Specification and Verification of Quantitative Properties: Expressions, Logics, and Automata

Benjamin MONMEGE
Ecole Normale Supérieure de Cachan
EDSP - LSV
Software Verification
Software Verification

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

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

TO BE VERIFIED
Software Verification

Property to be verified

Critical Software

• communication systems
• e-commerce
• health databases
• energy production
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?
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?
Formal Verification

Property to be verified

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

Is the property verified or not by the software?
Formal Verification

Property to be verified

Formal Model

ababcaabb

ababcaabb

Is the property verified or not by the model?

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

TO BE VERIFIED
Formal Verification

Property to be verified
Formal Specification

\[(a + b)^* c(a c)^+ \]

\[\forall x \forall y (x < y \Rightarrow \exists z (x < z < y))\]

\[FG (p U q)\]

Is the property verified or not by the model?

Formal Model

ababcaabb

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

ababcaabb

ababcaabb
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]
GOAL 1: describe and study versatile and expressive specification formalisms of quantitative properties, in a unified framework of graphs.
GOAL 1: describe and study versatile and expressive specification formalisms of quantitative properties, in a unified framework of graphs.

Efficient translations

Some kind of automata?

Model Checking / Evaluation
GOAL 1: describe and study versatile and expressive specification formalisms of quantitative properties, in a unified framework of graphs.
The Success Story of Automata

An Automata-Theoretic Approach to ...
The Success Story of Automata

An Automata-Theoretic Approach to ...

... Parsing Regular Expressions
Kleene’s theorem [Kleene 56]

1956
The Success Story of Automata

An Automata-Theoretic Approach to ...

... Parsing Regular Expressions
Kleene’s theorem [Kleene 56]

... Deciding MSO Over Finite Words
[Büchi 59, Elgot 61, Trakhtenbrot 61]

1956 1960
The Success Story of Automata

An Automata-Theoretic Approach to...

- Parsing Regular Expressions
  Kleene’s theorem [Kleene 56]
- Deciding MSO Over Finite Words
  [Büchi 59, Elgot 61, Trakhtenbrot 61]
- Model Checking a System Against LTL
  [Vardi&Wolper 86]

1956 1960 1986
The Success Story of Automata

An Automata-Theoretic Approach to ...

- Parsing Regular Expressions
  Kleene's theorem [Kleene 56]
- Deciding MSO Over Finite Words
  [Büchi 59, Elgot 61, Trakhtenbrot 61]
- Parsing Weighted Expressions
  Schützenberger's theorem [Schützenberger 61]
- Model Checking a System Against LTL
  [Vardi&Wolper 86]

1956 1960 1961 1986
The Success Story of Automata

An Automata-Theoretic Approach to...

- Parsing Regular Expressions
  - Kleene's theorem [Kleene 56]
- Deciding MSO Over Finite Words
  - Büchi 59, Elgot 61, Trakhtenbrot 61]
- Model Checking a System Against LTL
  - [Vardi&Wolper 86]


- Parsing Weighted Expressions
  - Schützenberger's theorem [Schützenberger 61]
- Modeling Probabilistic Properties
  - [Rabin 63]
The Success Story of Automata

An Automata-Theoretic Approach to ...

- Parsing Regular Expressions
  Kleene’s theorem [Kleene 56]

- Deciding MSO Over Finite Words
  [Büchi 59, Elgot 61, Trakhtenbrot 61]

- Model Checking a System Against LTL
  [Vardi&Wolper 86]


- Parsing Weighted Expressions
  Schützenberger’s theorem [Schützenberger 61]

- Deciding MSO Over Infinite Words

- Deciding MSO Over Finite Words

- Model Checking a System Against LTL

- Handling Weighted MSO Properties
  [Droste&Gastin 07]

- Modeling Probabilistic Properties
  [Rabin 63]
The Success Story of Automata

An Automata-Theoretic Approach to...

- Parsing Regular Expressions
  - Kleene's theorem [Kleene 56]
- Deciding MSO Over Finite Words
  - Büchi 59, Elgot 61, Trakhtenbrot 61]
- Model Checking a System Against LTL
  - Vardi&Wolper 86]


- Parsing
  - Schützenberger's theorem [Schützenberger 61]
- Modeling
  - Rabin 63]
- Handling
  - Droste&Gastin 07]

Weighted Expressions

Restricted Weighted Monadic Second Order Logic

Weighted Automata
The Success Story of Automata

An Automata-Theoretic Approach to ...

- Parsing Regular Expressions
  - Kleene’s theorem [Kleene 56]
- Deciding MSO Over Finite Words
  - Büchi 59, Elgot 61, Trakhtenbrot 61
- Model Checking a System Against LTL
  - Vardi&Wolper 86
- Parsing
  - Schützenberger’s theorem [Schützenberger 61]
- Modeling
  - Rabin 63
- Handling
  - Droste&Gastin 07


Weighted Expressions

Weighted Automata

Restricted Weighted Monadic Second Order Logic

cannot express certain natural properties
GOAL 1: describe and study versatile and expressive specification formalisms of quantitative properties, in a unified framework of graphs.
GOAL 1: describe and study versatile and expressive specification formalisms of quantitative properties, in a unified framework of graphs.

GOAL 2: introduce a more expressive class of weighted automata, able to compute a broader class of weighted properties over general classes of graphs.
GOAL 1: describe and study versatile and expressive specification formalisms of quantitative properties, in a unified framework of graphs.

GOAL 2: introduce a more expressive class of weighted automata, able to compute a broader class of weighted properties over general classes of graphs.

GOAL 3: give efficient procedures to translate specifications in automata, and evaluate automata as efficiently as possible.

High-Level Specification Languages
- First-Order Logic
- Temporal/Navigating Logics
- Hybrid Expressions

Low-Level Machinery
- Pebble Navigating Automata

Model
- Words, Trees, Graphs...

Model Checking / Evaluation
Paul Gastin and Benjamin Monmege, CIAA 2012.
Benedikt Bollig, Paul Gastin, and Benjamin Monmege, FoSSaCS 2013.
GOAL 1: describe and study versatile and expressive specification formalisms of quantitative properties, in a unified framework of graphs.

GOAL 2: introduce a more expressive class of weighted automata, able to compute a broader class of weighted properties over general classes of graphs.

GOAL 3: give efficient procedures to translate specifications in automata, and evaluate automata.
Logical Specifications: Query Examples
Logical Specifications: Query Examples

Is there a line of green pixels?
Logical Specifications: Query Examples

Is there a line of green pixels?

How many lines of green pixels are there?
Logical Specifications: Query Examples

Is there a line of green pixels?

How many lines of green pixels are there?

What is the size of the picture?
Logical Specifications: Query Examples

Is there a line of green pixels?

How many lines of green pixels are there?

What is the size of the picture?

