On two Notions of Higher-Order Model-Checking

Geocal Meeting

N. Kobayashi      E. Lozes      F. Bruse

November 29th, 2016
Two Higher-Order Extensions of Model-Checking

H. O. Recursion Schemes

- higher-order models
- functional programs verification
- model-checking is complicated

[Knapi&amp; al, 2001] [Ong, 2006]
[Haque&amp; al, 2008] [Kobayashi&amp; Ong, 2009]

H. O. Fixpoint Logic

- higher-order properties
- rely-guarantee reasoning
- non-regular properties
- model-checking is easy

[Viswanathan&amp; Viswanathan, 2004]
[Axelson,Lange,Somla, 2007] [Lange,Lozes, 2014]

How are they related?
Why the Question Matters

- we don’t have a **simple proof** of HORS decidability
  
  but if we can reduce HORS model-checking to HFL model-checking, we may give a new, simpler proof of the decidability of HORS model-checking.

- we don’t have an **efficient model-checker** for HFL
  
  but if we can reduce HFL model-checking to HORS model-checking, we can use existing HORS model-checkers.
A Simple Answer

**Theorem** [Ong, 2006] The HORS model-checking problem is $k$-EXPTIME complete at order $k$.

**Theorem** [Axelson, Lange, Somla, 2007] The HFL model-checking problem is $k$-EXPTIME complete at order $k$.

⇒ the two problems can be reduced one to each other.

But... encoding a $k$-EXPTIME Turing machine is not what we are looking for.
The Big Picture

recursion scheme \( \text{tree}(G) \) is accepted by altern. parity tree autom. \( \mathcal{A} \)

\[
S \models \varphi
\]

HFL formula

N. Kobayashi, E. Lozes, F. Bruse
On two Notions of Higher-Order Model-Checking
November 29th, 2016 5 / 31
The Big Picture

recursion scheme

$\text{tree}(G)$ is accepted by $A$

altern. parity tree autom.

$S \models \varphi$

HFL formula

N. Kobayashi, E. Lozes, F. Bruse
Recursion Schemes

\texttt{recursion scheme} \qquad \texttt{altern. parity tree autom.}

\texttt{tree}(G) \quad \texttt{is accepted by} \quad A

\begin{align*}
\mathcal{S} & \models \varphi \\
\text{ltz} & \quad \text{HFL formula}
\end{align*}
Recursion Schemes

terminals (order \( \leq 1 \))
\[
a : \ast \rightarrow \ast \rightarrow \ast \\
b : \ast \rightarrow \ast \\
c : \ast
\]

non-terminals
\[
S : \ast \\
F : (\ast \rightarrow \ast) \rightarrow \ast \\
B : (\ast \rightarrow \ast) \rightarrow \ast \rightarrow \ast
\]

rules
\[
S \rightarrow F b \\
F x \rightarrow a c (x (F (B b))) \\
B x y \rightarrow b (x y)
\]

reductions
\[
S \rightarrow F b \\
\quad \rightarrow a c (b (F (B b))) \\
\quad \rightarrow a c (b (a c (B b (F (B (B b))))) ) \\
\quad \rightarrow \ldots
\]

limit tree
\[
\begin{array}{c}
\text{a} \\
\text{c} \\
\text{b}
\end{array}
\quad \begin{array}{c}
\text{a} \\
\text{c} \text{b}^2 \\
\quad \begin{array}{c}
\text{a} \\
\text{c} \text{b}^3 \\
\ldots
\end{array}
\end{array}
\]
Alternating Parity Tree Automaton

recursion scheme

\( tree(G) \) is accepted by \( A \)

\( S \models \varphi \)

HFL formula

N. Kobayashi, E. Lozes, F. Bruse
On two Notions of Higher-Order Model-Checking
November 29th, 2016 8 / 31
Alternating Parity Tree Automaton

\[ A = (Q, \Sigma, \delta, q_0, \Omega) \]

- \( \delta(q, x) \in \text{Bool}^+(\text{Dir}(x) \times Q) \)
- \( \text{Dir}(x) = \{1, \ldots, \text{arity}(x)\} \)
- \( \Omega : Q \rightarrow \{0, \ldots, p - 1\} : \) priority function
- \( \Omega(q_0) = 0, \Omega(q_1) = 1 \)
acceptance game on a given tree $T$

- a play $\pi$ is a path of $T$ labeled with states
- parity condition: prover wins if
  - either $\pi$ is finite
  - or $\pi = s_0s_1\ldots s_i\ldots$ with $\limsup_{i \to \infty} \Omega(s_i)$ even

