Descriptive complexity vs. decidability for MSO

Michał Skrzypczak

University of Warsaw

Journées d'Informatique Fondamentale de Paris Diderot
25'th April 2013, Paris

$$w = \; a\; a\; a\; a\; b\; c\; c\; b \;\in \{a, b, c\}^*$$

Regular languages of finite words

$$w = \; a \; a \; a \; a \; b \; c \; c \; b \; \in \{a, b, c\}^*$$

Regular expressions

$$(aa)^* \; b \; (b \mid c)^*$$

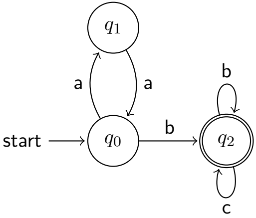
Regular languages of finite words

$$w = \ a\ a\ a\ a\ b\ c\ c\ b\ \in \{a,b,c\}^*$$

Regular expressions

$$(aa)^*\ b\ (b \mid c)^*$$

Finite automata



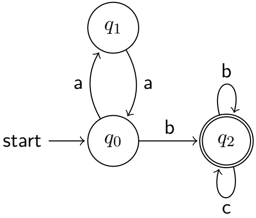
Regular languages of finite words

$$w = \ a\ a\ a\ a\ b\ c\ c\ b\ \in \{a,b,c\}^*$$

Regular expressions

$$(aa)^*\ b\ (b \mid c)^*$$

Finite automata



Monadic Second-Order logic

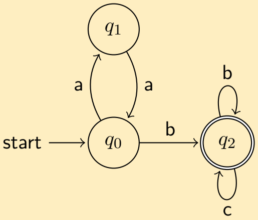
$$\exists_x\ b(x) \wedge \forall_{y<x}\ a(y)\ \wedge\ \ldots$$

$$w = \; a\ a\ a\ a\ b\ c\ c\ b\ \in \{a, b, c\}^*$$

Regular expressions

$$(aa)^*\ b\ (b \mid c)^*$$

Finite automata



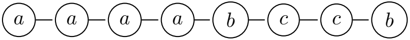
Monadic Second-Order logic

$$\exists_x\ b(x) \wedge \forall_{y<x}\ a(y)\ \wedge\ \ldots$$

Monadic Second-Order logic

$w =$ ⓐ—ⓐ—ⓐ—ⓐ—ⓑ—ⓒ—ⓒ—ⓑ

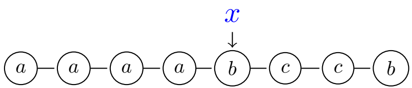
Monadic Second-Order logic

$w =$ 

$$\psi = \exists_x \; b(x) \wedge \forall_{y<x} \; a(y) \;\; \wedge \exists_P \; x \in P \;\; \wedge \ldots$$

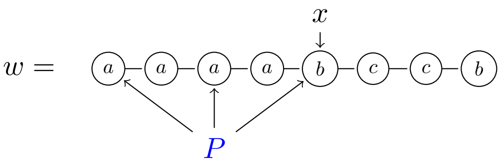
Monadic Second-Order logic

$w =$



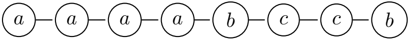
$$\psi = \exists_x\ b(x) \wedge \forall_{y<x}\ a(y)\ \wedge \exists_P\ x \in P\ \wedge \ldots$$

Monadic Second-Order logic



$$\psi = \exists_x \; b(x) \wedge \forall_{y<x} \; a(y) \;\; \wedge {\color{blue}\exists_P} \; x \in P \;\; \wedge \ldots$$

Monadic Second-Order logic

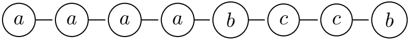
$w =$ 

$$\psi = \exists_x \; b(x) \wedge \forall_{y<x} \; a(y) \;\; \wedge \exists_P \; x \in P \;\; \wedge \ldots$$

Language defined by ψ

$$\mathrm{L}(\psi) = \{w \in A^* : \; w \text{ satisfies } \psi\}$$

Monadic Second-Order logic

$w =$ 

$$\psi = \exists_x \; b(x) \wedge \forall_{y<x} \; a(y) \;\; \wedge \exists_P \; x \in P \;\; \wedge \ldots$$

Language defined by ψ

$$\mathrm{L}(\psi) = \{w \in A^* : \; w \text{ satisfies } \psi\}$$

Satisfiability problem

Given an MSO formula ψ decide whether $\mathrm{L}(\psi) \neq \emptyset$?

Monadic Second-Order logic

$w =$ ⓐ—ⓐ—ⓐ—ⓐ—ⓑ—ⓒ—ⓒ—ⓑ

$$\psi = \exists_x\ b(x) \wedge \forall_{y<x}\ a(y)\ \wedge \exists_P\ x \in P\ \wedge \ldots$$

Language defined by ψ

$$\mathrm{L}(\psi) = \{w \in A^* :\ w \text{ satisfies } \psi\}$$

Satisfiability problem

Given an MSO formula ψ decide whether $\mathrm{L}(\psi) \neq \emptyset$?

Formula is a declarative definition of the language.
Automata are more operational.

