Logical Characterization of Weighted Pebble Walking Automata

Benjamin Monmege
Université libre de Bruxelles, Belgium

Benedikt Bollig and Paul Gastin (LSV, ENS Cachan, France)
Marc Zeitoun (LaBRI, Bordeaux University, France)

CSL-LICS 2014

Vienna - July 15, 2014

Equivalence between automata and logic

- Well-known and studied model of computation: NFA over words
- Existing extensions over trees, grids, graphs...
- Robustness of automata intrinsically linked to logical characterization

Equivalence between automata and logic

- Well-known and studied model of computation: NFA over words
- Existing extensions over trees, grids, graphs...
- Robustness of automata intrinsically linked to logical characterization

- Büchi-Elgot-Trakhtenbrot: NFA vs MSO
- Engelfriet-Hoogeboom: pebble walking automata vs FOposTC
- Droste-Gastin: weighted automata vs restricted weighted MSO

Equivalence between automata and logic

- Well-known and studied model of computation: NFA over words
- Existing extensions over trees, grids, graphs...
- Robustness of automata intrinsically linked to logical characterization

- Büchi-Elgot-Trakhtenbrot: NFA vs MSO
- Engelfriet-Hoogeboom: pebble walking automata vs FOposTC
- Droste-Gastin: weighted automata vs restricted weighted MSO

- Aim: **extend Engelfriet-Hoogeboom result to the quantitative setting, relating weighted pebble walking automata with weighted FOposTC**

Graphs as a general model

Words: $D = \{\rightarrow\}$
computations of sequential programs

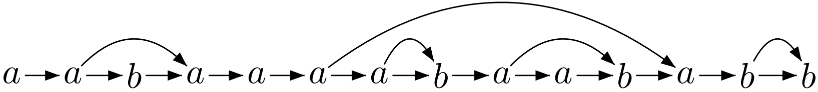
$$a \rightarrow a \rightarrow b \rightarrow a \rightarrow a \rightarrow a \rightarrow a \rightarrow b \rightarrow a \rightarrow a \rightarrow b \rightarrow a \rightarrow b \rightarrow b$$

Graphs as a general model

Words: $D = \{\rightarrow\}$
computations of sequential programs

$$a \rightarrow a \rightarrow b \rightarrow a \rightarrow a \rightarrow a \rightarrow a \rightarrow b \rightarrow a \rightarrow a \rightarrow b \rightarrow a \rightarrow b \rightarrow b$$

Nested words: $D = \{\rightarrow, \curvearrowright\}$
computations of recursive programs, XML documents



Graphs as a general model

Words: $D = \{\rightarrow\}$

computations of sequential programs

$$a \rightarrow a \rightarrow b \rightarrow a \rightarrow a \rightarrow a \rightarrow a \rightarrow b \rightarrow a \rightarrow a \rightarrow b \rightarrow a \rightarrow b \rightarrow b$$

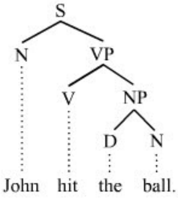
Nested words: $D = \{\rightarrow, \curvearrowright\}$

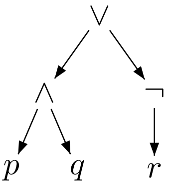
computations of recursive programs, XML documents

$$a \rightarrow a \rightarrow b \rightarrow a \rightarrow a \rightarrow a \rightarrow a \rightarrow b \rightarrow a \rightarrow a \rightarrow b \rightarrow a \rightarrow b \rightarrow b$$

Ranked trees: $D = \{\downarrow_1, \downarrow_2\}$

expressions, formulae, parse trees





Graphs as a general model

Definition: directed graphs

$G = (V, (E_d)_{d\in D}, \lambda, \iota)$ where

- V is a nonempty and finite set of vertices;
- for all edge label $d \in D$, $E_d \subseteq V \times V$ is an *irreflexive relation*, describing the d-edges of the graph, which is *deterministic and codeterministic*;
- $\lambda\colon V \to A$ is a vertex-labeling function;
- $\iota \in V$ is an initial vertex.

For all edge label d, we consider its reverse d^{-1} letting $E_{d^{-1}} = (E_d)^{-1}$

Graphs as a general model

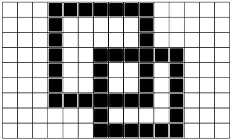
Definition: directed graphs

$G = (V, (E_d)_{d \in D}, \lambda, \iota)$ where

- V is a nonempty and finite set of vertices;
- for all edge label $d \in D$, $E_d \subseteq V \times V$ is an *irreflexive relation*, describing the d-edges of the graph, which is *deterministic and codeterministic*;
- $\lambda \colon V \to A$ is a vertex-labeling function;
- $\iota \in V$ is an initial vertex.

For all edge label d, we consider its reverse d^{-1} letting $E_{d^{-1}} = (E_d)^{-1}$

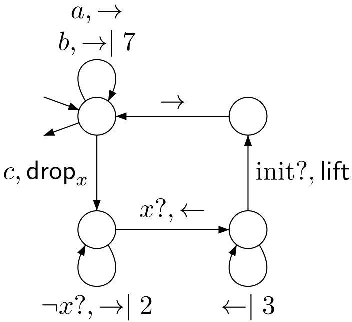
Grids: $D = \{\rightarrow, \uparrow\}$
pictures



Weighted pebble walking automata

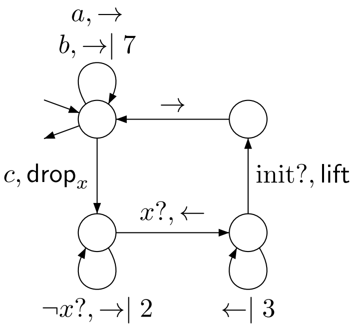
Definition:

- Finite number of states, initial and final states
- Ability to navigate in the graph (using the deterministic edge labels)
- Bounded supply of pebbles able to mark temporarily a position
- Pebbles are treated with a stack policy: first pebble to lift is the last dropped pebble
- Transitions equipped with weights in a complete semiring $(\mathbf{S}, +, \times, 0, 1)$



Examples of complete semirings:
$(\{0,1\}, \vee, \wedge, 0, 1)$
$(\mathbb{R}^+ \cup \{+\infty\}, +, \times, 0, 1)$
$(\mathbb{Z} \cup \{+\infty, -\infty\}, \min, +, +\infty, 0)$
$(\mathbb{Z} \cup \{+\infty, -\infty\}, \max, +, -\infty, 0)$
$([0,1], \min, \max, 1, 0)$
$(2^{\Sigma^\star}, \cup, \cdot, \emptyset, \{\varepsilon\})$

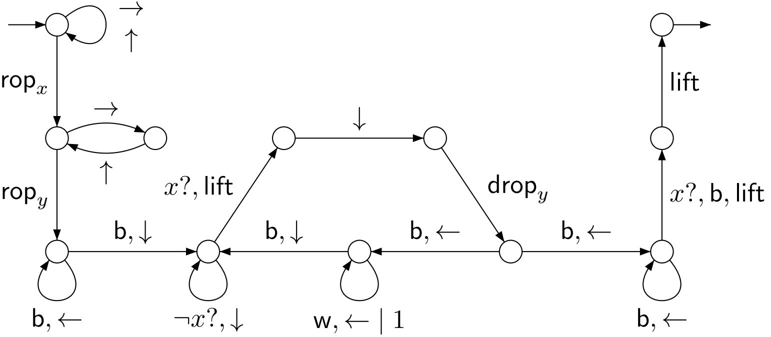
Weighted pebble walking automata

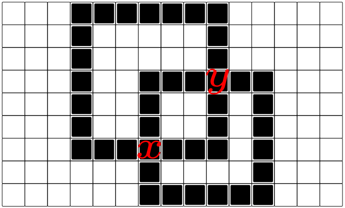


Definition: Semantics over $(\mathbf{S}, +, \times, 0, 1)$

- Configurations over a graph G: (G, q, π, v) with state q, stack π of pebble positions and current vertex v
- Weight of a run: multiplication of the weights of transitions
- Semantics $[\![\mathcal{A}]\!](G)$: sum of weights of accepting runs over G

Example of weighted pebble walking automaton

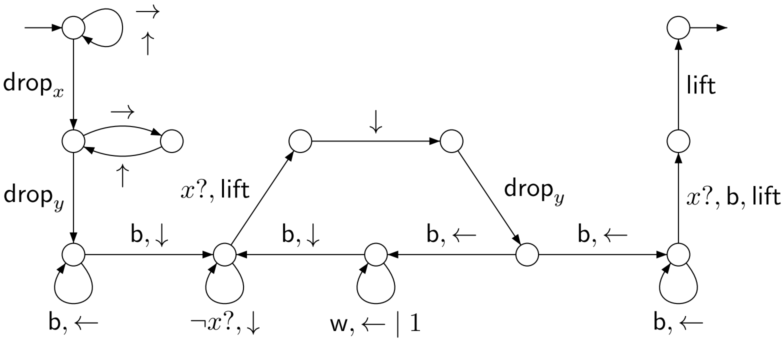


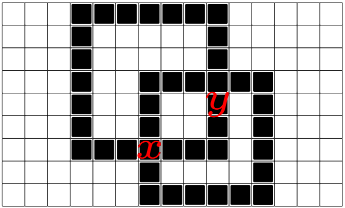


computes the biggest size of frames (empty black square)

$(\mathbb{N} \cup \{-\infty\}, \max, +, -\infty, 0)$

Example of weighted pebble walking automaton

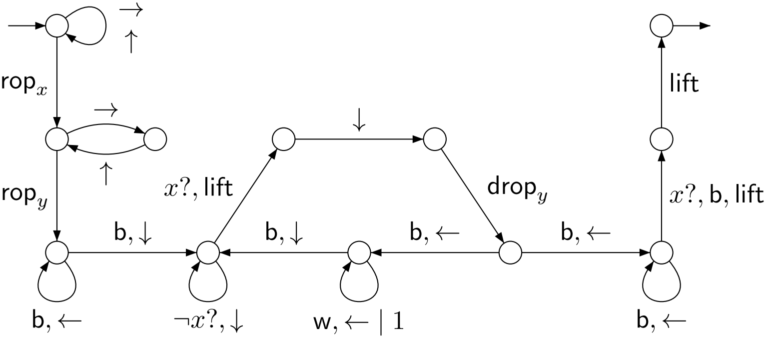


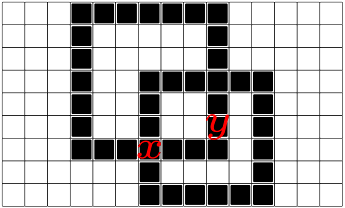


computes the biggest size of frames (empty black square)

