Two-way cost automata and cost logics over infinite trees

Achim Blumensath\textsuperscript{1}, Thomas Colcombet\textsuperscript{2},
Denis Kuperberg\textsuperscript{3}, Pawel Parys\textsuperscript{3}, and Michael Vanden Boom\textsuperscript{4}

\textsuperscript{1}TU Darmstadt, \textsuperscript{2}Université Paris Diderot,
\textsuperscript{3}University of Warsaw, \textsuperscript{4}University of Oxford

CSL-LICS 2014
Vienna, Austria
Boundedness questions

**Finite power property** [Simon ’78, Hashiguchi ’79]
given regular language $L$ of finite words,
is there $n \in \mathbb{N}$ such that $L^* = \{\varepsilon\} \cup L^1 \cup L^2 \cup \cdots \cup L^n$?

**Star-height problem** [Hashiguchi ’88, Kirsten ’05]
given regular language $L$ of finite words and $n \in \mathbb{N}$,
is there a regular expression for $L$ with at most $n$ nestings of Kleene star?

**Fixpoint closure boundedness** [Blumensath+Otto+Weyer ’09]
given an MSO formula $\varphi(x, X)$ positive in $X$,
is there $n \in \mathbb{N}$ such that the least fixpoint of $\varphi$ over finite words
is always reached within $n$ iterations?
Boundedness questions

The **theory of regular cost functions** is an extension of the theory of regular languages that can be used to solve these boundedness questions in a uniform way.
Boundedness questions

The **theory of regular cost functions** is an extension of the theory of regular languages that can be used to solve these boundedness questions in a uniform way.

**Boundedness problem**

**Instance:** function $f : \mathcal{D} \to \mathbb{N} \cup \{\infty\}$

($\mathcal{D}$ is set of words or trees over some fixed finite alphabet $\mathbb{A}$)

**Question:** Is there $n \in \mathbb{N}$ such that for all structures $s \in \mathcal{D}$, $f(s) \leq n$?
Cost functions over finite words [Colcombet’09]

Regular Cost Functions
- nondeterministic cost automata
- cost MSO
- BS expressions
- stabilization monoids

Boundedness decidable
[Colcombet’09, Bojańczyk+Colcombet’06]
Cost functions over finite words

Cost monadic second-order logic (CMSO)

Atomic formulas: \( a(x) \quad x \in X \quad |X| \leq N \)

must occur positively

Constructors: \( \wedge, \vee, \neg \)

Boolean connectives

\( \exists x \)

first-order quantification

\( \exists X \)

monadic second-order quantification
Cost functions over finite words

Cost monadic second-order logic (CMSO)

Atomic formulas: \( a(x) \quad x \in X \quad |X| \leq N \)

must occur positively

Constructors: \( \land, \lor, \neg \)

first-order quantification

Boolean connectives

monadic second-order quantification

Semantics \( \llbracket \varphi \rrbracket : \mathbb{A}^* \rightarrow \mathbb{N} \cup \{\infty\} \)

\( \llbracket \varphi \rrbracket (u) := \inf \{ n : u \models \varphi[n/N] \} \)
Cost functions over finite words

Cost monadic second-order logic (CMSO)

 Atomic formulas: \( a(x) \quad x \in X \quad |X| \leq N \)

 must occur positively

 Constructors: \( \land, \lor, \neg \)

 Boolean connectives

 \( \exists x \)

 first-order quantification

 \( \exists X \)

 monadic second-order quantification

 Semantics \( [\varphi] : A^* \rightarrow \mathbb{N} \cup \{\infty\} \)

 \( [\varphi](u) := \inf \{ n : u \models \varphi[n/N] \} \)

 Example

 If \( \varphi \) is in MSO, then \( [\varphi](u) := \begin{cases} 0 & \text{if } u \models \varphi \\ \infty & \text{otherwise} \end{cases} \)
Cost functions over finite words

Cost monadic second-order logic (CMSO)

Atomic formulas: \( a(x) \quad x \in X \quad |X| \leq N \)

Constructors: \( \land, \lor, \neg \) first-order quantification \( \forall X \) monadic second-order quantification

Semantics \( \llbracket \varphi \rrbracket : \mathbb{A}^* \rightarrow \mathbb{N} \cup \{\infty\} \)

\( \llbracket \varphi \rrbracket (u) := \inf \{ n : u \models \varphi[\frac{n}{N}] \} \)

Example

Maximum length of a block of \( a \)'s

\( \varphi := \forall X \left( (\text{block}(X) \land \forall x(x \in X \rightarrow a(x)) \rightarrow |X| \leq N \right) \)
Cost functions over finite words [Colcombet’09]

**Regular Cost Functions**

- Nondeterministic cost automata
- Cost MSO
- BS expressions
- Stabilization monoids