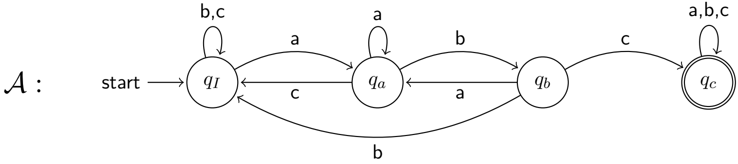
Finite automata

$$L = (a \mid b \mid c)^* \, abc \, (a \mid b \mid c)^*$$

Finite automata

$$L = (a \mid b \mid c)^* \; abc \; (a \mid b \mid c)^*$$

Deterministic automaton — only one transition to use

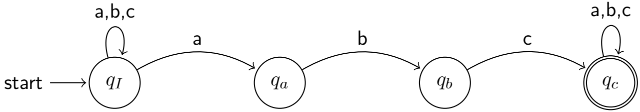


Finite automata

$$L = (a \mid b \mid c)^* \; abc \; (a \mid b \mid c)^*$$

Non-deterministic automaton — many possible transitions

$\mathcal{B}$:

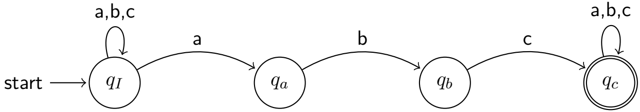


Finite automata

$$L = (a \mid b \mid c)^* \; abc \; (a \mid b \mid c)^*$$

Non-deterministic automaton — many possible transitions

$\mathcal{B}$:



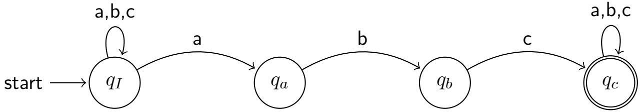
Run of an automaton

$$\begin{array}{lcl} w & = & \quad a \quad\; b \quad\; b \quad\; a \quad\; b \quad\; c \quad\; c \\ \rho & = & q_I \longrightarrow q_I \longrightarrow q_I \longrightarrow q_I \longrightarrow q_a \longrightarrow q_b \longrightarrow q_c \longrightarrow q_c \end{array}$$

$$L = (a \mid b \mid c)^* \; abc \; (a \mid b \mid c)^*$$

Non-deterministic automaton — many possible transitions

$\mathcal{B}$:



Run of an automaton

$$\begin{array}{lllllllll} w & = & a & b & b & a & b & c & c \\ \rho & = & q_I \to q_I & \to q_I & \to q_I & \to q_a & \to q_b & \to q_c & \to q_c \end{array}$$

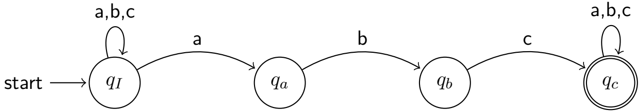
A run ρ is accepting if it ends in a final state.

Finite automata

$$L = (a \mid b \mid c)^* \, abc \, (a \mid b \mid c)^*$$

Non-deterministic automaton — many possible transitions

$\mathcal{B}$:



Run of an automaton

$$\begin{array}{rcccccccccccccccc} w & = & & a & & b & & b & & a & & b & & c & & c & \\ \rho & = & q_I & \longrightarrow & q_I & \longrightarrow & q_I & \longrightarrow & q_I & \longrightarrow & q_a & \longrightarrow & q_b & \longrightarrow & q_c & \longrightarrow & q_c \end{array}$$

A run ρ is accepting if it ends in a final state.

Language recognised by the automaton

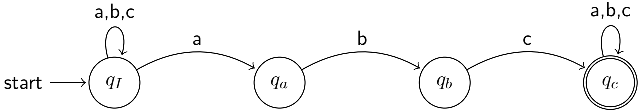
deterministic : $\{w \in A^* :$ the run of $\mathcal{A}$ on w is accepting$\}$

Finite automata

$$L = (a \mid b \mid c)^* \; abc \; (a \mid b \mid c)^*$$

Non-deterministic automaton — many possible transitions

$\mathcal{B}$:



Run of an automaton

$$\begin{array}{lcccccccccccccccc} w & = & & a & & b & & b & & a & & b & & c & & c & \\ \rho & = & q_I & \longrightarrow & q_I & \longrightarrow & q_I & \longrightarrow & q_I & \longrightarrow & q_a & \longrightarrow & q_b & \longrightarrow & q_c & \longrightarrow & q_c \end{array}$$

A run ρ is accepting if it ends in a final state.

Language recognised by the automaton

deterministic : $\{w \in A^* :$ the run of $\mathcal{A}$ on w is accepting$\}$

non-det. : $\{w \in A^* :$ exists an accepting run of $\mathcal{A}$ on $w\}$

Equivalence

Theorem (Büchi [1960], Elgot [1961], Trakhtenbrot [1962])

For $L \subseteq A^*$ the following conditions are (effectively) equivalent:

- $L = \mathrm{L}(\psi)$ *for an MSO formula* ψ,

Equivalence

Theorem (Büchi [1960], Elgot [1961], Trakhtenbrot [1962])

For $L \subseteq A^*$ the following conditions are (effectively) equivalent:

- $L = \mathrm{L}(\psi)$ *for an MSO formula* ψ,
- $L = \mathrm{L}(\mathcal{B})$ *for a non-deterministic automaton* $\mathcal{B}$,

Theorem (Büchi [1960], Elgot [1961], Trakhtenbrot [1962])

For $L \subseteq A^*$ the following conditions are (effectively) equivalent:

- $L = \mathrm{L}(\psi)$ *for an MSO formula* ψ*,*
- $L = \mathrm{L}(\mathcal{B})$ *for a non-deterministic automaton* $\mathcal{B}$*,*
- $L = \mathrm{L}(\mathcal{A})$ *for a deterministic automaton* $\mathcal{A}$*.*

Such a language is called regular.