What is the color with maximum number of pixels?
Logical Specifications: Query Examples

Is there a line of green pixels?

How many lines of green pixels are there?

What is the size of the picture?

What is the color with maximum number of pixels?

What is the size of the biggest monochromatic rectangle?
Modeling a picture as a graph
Modeling a picture as a graph
Modeling a picture as a graph
\[ G = (V, (E_d)_{d \in D}, \lambda) \]
\[ G = (V, (E_d)_{d \in D}, \lambda) \]

- \( V \): set of vertices
- \( \lambda \): labels of vertices
\[ G = (V, (E_d)_{d \in D}, \lambda) \]

- \( V \): set of vertices
- \( \lambda \): labels of vertices
- \( D \): set of directions

\[ D = \{\rightarrow, \downarrow\} \cup \{\leftarrow, \uparrow\} \]
\[ G = (V, (E_d)_{d \in D}, \lambda) \]

- \( V \): set of vertices
- \( \lambda \): labels of vertices
- \( D \): set of directions
- \( E_d \): set of \( d \)-edges

\[ D = \{ \rightarrow, \downarrow \} \cup \{ \leftarrow, \uparrow \} \]
A deterministic (hence bounded degree) graph $G = (V, (E_d)_{d \in D}, \lambda)$ is defined as follows:

- **V**: set of vertices
- **\lambda**: labels of vertices
- **D**: set of directions
- **E_d**: set of $d$-edges

The set of directions $D$ is defined as $D = \{\rightarrow, \downarrow\} \cup \{\leftarrow, \uparrow\}$.
Examples

Words

Ranked Trees

Nested Words (Unranked Trees)

Mazurkiewicz Traces

Pictures

Graphs of bounded degree

Labyrinths

...
Logical Specifications: Query Examples

Is there a line of green pixels?

How many lines of green pixels are there?

What is the size of the picture?

What is the color with maximum number of pixels?

What is the size of the biggest monochromatic rectangle?
Logical Specifications: Query Examples

Is there a line of green pixels?
\[ \exists x \forall y \left[ (R^\rightarrow(x, y) \lor R^\rightarrow(y, x)) \Rightarrow P_\square(y) \right] \]

How many lines of green pixels are there?

What is the size of the picture?

What is the color with maximum number of pixels?

What is the size of the biggest monochromatic rectangle?
Logical Specifications: Query Examples

Is there a line of green pixels?

$$\exists x \forall y \left[ (R^*_\rightarrow(x, y) \lor R^*_\rightarrow(y, x)) \Rightarrow P_{\bullet}(y) \right]$$

How many lines of green pixels are there?

What is the size of the picture?

Boolean fragment: first-order logic

$$\varphi ::= \top \mid (x = y) \mid \text{init}(x) \mid \text{final}(x) \mid P_a(x) \mid R_d(x, y) \mid R^*_d(x, y) \mid$$

$$\neg \varphi \mid \varphi \lor \varphi \mid \varphi \land \varphi \mid \exists x \ \varphi \mid \forall x \ \varphi$$

$$G, \sigma \models P_a(x) \iff \lambda(\sigma(x)) = a$$

$$G, \sigma \models R_d(x, y) \iff (\sigma(x), \sigma(y)) \in E_d$$

$$G, \sigma \models R^*_d(x, y) \iff \text{there is a } d\text{-path from } \sigma(x) \text{ to } \sigma(y)$$
Logical Specifications: Query Examples

Is there a line of green pixels?

\[ \exists x \forall y \left( R_\rightarrow(x, y) \lor R_\rightarrow(y, x) \right) \Rightarrow P(y) \]

How many lines of green pixels are there?

What is the size of the picture?

What is the color with maximum number of pixels?

What is the size of the biggest monochromatic rectangle?
Logical Specifications: Query Examples

Is there a line of green pixels?

\[ \exists x \forall y \left[ (R^*_{\rightarrow} (x, y) \lor R^*_{\rightarrow} (y, x)) \implies P_\square (y) \right] \]

How many lines of green pixels are there?

\[ \sum_{x} \forall y \left[ (R^*_{\rightarrow} (x, y) \lor R^*_{\rightarrow} (y, x)) \implies (R^*_{\rightarrow} (x, y) \land P_\square (y)) \right] \]

What is the size of the picture?

What is the color with maximum number of pixels?

What is the size of the biggest monochromatic rectangle?
Logical Specifications: Query Examples

Is there a line of green pixels?

$$\exists x \forall y \left[ (R^{\rightarrow} (x, y) \lor R^{\rightarrow} (y, x)) \Rightarrow P_e(y) \right]$$

How many lines of green pixels are there?

$$\sum_x \forall y \left[ (R^{\rightarrow} (x, y) \lor R^{\rightarrow} (y, x)) \Rightarrow (R^{\rightarrow} (x, y) \land P_e(y)) \right]$$

What is the size of the picture?

$$\left( \sum_x \neg \exists y R^{\rightarrow} (y, x) \right) \times \left( \sum_x \neg \exists y R^{\downarrow} (y, x) \right)$$

What is the color with maximum number of pixels?

What is the size of the biggest monochromatic rectangle?
Logical Specifications: Query Examples

- Is there a line of green pixels?
  \[ \exists x \forall y \left[ (R^*_{\rightarrow} (x, y) \lor R^*_{\rightarrow} (y, x)) \Rightarrow P_{\square} (y) \right] \]

- How many lines of green pixels are there?
  \[ \sum_x \forall y \left[ (R^*_{\rightarrow} (x, y) \lor R^*_{\rightarrow} (y, x)) \Rightarrow (R^*_{\rightarrow} (x, y) \land P_{\square} (y)) \right] \]

- What is the size of the picture?
  \[ \left( \sum_x \neg \exists y R_{\rightarrow} (y, x) \right) \times \left( \sum_x \neg \exists y R_{\downarrow} (y, x) \right) \]

- What is the color with maximum number of pixels?

