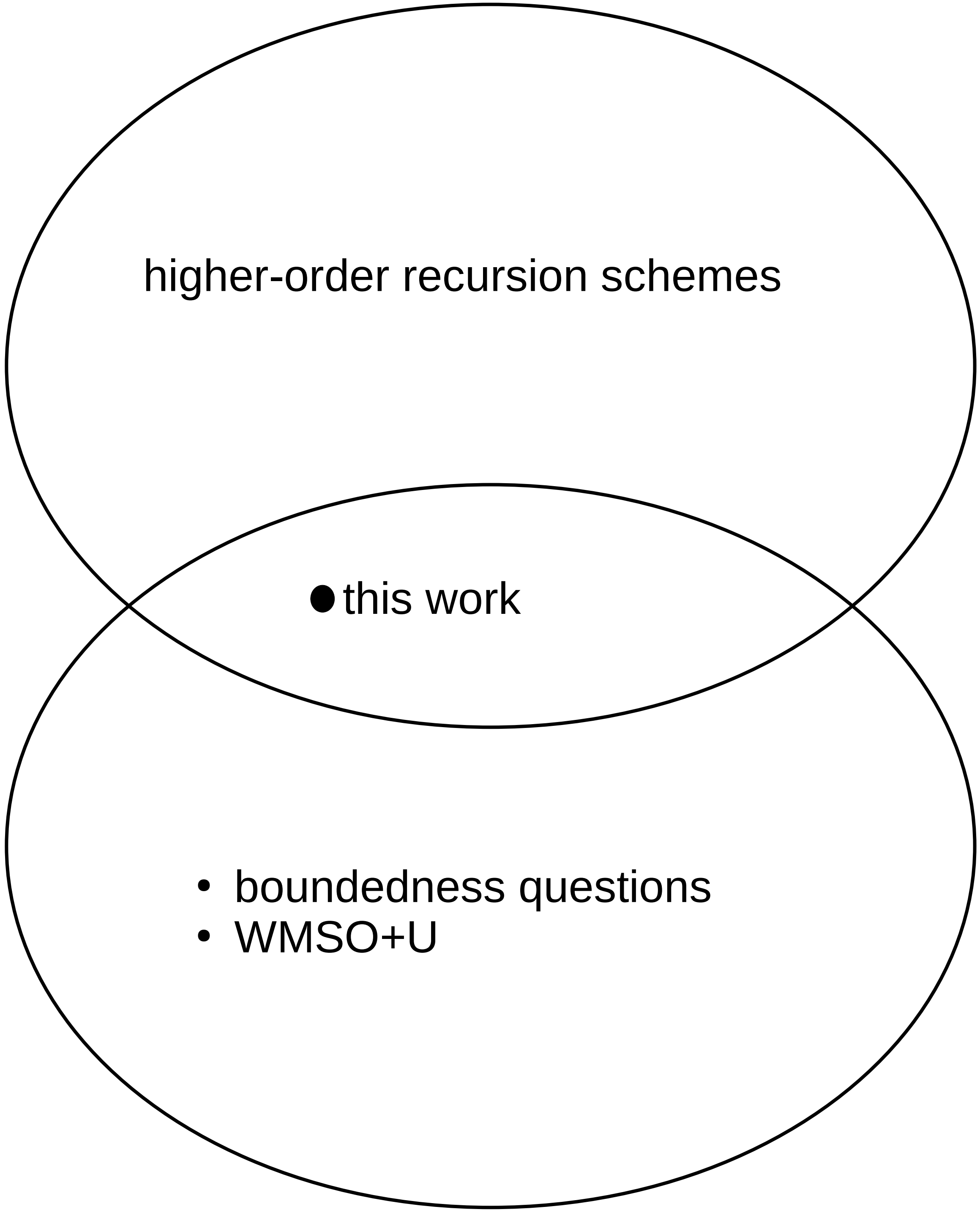
Extending the WMSO+U Logic with Quantification over Tuples

Anita Badyl **Paweł Parys**

University of Warsaw

CSL 2024


higher-order recursion schemes
this work
boundedness questions
WMSO+U

Higher-order recursion schemes – what is this?