Equivalence

Theorem (Büchi [1960], Elgot [1961], Trakhtenbrot [1962])

For $L \subseteq A^*$ the following conditions are (effectively) equivalent:

- $L = \mathrm{L}(\psi)$ *for an MSO formula* ψ,
- $L = \mathrm{L}(\mathcal{B})$ *for a non-deterministic automaton* $\mathcal{B}$,
- $L = \mathrm{L}(\mathcal{A})$ *for a deterministic automaton* $\mathcal{A}$.

Such a language is called regular.

Crucial step - determinization

Given a non-deterministic automaton compute an equivalent deterministic one (via powerset construction).

Equivalence

Theorem (Büchi [1960], Elgot [1961], Trakhtenbrot [1962])

For $L \subseteq A^*$ the following conditions are (effectively) equivalent:

- $L = \mathrm{L}(\psi)$ *for an MSO formula* ψ*,*
- $L = \mathrm{L}(\mathcal{B})$ *for a non-deterministic automaton* $\mathcal{B}$*,*
- $L = \mathrm{L}(\mathcal{A})$ *for a deterministic automaton* $\mathcal{A}$*.*

Such a language is called regular.

Crucial step - determinization

Given a non-deterministic automaton compute an equivalent deterministic one (via powerset construction).

Deciding satisfiability

Take ψ, compute an equivalent automaton $\mathcal{A}$, check if $\mathcal{A}$ accepts some word.

Simplifying the formulæ

ψ — MSO formula (many quantifiers inside)

Simplifying the formulæ

ψ — MSO formula (many quantifiers inside)

$\rightsquigarrow$

$\mathcal{A}$ — deterministic automaton

Simplifying the formulæ

ψ — MSO formula (many quantifiers inside)

$\rightsquigarrow$

$\mathcal{A}$ — deterministic automaton

$\rightsquigarrow$

ψ' — simplified formula

Simplifying the formulæ

ψ — MSO formula (many quantifiers inside)

$\rightsquigarrow$

$\mathcal{A}$ — deterministic automaton

$\rightsquigarrow$

$\psi' = \exists_\rho \; (\rho \text{ is a run of } \mathcal{A}) \;\wedge\; (\rho \text{ is accepting})$

Simplifying the formulæ

ψ — MSO formula (many quantifiers inside)

$\rightsquigarrow$

$\mathcal{A}$ — deterministic automaton

$\rightsquigarrow$

$$\psi' = \exists_\rho \underbrace{(\rho \text{ is a run of } \mathcal{A}) \wedge (\rho \text{ is accepting})}_{\text{no set quantifiers here}}$$

Simplifying the formulæ

ψ — MSO formula (many quantifiers inside)

$\rightsquigarrow$

$\mathcal{A}$ — deterministic automaton

$\rightsquigarrow$

$$\psi' = \underbrace{\exists_\rho \; \underbrace{(\rho \text{ is a run of } \mathcal{A}) \;\wedge\; (\rho \text{ is accepting})}_{\text{no set quantifiers here}}}_{\exists_{\vec{X}} \; (\ldots) \text{ — existential formula}}$$

Simplifying the formulæ

ψ — MSO formula (many quantifiers inside)

$\mathcal{A}$ — deterministic automaton

$$\psi' = \underbrace{\exists_\rho \ (\rho \text{ is a run of } \mathcal{A}) \ \wedge \ (\rho \text{ is accepting})}_{\exists_{\vec{X}} \ (\ldots) \text{ — existential formula}}$$

$$\psi'' = \underbrace{\forall_\rho \ (\rho \text{ is a run of } \mathcal{A}) \ \Rightarrow \ (\rho \text{ is accepting})}_{\forall_{\vec{X}} \ (\ldots) \text{ — universal formula}}$$

$$w = \quad a - a - a - a - b - c - c - b - \cdots \quad \in A^{\omega}$$

Infinite words

$$w = \quad a - a - a - a - b - c - c - b - \cdots \quad \in A^\omega$$

ω-language defined by ψ — the same definition

$$\mathrm{L}(\psi) = \{w \in A^\omega :\ w \text{ satisfies } \psi\}\,.$$

Infinite words

$$w = \quad (a)-(a)-(a)-(a)-(b)-(c)-(c)-(b)- \cdots \quad \in A^\omega$$

ω-language defined by ψ — the same definition

$$\mathrm{L}(\psi) = \{w \in A^\omega : \ w \text{ satisfies } \psi\}\,.$$

What about automata?

When an infinite run ρ is accepting?
There is no *last* state!

Infinite words

$$w = \quad (a)-(a)-(a)-(a)-(b)-(c)-(c)-(b)- \cdots \quad \in A^\omega$$

ω-language defined by ψ — the same definition

$$\mathrm{L}(\psi) = \{w \in A^\omega : \; w \text{ satisfies } \psi\}\,.$$

Büchi automata

A run of a Büchi automaton is accepting if it visits a final state infinitely many times.

Infinite words

$$w = \quad (a)-(a)-(a)-(a)-(b)-(c)-(c)-(b)- \cdots \quad \in A^\omega$$

ω-language defined by ψ — the same definition

$$\mathrm{L}(\psi) = \{w \in A^\omega : \ w \text{ satisfies } \psi\}\,.$$

Büchi automata

A run of a Büchi automaton is accepting if it visits a final state infinitely many times.

Parity automata

Each state q has priority $\Omega(q) \in \mathbb{N}$.
A run ρ of a parity automaton is accepting if:

the highest priority visited infinitely often is even.