- What is the size of the biggest monochromatic rectangle?
Logical Specifications: Query Examples

Is there a line of green pixels?
\[ \exists x \forall y \left[ (R^\rightarrow (x, y) \lor R^\rightarrow (y, x)) \Rightarrow P(y) \right] \]

How many lines of green pixels are there?
\[ \sum_x \forall y \left[ (R^\rightarrow (x, y) \lor R^\rightarrow (y, x)) \Rightarrow (R^\rightarrow (x, y) \land P(y)) \right] \]

What is the size of the picture?
\[ \left( \sum_x \neg \exists y \ R(y, x) \right) \times \left( \sum_x \exists y \ R(y, x) \right) \]

What is the color with maximum number of pixels?
\[ (\mathbb{N} \cup \{-\infty\}, \max, +, -\infty, 0) \]

What is the size of the biggest monochromatic rectangle?
Logical Specifications: Query Examples

Is there a line of green pixels?
$$\exists x \forall y \left[ (R^\to(x, y) \lor R^\to(y, x)) \Rightarrow P(x) \right]$$

How many lines of green pixels are there?
$$\sum \forall \left[ (R^\to(x, y) \lor R^\to(y, x)) \Rightarrow (R^\to(x, y) \land P(y)) \right]$$

What is the size of the picture?
$$\left( \sum \neg \exists y R^\to(y, x) \right) \times \left( \sum \neg \exists y R(y, x) \right)$$

What is the color with maximum number of pixels?
$$\max \left( \sum P(x), \sum P(x), \ldots, \sum P(x) \right)$$

What is the size of the biggest monochromatic rectangle?
Logical Specifications: Query Examples

Is there a line of green pixels?
$$\exists x \forall y \left[ (R_\rightarrow(x, y) \lor R_\rightarrow(y, x)) \Rightarrow P(y) \right]$$

How many lines of green pixels are there?
$$\sum_x \forall y \left[ (R_\rightarrow(x, y) \lor R_\rightarrow(y, x)) \Rightarrow (R_\rightarrow(x, y) \land P(y)) \right]$$

What is the size of the picture?
$$\left( \sum_x \neg \exists y \ R_\rightarrow(y, x) \right) \times \left( \sum_x \neg \exists y \ R_\rightarrow(y, x) \right)$$

What is the color with maximum number of pixels?
$$\max \left( \sum_x P^\square(x), \sum_x P^\square(x), \ldots, \sum_x P^\square(x) \right)$$

What is the size of the biggest monochromatic rectangle?
$$\max_{x,y} \left[ \varphi_{\text{mono}}(x, y) + \left( \sum_z \varphi_{\text{rect}}(x, y, z) \right) \right]$$
Logical Specifications: Query Examples

Is there a line of green pixels?

$$\exists x \forall y \left[ (R_\rightarrow(x, y) \lor R_\rightarrow(y, x)) \Rightarrow P_\square(y) \right]$$

How many lines of green pixels are there?

$$\sum_x \forall y \left[ (R_\rightarrow(x, y) \lor R_\rightarrow(y, x)) \Rightarrow (R_\rightarrow(x, y) \land P_\square(y)) \right]$$

What is the size of the picture?

$$\left( \sum_x \neg \exists y R_\rightarrow(y, x) \right) \times \left( \sum_x \neg \exists y R_\downarrow(y, x) \right)$$

What is the color with maximum number of pixels?

$$\max \left( \sum_x P_\square(x), \sum_x P_\square(x), \ldots, \sum_x P_\square(x) \right)$$

What is the size of the biggest monochromatic rectangle?

$$\max_{x,y} \left[ \varphi_{\text{mono}}(x, y) + \left( \sum_z \varphi_{\text{rect}}(x, y, z)?1:0 \right) \right]$$
Logical Specifications: Query Examples

Is there a line of green pixels?

$$\exists x \forall y \left[ (R_{\rightarrow}(x, y) \lor R_{\rightarrow}(y, x)) \Rightarrow P_{\downarrow}(y) \right]$$

How many lines of green pixels are there?

$$\sum_{x} \forall y \left[ (R_{\rightarrow}(x, y) \lor R_{\rightarrow}(y, x)) \Rightarrow (R_{\rightarrow}(x, y) \land P_{\downarrow}(y)) \right]$$

What is the size of the picture?

$$\left( \sum_{x} \neg \exists y \ R_{\rightarrow}(y, x) \right) \times \left( \sum_{x} \neg \exists y \ R_{\downarrow}(y, x) \right)$$

What is the color with maximum number of pixels?

$$\max \left( \sum_{x} P_{\downarrow}(x), \sum_{x} P_{\square}(x), \ldots, \sum_{x} P_{\triangle}(x) \right)$$

What is the size of the biggest monochromatic rectangle?

$$\max_{x, y} \left[ \varphi_{\text{mono}}(x, y) + \left( \sum_{z} \varphi_{\text{rect}}(x, y, z) \cdot 1 : 0 \right) \right]$$
Weighted Logic

**Boolean fragment: first order logic**

\[ \varphi ::= \top \mid (x = y) \mid \text{init}(x) \mid \text{final}(x) \mid P_a(x) \mid R_d(x, y) \mid R_d^*(x, y) \mid \neg \varphi \mid \varphi \lor \varphi \mid \varphi \land \varphi \mid \exists x \varphi \mid \forall x \varphi \]

**Weighted fragment: weighted first order logic**

\[ \Phi ::= s \mid \varphi \mid \Phi \oplus \Phi \mid \Phi \otimes \Phi \mid \bigoplus x \Phi \mid \bigotimes x \Phi \]

*generic notations for additive and multiplicative operations*
Weight Domains: Semirings

\((\mathbb{S}, +, \times, 0, 1)\)

- associative and commutative, with neutral element 0
- zero of the multiplicative operation
- associative, with neutral element 1, distributive over addition
Weight Domains: Semirings

\((\mathbb{S}, +, \times, 0, 1)\)

- associative and commutative, with neutral element 0
- associative, with neutral element 1, distributive over addition
- zero of the multiplicative operation

\((\{0, 1\}, \lor, \land, 0, 1)\)
Weight Domains: Semirings

\[(\mathbb{S}, +, \times, 0, 1)\]

- associative and commutative, with neutral element 0
- zero of the multiplicative operation
- associative, with neutral element 1, distributive over addition

\[(\mathbb{R}, +, \times, 0, 1)\]
\[(\mathbb{Q}, +, \times, 0, 1)\]
\[(\mathbb{Z}, +, \times, 0, 1)\]
\[(\mathbb{N}, +, \times, 0, 1)\]
\[(\{0, 1\}, \lor, \land, 0, 1)\]
Weight Domains: Semirings

\[(S, +, \times, 0, 1)\]

- associative and commutative, with neutral element 0
- associative, with neutral element 1, distributive over addition

\[(\mathbb{R}, +, \times, 0, 1)\]
\[(\mathbb{Q}, +, \times, 0, 1)\]
\[(\mathbb{Z}, +, \times, 0, 1)\]
\[(\mathbb{N}, +, \times, 0, 1)\]

\[(\{0, 1\}, \lor, \land, 0, 1)\]

\[(\mathbb{R} \cup \{+\infty\}, \min, +, +\infty, 0)\]
\[(\mathbb{R} \cup \{-\infty\}, \max, +, -\infty, 0)\]
Weight Domains: Semirings

\((\mathbb{S}, +, \times, 0, 1)\)

- **associative and commutative**, with neutral element 0
- **associative**, with neutral element 1,
  distributive over addition