$(\mathbb{N} \cup \{-\infty\}, \max, +, -\infty, 0)$

Example of weighted pebble walking automaton

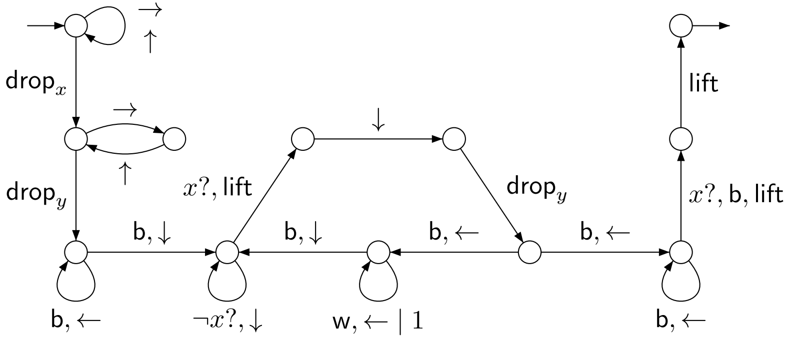


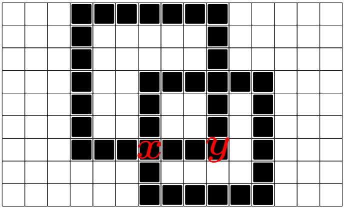


computes the biggest size of frames (empty black square)

$(\mathbb{N} \cup \{-\infty\}, \max, +, -\infty, 0)$

Example of weighted pebble walking automaton





computes the biggest size of frames (empty black square)

$(\mathbb{N} \cup \{-\infty\}, \max, +, -\infty, 0)$

Logical characterization

Classical weighted automata are **one-way** (sometimes branching) and **without pebbles**

Logical characterization for them in terms of a restricted weighted MSO logic:

- over words [Droste and Gastin, 2009]
- over trees [Droste and Vogler, 2006]
- over grids [Fichtner, 2011]
- over nested words [Mathissen, 2010]...

Logical characterization

Classical weighted automata are **one-way** (sometimes branching) and **without pebbles**

Logical characterization for them in terms of a restricted weighted MSO logic:

- over words [Droste and Gastin, 2009]
- over trees [Droste and Vogler, 2006]
- over grids [Fichtner, 2011]
- over nested words [Mathissen, 2010]...

Restricted weighted MSO does not even contain full weighted FO a priori

Theorem: Our contribution

Weighted pebble walking automata over graphs $(\mathrm{wPWA}) = \mathrm{wFOTC}$

Weighted first-order logic

Definition:

Classical first-order logic

$$\varphi ::= \top \mid (x = y) \mid \mathrm{init}(x) \mid P_a(x) \mid R_d(x,y) \mid R_d^+(x,y) \mid \neg\varphi \mid \varphi \vee \varphi \mid \exists x\, \varphi$$

Weighted first-order logic

Definition:

Classical first-order logic

$$\varphi ::= \top \mid (x = y) \mid \mathrm{init}(x) \mid P_a(x) \mid R_d(x, y) \mid R_d^+(x, y) \mid \neg\varphi \mid \varphi \vee \varphi \mid \exists x\, \varphi$$

Weighted first-order logic over graphs with weights in a semiring $(\mathbf{S}, +, \times, 0, 1)$

$$\Phi ::= s \mid \varphi \,?\, \Phi : \Phi \mid \Phi \oplus \Phi \mid \Phi \otimes \Phi \mid \bigoplus_x \Phi \mid \bigotimes_x \Phi$$

Weighted first-order logic

Definition:

Classical first-order logic

$$\varphi ::= \top \mid (x = y) \mid \mathrm{init}(x) \mid P_a(x) \mid R_d(x,y) \mid R_d^+(x,y) \mid \neg\varphi \mid \varphi \vee \varphi \mid \exists x\, \varphi$$

Weighted first-order logic over graphs with weights in a semiring $(\mathbf{S}, +, \times, 0, 1)$

$$\Phi ::= s \mid \varphi \,?\, \Phi : \Phi \mid \Phi \oplus \Phi \mid \Phi \otimes \Phi \mid \bigoplus_x \Phi \mid \bigotimes_x \Phi$$

Semantics over a graph G and a valuation σ of free variables

$$[\![\varphi \,?\, \Phi_1 : \Phi_2]\!](G,\sigma) = \begin{cases} [\![\Phi_1]\!](G,\sigma) & \text{if } G,\sigma \models \varphi \\ [\![\Phi_2]\!](G,\sigma) & \text{otherwise} \end{cases}$$

$$[\![\Phi_1 \oplus \Phi_2]\!](G,\sigma) = [\![\Phi_1]\!](G,\sigma) + [\![\Phi_2]\!](G,\sigma) \qquad [\![\bigoplus_x \Phi]\!](G,\sigma) = \sum_{v \in V} [\![\Phi]\!](G, \sigma[x \mapsto v])$$

$$[\![\Phi_1 \otimes \Phi_2]\!](G,\sigma) = [\![\Phi_1]\!](G,\sigma) \times [\![\Phi_2]\!]G,\sigma) \qquad [\![\bigotimes_x \Phi]\!](G,\sigma) = \prod_{v \in V} [\![\Phi]\!](G, \sigma[x \mapsto v])$$

Weighted first-order logic

Definition:

Classical first-order logic

$$\varphi ::= \top \mid (x = y) \mid \mathrm{init}(x) \mid P_a(x) \mid R_d(x,y) \mid R_d^+(x,y) \mid \neg\varphi \mid \varphi \vee \varphi \mid \exists x\, \varphi$$

Weighted first-order logic over graphs with weights in a semiring $(\mathbf{S}, +, \times, 0, 1)$

$$\Phi ::= s \mid \varphi \,?\, \Phi : \Phi \mid \Phi \oplus \Phi \mid \Phi \otimes \Phi \mid \bigoplus_x \Phi \mid \bigotimes_x \Phi$$

Semantics over a graph G and a valuation σ of free variables

$$[\![\varphi \,?\, \Phi_1 : \Phi_2]\!](G,\sigma) = \begin{cases} [\![\Phi_1]\!](G,\sigma) & \text{if } G,\sigma \models \varphi \\ [\![\Phi_2]\!](G,\sigma) & \text{otherwise} \end{cases}$$