Equivalence

Theorem (Büchi [1962], McNaughton [1966], . . .)

For $L \subseteq A^\omega$ the following conditions are (effectively) equivalent:

- $L = \mathrm{L}(\psi)$ *for an MSO formula* ψ*,*

Equivalence

Theorem (Büchi [1962], McNaughton [1966], . . .)

For $L \subseteq A^\omega$ the following conditions are (effectively) equivalent:

- $L = \mathrm{L}(\psi)$ *for an MSO formula* ψ*,*
- $L = \mathrm{L}(\mathcal{B})$ *for a non-deterministic Büchi automaton* $\mathcal{B}$*,*

Equivalence

Theorem (Büchi [1962], McNaughton [1966], . . .)

For $L \subseteq A^\omega$ the following conditions are (effectively) equivalent:

- $L = \mathrm{L}(\psi)$ *for an MSO formula* ψ*,*
- $L = \mathrm{L}(\mathcal{B})$ *for a non-deterministic Büchi automaton* $\mathcal{B}$*,*
- $L = \mathrm{L}(\mathcal{A})$ *for a deterministic parity automaton* $\mathcal{A}$*.*

Equivalence

Theorem (Büchi [1962], McNaughton [1966], . . .)

For $L \subseteq A^\omega$ the following conditions are (effectively) equivalent:

- $L = \mathrm{L}(\psi)$ *for an MSO formula* ψ*,*
- $L = \mathrm{L}(\mathcal{B})$ *for a non-deterministic Büchi automaton* $\mathcal{B}$*,*
- $L = \mathrm{L}(\mathcal{A})$ *for a deterministic parity automaton* $\mathcal{A}$*.*

Such a language is called ω*-regular.*

Equivalence

Theorem (Büchi [1962], McNaughton [1966], . . .)

For $L \subseteq A^\omega$ the following conditions are (effectively) equivalent:

- $L = \mathrm{L}(\psi)$ *for an MSO formula* ψ*,*
- $L = \mathrm{L}(\mathcal{B})$ *for a non-deterministic Büchi automaton* $\mathcal{B}$*,*
- $L = \mathrm{L}(\mathcal{A})$ *for a deterministic parity automaton* $\mathcal{A}$*.*

Such a language is called ω-regular.

Original proof (Büchi)

Only Büchi automata, no determinization, direct complementation:

$$\text{given } \mathcal{A} \text{ compute } \mathcal{B} \text{ such that } \mathrm{L}(\mathcal{B}) = A^\omega \setminus \mathrm{L}(\mathcal{A}).$$

Equivalence

Theorem (Büchi [1962], McNaughton [1966], . . .)

For $L \subseteq A^\omega$ the following conditions are (effectively) equivalent:

- $L = \mathrm{L}(\psi)$ *for an MSO formula* ψ*,*
- $L = \mathrm{L}(\mathcal{B})$ *for a non-deterministic Büchi automaton* $\mathcal{B}$*,*
- $L = \mathrm{L}(\mathcal{A})$ *for a deterministic parity automaton* $\mathcal{A}$*.*

Such a language is called ω*-regular.*

Modern proof — determinization

Given a non-deterministic Büchi automaton compute an equivalent deterministic parity automaton.

Equivalence

Theorem (Büchi [1962], McNaughton [1966], ...)

For $L \subseteq A^\omega$ the following conditions are (effectively) equivalent:

- $L = \mathrm{L}(\psi)$ *for an MSO formula* ψ,
- $L = \mathrm{L}(\mathcal{B})$ *for a non-deterministic Büchi automaton* $\mathcal{B}$,
- $L = \mathrm{L}(\mathcal{A})$ *for a deterministic parity automaton* $\mathcal{A}$.

Such a language is called ω-regular.

Modern proof — determinization

Given a non-deterministic Büchi automaton compute an equivalent deterministic parity automaton.

Problems

Powerset construction is not enough to determinize!
Deterministic Büchi automata are too weak.
Determinization procedure is complicated.

Equivalence continued

Using non-deterministic Büchi automata and complementation

Decidability of $\mathrm{L}(\psi) \neq \emptyset$.

Equivalence continued

Using non-deterministic Büchi automata and complementation

Decidability of $\mathrm{L}(\psi) \neq \emptyset$.

Every MSO formula over infinite words is equivalent to:

- an existential formula $(\exists_{\vec{X}}\ (\ldots))$,
- a universal formula $(\forall_{\vec{X}}\ (\ldots))$.

Equivalence continued

Using non-deterministic Büchi automata and complementation

Decidability of $\mathrm{L}(\psi) \neq \emptyset$.
Every MSO formula over infinite words is equivalent to:

- an existential formula ($\exists_{\vec{X}}\ (\ldots)$),
- a universal formula ($\forall_{\vec{X}}\ (\ldots)$).

Verification of interactive systems

- model a system as a finite automaton,
- write an MSO formula ψ specifying allowed behaviours,
- construct an automaton $\mathcal{B}$ recognising bad behaviours
- check for emptiness of $\mathcal{B}$.

Equivalence continued

Using non-deterministic Büchi automata and complementation

Decidability of $\mathrm{L}(\psi) \neq \emptyset$.
Every MSO formula over infinite words is equivalent to:

- an existential formula ($\exists_{\vec{X}} (\ldots)$),
- a universal formula ($\forall_{\vec{X}} (\ldots)$).