- \((\mathbb{R}, +, \times, 0, 1)\)
- \((\mathbb{Q}, +, \times, 0, 1)\)
- \((\mathbb{Z}, +, \times, 0, 1)\)
- \((\mathbb{N}, +, \times, 0, 1)\)
- \(\mathcal{P}(A^*)\)

- \((\mathbb{R} \cup \{+\infty\}, \min, +, +\infty, 0)\)
- \((\mathbb{R} \cup \{-\infty\}, \max, +, -\infty, 0)\)
Weight Domains: Semirings

\((\mathbb{S}, +, \times, 0, 1)\)

- associative and commutative, with neutral element 0
- associative, with neutral element 1, distributive over addition

\((\mathbb{R}, +, \times, 0, 1)\)
\((\mathbb{Q}, +, \times, 0, 1)\)
\((\mathbb{Z}, +, \times, 0, 1)\)
\((\mathbb{N}, +, \times, 0, 1)\)

\((\mathcal{P}(A^*), \cup, \cdot, \emptyset, \{\varepsilon\})\)

- zero of the multiplicative operation

\((\{0, 1\}, \lor, \land, 0, 1)\)
\(([0, 1], \max, \min, 0, 1)\)
\((\mathbb{R} \cup \{-\infty, +\infty\}, \max, \min, -\infty, +\infty)\)

\((\mathbb{R} \cup \{+\infty\}, \min, +, +\infty, 0)\)
\((\mathbb{R} \cup \{-\infty\}, \max, +, -\infty, 0)\)
Weighted Logics: Semantics

\[ [s](G, \sigma) = s \]

\[ [\varphi](G, \sigma) = \begin{cases} 1 & \text{if } G, \sigma \models \varphi \\ 0 & \text{otherwise} \end{cases} \]

\[ [\Phi_1 \oplus \Phi_2](G, \sigma) = [\Phi_1](G, \sigma) + [\Phi_2](G, \sigma) \]

\[ [\Phi_1 \otimes \Phi_2](G, \sigma) = [\Phi_1](G, \sigma) \times [\Phi_2](G, \sigma) \]

\[ [\bigoplus_x \Phi](G, \sigma) = \sum_{v \in V} [\Phi](G, \sigma[x \mapsto v]) \]

\[ [\bigotimes_x \Phi](G, \sigma) = \prod_{v \in V} [\Phi](G, \sigma[x \mapsto v]) \]
GOAL 1: describe and study versatile and expressive specification formalisms of quantitative properties, in a unified framework of graphs.

GOAL 2: introduce a more expressive class of weighted automata, able to compute a broader class of weighted properties over general classes of graphs.

GOAL 3: give efficient procedures to translate specifications in automata, and evaluate automata. As efficient as possible.
GOAL 1: describe and study versatile and expressive specification formalisms of quantitative properties, in a unified framework of graphs.

GOAL 2: introduce a more expressive class of weighted automata, able to compute a broader class of weighted properties over general classes of graphs.

GOAL 3: give efficient procedures to translate specifications in automata, and evaluate automata.
Pebble Weighted Automata: An Example

\[ \gamma = \sum_c (+c)c ? \]

\[(\mathbb{Z} \cup \{-\infty\}, \max, +, -\infty, 0)\]

Pebble automata over words and trees: [Globerman&Harel 96], [Engelfriet&Hoogeboom 99]
Pebble Weighted Automata: An Example

\( \gamma = \sum_c (c+e)c? \)

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

Pebble automata over words and trees: [Globerman&Harel 96], [Engelfriet&Hoogeboom 99]
Pebble Weighted Automata: An Example

\[ (\mathbb{Z} \cup \{-\infty\}, \max, +, -\infty, 0) \]

\[ \gamma = \sum_e (\pm e) e? \]

Pebble automata over words and trees: [Globerman&Harel 96], [Engelfriet&Hoogeboom 99]
Pebble Weighted Automata: An Example

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

\[ \gamma = \sum_c (\text{color})c \]

\begin{align*}
\text{drop}_x & \quad \rightarrow + \downarrow \\
\rightarrow + \downarrow & \quad \rightarrow + \downarrow \\
\text{drop}_y & \\
\rightarrow + \downarrow & \quad \rightarrow + \downarrow \\
\end{align*}

\begin{align*}
\text{lift} & \\
\rightarrow + \downarrow & \quad \rightarrow + \downarrow \\
\text{lift} & \\
\end{align*}

Pebble automata over words and trees: [Globerman&Harel 96], [Engelfriet&Hoogeboom 99]
Pebble Weighted Automata: An Example

\[ (\mathbb{Z} \cup \{-\infty\}, \max, +, -\infty, 0) \]

\[ + 255 \]

\[ \gamma = \sum_e (e + c)e? \]

Pebble automata over words and trees: [Globerman&Harel 96], [Engelfriet&Hoogeboom 99]
Pebble Weighted Automata: An Example

\( \gamma = \sum_{c} (\pm c) e \cdot ? \)

\((Z \cup \{ -\infty \}, \max, +, -\infty, 0) + 255 - 0\)

Pebble automata over words and trees: [Globerman&Harel 96], [Engelfriet&Hoogeboom 99]
Pebble Weighted Automata: An Example

\[ \mathbb{Z} \cup \{-\infty\}, \max, +, -\infty, 0 \]

\[ + 255 - 0 \]

\[ \gamma = \sum c (e) e ? \]

Pebble automata over words and trees: [Globerman&Harel 96], [Engelfriet&Hoogeboom 99]
Pebble Weighted Automata: An Example

\[(\mathbb{Z} \cup \{-\infty\}, \text{max}, +, -\infty, 0)\]

\[+ 255 - 0\]

\[\gamma = \sum_c (c + c) c?\]

Pebble automata over words and trees: [Globerman&Harel 96], [Engelfriet&Hoogeboom 99]
Pebble Weighted Automata: An Example

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

\[ + 255 - 0 \]

\( \gamma = \sum_e ( + e ) e ? \)

Pebble automata over words and trees: [Globerman&Harel 96], [Engelfriet&Hoogeboom 99]
Pebble Weighted Automata: An Example

\[(\mathbb{Z} \cup \{-\infty\}, \text{max}, +, -\infty, 0)\]

\[+ 255 - 0\]

---

\[\gamma = \sum e \cdot (x \vee y)\?