Definition

Recursion schemes = simply-typed lambda-calculus + recursion

In other words:

- programs with recursion
- higher-order functions (i.e., functions taking other functions as parameters)
- every function/parameter has a fixed type
- no data values, only functions

Higher-order recursion schemes – example

fun f(x) {
    a(x);
    *if * then f(x);*
    b(x);
}
f(x)

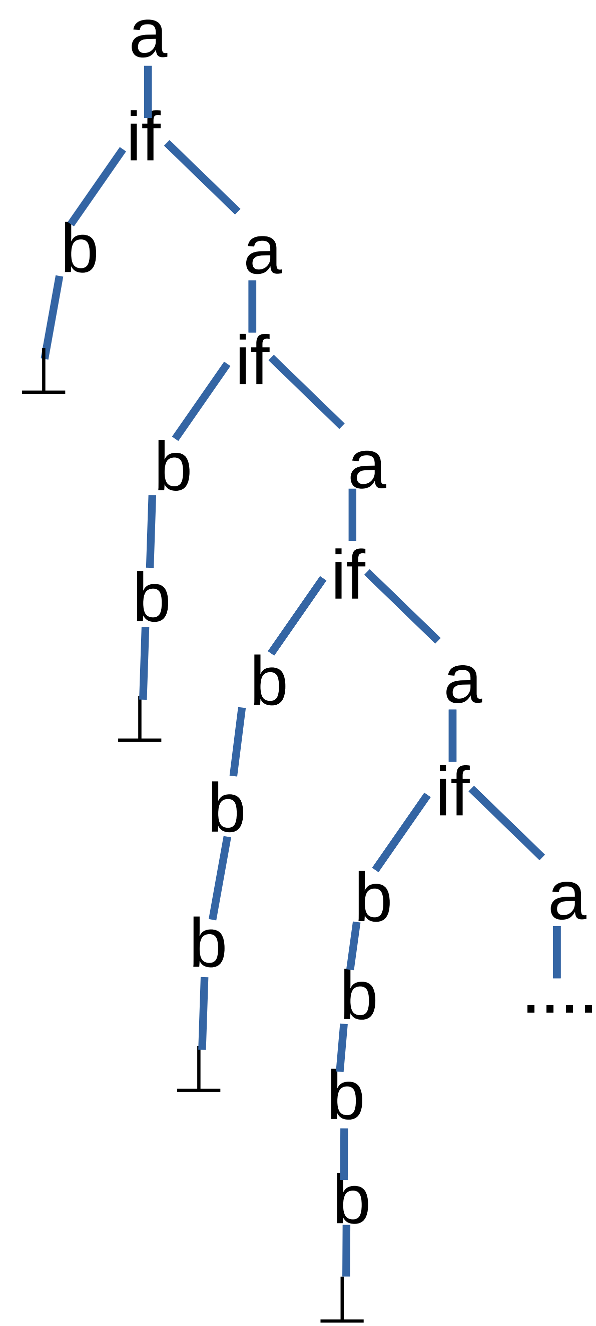
branching (we are not sure what will be chosen)

recursion

uniterpreted constants
(unknown functions)

Higher-order recursion schemes – example

```
fun f(x) {
    a(x);
    if * then f(x);
    b(x);
}
f(x)
```