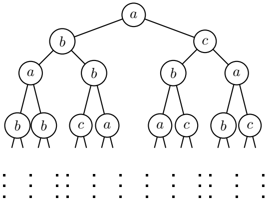
Verification of interactive systems

- model a system as a finite automaton,
- write an MSO formula ψ specifying allowed behaviours,
- construct an automaton $\mathcal{B}$ recognising bad behaviours
- check for emptiness of $\mathcal{B}$.

Using deterministic parity automata

Even more: memoryless winning strategies, Wagner hierarchy, . . .

Infinite trees — labellings of the full binary tree

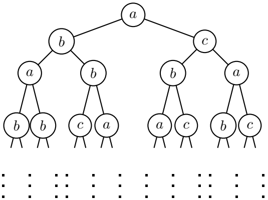


Infinite trees — labellings of the full binary tree

Trees are expressive

One infinite tree can encode:

- an arbitrary set of finite words,

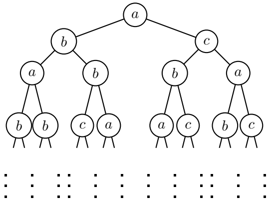


Infinite trees — labellings of the full binary tree

Trees are expressive

One infinite tree can encode:

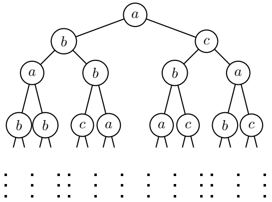
- an arbitrary set of finite words,
- all futures of a non-deterministic system,



Trees are expressive

One infinite tree can encode:

- an arbitrary set of finite words,
- all futures of a non-deterministic system,
- a strategy in an infinite-duration game.

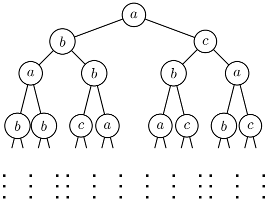


Infinite trees — labellings of the full binary tree

Trees are expressive

One infinite tree can encode:

- an arbitrary set of finite words,
- all futures of a non-deterministic system,
- a strategy in an infinite-duration game.



Regular tree languages?

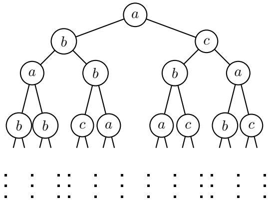
Those definable in MSO logic.

Infinite trees — labellings of the full binary tree

Trees are expressive

One infinite tree can encode:

- an arbitrary set of finite words,
- all futures of a non-deterministic system,
- a strategy in an infinite-duration game.



Regular tree languages?

Those definable in MSO logic.

What about decidability?

Given ψ check if there is a tree satisfying ψ?

Mother of all decidability results

Theorem (Rabin [1969])

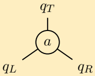
The satisfiability problem is decidable for MSO formulæ over infinite trees.

Mother of all decidability results

Theorem (Rabin [1969])

The satisfiability problem is decidable for MSO formulæ over infinite trees.

Proof — non-deterministic tree automata

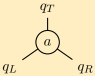
Branching transitions: 

ρ is accepting if it satisfies the acceptance condition on all infinite branches of the tree.

Theorem (Rabin [1969])

The satisfiability problem is decidable for MSO formulæ over infinite trees.

Proof — non-deterministic tree automata

Branching transitions: 

ρ is accepting if it satisfies the acceptance condition on all infinite branches of the tree.

$$\mathrm{L}(\mathcal{A}) = \left\{ t : \underset{\text{run } \rho}{\exists} \ \underset{\text{branch } \pi}{\forall} \ \rho \text{ is accepting on } \pi \right\}$$

No simple automata for infinite trees

Theorem (Rabin [1969], Emerson&Jutla [1991], Mostowski [1991])

The following conditions are (effectively) equivalent for a language of trees L:

- $L = \mathrm{L}(\psi)$ *for an MSO formula over trees* ψ,

No simple automata for infinite trees

Theorem (Rabin [1969], Emerson&Jutla [1991], Mostowski [1991])

The following conditions are (effectively) equivalent for a language of trees L:

- $L = \mathrm{L}(\psi)$ *for an MSO formula over trees* ψ*,*
- $L = \mathrm{L}(\mathcal{B})$ *for a non-deterministic parity tree automaton* $\mathcal{B}$*.*

No simple automata for infinite trees

Theorem (Rabin [1969], Emerson&Jutla [1991], Mostowski [1991])

The following conditions are (effectively) equivalent for a language of trees L:

- $L = \mathrm{L}(\psi)$ *for an MSO formula over trees* ψ*,*
- $L = \mathrm{L}(\mathcal{B})$ *for a non-deterministic parity tree automaton* $\mathcal{B}$*.*

Such L is called regular tree language.

No simple automata for infinite trees

Theorem (Rabin [1969], Emerson&Jutla [1991], Mostowski [1991])

The following conditions are (effectively) equivalent for a language of trees L:

- $L = \mathrm{L}(\psi)$ *for an MSO formula over trees* ψ*,*
- $L = \mathrm{L}(\mathcal{B})$ *for a* **non-deterministic** *parity tree automaton* $\mathcal{B}$*.*

Such L is called regular tree language.

It is the best we can have...

1) Deterministic top–down automata are strictly weaker.

No simple automata for infinite trees

Theorem (Rabin [1969], Emerson&Jutla [1991], Mostowski [1991])

The following conditions are (effectively) equivalent for a language of trees L:

- $L = \mathrm{L}(\psi)$ *for an MSO formula over trees* ψ*,*
- $L = \mathrm{L}(\mathcal{B})$ *for a non-deterministic* **parity** *tree automaton* $\mathcal{B}$*.*

Such L is called regular tree language.