---

Pebble automata over words and trees: [Globerman&Harel 96], [Engelfriet&Hoogeboom 99]
Pebble Weighted Automata: An Example

\[ \mathbb{Z} \cup \{-\infty\}, \max, +, -\infty, 0 \]

\[ + 255 - 0 \]
Pebble Weighted Automata: An Example

\[ (\mathbb{Z} \cup \{-\infty\}, \max, +, -\infty, 0) \]

\[ + 255 - 0 \]

Pebble automata over words and trees: [Globerman & Harel 96], [Engelfriet & Hoogeboom 99]

\[ \gamma = \sum_e (c+e) e? \]
Pebble Weighted Automata: An Example

\[ (\mathbb{Z} \cup \{-\infty\}, \text{max}, +, -\infty, 0) \]

\[ + 255 - 0 \]

\[ \gamma = \sum_c (c^e) e? \]

\[ \text{drop}_x \]

\[ \text{drop}_y \]

\[ (x? \land ?) \rightarrow \]

\[ ? \rightarrow \]

\[ ? \downarrow \]

\[ \text{lift} \]

\[ \text{lift} \]

\[ y? \]

Pebble automata over words and trees: [Globerman&Harel 96], [Engelfriet&Hoogeboom 99]
Pebble Weighted Automata: An Example

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

\[ + 255 - 0 \]

\[ \gamma = \sum_c (c) e ? \]

\[ \text{drop}_x \]

\[ \text{drop}_y \]

\[ \text{lift} \]

\[ \text{lift} \]

\[ x ? \land ? \rightarrow \]

\[ y ? \rightarrow \]

\[ ? \leftarrow \]

\[ ? \uparrow \]

\[ ? \rightarrow \]

\[ ? \downarrow \]

Pebble automata over words and trees: [Globerman&Harel 96], [Engelfriet&Hoogeboom 99]
Pebble Weighted Automata: An Example

\[ \mathbb{Z} \cup \{ -\infty \}, \max, +, -\infty, 0 \]

\[ + 255 - 0 \]

Weight of the run: 255

\[ \gamma = \sum \left\{ c \right\} c \]

Pebble automata over words and trees: [Globerman&Harel 96], [Engelfriet&Hoogeboom 99]
Pebble Weighted Automata: An Example

Non determinism

\[ \gamma = \sum_c (+c)c? \]

\[ \gamma = \sum_c (+c)c? \]

Non determinism resolved by max

Weight of the run: 255

\[ (\mathbb{Z} \cup \{-\infty\}, \max, +, -\infty, 0) \]

+ 255 - 0

Pebble automata over words and trees: [Globerman&Harel 96], [Engelfriet&Hoogeboom 99]

\[ Xc(\pm c)c? \]
Pebble Weighted Automata: An Example

\[ (\mathbb{Z} \cup \{-\infty\}, \text{max}, +, -\infty, 0) \]

\[ +255 - 0 \]

\[ \gamma = \sum_{c} (+c) e? \]

\[ \text{Max of the weights of the runs: biggest contrast in a green rectangle} \]

\[ \text{Non determinism resolved by max} \]
Pebble Weighted Automata

\[ A = (Q, A, D, I, \Delta, F) \]

finite set of states
alphabet
set of directions
matrix of transition
vector of initial weights
vector of final weights

transitions: linear combination of \((\text{test}, \text{action})\)

boolean combination of \(T, a?, x?, d?, \text{init}?, \text{final}?)\)

move \(d\), drop a pebble, lift the last dropped pebble
Semantics of Pebble Weighted Automata

Configuration of $A$: $(G, q, \pi, v)$

- state
- stack of dropped pebbles
- current vertex
Semantics of Pebble Weighted Automata

Configuration of $\mathcal{A}$: $(G, q, \pi, v)$

- state
- stack of dropped pebbles
- current vertex

Run over a graph $G$: finite sequence of configurations

Weight of a run: multiplication of the weights of the transitions and the initial and final weights

Semantics $[\mathcal{A}](G)$: sum of the weights of the runs
Semantics of Pebble Weighted Automata

Configuration of $\mathcal{A}$: $(G, q, \pi, v)$

- State
- Stack of dropped pebbles
- Current vertex

Run over a graph $G$: finite sequence of configurations

Weight of a run: multiplication of the weights of the transitions and the initial and final weights

Semantics $\semantics{\mathcal{A}}(G)$: sum of the weights of the runs

Navigation possibly leads to infinitely many runs!

Restrict to continuous semirings
Weight Domains: **Continuous Semirings**

\[(\mathbb{S}, +, \times, 0, 1)\]

*every infinite sum exists and is the limit of finite approximate sums, keeping good properties of usual semiring* 

\[
\begin{align*}
(\mathbb{R}, +, \times, 0, 1) \\
(\mathbb{Q}, +, \times, 0, 1) \\
(\mathbb{Z}, +, \times, 0, 1) \\
(\mathbb{N}, +, \times, 0, 1) \\
(\mathbb{R} \cup \{0, 1\}, \vee, \wedge, 0, 1) \\
([0, 1], \max, \min, 0, 1) \\
(\mathbb{R} \cup \{-\infty, +\infty\}, \max, \min, -\infty, +\infty) \\
\mathbb{P}(A^*) \cup \{0, 1\} \\
(\mathbb{R} \cup \{+\infty\}, \min, +, +\infty, 0) \\
(\mathbb{R} \cup \{-\infty\}, \max, +, -\infty, 0)
\end{align*}
\]
Weight Domains: **Continuous Semirings**

\[(\mathbb{S}, +, \times, 0, 1)\]

*every infinite sum exists and is the limit of finite approximate sums, keeping good properties of usual semiring*
Weight Domains: **Continuous Semirings**

\[(\mathbb{S}, +, \times, 0, 1)\]

Every infinite sum exists and is the limit of finite approximate sums, keeping good properties of usual semiring

\[
\begin{align*}
(\mathbb{R}, +, \times, 0, 1) \\
(\mathbb{Q}, +, \times, 0, 1) \\
(\mathbb{Z}, +, \times, 0, 1) \\
(\mathbb{N}, +, \times, 0, 1)
\end{align*}
\]

\[
\begin{align*}
(\mathbb{R}^+ \cup \{+\infty\}, +, \times, 0, 1) \\
(\mathbb{N} \cup \{+\infty\}, +, \times, 0, 1) \\
(\mathcal{P}(A^*), \cup, \cdot, \emptyset, \{\varepsilon\})
\end{align*}
\]

\[
\begin{align*}
(\{0, 1\}, \lor, \land, 0, 1) \\
([0, 1], \max, \min, 0, 1) \\
(\mathbb{R} \cup \{-\infty, +\infty\}, \max, \min, -\infty, +\infty)
\end{align*}
\]

\[
\begin{align*}
(\mathbb{R} \cup \{-\infty, +\infty\}, \min, +, +\infty, 0) \\
(\mathbb{R} \cup \{-\infty\}, \max, +, -\infty, 0)
\end{align*}
\]
Theorem: Consider a searchable class of graph. Every Weighted First-Order formula can then be translated into a Pebble Weighted Automaton equivalent over this class of graphs.