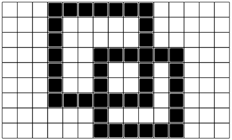
$$[\![\Phi_1 \oplus \Phi_2]\!](G,\sigma) = [\![\Phi_1]\!](G,\sigma) + [\![\Phi_2]\!](G,\sigma) \qquad [\![\bigoplus_x \Phi]\!](G,\sigma) = \sum_{v\in V} [\![\Phi]\!](G,\sigma[x\mapsto v])$$

$$[\![\Phi_1 \otimes \Phi_2]\!](G,\sigma) = [\![\Phi_1]\!](G,\sigma) \times [\![\Phi_2]\!]G,\sigma) \qquad [\![\bigotimes_x \Phi]\!](G,\sigma) = \prod_{v\in V} [\![\Phi]\!](G,\sigma[x\mapsto v])$$

Examples in $(\mathbb{N}\cup\{-\infty\}, \max, +, -\infty, 0)$

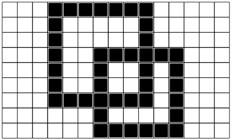
$\Phi_{\mathrm{b}} = \bigotimes_x P_{\mathrm{b}}(x) \,?\, 1 : 0$ $\qquad$ $\Phi_{\mathrm{w}} = \bigotimes_x P_{\mathrm{w}}(x) \,?\, 1 : 0$ $\qquad$ $\Phi_{\mathrm{b}} \oplus \Phi_{\mathrm{w}}$

Transitive closure in graphs



Binary predicate $R_{\nearrow}(x,y) = \exists z[R_{\rightarrow}(x,z) \wedge R_{\uparrow}(z,y)]$
Transitive Closure $\mathrm{TC}_{x,y} R_{\nearrow}(x,y)$
test if square (not doable in FO)

Transitive closure in graphs



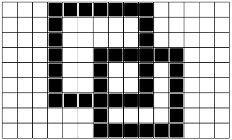
Binary predicate $R_{\nearrow}(x,y) = \exists z[R_{\rightarrow}(x,z) \wedge R_{\uparrow}(z,y)]$
Transitive Closure $\mathrm{TC}_{x,y} R_{\nearrow}(x,y)$
test if square (not doable in FO)

Weighted transitive closure: semiring $(\mathbb{N} \cup \{-\infty\}, \max, +, -\infty, 0)$

$$\mathrm{TC}_{x,y}[R_{\nearrow}(x,y)\ ?\ 1 : -\infty]$$

verifies that it is a square **and** computes the length of its diagonal

Transitive closure in graphs



Binary predicate $R_{\nearrow}(x,y) = \exists z [R_{\rightarrow}(x,z) \wedge R_{\uparrow}(z,y)]$
Transitive Closure $\mathrm{TC}_{x,y} R_{\nearrow}(x,y)$
test if square (not doable in FO)

Weighted transitive closure: semiring $(\mathbb{N} \cup \{-\infty\}, \max, +, -\infty, 0)$

$$\mathrm{TC}_{x,y}[R_{\nearrow}(x,y)\,?\,1 : -\infty]$$

verifies that it is a square **and** computes the length of its diagonal

Semantics given in a complete semiring $(\mathbf{S}, +, \times, 0, 1)$

$$[\![[\mathrm{TC}_{x,y}\Phi](x',y')]\!](G,\sigma) = \sum_{\substack{v_0,v_1,\dots,v_m\ (m>0)\\ \sigma(x')=v_0,\sigma(y')=v_m}} \prod_{0\leqslant k\leqslant m-1} [\![\Phi]\!](G,\sigma[x\mapsto v_k, y\mapsto v_{k+1}])$$

sum along
sequences of stop-vertices

multiplication along
the sequence

Bounding the Transitive Closure

- A necessary restriction to obtain a fragment of logic expressively equivalent to wPWA
- But not so restrictive in most of the cases!

$$\mathrm{TC}^{N}_{x,y}\Phi(x,y) = \mathrm{TC}_{x,y}[\mathsf{dist}(x,y) \leqslant N \,?\, \Phi(x,y) : 0]$$

Bounding the Transitive Closure

- A necessary restriction to obtain a fragment of logic expressively equivalent to wPWA
- But not so restrictive in most of the cases!

$$\mathrm{TC}^{N}_{x,y}\Phi(x,y) = \mathrm{TC}_{x,y}[\mathsf{dist}(x,y) \leqslant N \,?\, \Phi(x,y) : 0]$$

Previous example: $\mathrm{TC}_{x,y}[R_{\nearrow}(x,y)\,?\,1 : -\infty] = \mathrm{TC}^{2}_{x,y}[R_{\nearrow}(x,y)\,?\,1 : -\infty]$

Bounding the Transitive Closure

- A necessary restriction to obtain a fragment of logic expressively equivalent to wPWA
- But not so restrictive in most of the cases!

$$\mathrm{TC}^{N}_{x,y}\Phi(x,y) = \mathrm{TC}_{x,y}[\mathsf{dist}(x,y) \leqslant N \, ? \, \Phi(x,y) : 0]$$

Previous example: $\mathrm{TC}_{x,y}[R_{\nearrow}(x,y) \, ? \, 1 : -\infty] = \mathrm{TC}^{2}_{x,y}[R_{\nearrow}(x,y) \, ? \, 1 : -\infty]$

Definition: Logic wFOTC

$$\Phi ::= s \mid \varphi \, ? \, \Phi : \Phi \mid \Phi \oplus \Phi \mid \Phi \otimes \Phi \mid \bigoplus_x \Phi \mid \bigotimes_x \Phi \mid \mathrm{TC}^{N}_{x,y}\Phi$$

with $s \in \mathbf{S}$, $\varphi \in \mathsf{FO}$, $x, y \in \mathrm{Var}$ and $N \in \mathbb{N} \setminus \{0\}$.