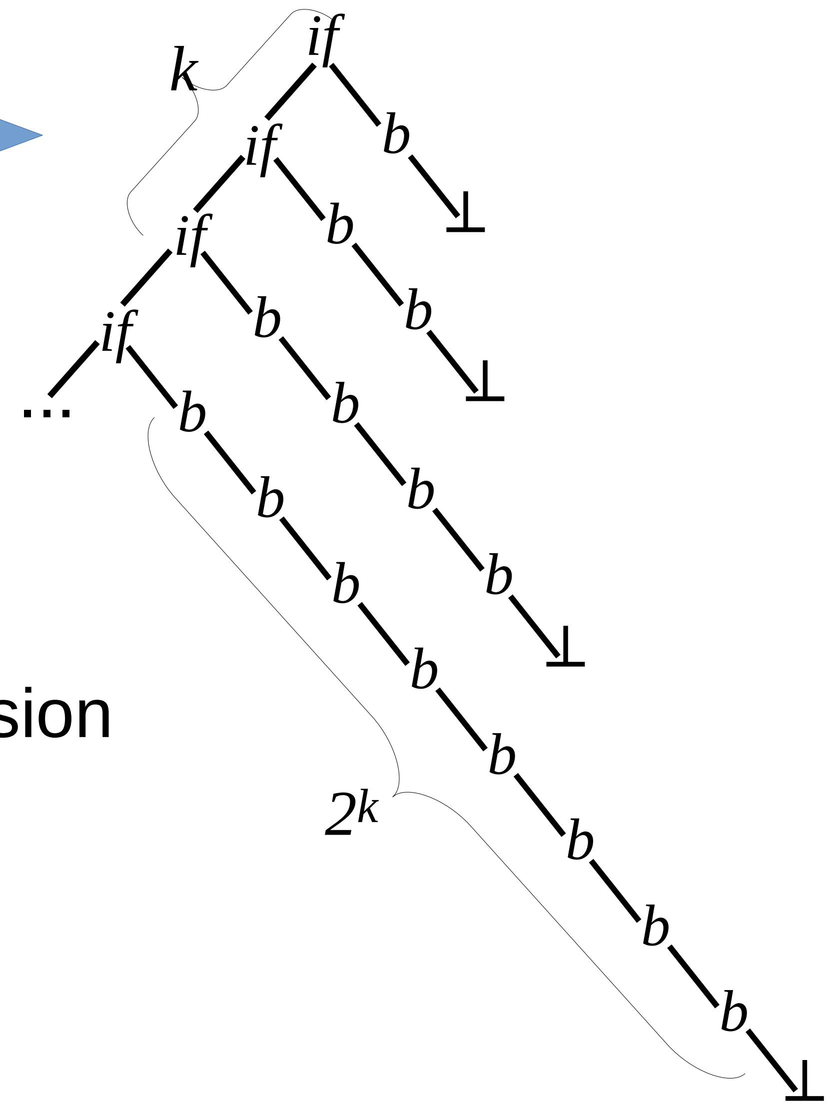


We are interested in trees representing the control flow of such programs.

Observation: these trees need not to be regular

Higher-order recursion schemes – example

```
fun A(f,x) {
   if * then A(D(f),x) else f(x);
}
fun D(f)(x) {
   f(x); f(x);
}
fun P(x) {
   b(x);
}
A(P,x)
```



This program uses higher-order recursion (passes functions as parameters)

Model-checking

Theorem [Ong 2006]
MSO model-checking on trees generated by recursion schemes is decidable.

Input: MSO formula ϕ, recursion scheme $\mathcal{G}$
Question: is ϕ true in the (infinite) tree generated by $\mathcal{G}$?

Model-checking

- a program in a functional programming language (e.g. OCAML)
- a property ψ

does the program satisfy ψ?

Model-checking

- a program in a functional programming language (e.g. OCAML)
- a property ψ

does the program satisfy ψ?

Approximation

ignore some details,

simulate some details using functions

- a recursion scheme $\mathcal{G}$
- a formula ϕ

is ϕ true in the tree generated by $\mathcal{G}$?

decidable

Model-checking

- a program in a functional programming language (e.g. OCAML)
- a property ψ

Approximation

ignore some details,

simulate some details using functions

- a recursion scheme $\mathcal{G}$
- a formula ϕ

does the program satisfy ψ?

- yes
- ?

is ϕ true in the tree generated by $\mathcal{G}$?

- yes
- no

decidable

There exist tools that take (short) programs in Ocaml and can verify some useful properties.

Can we go beyond MSO?