WFO \rightarrow PWA

Over words: Benedikt Bollig, Paul Gastin, Benjamin Monmege, and Marc Zeitoun, ICALP 2010.
Over nested words: Benedikt Bollig, Paul Gastin, and Benjamin Monmege, FoSSaCS 2013.
Translation from Logics to Automata

**Theorem:** Consider a *searchable* class of graph. Every Weighted First-Order formula can then be translated into a Pebble Weighted Automaton equivalent over this class of graphs.

WFO \(\rightarrow\) PWA

Which complexity?

Over words: Benedikt Bollig, Paul Gastin, Benjamin Monmege, and Marc Zeitoun, ICALP 2010.
Over nested words: Benedikt Bollig, Paul Gastin, and Benjamin Monmege, FoSSaCS 2013.
Translation from Logics to Automata

\[ \bigoplus_{x} P(x) \]

\((\mathbb{N} \cup \{+\infty\}, +, \times, 0, 1)\)
Translation from Logics to Automata

\[ \bigoplus_x P_x(x) \]

use **non-determinism** to count

- a run **per position**
- each run has the value of the subformula

\[ (N \cup \{+\infty\}, +, \times, 0, 1) \]
Translation from Logics to Automata

\[ \bigoplus_x \Phi(x) \]

use non-determinism to count

- a run per position
- each run has the value of the subformula

\[ (\mathbb{N} \cup \{+\infty\}, +, \times, 0, 1) \]
Translation from Logics to Automata

\[ \bigoplus_x \Phi(x) \]

use non-determinism to count

- a run per position
- each run has the value of the subformula

\[ (\mathbb{N} \cup \{+\infty\}, +, \times, 0, 1) \]

\( \neg \text{final?} \rightarrow \text{next} \)

\( \neg \text{final?} \rightarrow \text{next} \)

\( \text{drop}_x \)

\( \text{lift} \)

\( \mathcal{A}_\Phi \)
Translation from Logics to Automata

\[ \bigoplus_x \Phi(x) \]

use non-determinism to count

- a run per position
- each run has the value of the subformula

\[ (N \cup \{+\infty\}, +, x, 0, 1) \]

explore the graph once
Translation from Logics to Automata

Searchable Graphs

\[ G = (V, (E_d)_{d \in D}, \lambda, v^{(i)}, v^{(f)}, \leq) \]

- \(v^{(i)}\) initial vertex
- \(v^{(f)}\) final vertex

\( \leq \) total order over vertices, computable with navigating automata

\(-\text{final?}\) \text{next}

\text{drop}_x

\text{lift}

explore the graph once
Translation from Logics to Automata

Searchable Graphs

$$G = (V, (E_d)_{d \in D}, \lambda, v^{(i)}, v^{(f)}, \leq)$$

- $v^{(i)}$: initial vertex
- $v^{(f)}$: final vertex
- $\leq$: total order over vertices, computable with navigating automata

Examples: words, trees, nested words, Mazurkiewicz traces, pictures, ...

Explore the graph once

- $\neg$final?next
- $\neg$final
- drop$_x$
- lift

$$A^\Phi$$
Translation from Logics to Automata

\[ \bigoplus_x \Phi(x) \]

use non-determinism to count

- a run per position
- each run has the value of the subformula

\[ (\mathbb{N} \cup \{+\infty\}, +, \times, 0, 1) \]

\[ \neg\text{final?} \text{next} \]

\[ \neg\text{final?} \text{next} \]

\[ \text{drop}_x \]

\[ \text{lift} \]

explore the graph once
Translation from Logics to Automata

\[ \bigotimes_x \Phi(x) \]
Translation from Logics to Automata

\[ \bigotimes_x \Phi(x) \]

use sequentialization to multiply

- a single accepting run
- multiply the values of subformula along this run
Translation from Logics to Automata

$$\bigotimes_x \Phi(x)$$

use sequentialization to multiply

- a single accepting run
- multiply the values of subformula along this run

![Diagram](image)

terminal

$-\text{final?} \rightarrow \text{next}$

$\text{drop}_x$  $\rightarrow$  $A^\Phi$

$\rightarrow$  $\text{lift}$
Translation from Logics to Automata

\[ \forall x \Phi(x) \]

use sequentialization to multiply

- a single accepting run
- multiply the values of subformula along this run
Translation from Logics to Automata

Challenging for the Boolean part: we need unambiguous automata
Translation from Logics to Automata

Challenging for the *Boolean* part: we need **unambiguous** automata

Use **deterministic** automata of size **non-elementary**...
Translation from Logics to Automata

Challenging for the Boolean part: we need unambiguous automata

Use deterministic automata of size non-elementary...

Take advantage of the navigation and the pebbles to build linear sized automata
Translation from Logics to Automata

\[ \exists x \varphi(x) \]

\( \mathbb{N} \cup \{+\infty\}, +, \times, 0, 1 \)
Translation from Logics to Automata

\[ \exists x \varphi(x) \]

use *unambiguous non-determinism* to check

- a **single** accepting run
- run has value 1 or 0 depending on the truth value of the Boolean subformula
Translation from Logics to Automata

\[ \exists x \varphi(x) \]

use unambiguous non-determinism to check
- a single accepting run
- run has value 1 or 0 depending on the truth value of the Boolean subformula

\[ B \varphi \]

\[ (\mathbb{N} \cup \{+\infty\}, +, \times, 0, 1) \]
Translation from Logics to Automata

\[ \exists x \varphi(x) \]

use **unambiguous non-determinism** to check

- a **single** accepting run
- run has value 1 or 0 depending on the truth value of the Boolean subformula

unambiguous automaton for formula \( \varphi(x) \)
Theorem: Consider a searchable class of graph. Every Weighted First-Order formula can then be translated into a Pebble Weighted Automaton equivalent over this class of graphs. Obtained automata are of linear size with respect to the size of the formula.
Model Checking / Evaluation

High-Level Specification Languages

First-Order Logic

Temporal/Navigating Logics

Hybrid Expressions

Low-Level Machinery

Pebble Navigating Automata

Efficient translations

GOAL 1: describe and study versatile and expressive specification formalisms of quantitative properties, in a unified framework of graphs.

GOAL 2: introduce a more expressive class of weighted automata, able to compute a broader class of weighted properties over general classes of graphs.

GOAL 3: give efficient procedures to translate specifications in automata, and evaluate automata.

Words, Trees, Graphs...

As efficient as possible
Evaluation of Pebble Weighted Automata

**Theorem:** A Weighted Automaton $A$ can be evaluated over a graph $G$ in time $O(|Q|^3 \times |V|^3)$.

**Theorem:** A Weighted Automaton $A$ can be evaluated over a word $w$ in time $O(|Q|^3 \times |w|)$. 

idem for trees and nested words
Evaluation of Pebble Weighted Automata

**Theorem:** A Weighted Automaton $A$ can be evaluated over a graph $G$ in time $O(|Q|^3 \times |V|^3)$.

**Theorem:** A Weighted Automaton $A$ can be evaluated over a word $w$ in time $O(|Q|^3 \times |w|)$.

idem for trees and nested words

**Theorem:** A Pebble Weighted Automaton $A$ using $p$ pebble names can be evaluated over a graph $G$ in time $O(|Q|^3 \times |V|^{p+3})$.