Comparison with restricted wMSO:

- unrestricted use of $\bigoplus_x$ and $\otimes$, presence of $\mathrm{TC}_{x,y}$, absence of $\bigoplus_X$

Contribution

Theorem:

over searchable graphs: $\mathrm{wFOTC} \longrightarrow$ weighted pebble walking automata

over zonable graphs: weighted pebble walking automata $\longrightarrow \mathrm{wFOTC}$

Contribution

Theorem:

over searchable graphs: $\mathrm{wFOTC} \longrightarrow$ weighted pebble walking automata

over zonable graphs: weighted pebble walking automata $\longrightarrow \mathrm{wFOTC}$

$\Longrightarrow$ (un)decidability and complexity results over automata transfer to wFOTC

Translation of wFOTC in wPWA

Definition: Hypothesis: searchable graphs

- linear order $\leqslant$ on vertices with ι (initial vertex) as minimal element
- existence of a guide: walking automaton $\mathcal{A}_{\mathcal{G}}$ computing $\leqslant$

All previously classes of graphs are searchable

Translation of wFOTC in wPWA

Definition: Hypothesis: searchable graphs

- linear order $\leqslant$ on vertices with ι (initial vertex) as minimal element
- existence of a guide: walking automaton $\mathcal{A}_{\mathcal{G}}$ computing $\leqslant$

All previously classes of graphs are searchable

Inductive translation:

$\Phi \oplus \Psi$ disjoint union of automata

$\Phi \otimes \Psi$ $\longrightarrow \boxed{\mathcal{A}_\Phi} \xrightarrow{\text{reset to } \iota} \boxed{\mathcal{A}_\Psi} \longrightarrow$

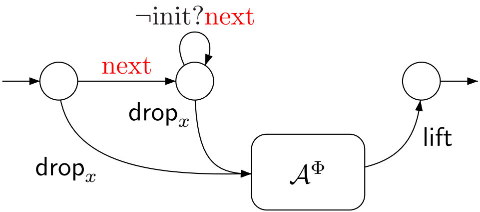
Translation of wFOTC in wPWA

Definition: Hypothesis: searchable graphs

- linear order $\leqslant$ on vertices with ι (initial vertex) as minimal element
- existence of a guide: walking automaton $\mathcal{A}_{\mathcal{G}}$ computing $\leqslant$

All previously classes of graphs are searchable

$\bigoplus_x \Phi$



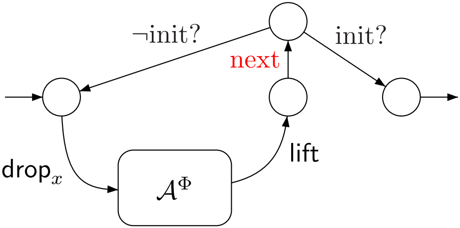
Translation of wFOTC in wPWA

Definition: Hypothesis: **searchable** graphs

- linear order $\leqslant$ on vertices with ι (initial vertex) as minimal element
- existence of a guide: walking automaton $\mathcal{A}_{\mathcal{G}}$ computing $\leqslant$

All previously classes of graphs are searchable

$\bigotimes_x \Phi$



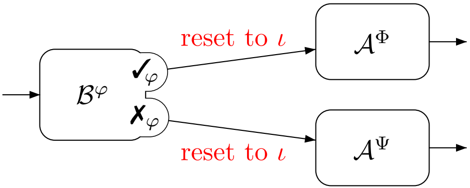
Translation of wFOTC in wPWA

Definition: Hypothesis: searchable graphs

- linear order $\leqslant$ on vertices with ι (initial vertex) as minimal element
- existence of a guide: walking automaton $\mathcal{A}_{\mathcal{G}}$ computing $\leqslant$

All previously classes of graphs are searchable

$\varphi \mathbin{?} \Phi : \Psi$



Boolean fragment: **linear size automata** (pebble and navigation)

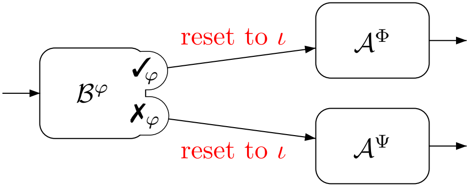
Translation of wFOTC in wPWA

Definition: Hypothesis: searchable graphs

- linear order $\leqslant$ on vertices with ι (initial vertex) as minimal element
- existence of a guide: walking automaton $\mathcal{A}_\mathcal{G}$ computing $\leqslant$

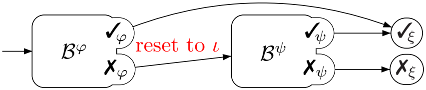
All previously classes of graphs are searchable

$\varphi \mathrel{?} \Phi : \Psi$

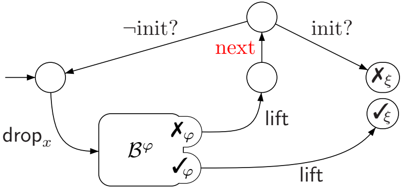


Boolean fragment: **linear size automata** (pebble and navigation)

disjunction $\xi = \varphi \vee \psi$



existential quantification $\xi = \exists x\ \varphi$



Translation of wFOTC in wPWA

Case of a formula $[\mathrm{TC}^{N}_{x,y}\Phi(x,y)]\underbrace{(x',y')}_{\text{fresh free variables}}$ with $\mathcal{A}$ a wPWA for Φ: construction of a wPWA $\mathcal{A}'$ with two more layers of pebbles that does the following