It is the best we can have. . .

1) Deterministic top–down automata are strictly weaker.
2) Non-deterministic **Büchi** tree automata do not recognise all regular tree languages.

No simple automata for infinite trees

Theorem (Rabin [1969], Emerson&Jutla [1991], Mostowski [1991])

The following conditions are (effectively) equivalent for a language of trees L:

- $L = \mathrm{L}(\psi)$ *for an MSO formula over trees* ψ,
- $L = \mathrm{L}(\mathcal{B})$ *for a non-deterministic parity tree automaton* $\mathcal{B}$.

Such L is called regular tree language.

It is the best we can have...

1) Deterministic top–down automata are strictly weaker.
2) Non-deterministic Büchi tree automata do not recognise all regular tree languages.

Proof of 2)

Pumping argument **OR** descriptive complexity argument...

Descriptive complexity

Idea

How many set quantifiers need to appear in a definition of $L \subseteq X$:

$$L = \{x : \varphi\}?$$

(We don't restrict φ to MSO, it can be any formula of arithmetic.)

Descriptive complexity

Idea

How many set quantifiers need to appear in a definition of $L \subseteq X$:

$$L = \{x : \varphi\}?$$

(We don't restrict φ to MSO, it can be any formula of arithmetic.)

How to compare descriptive complexity of languages?

As for **NP**: known complete sets and appropriate reductions.

Descriptive complexity

Idea

How many set quantifiers need to appear in a definition of $L \subseteq X$:

$$L = \{x : \varphi\}?$$

(We don't restrict φ to MSO, it can be any formula of arithmetic.)

How to compare descriptive complexity of languages?

As for **NP**: known complete sets and appropriate reductions.

Better than NP

The hierarchy is strict: for every n there is a set L that requires n set quantifiers: $\underbrace{\exists\forall\ldots\exists\forall}_{n}$.

Upper bounds

Intuition

Any machine recognising L gives an upper bound on the descriptive complexity of L.

Upper bounds

Intuition

Any machine recognising L gives an upper bound on the descriptive complexity of L.

Example

If $\mathcal{M}$ is a non-deterministic ω-word machine with a Borel acceptance condition then

$$\mathrm{L}(\mathcal{M}) = \{w : \exists_\rho \, (\rho \text{ is a run}) \wedge (\rho \text{ is accepting})\}$$

Upper bounds

Intuition

Any machine recognising L gives an upper bound on the descriptive complexity of L.

Example

If $\mathcal{M}$ is a non-deterministic ω-word machine with a Borel acceptance condition then

$$\mathrm{L}(\mathcal{M}) = \{w :\ \exists_\rho \underbrace{(\rho \text{ is a run})}_{\text{Borel}} \wedge \underbrace{(\rho \text{ is accepting})}_{\text{Borel}}\}$$

Upper bounds

Intuition

Any machine recognising L gives an upper bound on the descriptive complexity of L.

Example

If $\mathcal{M}$ is a non-deterministic ω-word machine with a Borel acceptance condition then

$$\mathrm{L}(\mathcal{M}) = \{w :\ \underbrace{\exists_\rho\, \underbrace{(\rho \text{ is a run})}_{\text{Borel}} \wedge \underbrace{(\rho \text{ is accepting})}_{\text{Borel}}}_{\text{existential}}\}$$

Upper bounds

Intuition

Any machine recognising L gives an upper bound on the descriptive complexity of L.

Example

If $\mathcal{M}$ is a non-deterministic ω-word machine with a Borel acceptance condition then

$$\mathrm{L}(\mathcal{M}) = \{w : \underbrace{\exists_\rho\, (\rho \text{ is a run}) \wedge (\rho \text{ is accepting})}_{\text{existential}}\}$$

Note

All additional features allowed in $\mathcal{M}$: counters, stacks, tapes, ...

The other way round

Research plan

Look for examples of languages requiring many set quantifiers.
Derive some undefinability / undecidability results.

The other way round

Research plan

Look for examples of languages requiring many set quantifiers.
Derive some undefinability / undecidability results.

Playground

Extensions of MSO logic.

The other way round

Research plan

Look for examples of languages requiring many set quantifiers.
Derive some undefinability / undecidability results.

Playground

Extensions of MSO logic.

Behind the scenes

Theorem (Gurevich, Shelah [1982])

The MSO theory of $(\mathbb{R}, <)$ *is undecidable.*

The other way round

Research plan

Look for examples of languages requiring many set quantifiers.
Derive some undefinability / undecidability results.

Playground

Extensions of MSO logic.

Behind the scenes

Theorem (Gurevich, Shelah [1982])

The MSO theory of $(\mathbb{R}, <)$ *is undecidable.*

Conjecture (Shelah [1975])

The MSO theory of $(\mathbb{R}, <, \mathbf{Borel})$ *where set quantifiers range over Borel subsets of* $\mathbb{R}$ *is decidable.*

The other way round — a classical example

Theorem (Rabin)

There is a regular tree language that is not recognised by any non-deterministic Büchi tree automaton.

The other way round — a classical example

Theorem (Rabin)

There is a regular tree language that is not recognised by any non-deterministic Büchi tree automaton.

Original proof

A pumping argument.

The other way round — a classical example

Theorem (Rabin)

There is a regular tree language that is not recognised by any non-deterministic Büchi tree automaton.