**Theorem:** A Pebble Weighted Automaton $A$ using $p$ pebble names can be evaluated over a word $w$ in time $O(|Q|^3 \times |w|^{p+1})$.

$p$ similar to the number of variables $\neq$ quantifier depth (reusability)

Over words: Paul Gastin and Benjamin Monmege, CIAA 2012.
Over nested words: Benedikt Bollig, Paul Gastin, and Benjamin Monmege, FoSSaCS 2013.
High-Level Specification Languages

First-Order Logic

Temporal/Navigating Logics

Hybrid Expressions

GOAL 1: describe and study versatile and expressive specification formalisms of quantitative properties, in a unified framework of graphs.

Low-Level Machinery

Efficient translations

Pebble Navigating Automata

GOAL 2: introduce a more expressive class of weighted automata, able to compute a broader class of weighted properties over general classes of graphs.

GOAL

Words, Trees, Graphs...

Model

As efficient as possible

GOAL 3: give efficient procedures to translate specifications in automata, and evaluate automata.
GOAL 1: describe and study versatile and expressive specification formalisms of quantitative properties, in a unified framework of graphs.

GOAL 2: introduce a more expressive class of weighted automata, able to compute a broader class of weighted properties over general classes of graphs.

GOAL 3: give efficient procedures to translate specifications in automata, and evaluate automata.
Expressing more properties: path expressions
Expressing more properties: path expressions

In First-Order Logic, you cannot follow *unbounded complex paths*.
Expressing more properties: path expressions

In First-Order Logic, you cannot follow *unbounded* complex paths
Expressing more properties: path expressions

In First-Order Logic, you cannot follow unbounded complex paths

A path: $\rightarrow \downarrow \downarrow \downarrow \rightarrow \rightarrow \uparrow$
Expressing more properties: path expressions

In First-Order Logic, you cannot follow unbounded complex paths

A path: $\rightarrow \downarrow \downarrow \downarrow \rightarrow \rightarrow \uparrow$

A pattern: $(\downarrow + \rightarrow)^* \cdot \uparrow \cdot (\downarrow + \leftarrow)^*$
Expressing more properties: path expressions

In First-Order Logic, you cannot follow unbounded complex paths

A path: $\rightarrow \downarrow \downarrow \downarrow \rightarrow \rightarrow \uparrow$

A pattern: $(\downarrow + \rightarrow)^* \cdot \uparrow \cdot (\downarrow + \leftarrow)^*$

What is the length of the shortest path from $\blacksquare$ to $\square$?

$(\mathbb{N} \cup \{+\infty\}, \min, +, +\infty, 0) \cdot (\blacksquare \lor \square) \cdot 1 \cdot (\rightarrow + \leftarrow + \downarrow + \uparrow))^* \cdot \square$
Expressing more properties: path expressions

In First-Order Logic, you cannot follow unbounded complex paths.

A path: \( \rightarrow \downarrow \downarrow \downarrow \rightarrow \rightarrow \uparrow \)

A pattern: \((\downarrow + \rightarrow)^* \cdot \uparrow \cdot (\downarrow + \leftarrow)^*\)

What is the length of the shortest path from blue to red?

\(((\text{blue} \lor \text{yellow}) \cdot 1 \cdot (\rightarrow + \leftarrow + \downarrow + \uparrow))^* \cdot \text{red}\)
Translation from Expressions to Automata

**Theorem:** Weighted Expressions and Weighted Automata are effectively equivalent.

WExp $\rightarrow$ WA

Over words: Paul Gastin and Benjamin Monmege, CIAA 2012.
Over nested words: Benedikt Bollig, Paul Gastin, and Benjamin Monmege, FoSSaCS 2013.
Translation from Expressions to Automata

**Theorem:** Weighted Expressions and Weighted Automata are effectively equivalent.

\[
\text{WExp} \quad \rightarrow \quad \text{WA}
\]

**Theorem:** Hybrid Weighted Expressions and Pebble Weighted Automata are effectively equivalent.

\[
\text{HybWExp} \quad \rightarrow \quad \text{PWA}
\]

Over words: Paul Gastin and Benjamin Monmege, CIAA 2012.
Over nested words: Benedikt Bollig, Paul Gastin, and Benjamin Monmege, FoSSaCS 2013.
Translation from Expressions to Automata

**Theorem:** Weighted Expressions and Weighted Automata are effectively equivalent.

\[
\text{WExp} \quad \xrightarrow{} \quad \text{WA}
\]

**Theorem:** Hybrid Weighted Expressions and Pebble Weighted Automata are effectively equivalent.

\[
\text{HybWExp} \quad \xrightarrow{} \quad \text{PWA}
\]

Extensions of **Kleene-Schützenberger theorems** for generic classes of graphs and expressions/automata navigating in graphs, with pebbles.

Over words: Paul Gastin and Benjamin Monmege, CIAA 2012.
Over nested words: Benedikt Bollig, Paul Gastin, and Benjamin Monmege, FoSSaCS 2013.
Translation from Expressions to Automata

**Theorem:** Weighted Expressions and Weighted Automata are effectively equivalent.

\[ \text{WExp} \xrightarrow{\text{linear time}} \text{WA} \]

**Theorem:** Hybrid Weighted Expressions and Pebble Weighted Automata are effectively equivalent.

\[ \text{HybWExp} \xrightarrow{\text{linear time}} \text{PWA} \]

Extensions of **Kleene-Schützenberger theorems** for generic classes of graphs and expressions/automata navigating in graphs, with pebbles

Translators from expressions to automata are efficient, and produce automata in linear time

Over words: Paul Gastin and Benjamin Monmege, CIAA 2012.
Over nested words: Benedikt Bollig, Paul Gastin, and Benjamin Monmege, FoSSaCS 2013.
Translation from Expressions to Automata

**Theorem:** *Weighted Expressions* and *Weighted Automata* are effectively equivalent.

\[
\begin{align*}
\text{WExp} & \quad \text{linear time} \quad \rightarrow \quad \text{WA}
\end{align*}
\]

**Theorem:** *Hybrid Weighted Expressions* and *Pebble Weighted Automata* are effectively equivalent.

\[
\begin{align*}
\text{HybWExp} & \quad \text{linear time} \quad \rightarrow \quad \text{PWA}
\end{align*}
\]

Prototype implemented: **QuantiS tool**


Over words: Paul Gastin and Benjamin Monmege, CIAA 2012.
Over nested words: Benedikt Bollig, Paul Gastin, and Benjamin Monmege, FoSSaCS 2013.
High-Level Specification Languages

- First-Order Logic
- Temporal/Navigating Logics
- Hybrid Expressions

Low-Level Machinery

- Pebble Navigating Automata

Efficient translations

Probabilistic case

GOAL

Words, Trees, Graphs...