Translation of wFOTC in wPWA

Case of a formula $[\mathrm{TC}^{N}_{x,y}\Phi(x,y)]\underbrace{(x',y')}_{\text{fresh free variables}}$ with $\mathcal{A}$ a wPWA for Φ: construction of a wPWA $\mathcal{A}'$ with two more layers of pebbles that does the following

1. search free variable x', and drop pebble x
2. guess a sequence of moves of length $\leqslant N$, follow it, and drop pebble y (*then flush the sequence to save memory*)
3. reset to ι and simulate $\mathcal{A}$
4. search pebble y
5. guess sequence π of moves of length $\leqslant N$, follow it, check that it holds x
6. lift pebbles y and x (hence returning to the vertex of x)
7. follow π^R to reach back the vertex that held y, and drop pebble x
8. if y' is held by the current vertex, enter a final state
9. in every case, go back to step 2

Translation of wFOTC in wPWA

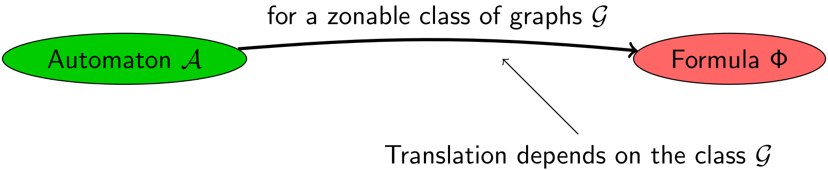
Case of a formula $[\mathrm{TC}^{N}_{x,y}\Phi(x,y)]\underbrace{(x',y')}_{\text{fresh free variables}}$ with $\mathcal{A}$ a wPWA for Φ: construction of a wPWA $\mathcal{A}'$ with two more layers of pebbles that does the following

1. search free variable x', and drop pebble x
2. guess a sequence π of moves of length $\leqslant N$, follow it, and drop pebble y (*then flush the sequence to save memory*)
    - test that π is minimal amongst all sequences going from x to y
3. reset to ι and simulate $\mathcal{A}$
4. search pebble y
5. guess sequence π of moves of length $\leqslant N$, follow it, check that it holds x
    - test that π is minimal amongst all sequences going q from y to x
6. lift pebbles y and x (hence returning to the vertex of x)
7. follow π^R to reach back the vertex that held y, and drop pebble x
8. if y' is held by the current vertex, enter a final state
9. in every case, go back to step 2

Translation of wPWA in wFOTC

Theorem:

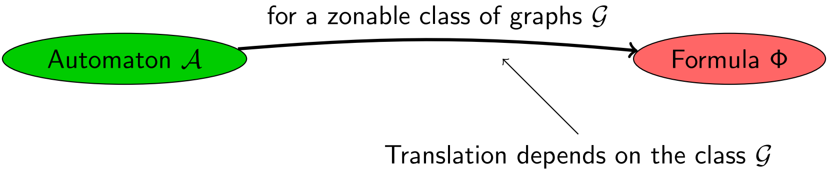
Let $\mathcal{G}$ be a **zonable** class of graphs. Then, for every wPWA $\mathcal{A}$, we can construct a formula Φ of wFOTC such that for every graph $G \in \mathcal{G}$, and valuation σ of free variables, $[\![\mathcal{A}]\!](G, \sigma) = [\![\Phi]\!](G, \sigma)$.



Translation of wPWA in wFOTC

Theorem:

Let $\mathcal{G}$ be a **zonable** class of graphs. Then, for every wPWA $\mathcal{A}$, we can construct a formula Φ of wFOTC such that for every graph $G \in \mathcal{G}$, and valuation σ of free variables, $[\![\mathcal{A}]\!](G,\sigma) = [\![\Phi]\!](G,\sigma)$.



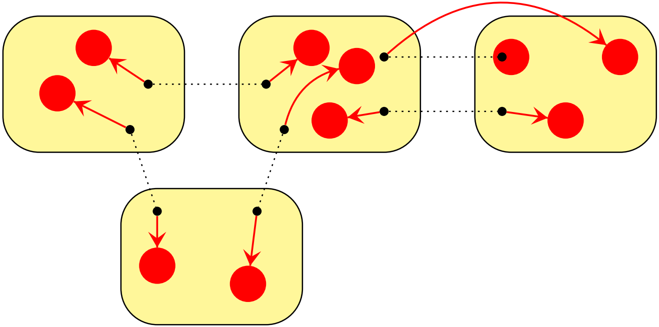
Proof in two steps:

- For the considered class of graphs, prove the **zonability**;
- **Generic** translation of automata into formulae for zonable class of graphs

Zonable classes of graphs

A zoning of a graph G with parameter N:

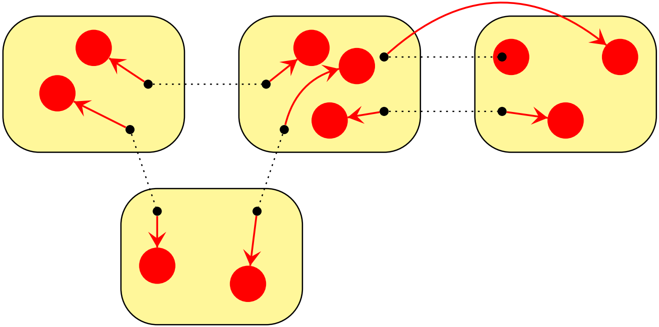
- an equivalence relation $\sim$, decomposing a graph into *zones* of diameter bounded by a constant M;
- set $\mathcal{W}$ of wires = (directed) edges relating different zones;
- an injective encoding function $enc\colon \mathcal{W} \times \{0, \dots, N-1\} \to V$