Modern proof

Show that if $\mathcal{B}$ is a non-deterministic Büchi tree automaton then there is an existential formula of arithmetic φ such that

$$\mathrm{L}(\mathcal{B}) = \{t : \ \varphi\}.$$

The other way round — a classical example

Theorem (Rabin)

There is a regular tree language that is not recognised by any non-deterministic Büchi tree automaton.

Modern proof

Show that if $\mathcal{B}$ is a non-deterministic Büchi tree automaton then there is an existential formula of arithmetic φ such that

$$\mathrm{L}(\mathcal{B}) = \{t : \varphi\}.$$

Find a regular tree language that requires a universal set quantifier.

The other way round — a classical example

Theorem (Rabin)

There is a regular tree language that is not recognised by any non-deterministic Büchi tree automaton.

Modern proof

Show that if $\mathcal{B}$ is a non-deterministic Büchi tree automaton then there is an existential formula of arithmetic φ such that

$$\mathrm{L}(\mathcal{B}) = \{t :\ \varphi\}.$$

Find a regular tree language that requires a universal set quantifier. For example:

$$\left\{ t : \ \forall_{\text{branch } \pi} \ \text{almost all letters on } \pi \text{ are } a \right\}$$

Our victim: asymptotic extensions of MSO

Our victim: asymptotic extensions of MSO

Idea (Bojańczyk [2004])

Extend MSO with an additional quantifier U such that

$$\mathsf{U}_X\ \psi(X) \iff \psi(X) \text{ holds for arbitrarily big finite sets } X.$$

Our victim: asymptotic extensions of MSO

Idea (Bojańczyk [2004])

Extend MSO with an additional quantifier U such that

$$\mathsf{U}_X\ \psi(X) \iff \psi(X) \text{ holds for arbitrarily big finite sets } X.$$

Related work (Bojańczyk, Colcombet, ...)

ω-B, ω-S, ω-BS automata, regular cost functions, domination games, asymptotic MSO, ...

Our victim: asymptotic extensions of MSO

Idea (Bojańczyk [2004])

Extend MSO with an additional quantifier U such that

$$\mathsf{U}_X\ \psi(X) \iff \psi(X) \text{ holds for arbitrarily big finite sets } X.$$

Related work (Bojańczyk, Colcombet, . . .)

ω-B, ω-S, ω-BS automata, regular cost functions, domination games, asymptotic MSO, . . .

Our victim: asymptotic extensions of MSO

Idea (Bojańczyk [2004])

Extend MSO with an additional quantifier U such that

$$\mathsf{U}_X\ \psi(X) \iff \psi(X) \text{ holds for arbitrarily big finite sets } X.$$

Related work (Bojańczyk, Colcombet, ...)

ω-B, ω-S, ω-BS automata, regular cost functions, domination games, asymptotic MSO, ...

Open problem [2004]

Is MSO+U decidable over infinite words / trees?

Our victim: asymptotic extensions of MSO

Idea (Bojańczyk [2004])

Extend MSO with an additional quantifier U such that

$$\mathsf{U}_X\ \psi(X) \iff \psi(X) \text{ holds for arbitrarily big finite sets } X.$$

Related work (Bojańczyk, Colcombet, ...)

ω-B, ω-S, ω-BS automata, regular cost functions, domination games, asymptotic MSO, ...

Open problem [2004]

Is MSO+U decidable over infinite words / trees?

Theorem (Bojańczyk, Toruńczyk [2012])

Satisfiability problem of weak MSO+U is decidable over infinite trees.

Round 1 (joint with Hummel)

Theorem (Hummel, S. [2010])

There is an ω-word language definable in MSO+U that requires at least one universal set quantifier.
This language is not recognised by any non-deterministic machine with a Borel acceptance condition.

Round 1 (joint with Hummel)

Theorem (Hummel, S. [2010])

There is an ω-word language definable in MSO+U that requires at least one universal set quantifier.
This language is not recognised by any non-deterministic machine with a Borel acceptance condition.

Theorem (Hummel, S. [2012])

For every n there is an ω-word language definable in MSO+U that requires at least n alternations of set quantifiers: $\underbrace{\exists\forall\ldots\exists\forall}_{n}$.

Round 1 (joint with Hummel)

Theorem (Hummel, S. [2010])

There is an ω-word language definable in MSO+U that requires at least one universal set quantifier.
This language is not recognised by any non-deterministic machine with a Borel acceptance condition.

Theorem (Hummel, S. [2012])

For every n there is an ω-word language definable in MSO+U that requires at least n alternations of set quantifiers: $\underbrace{\exists\forall\ldots\exists\forall}_{n}$.

Corollary

There is no model of alternating machines on ω-words with a fixed projective acceptance condition capturing MSO+U.

Round 2 (joint with Bojańczyk, Gogacz, and Michalewski)

Theorem (B., G., M., S. [2013] (unpublished))

Under the assumption that V=L, MSO+U logic is undecidable over infinite trees.

Round 2 (joint with Bojańczyk, Gogacz, and Michalewski)

Theorem (B., G., M., S. [2013] (unpublished))

Under the assumption that V=L, MSO+U logic is undecidable over infinite trees.

Set-theoretic swamp...

V=L states that we work in the Gödel's constructible universe of Set Theory.

Round 2 (joint with Bojańczyk, Gogacz, and Michalewski)

Theorem (B., G., M., S. [2013] (unpublished))

Under the assumption that V=L, MSO+U logic is undecidable over infinite trees.

Set-theoretic swamp...