**Boundedness decidable**

[Colcombet’09, Bojańczyk+Colcombet’06]
Cost functions over finite words [Colcombet’09]

Regular Cost Functions
- nondeterministic cost automata
- cost MSO
- BS expressions
- stabilization monoids

Boundedness decidable
[Colcombet’09, Bojańczyk+Colcombet’06]

Language universality, inclusion, and emptiness decidable
Cost functions over finite words [Colcombet’09]

Regular Cost Functions

- nondeterministic cost automata
- cost MSO
- BS expressions
- stabilization monoids

Boundedness decidable
[Colcombet’09, Bojańczyk+Colcombet’06]

- language universality, inclusion, and emptiness decidable
- finite power property, star height problem, fixpoint closure boundedness, ...
  decidable
The theory of regular cost functions is a robust decidable extension of the theory of regular languages over:

- **finite words** [Colcombet ’09, Bojanczyk+Colcombet ’06]

- **infinite words** [Kuperberg+VB’12, Colcombet unpublished]

- **finite trees** [Colcombet+Löding ’10]
The theory of regular cost functions is a robust decidable extension of the theory of regular languages over:

- **finite words** [Colcombet ’09, Bojanczyk+Colcombet ’06]
- **infinite words** [Kuperberg+VB’12, Colcombet unpublished]
- **finite trees** [Colcombet+Löding ’10]
- **infinite trees**
Motivating open problem

Mostowski index problem

**Instance:** regular language $L$ of infinite trees, and set $\{i, i+1, \ldots, j\}$

**Question:** Is there a nondeterministic parity automaton $\mathcal{A}$ using only priorities $\{i, i+1, \ldots, j\}$ such that $L = L(\mathcal{A})$?
Motivating open problem

**Mostowski index problem**

**Instance:** regular language $L$ of infinite trees, and set $\{i, i + 1, \ldots, j\}$

**Question:** Is there a nondeterministic parity automaton $\mathcal{A}$ using only priorities $\{i, i + 1, \ldots, j\}$ such that $L = L(\mathcal{A})$?

Reduced to deciding boundedness for certain cost functions over infinite trees [Colcombet+Löding ’08]
Cost functions over infinite trees

- Regular Cost Functions
  - alternating cost-parity automata

- QW Cost Functions
  - quasi-weak cost automata

- Boundedness decidable
  - [Kuperberg+VB’11]

- Weak cost automata
  - WCMSO

Special case of Mostowski index problem
Cost functions over infinite trees

Regular Cost Functions
alternating cost-parity automata

QW Cost Functions
quasi-weak cost automata
QWCMSO

Boundedness decidable
[Kuperberg+VB’11]

weak cost automata
WCMSO

special case of Mostowski index problem
Cost functions over infinite trees

Regular Cost Functions
alternating 2-way/1-way cost-parity automata

QW Cost Functions
2-way/1-way qw cost automata
QWCMSO

Boundedness decidable
[Kuperberg+VB’11]

weak cost automata
WCMSO

special case of Mostowski index problem
Cost parity automata on infinite trees

\[ \mathcal{A} = \langle A, Q, q_0, \delta, \Omega \rangle \]

- \( \delta \) describes possible moves for Eve and Adam, and associated counter actions (increment, reset, leave unchanged)
- \( \Omega : Q \rightarrow P \) for a finite set of priorities \( P \)

**n-acceptance game** \( \mathcal{A} \times t \)

- Positions in the game are \( Q \times \text{dom}(t) \).
- Eve and Adam select the next position in the play based on \( \delta \).
- Eve is trying to ensure the play has counter value at most \( n \) and the maximum priority occurring infinitely often in the play is even.

**Semantics**

\[ [\mathcal{A}](t) := \inf \{ n : \text{Eve wins the } n\text{-acceptance game } \mathcal{A} \times t \} \]
Weak cost automata and logic over infinite trees

Weak cost automaton
alternating cost-parity automaton such that no cycle visits both even and odd priorities
Weak cost automata and logic over infinite trees

Weak cost automaton
alternating cost-parity automaton such that no cycle visits both even and odd priorities

Weak cost monadic second-order logic (WCMSO)
Syntax like CMSO, but interpret second-order quantification over finite sets
Weak cost automata and logic over infinite trees

Weak cost automaton
alternating cost-parity automaton such that no cycle visits both even and odd priorities

Weak cost monadic second-order logic (WCMSO)
Syntax like CMSO, but interpret second-order quantification over finite sets
Quasi-weak cost automata and logic over infinite trees

Quasi-weak cost automaton
alternating cost-parity automaton such that
in any cycle with both even and odd priorities,
there is a counter which is incremented but not reset
Quasi-weak cost automata and logic over infinite trees

Quasi-weak cost automaton
alternating cost-parity automaton such that
in any cycle with both even and odd priorities,
there is a counter which is incremented but not reset

Quasi-weak cost monadic second-order logic (QWCMSO)
Add bounded expansion operator to WCMSO:

\[ z \in \mu^N Y. \{x : \varphi(x, Y)\} \]

where \( Y \) occurs positively in \( \varphi(x, Y) \),
and this operator occurs positively in the enclosing formula.
**Quasi-weak cost automata and logic over infinite trees**

**Quasi-weak cost automaton**

alternating cost-parity automaton such that in any cycle with both even and odd priorities, there is a counter which is incremented but not reset.

