Logical Characterization of Weighted Pebble Automata Navigating over Graphs

Benjamin Monmege
Université Libre de Bruxelles, Belgium

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

7th International Workshop WATA 2014

Leipzig - May 5, 2014

To be presented at CSL-LICS 2014

Weighted Pebble Walking Automata

- Unusual mechanism
- Expressive power not fully clear

AIM: study expressive power in terms of other formalisms, e.g., of **logic**

Weighted Pebble Walking Automata

- Unusual mechanism
- Expressive power not fully clear

AIM: study expressive power in terms of other formalisms, e.g., of **logic**

Many such results for weighted automata: over words [Droste and Gastin, 2009], over trees [Droste and Vogler, 2006], over grids [Fichtner, 2011], over nested words [Mathissen, 2010]...

Weighted Pebble Walking Automata

- Unusual mechanism
- Expressive power not fully clear

AIM: study expressive power in terms of other formalisms, e.g., of **logic**

Many such results for weighted automata: over words [Droste and Gastin, 2009], over trees [Droste and Vogler, 2006], over grids [Fichtner, 2011], over nested words [Mathissen, 2010]...

Boolean setting [Engelfriet and Hoogeboom, 2007]

Pebble Walking Automata = FO + posTC

Weighted Pebble Walking Automata

- Unusual mechanism
- Expressive power not fully clear

AIM: study expressive power in terms of other formalisms, e.g., of **logic**

Many such results for weighted automata: over words [Droste and Gastin, 2009], over trees [Droste and Vogler, 2006], over grids [Fichtner, 2011], over nested words [Mathissen, 2010]...

Boolean setting [Engelfriet and Hoogeboom, 2007]

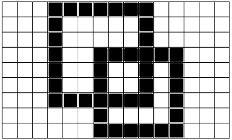
Pebble Walking Automata = FO + posTC

Extension in the quantitative setting

Theorem:

Weighted Pebble Walking Automata (wPWA) = wFOTC

Transitive Closure in Graphs

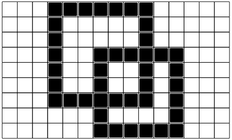


Binary predicate $R_{\nearrow}(x, y) = \exists z[R_{\rightarrow}(x, z) \wedge R_{\uparrow}(z, y)]$

Transitive Closure $\mathrm{TC}_{x,y} R_{\nearrow}(x, y)$

test if square (not doable in FO)

Transitive Closure in Graphs



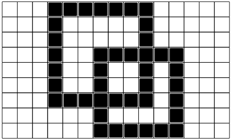
Binary predicate $R_{\nearrow}(x,y) = \exists z[R_{\rightarrow}(x,z) \wedge R_{\uparrow}(z,y)]$
Transitive Closure $\mathrm{TC}_{x,y} R_{\nearrow}(x,y)$
test if square (not doable in FO)

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

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

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 of Weighted Transitive Closure: complete semiring $(\mathbf{S}, +, \times, 0, 1)$

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

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!

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

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!

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

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

Definition: Logic wFOTC

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

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

Translation of wFOTC in wPWA

Inductive construction for **searchable** graphs

- For the wFO fragment, see Paul's talk
- 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

Inductive construction for **searchable** graphs

- For the wFO fragment, see Paul's talk
- 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. **goes back to the initial vertex** and simulate $\mathcal{A}$
  4. **search** pebble y
  5. guess a 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

Inductive construction for **searchable** graphs

- For the wFO fragment, see Paul's talk
- 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:

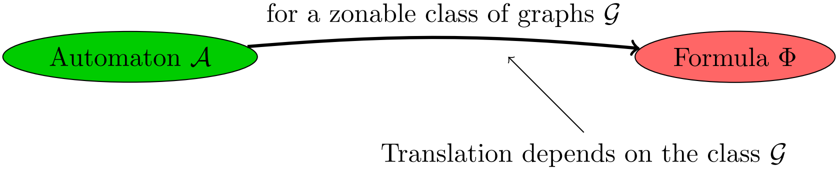
  construction of a wPWA $\mathcal{A}'$ with two more layers of pebbles that does the following

  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. **goes back to the initial vertex** and simulate $\mathcal{A}$
  4. **search** pebble y
  5. guess a sequence π of moves of length $\leqslant N$, follow it, check that it holds x
     - test that π is minimal amongst all sequences going 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:

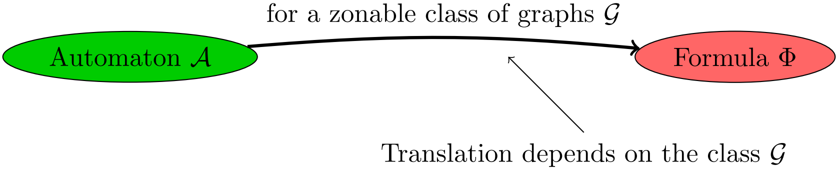
Let $\mathcal{G}$ be a **zonable** class of graphs. Then, for every wPWA $\mathcal{A}$, we can construct a formula Φ of wFOTC such that for every graph $G \in \mathcal{G}$, and valuation σ of free variables, $[\![\mathcal{A}]\!](G, \sigma) = [\![\Phi]\!](G, \sigma)$.