What about checking properties not expressible in MSO,
e.g., talking about boundedness?

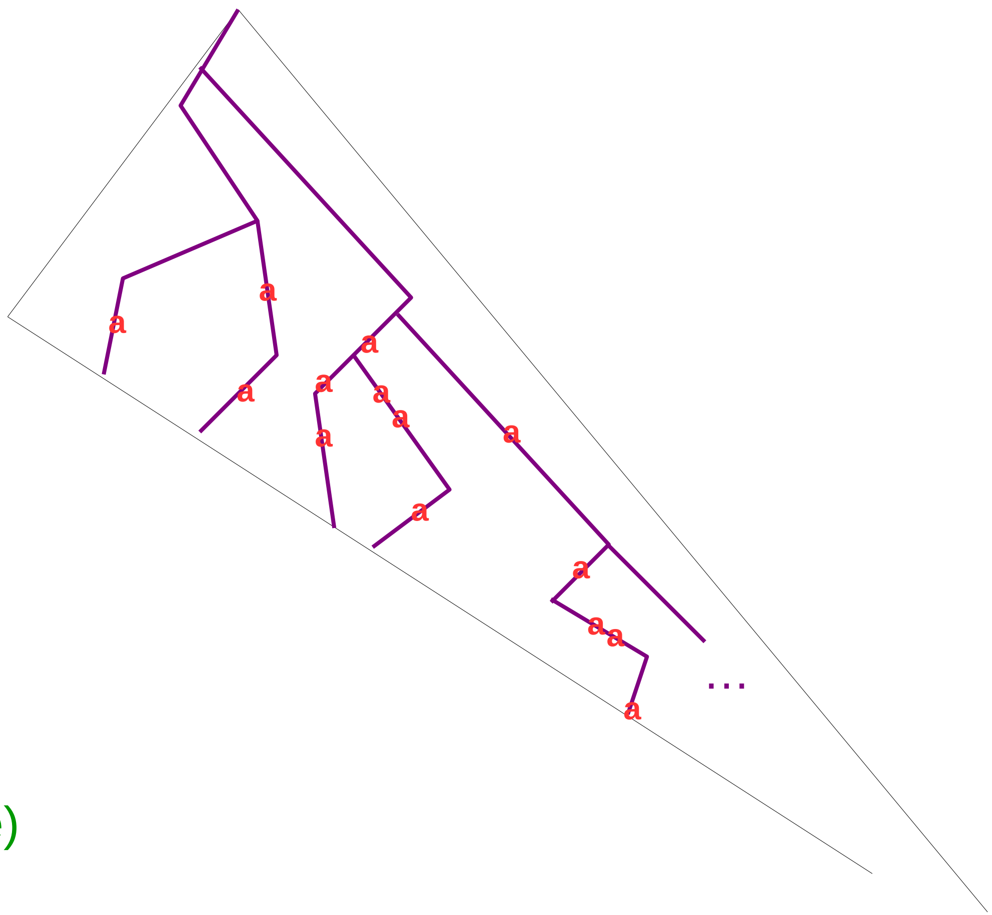
Can we go beyond MSO?

What about checking properties not expressible in MSO,
e.g., talking about boundedness?

Unboundedness – basic problem

Input: recursion scheme G, symbol a
Question: In the tree generated by G,
are there (finite) branches
with arbitrarily many occurrences
of symbol a?

($\forall n$ $\exists$branch with $>n$ occurrences of a)



Notice:
There may be no path with infinitely many „*a*".
This property **is not regular!!!**
(the result [Ong – LICS 2006] does not help here)

Can we go beyond MSO?