Zonable classes of graphs

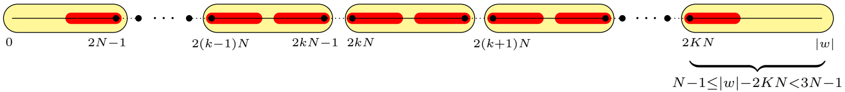
A zoning of a graph G with parameter N:

- an equivalence relation $\sim$, decomposing a graph into *zones* of diameter bounded by a constant M;
- set $\mathcal{W}$ of wires = (directed) edges relating different zones;
- an injective encoding function $enc\colon \mathcal{W} \times \{0, \dots, N-1\} \to V$

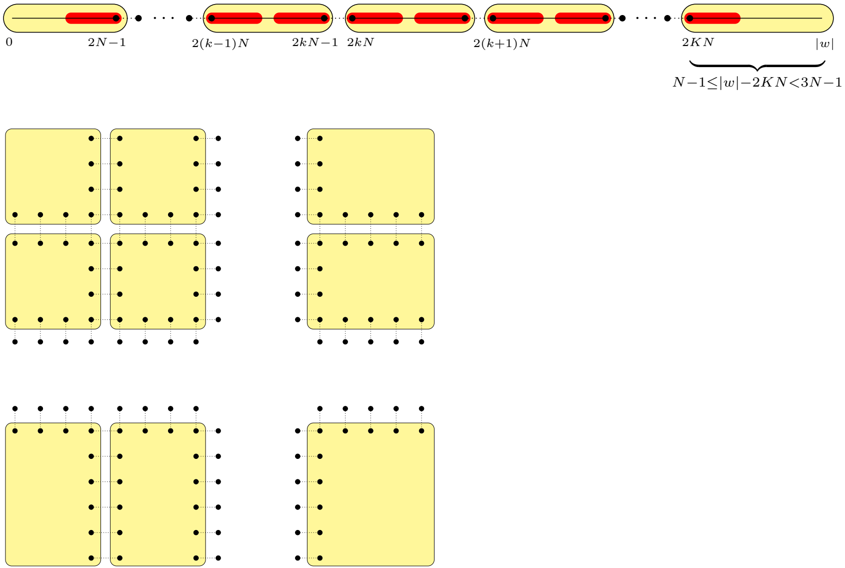


and $\sim$ and *enc* must be expressible by some formulae $\mathsf{zone}(z, z')$ and $\mathsf{enc}_n(z, z', x)$ (for $n \in \{0, \dots, N-1\}$) in wFOTC

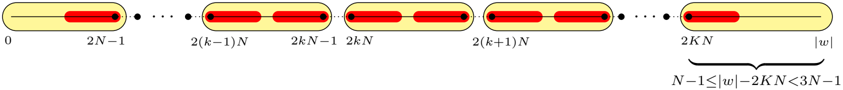
Examples

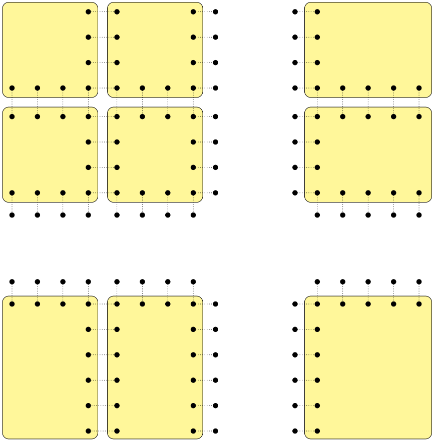


Examples



Examples

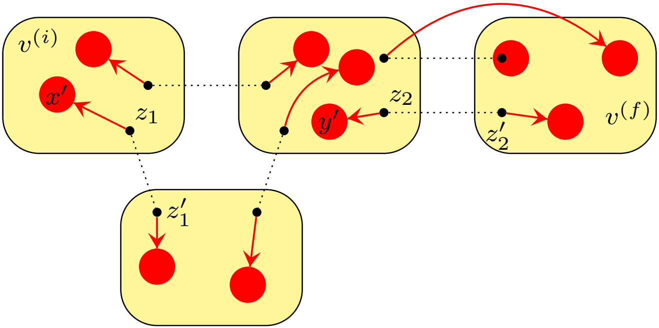




but also trees, nested words, Mazurkiewicz traces, rings...

Translation in a zonable class of graphs

- External (bounded) transitive closure jumping from zone to zone: state at the wires encoded using *enc*;
- Internal (bounded) transitive closures to compute the weights of the infinite set of runs restricted to a zone: computation by McNaughton-Yamada algorithm, state directly encoded in the formulae.



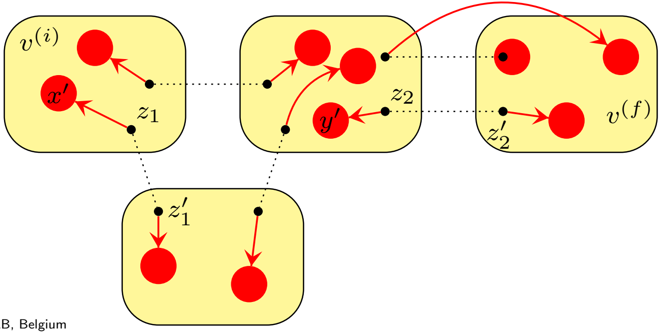
Translation in a zonable class of graphs

