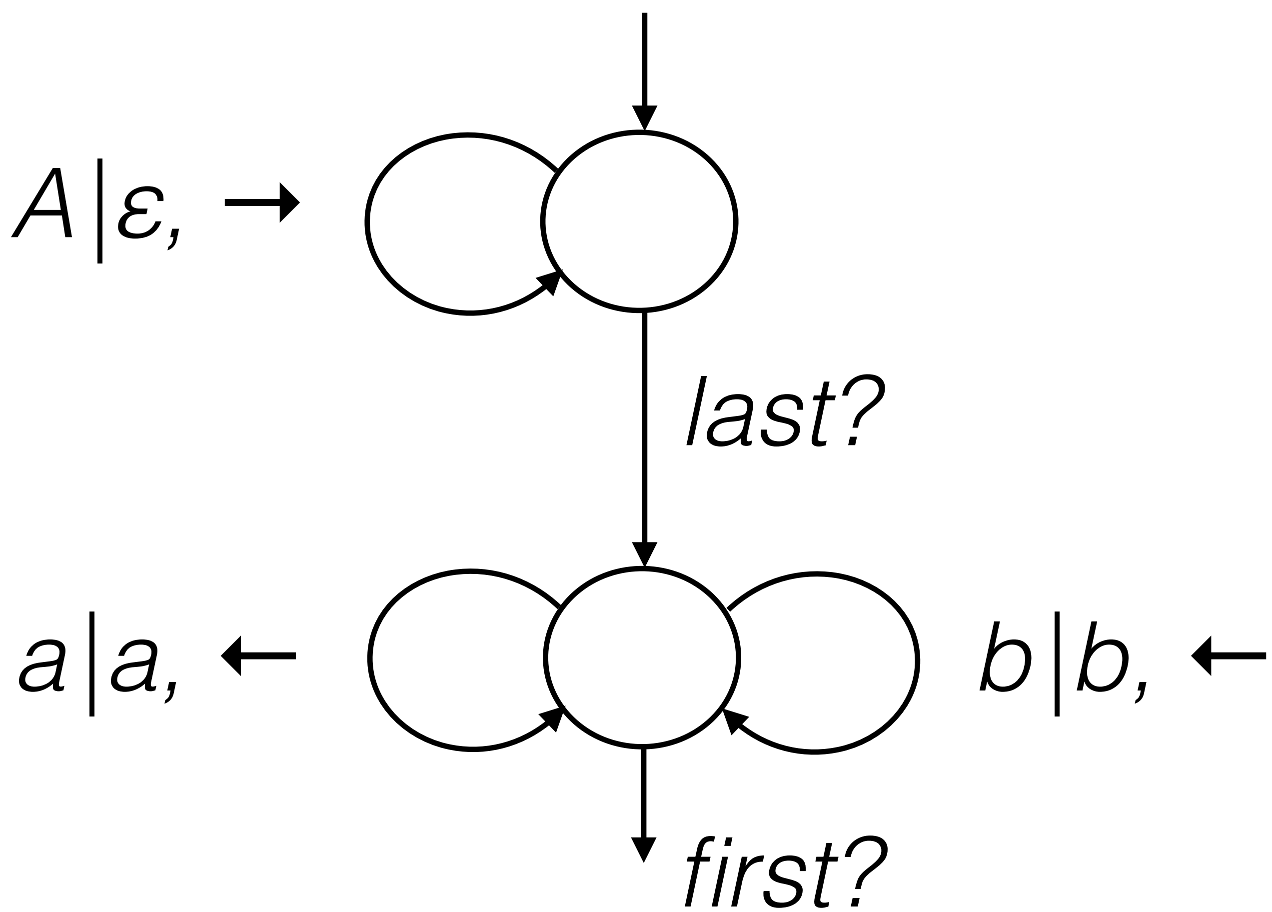
Reverse?



Impossible in FO…
… because of order of interpretation of product

Solution: in this non-commutative setting, add right-to-left products

Reverse?