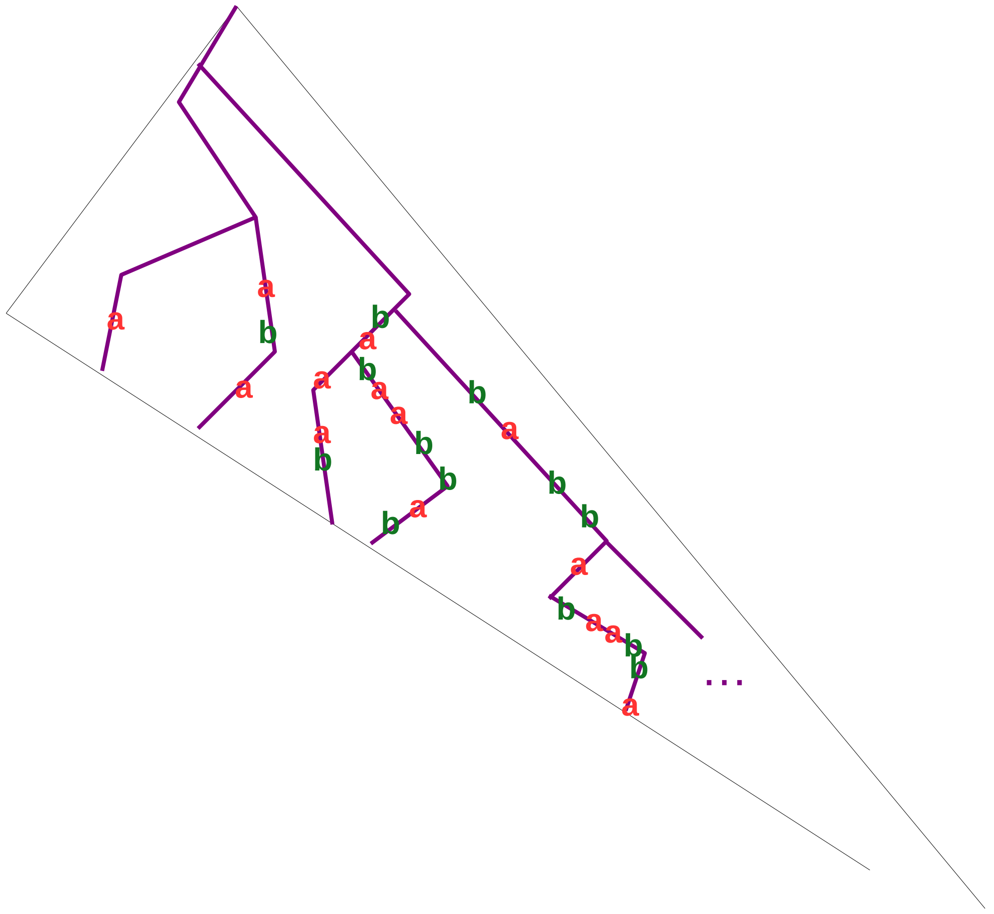
What about checking properties not expressible in MSO,
e.g., talking about boundedness?

Simultaneous unboundedness

Input: recursion scheme G, symbols $a_1,\ldots,a_k$

Question: In the tree generated by G,
are there (finite) branches
with arbitrarily many occurrences
of all symbols from $a_1,\ldots,a_k$?

($\forall n$ $\exists$branch $\forall i$ there are $>n$
occurrences of a_i on the branch)



Thm. This is decidable [Clemente, P., Salvati, Walukiewicz 2016]

Can we go beyond MSO?

What about checking properties not expressible in MSO,
e.g., talking about boundedness?

General approach - logic

We consider the WMSO+U logic.

“+U” = we add a new quantifier „U” [Bojańczyk, 2004]