Weight of the runs from z_i in state q_i to z_f in state q_f:

$$\bigoplus_{x',y'}\Big[\bigoplus_{z_1,z_1'}\bigoplus_{q_1\in Q}\mathsf{enc}_{q_1}(z_1,z_1',x')\otimes\Phi_{q_i,q_1}(z_i,z_1)\Big]\otimes[\mathrm{TC}^{3M}_{y_1,y_2}\Psi](x',y')$$
$$\otimes\bigoplus_{z_2,z_2'}\bigoplus_{q_2,q_2'\in Q}\Big[\mathsf{enc}_{q_2}(z_2,z_2',y')\otimes\mathsf{tr}_{q_2,q_2'}(z_2,z_2')\otimes\Phi_{q_2',q_f}(z_2',z_f)\Big]$$

with $\Psi(y_1,y_2)$ the formula

$$\bigoplus_{\substack{z_1,z_1',\\ z_2,z_2'}}\bigoplus_{\substack{q_1,q_1',\\ q_2\in Q}}\Big[\mathsf{enc}_{q_1}(z_1,z_1',y_1)\otimes\mathsf{tr}_{q_1,q_1'}(z_1,z_1')\otimes\mathsf{enc}_{q_2}(z_2,z_2',y_2)\otimes\Phi_{q_1',q_2}(z_1',z_2)\Big]$$



Translation in a zonable class of graphs

Weight of the runs from z_i in state q_i to z_f in state q_f:

$$\bigoplus_{x',y'} \Big[\bigoplus_{z_1,z_1'} \bigoplus_{q_1\in Q} \mathsf{enc}_{q_1}(z_1,z_1',x') \otimes \Phi_{q_i,q_1}(z_i,z_1)\Big] \otimes [\mathrm{TC}^{3M}_{y_1,y_2}\Psi](x',y')$$
$$\otimes \bigoplus_{z_2,z_2'} \bigoplus_{q_2,q_2'\in Q} \Big[\mathsf{enc}_{q_2}(z_2,z_2',y') \otimes \mathsf{tr}_{q_2,q_2'}(z_2,z_2') \otimes \Phi_{q_2',q_f}(z_2',z_f)\Big]$$

with $\Psi(y_1,y_2)$ the formula

$$\bigoplus_{\substack{z_1,z_1',\\ z_2,z_2'}} \bigoplus_{\substack{q_1,q_1',\\ q_2\in Q}} \Big[\mathsf{enc}_{q_1}(z_1,z_1',y_1) \otimes \mathsf{tr}_{q_1,q_1'}(z_1,z_1') \otimes \mathsf{enc}_{q_2}(z_2,z_2',y_2) \otimes \Phi_{q_1',q_2}(z_1',z_2)\Big]$$

$\Phi_{q,q'}(x,x')$ formula computing the weight of the runs from x in q to x' in q', staying in the zone containing both x and x'

- built by McNaughton-Yamada algorithm, with cascade of **bounded** transitive closures (**since zones have bounded diameter**)

Conclusion and Perspectives

- Expressive equivalence between weighted pebble walking automata and weighted first-order logic with bounded transitive closure, over arbitrary complete semirings
- Additional reasonable requirements on the classes of graphs (searchable and zonable), met by usual examples of graphs (words, nested words, trees, grids, Mazurkiewicz traces, rings...)
- Interesting special case: a logic for **graph-to-word transducers** (non-commutative semiring of languages over an alphabet Σ)

Conclusion and Perspectives

- Expressive equivalence between weighted pebble walking automata and weighted first-order logic with bounded transitive closure, over arbitrary complete semirings
- Additional reasonable requirements on the classes of graphs (searchable and zonable), met by usual examples of graphs (words, nested words, trees, grids, Mazurkiewicz traces, rings...)
- Interesting special case: a logic for **graph-to-word transducers** (non-commutative semiring of languages over an alphabet Σ)

- Translation from automata to logic with less transitive closures? as in [Bollig, Gastin, Monmege, and Zeitoun, 2010] for words and the non-looping semantics
- Case of **strong pebbles** to deal with unbounded transitive closure?
- Extension to infinite structures?

Conclusion and Perspectives

- Expressive equivalence between weighted pebble walking automata and weighted first-order logic with bounded transitive closure, over arbitrary complete semirings
- Additional reasonable requirements on the classes of graphs (searchable and zonable), met by usual examples of graphs (words, nested words, trees, grids, Mazurkiewicz traces, rings...)
- Interesting special case: a logic for **graph-to-word transducers** (non-commutative semiring of languages over an alphabet Σ)

- Translation from automata to logic with less transitive closures? as in [Bollig, Gastin, Monmege, and Zeitoun, 2010] for words and the non-looping semantics
- Case of **strong pebbles** to deal with unbounded transitive closure?
- Extension to infinite structures?

Thank you!

References

Benedikt Bollig, Paul Gastin, Benjamin Monmege, and Marc Zeitoun. Pebble weighted automata and transitive closure logics. In *Proceedings of ICALP'10*, volume 6199 of *LNCS*, pages 587–598. Springer, 2010.

Manfred Droste and Paul Gastin. Weighted automata and weighted logics. EATCS Monographs in TCS, chapter 5, pages 175–211. Springer, 2009.

Manfred Droste and Heiko Vogler. Weighted tree automata and weighted logics. *Theoretical Computer Science*, 366(3):228–247, 2006.

Ina Fichtner. Weighted picture automata and weighted logics. *Theory of Computing Systems*, 48(1):48–78, 2011.

Christian Mathissen. Weighted logics for nested words and algebraic formal power series. *Logical Methods in Computer Science*, 6(1), 2010.

Benjamin Monmege. *Specification and Verification of Quantitative Properties: Expressions, Logics, and Automata*. Phd thesis, ENS de Cachan, 2013.