V=L states that we work in the Gödel's constructible universe of Set Theory.
V=L has similar status to Continuum Hypothesis:

$$\text{(Set Theory has a model)} \Longrightarrow \text{(it has a model satisfying V=L)}$$

Round 2 (joint with Bojańczyk, Gogacz, and Michalewski)

Theorem (B., G., M., S. [2013] (unpublished))

Under the assumption that V=L, MSO+U logic is undecidable over infinite trees.

Set-theoretic swamp...

V=L states that we work in the Gödel's constructible universe of Set Theory.
V=L has similar status to Continuum Hypothesis:

(Set Theory has a model) $\Longrightarrow$ (it has a model satisfying V=L)

Corollary

If there exists a proof that MSO+U is decidable over infinite trees

then

Set Theory is inconsistent.

Round 2 (joint with Bojańczyk, Gogacz, and Michalewski)

Intermediate statement

If $\mathbf{\Lambda}$ is any extension of MSO that defines some ω-word language requiring 6 alternations of set quantifiers

then (assuming V=L)

the $\mathbf{\Lambda}$-theory of the full binary tree is undecidable.

Round 2 (joint with Bojańczyk, Gogacz, and Michalewski)

Intermediate statement

If $\mathbf{\Lambda}$ is any extension of MSO that defines some ω-word language requiring 6 alternations of set quantifiers

then (assuming V=L)

the $\mathbf{\Lambda}$-theory of the full binary tree is undecidable.

Proof.

By a reduction to Shelah's undecidability of MSO on $(\mathbb{R}, <)$. ∎

Round 2 (joint with Bojańczyk, Gogacz, and Michalewski)

Intermediate statement

If $\mathbf{\Lambda}$ is any extension of MSO that defines some ω-word language requiring 6 alternations of set quantifiers

then (assuming V=L)

the $\mathbf{\Lambda}$-theory of the full binary tree is undecidable.

Proof.

By a reduction to Shelah's undecidability of MSO on $(\mathbb{R}, <)$. ■

MSO+U fits ideally in — it defines ω-word languages requiring arbitrarily many set quantifiers.

Summary & further work

Making long story short. . .

Decidability

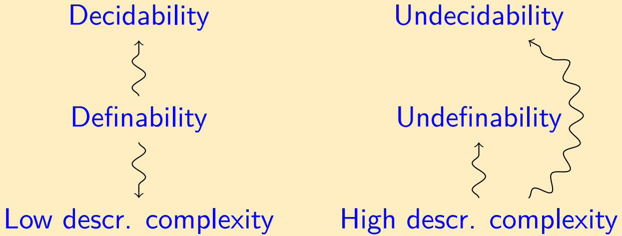
$\updownarrow$

Definability

$\updownarrow$

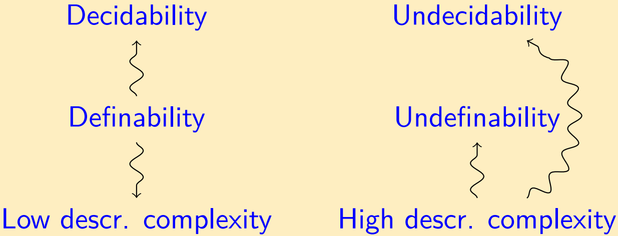
Low descr. complexity

Making long story short...



Summary & further work

Making long story short...



Getting rid of V=L

If there is no proof that MSO+U is decidable, there should be a direct proof of undecidability.

Summary & further work

Making long story short...

Decidability ⇝ Definability ⇝ Low descr. complexity

Undecidability ⇜ Undefinability ⇜ High descr. complexity

Getting rid of V=L

If there is no proof that MSO+U is decidable, there should be a direct proof of undecidability.

What about Borel quantifiers?

Rabin's theorem implies that $\mathrm{MSO}(\mathbb{R}, <, \mathbf{\Sigma}^0_2)$ is decidable.

Summary & further work

Making long story short...

Decidability ⟷ Definability ⟷ Low descr. complexity

Undecidability ⟵ Undefinability ⟵ High descr. complexity

Getting rid of V=L

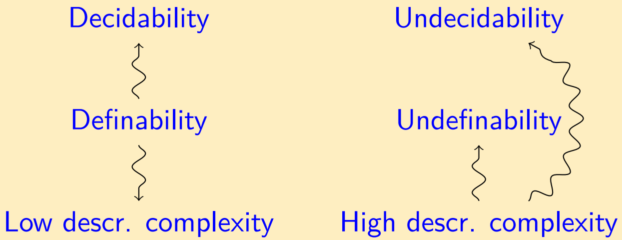
If there is no proof that MSO+U is decidable, there should be a direct proof of undecidability.

What about Borel quantifiers?

Rabin's theorem implies that $\mathrm{MSO}(\mathbb{R}, <, \boldsymbol{\Sigma}^0_2)$ is decidable.
What about $\mathrm{MSO}(\mathbb{R}, <, \boldsymbol{\Sigma}^0_3)$?

Summary & further work

Making long story short...



Getting rid of V=L

If there is no proof that MSO+U is decidable, there should be a direct proof of undecidability.

What about Borel quantifiers?

Rabin's theorem implies that $\mathrm{MSO}(\mathbb{R}, <, \mathbf{\Sigma}^0_2)$ is decidable.
What about $\mathrm{MSO}(\mathbb{R}, <, \mathbf{\Sigma}^0_3)$? Or $\mathrm{MSO}(\mathbb{R}, <, \mathbf{Borel})$?

Thank you for your attention!