$$UX.\phi(X)$$

$\phi(X)$ holds for finite sets of arbitrarily large size

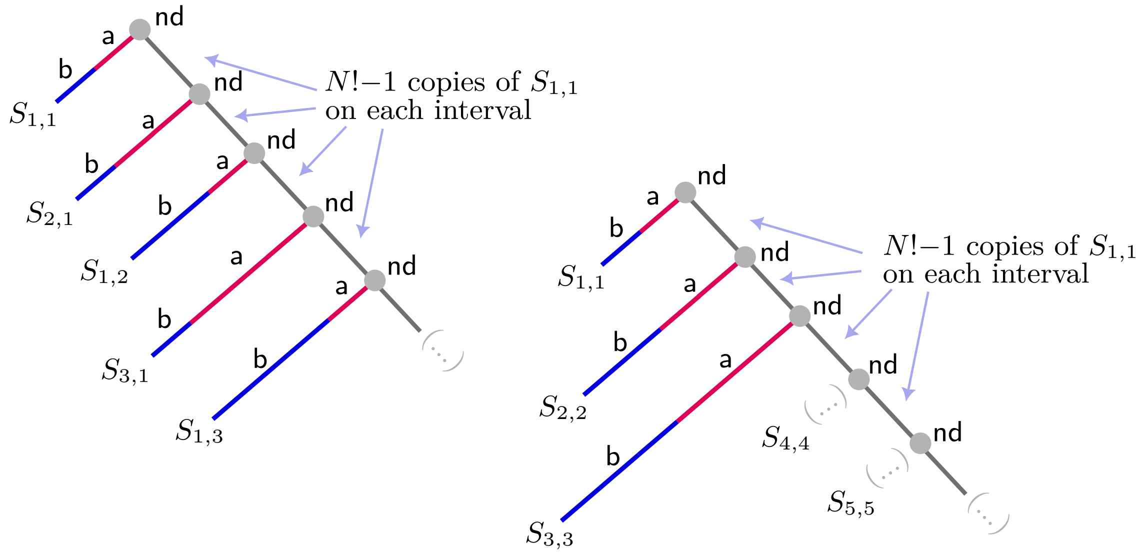
$$\forall n\in\mathbb{N}\ \exists X\ (\ n<|X|<\infty\ \wedge\ \phi(X)\ )$$

“W” = weak – we can quantify only over finite sets
($\exists X$ / $\forall X$ means: exists a finite set X / for all finite sets X)

Contribution 1

Can WMSO+U express the simultaneous unboundedness?

Thm. NO, it cannot. WMSO+U cannot distinguish the following trees:



Contribution 2 – more expressive logic

WMSO+U_{tup} – We add „U" quantification for tuples of sets

$$U(X_1,\ldots,X_k).\phi(X_1,\ldots,X_k)$$

$\phi(X)$ holds for tuples of arbitrarily large finite sets

$$\forall n\in\mathbb{N}\ \exists X_1\ldots\exists X_k\,(n<|X_1|,\ldots,|X_k|<\infty\ \wedge\ \phi(X_1,\ldots,X_k))$$

Note: This is different from saying $UX_1\ldots UX_k.\phi(X_1,\ldots,X_k)$

Contribution 2 – more expressive logic

WMSO+U_{tup} – We add „U" quantification for tuples of sets

$$U(X_1,\ldots,X_k).\phi(X_1,\ldots,X_k)$$

$\phi(X)$ holds for tuples of arbitrarily large finite sets

$$\forall n\in\mathbb{N}\ \exists X_1\ldots\exists X_k\ (n<|X_1|,\ldots,|X_k|<\infty\ \land\ \phi(X_1,\ldots,X_k))$$

Note: This is different from saying $UX_1\ldots UX_k.\phi(X_1,\ldots,X_k)$

Thm. The following problem is decidable:

Input: recursion scheme *G*, WMSO+U_{tup} sentence ϕ
Question: Is ϕ true in the tree generated by *G*?

About the proof

Theorem – the following problem is decidable:
input: formula ϕ, recursion scheme $\mathcal{G}$,
question: is ϕ true in the tree generated by $\mathcal{G}$?