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:

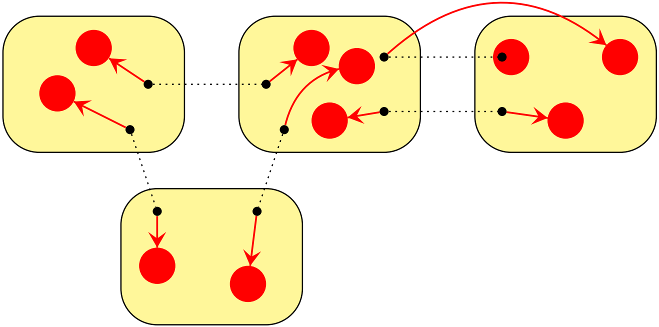
- For the considered class of graphs, prove the **zonability**;
- **Generic** translation of automata into formulae for zonable class of graphs

Example of zonable classes of graphs: words, trees, grids/pictures, nested words, Mazurkiewicz traces...

Zonable classes of graphs

A zoning of a graph G with parameter N:

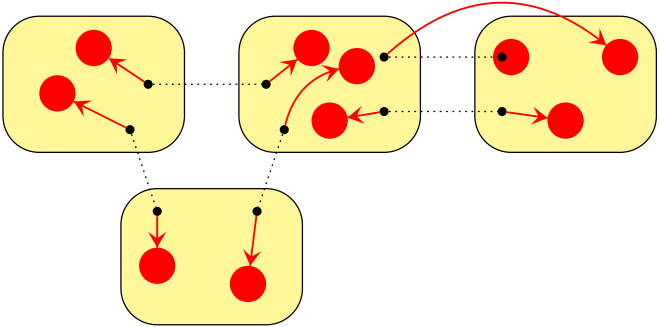
- an equivalence relation $\sim$, decomposing a graph into *zones* of diameter bounded by a constant M;
- set $\mathcal{W}$ of wires = (directed) edges relating different zones;
- an injective encoding function $enc\colon \mathcal{W} \times \{0, \ldots, N-1\} \to V$



Zonable classes of graphs

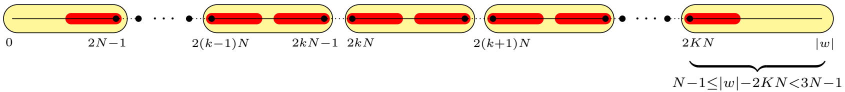
A zoning of a graph G with parameter N:

- 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 $enc\colon \mathcal{W} \times \{0, \dots, N-1\} \to V$

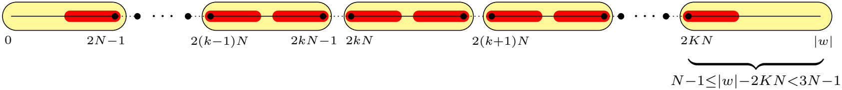


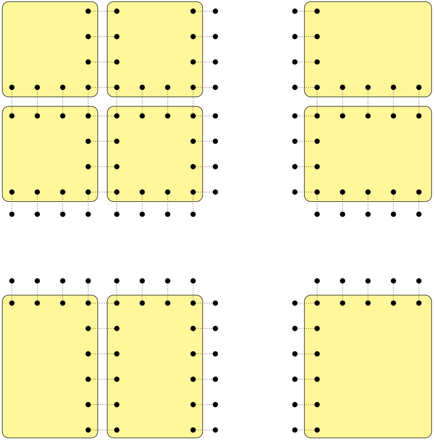
and $\sim$ and enc must be expressible by some formulae $\mathsf{zone}(z, z')$ and $\mathsf{enc}_n(z, z', x)$ (for $n \in \{0, \dots, N-1\}$) in wFOTC

Examples: words and grids



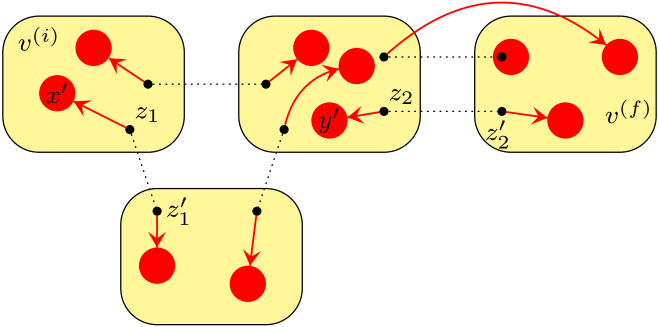
Examples: words and grids





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

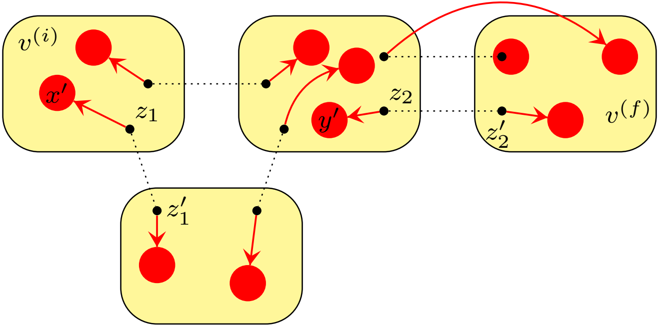
Weight of the runs from z_i in state q_i to z_f in state q_f:

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

$$\otimes \bigoplus_{z_2,z_2'} \bigoplus_{q_2,q_2' \in Q} \Big[\mathsf{enc}_{q_2}(z_2, z_2', y') \otimes \mathsf{tr}_{q_2,q_2'}(z_2, z_2') \otimes \Phi_{q_2',q_f}(z_2', z_f) \Big]$$

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 continuous 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...)
- Interesting special case: **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?

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.

Joost Engelfriet and Hendrik Jan Hoogeboom. Automata with nested pebbles capture first-order logic with transitive closure. *LMCS*, 3:1–27, 2007.

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.