Impossible in FO…
… because of order of interpretation of product

Solution: in this non-commutative setting, add right-to-left products

$$\coprod_x P_x(a)?\{a\} : (P_x(b)?\{b\})$$

Transitive closure

$$\varphi ::= \top \mid P_a(x) \mid x \leqslant y \mid x \in X \mid \neg\varphi \mid \varphi \wedge \varphi \mid \forall x\, \varphi$$

$$\Phi ::= L \mid \varphi ? \Phi : \Phi \mid \Phi + \Phi \mid \Phi \times \Phi \mid \sum_x \Phi \mid \prod_x \Phi \mid N\text{-}TC_{x,y} \Phi$$

Transitive closure

$$\varphi ::= \top \mid P_a(x) \mid x \leqslant y \mid x \in X \mid \neg\varphi \mid \varphi \wedge \varphi \mid \forall x\, \varphi$$

$$\Phi ::= L \mid \varphi ? \Phi : \Phi \mid \Phi + \Phi \mid \Phi \times \Phi \mid \sum_x \Phi \mid \prod_x \Phi \mid N\text{-}TC_{x,y} \Phi$$

Regular language

Transitive closure

$$\varphi ::= \top \mid P_a(x) \mid x \leqslant y \mid x \in X \mid \neg\varphi \mid \varphi \wedge \varphi \mid \forall x\, \varphi$$

$$\Phi ::= L \mid \varphi ? \Phi : \Phi \mid \Phi + \Phi \mid \Phi \times \Phi \mid \sum_x \Phi \mid \prod_x \Phi \mid N\text{-}TC_{x,y} \Phi$$

Regular language

Able to define right-to-left product

$$\coprod_x \Phi(x) \coloneqq [1\text{-}TC_{x,y}(y = x - 1 ? \Phi(x))](\mathit{last}, \mathit{first}) \times \Phi(\mathit{first})$$

Transitive closure

$$\varphi ::= \top \mid P_a(x) \mid x \leqslant y \mid x \in X \mid \neg\varphi \mid \varphi \wedge \varphi \mid \forall x\, \varphi$$

$$\Phi ::= L \mid \varphi ?\Phi : \Phi \mid \Phi + \Phi \mid \Phi \times \Phi \mid \sum_x \Phi \mid \prod_x \Phi \mid N\text{-}TC_{x,y}\Phi$$

Regular language

Able to define right-to-left product

$$\coprod_x \Phi(x) \coloneqq [1\text{-}TC_{x,y}(y = x - 1 ? \Phi(x))](last, first) \times \Phi(first)$$

Theorem: Pebble Transducers = FO + bounded-TC

with regular language productions

linear transformation from logic to transducers

Algorithmic questions

Algorithmic questions

Theorem: Evaluation of FO + bounded-TC with complexity $O(|formula| \times |input|^{\#variables})$

Algorithmic questions

Theorem: Evaluation of FO + bounded-TC with complexity $O(|formula| x |input|^{\#variables})$

- In functional setting: it is decidable if a 2-way transducer is recognisable by a 1-way transducer [Filiot,Gauwin,Reynier, Servais, 13]

Algorithmic questions

Theorem: Evaluation of FO + bounded-TC with complexity $O(|\text{formula}| \times |\text{input}|^{\#\text{variables}})$

- In functional setting: it is decidable if a 2-way transducer is recognisable by a 1-way transducer [Filiot,Gauwin,Reynier, Servais, 13]
  - What about the general setting, with pebbles?

Algorithmic questions

Theorem: Evaluation of FO + bounded-TC with complexity O(|formula|x|input|$^{\#variables}$)

- In functional setting: it is decidable if a 2-way transducer is recognisable by a 1-way transducer [Filiot,Gauwin,Reynier, Servais, 13]
  - What about the general setting, with pebbles?
- Determinisability of functional non-det transducers is decidable in PTIME [Schützenberger, 75]

Algorithmic questions

Theorem: Evaluation of FO + bounded-TC with complexity O(|formula|x|input|$^{\#variables}$)

- In functional setting: it is decidable if a 2-way transducer is recognisable by a 1-way transducer [Filiot,Gauwin,Reynier, Servais, 13]
  - What about the general setting, with pebbles?
- Determinisability of functional non-det transducers is decidable in PTIME [Schützenberger, 75]
  - Extension to the general setting?

Algorithmic questions

Theorem: Evaluation of FO + bounded-TC with complexity O(|formula|x|input|$^{\#variables}$)

- In functional setting: it is decidable if a 2-way transducer is recognisable by a 1-way transducer [Filiot,Gauwin,Reynier, Servais, 13]
  - What about the general setting, with pebbles?
- Determinisability of functional non-det transducers is decidable in PTIME [Schützenberger, 75]
  - Extension to the general setting?

Thank you!