Key ingredients:

- decidability of “simultaneous unboundedness” for HORSes:
  input: recursion scheme $\mathcal{G}$, letters $a_1,\dots,a_k$
  question: are there paths with arbitrarily many occurrences of all letters from $a_1,\dots,a_k$ in the tree generated by $\mathcal{G}$?
  [Hague, Kochems, Ong 2016, Clemente, P., Salvati, Walukiewicz 2016]

- „reflection” for simultaneous unboundedness: [P. 2017]
  input: recursion scheme $\mathcal{G}$, letters $a_1,\dots,a_k$
  output: recursion scheme $\mathcal{H}$, generating the same tree as $\mathcal{G}$, but with additional labels – in each node it is written whether simultaneous unboundedness wrt $a_1,\dots,a_k$ holds in this node

About the proof

Theorem – the following problem is decidable:
input: formula ϕ, recursion scheme $\mathcal{G}$,
question: is ϕ true in the tree generated by $\mathcal{G}$?

Key ingredients:

- „reflection" for simultaneous unboundedness: [P. 2017]
  input: recursion scheme $\mathcal{G}$, letters $a_1,\dots,a_k$
  output: recursion scheme $\mathcal{H}$, generating the same tree as $\mathcal{G}$, but with additional labels – in each node it is written whether simultaneous unboundedness wrt $a_1,\dots,a_k$ holds in this node

- „reflection" for (W)MSO: [Broadbent, Carayol, Ong, Serre 2010]
  input: recursion scheme $\mathcal{G}$, formula $\psi(x)\in$WMSO
  output: recursion scheme $\mathcal{H}$, generating the same tree as $\mathcal{G}$, but with additional labels – in each node it is written whether ψ holds in this node

- additionally: recursion schemes can be composed with finite tree transducers

About the proof

Theorem – the following problem is decidable:
input: formula ϕ, recursion scheme $\mathcal{G}$,
question: is ϕ true in the tree generated by $\mathcal{G}$?

Key ingredients:

- translation: formula $\Rightarrow$ sequence of operations $O_1, O_2, \ldots, O_k$
- Three kinds of operations:
  - apply reflection for simultaneous unboundedness
  - apply reflection for an MSO formula
  - apply finite tree transducer

About the proof

Theorem – the following problem is decidable:
input: formula ϕ, recursion scheme $\mathcal{G}$,
question: is ϕ true in the tree generated by $\mathcal{G}$?

Key ingredients:

- translation: formula $\Rightarrow$ sequence of operations $O_1, O_2, \ldots, O_k$
- Three kinds of operations:
  - apply reflection for simultaneous unboundedness
  - apply reflection for an MSO formula
  - apply finite tree transducer

Proof: by induction on the formula

How to deal with the $U(X_1,\ldots,X_k)$ quantifier?

Step 1: Create a variant of the tree "with all possible choices for $X_1,\ldots,X_k$" (on additional new branches below every node) - transducer

Step 2: Apply reflection for sim. unb. - check which choices are "good"

Step 3: Move the information to a correct place in the tree

Step 4: Remove additional branches

About the proof

Theorem – the following problem is decidable:
input: formula ϕ, recursion scheme $\mathcal{G}$,
question: is ϕ true in the tree generated by $\mathcal{G}$?

Key ingredients:

- translation: formula $\Rightarrow$ sequence of operations $O_1, O_2, ..., O_k$
- Three kinds of operations:
  - apply reflection for simultaneous unboundedness
  - apply reflection for an MSO formula
  - apply finite tree transducer

Then: For every O_i and rec. scheme $\mathcal{G}_i$ generating a tree t_i we create a rec. scheme $\mathcal{G}_{i+1}$ generating $t_{i+1}=O_i(t_i)$ (the effect of applying A_i to t_i), using appropriate theorem

About the proof

Theorem – the following problem is decidable:
input: formula ϕ, recursion scheme $\mathcal{G}$,
question: is ϕ true in the tree generated by $\mathcal{G}$?

Conclusion of the proof:

- The proof consists of a few (clearly separated) steps
- The technical difficulty is hidden in the “simultaneous unboundednes reflection” theorem

Summary

- Thm 1. WMSO+U can not express simultaneous unboundedness
- We introduce WMSO+U_{tup}, a logic which can express simultaneous unboundedness
- Thm 2. Given an WMSO+U_{tup} sentence ϕ and a recursion scheme G, we can decide whether ϕ holds in the tree generated by G

Thank you!