Model Checking / Evaluation

As efficient as possible
High-Level Specification Languages

First-Order Logic
Temporal/Navigating Logics
Hybrid Expressions

Efficient translations

Low-Level Machinery

Pebble Navigating Automata
extensions of probabilistic automata

Probabilistic case

As efficient as possible

GOAL

Model Checking / Evaluation

Words, Trees, Graphs...
High-Level Specification Languages

First-Order Logic
Temporal/Navigating Logics
Hybrid Expressions
probabilistic hybrid expressions
\( \frac{1}{2} \cdot a? \cdot \rightarrow^* + \frac{1}{2} \cdot b? \cdot \downarrow \\
\cdot \left( \frac{1}{2} \cdot c \cdot \rightarrow \right)^* \cdot \frac{1}{2} \cdot a \)

Low-Level Machinery
Pebble Navigating Automata
extensions of probabilistic automata

Efficient translations

Probabilistic case

Words, Trees, Graphs...

Model Checking / Evaluation

As efficient as possible

GOAL

Efficient translations

As efficient as possible
High-Level Specification Languages

First-Order Logic
Temporal/Navigating Logics
Hybrid Expressions
probabilistic hybrid expressions
\[ \frac{1}{2} \cdot a? \cdot \xrightarrow{*} + \frac{1}{2} \cdot b? \cdot \downarrow \cdot \left( \frac{1}{2} \cdot c \cdot \xrightarrow{*} \right) \cdot \frac{1}{2} \cdot a \]

Low-Level Machinery

Pebble Navigating Automata
extensions of probabilistic automata

Efficient translations

Words, Trees, Graphs...

Theorem: Probabilistic Pebble Automata and Probabilistic Hybrid Expressions are equivalent.

GOAL

As efficient as possible

Over words: Benedikt Bollig, Paul Gastin, Benjamin Monmege, and Marc Zeitoun, ATVA 2012.
High-Level Specification Languages

First-Order Logic
Temporal/Navigating Logics
Hybrid Expressions

Low-Level Machinery

Pebble Navigating Automata

Efficient translations

Non-Continuous Semirings
\((\mathbb{R}, +, \times, 0, 1)\)

Model Checking / Evaluation

Over words: Benedikt Bollig, Paul Gastin, Benjamin Monmege, and Marc Zeitoun, ICALP 2010.
High-Level Specification Languages

First-Order Logic
Temporal/Navigating Logics
Hybrid Expressions

Low-Level Machinery

Pebble Navigating Automata
restrict the semantics (remove loops in runs)

Non-Continuous Semirings
\((\mathbb{R}, +, \times, 0, 1)\)

Efficient translations

Model Checking / Evaluation

Over words: Benedikt Bollig, Paul Gastin, Benjamin Monmege, and Marc Zeitoun, ICALP 2010.
High-Level Specification Languages

- First-Order Logic
- Temporal/Navigating Logics
- Hybrid Expressions

Low-Level Machinery

- Pebble Navigating Automata
  - restrict the semantics (remove loops in runs)
  - restrict the syntax (one-way automata)

Efficient translations

Non-Continuous Semirings

\((\mathbb{R}, +, \times, 0, 1)\)

Over words: Benedikt Bollig, Paul Gastin, Benjamin Monmege, and Marc Zeitoun, ICALP 2010.
High-Level Specification Languages
- First-Order Logic
- Temporal/Navigating Logics
- Hybrid Expressions
  
  similar restrictions over expressions and logics

Low-Level Machinery
- Pebble Navigating Automata
  restrict the semantics
  (remove loops in runs)
  or
  restrict the syntax
  (one-way automata)

Efficient translations

Non-Continuous Semirings
\((\mathbb{R}, +, \times, 0, 1)\)

GOAL

Words, trees, graphs...

Model Checking / Evaluation

As efficient as possible

Over words: Benedikt Bollig, Paul Gastin, Benjamin Monmege, and Marc Zeitoun, ICALP 2010.
High-Level Specification Languages

- First-Order Logic
- Temporal/Navigating Logics
- Hybrid Expressions

similar restrictions over expressions and logics

Efficient translations

Non-Continuous Semirings
\((\mathbb{R}, +, \times, 0, 1)\)

Low-Level Machinery

- Pebble Navigating Automata
- restrict the semantics (remove loops in runs)
- restrict the syntax (one-way automata)
- for words

GOAL

Model Checking / Evaluation

As efficient as possible

Words, Trees, Graphs...

Over words: Benedikt Bollig, Paul Gastin, Benjamin Monmege, and Marc Zeitoun, ICALP 2010.
High-Level Specification Languages

- First-Order Logic
- Temporal/Navigating Logics
- Hybrid Expressions

Efficient translations

Low-Level Machinery

- Pebble Navigating Automata

Model Checking / Evaluation

GOAL

As efficient as possible

Model

Words, Trees, Graphs...
High-Level Specification Languages

First-Order Logic

Temporal/Navigating Logics

Hybrid Expressions

Efficient translations

Low-Level Machinery

Pebble Navigating Automata

already mentioned

Words, Trees, Graphs...

Model Checking / Evaluation

As efficient as possible

GOAL

Model
Theorem: Weighted First Order logic with weighted transitive closure and Pebble Weighted Automata are equivalent for zonable and searchable classes of graphs.

generalizes [Thomas 82], [Matz 98], [Engelfriet&Hoogeboom 07]
Perspectives

• **Applications** in various areas: Natural Language Processing, Image compression, Quantitative verification of software, ...

  restrict the power to obtain even more *intuitive* formalisms?
Perspectives

• **Applications** in various areas: Natural Language Processing, Image compression, Quantitative verification of software, ...

  restrict the power to obtain even more *intuitive* formalisms?

• Add *data* or *weights* in the graphical models
Perspectives

• **Applications** in various areas: Natural Language Processing, Image compression, Quantitative verification of software, ...

  restrict the power to obtain even more *intuitive* formalisms?

• **Add data or weights** in the graphical models

• **Consider system-evaluation** of a system generating graphs, against a quantitative specification
Perspectives

• **Applications** in various areas: Natural Language Processing, Image compression, Quantitative verification of software, ...

  restrict the power to obtain even more *intuitive* formalisms?

• **Add data or weights** in the graphical models

• **Consider** *system-evaluation* of a system generating graphs, against a quantitative specification

• **Consider** logical specifications able to *compare* several computed weights like in PCTL for example