**Quasi-weak cost monadic second-order logic (QWCMSO)**

Add **bounded expansion operator** to WCMSO:

\[ z \in \mu^N Y. \{ x : \varphi(x, Y) \} \]

where \( Y \) occurs positively in \( \varphi(x, Y) \), and this operator occurs positively in the enclosing formula.

**Example**

Maximal size of block of \( a \)'s on a branch starting at the root:

\[ \exists w[ \text{root}(w) \land w \in \mu^N X. \{ x : \exists yz[ b(x, y, z) \lor (a(x, y, z) \land y \in X \land z \in X) ] \} ] \]
Game for testing
\( z \in \mu^N \cdot \{ x : \varphi(x, Y) \} \) for \( n \in \mathbb{N} \).

Initial position \( x := z \).

Game from position \( x \):

- Eve chooses set \( Y \) such that \( \varphi(x, Y) \) holds
  (if it is not possible, she loses).
- Adam chooses some new \( y \in Y \)
  (if it is not possible, he loses).
- Game continues in next round with \( x := y \)

If the game exceeds \( n \) rounds, Adam wins.
Bounded expansion operator and 2-way automata

Game for testing
\[ z \in \mu^N \forall x: \varphi(x, Y) \] for \( n \in \mathbb{N} \).

Initial position \( x := z \).

Game from position \( x \):

- **Eve chooses** set \( Y \) such that \( \varphi(x, Y) \) holds
  (if it is not possible, she loses).
- Adam chooses some new \( y \in Y \)
  (if it is not possible, he loses).
- Game continues in next round with \( x := y \)

If the game exceeds \( n \) rounds, Adam wins.
Game for testing \( z \in \mu^N \mathcal{Y}. \{x : \varphi(x, \mathcal{Y})\} \) for \( n \in \mathbb{N} \).

Initial position \( x := z \).

Game from position \( x \):

- Eve chooses set \( \mathcal{Y} \) such that \( \varphi(x, \mathcal{Y}) \) holds
  (if it is not possible, she loses).

- Adam chooses some new \( y \in \mathcal{Y} \)
  (if it is not possible, he loses).

- Game continues in next round with \( x := y \)

If the game exceeds \( n \) rounds, Adam wins.
Bounded expansion operator and 2-way automata

Game for testing
\[ z \in \mu^N \{ x : \phi(x, Y) \} \text{ for } n \in \mathbb{N}. \]

Initial position \( x := z \).

Game from position \( x \):  
- Eve chooses set \( Y \) such that \( \phi(x, Y) \) holds  
  (if it is not possible, she loses).
- Adam chooses some new \( y \in Y \)  
  (if it is not possible, he loses).
- Game continues in **next round** with \( x := y \)

If the game exceeds \( n \) rounds, Adam wins.
Game for testing
\( z \in \mu^N Y.\{x : \phi(x, Y)\} \) for \( n \in \mathbb{N} \).

Initial position \( x := z \).

Game from position \( x \):
- **Eve chooses** set \( Y \) such that 
  \( \phi(x, Y) \) holds
  (if it is not possible, she loses).
- Adam chooses some new \( y \in Y \)
  (if it is not possible, he loses).
- Game continues in next round with \( x := y \)

If the game exceeds \( n \) rounds, Adam wins.
Bounded expansion operator and 2-way automata

Game for testing
\[ z \in \mu^N \{ x : \varphi(x, Y) \} \text{ for } n \in \mathbb{N}. \]

Initial position \( x := z \).

Game from position \( x \):

- Eve chooses set \( Y \) such that \( \varphi(x, Y) \) holds
  (if it is not possible, she loses).
- **Adam chooses** some new \( y \in Y \)
  (if it is not possible, he loses).
- Game continues in next round with \( x := y \)

If the game exceeds \( n \) rounds, Adam wins.
**Summary**

*Regular Cost Functions*
- alternating 2-way/1-way cost-parity automata
- cost $\mu$-calculus

*QW Cost Functions*
- 2-way/1-way qw cost automata
- alternation-free cost $\mu$-calculus
- QWCMSO

*Boundedness decidable*
- weak cost automata
- WCMSO