- $T \in L(\mathcal{A})$ if prover has a winning strategy
Alternating Parity Tree Automaton

\[
\begin{array}{c}
q_0 \\
\downarrow \\
q_1
\end{array}
\]

\[
\begin{array}{c}
0 \\
\downarrow \\
1
\end{array}
\]

\[
\begin{array}{c}
\bigwedge \\
\downarrow \\
\top
\end{array}
\]

\[
\begin{array}{c}
a \\
\downarrow \\
b
\end{array}
\]

\[
\begin{array}{c}
b \\
\downarrow \\
a
\end{array}
\]

\[
\begin{array}{c}
c \\
\downarrow \\
b
\end{array}
\]

\[
\begin{array}{c}
c \\
\downarrow \\
a
\end{array}
\]

\[
\begin{array}{c}
abq_0 \\
cq_0 \\
b
\end{array}
\]

\[
\begin{array}{c}
baq_1 \\
caq_0 \\
b
\end{array}
\]

\[
\begin{array}{c}
bcq_0 \\
b
\end{array}
\]

\[
\begin{array}{c}
b
\end{array}
\]

ex: accepting
Alternating Parity Tree Automaton

ex: non-accepting
Higher-Order Fixpoint Logic

recursion scheme

\[ \text{tree}(G) \]

is accepted by

\[ A \]

altern. parity tree autom.

\[ S \models \varphi \]

HFL formula
Higher-Order Fixpoint Logic

\[ \eta ::= \bullet | \eta_1 \to \eta_2 \quad \text{(simple types)} \]

\[ \varphi, \psi ::= \top | \bot \quad \text{(true,false)} \]
\[ \varphi \lor \psi \quad \text{(disjunction)} \]
\[ \varphi \land \psi \quad \text{(conjunction)} \]
\[ \langle a \rangle \varphi \quad \text{(may modality)} \]
\[ [a] \varphi \quad \text{(must modality)} \]
\[ X \quad \text{(variable)} \]
\[ \mu X^\eta. \varphi \quad \text{(h.o least fixed point)} \]
\[ \nu X^\eta. \varphi \quad \text{(h.o greatest fixed point)} \]
\[ \lambda X^\eta. \varphi \quad \text{(abstraction)} \]
\[ \varphi \psi \quad \text{(function application)} \]

remark: negation is admissible [Lozes, FICS’2015]
Examples

- predicate transformers

\[ \lambda X. p \lor \langle a \rangle X \quad \lambda X. \lambda Y. X \lor \langle a \rangle Y \]
Examples

- predicate transformers

\[ \lambda X. p \lor \langle a \rangle X \quad \lambda X. \lambda Y. X \lor \langle a \rangle Y \]

- higher-order predicate transformers

\[ \lambda F. \lambda X. F (F X) \]
Examples

- predicate transformers

\[ \lambda X. \; p \lor \langle a \rangle X \quad \lambda X. \; \lambda Y. \; X \lor \langle a \rangle Y \]

- higher-order predicate transformers

\[ \lambda F. \; \lambda X. \; F \left( F \; X \right) \]

- recursive predicate transformers

\[ \mu F. \; \lambda X. \; X \lor \bigvee_{a \in \Sigma} \langle a \rangle \left( F \left( \langle a \rangle \; X \right) \right) \]
Non-Regular Properties

The semantics of

\[ \mu F. \lambda X. X \lor \bigvee_{a \in \Sigma} \langle a \rangle (F (\langle a \rangle X)) \]

can be computed by its approximants

\[
\begin{align*}
F^0 X &= \bot \\
F^1 X &= X \\
F^2 X &= X \lor \bigvee_{a \in \Sigma} \langle a \rangle \langle a \rangle X \\
F^3 X &= F^2 X \lor \bigvee_{a,b \in \Sigma} \langle a \rangle \langle b \rangle \langle b \rangle \langle a \rangle X \\
\vdots \\
F^\omega X &= \bigvee \langle w \rangle X
\end{align*}
\]

palindrome \( w \)
Global Model-Checking

Recursion scheme $\text{tree}(G)$ is accepted by alternate parity tree automaton $A$.

$S \models \varphi$

$\text{Lts}$ \quad \text{HFL formula}
Global Model-Checking

- represent functions in extension
- compute fixpoints by their approximants

\[ \mu F. \lambda X. X \land [a]F \langle b\rangle X \]

<table>
<thead>
<tr>
<th>( X )</th>
<th>( F^0 \ X )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \emptyset )</td>
<td>( \emptyset )</td>
</tr>
<tr>
<td>( {s_0} )</td>
<td>( {s_0} )</td>
</tr>
<tr>
<td>( {s_1} )</td>
<td>( {s_1} )</td>
</tr>
<tr>
<td>( {s_0, s_1} )</td>
<td>( {s_0, s_1} )</td>
</tr>
</tbody>
</table>
Global Model-Checking

- represent functions in extension
- compute fixpoints by their approximants

\[ \mu F. \lambda X. X \wedge [a] F \langle b \rangle X \]

<table>
<thead>
<tr>
<th>X</th>
<th>( F^0 X )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \emptyset )</td>
<td>( \emptyset )</td>
</tr>
<tr>
<td>( {s_0} )</td>
<td></td>
</tr>
<tr>
<td>( {s_1} )</td>
<td></td>
</tr>
<tr>
<td>( {s_0, s_1} )</td>
<td></td>
</tr>
</tbody>
</table>
Global Model-Checking

- represent functions in extension
- compute fixpoints by their approximants

\[ \mu F. \lambda X. X \land [a]F \langle b \rangle X \]

<table>
<thead>
<tr>
<th>X</th>
<th>( F^0 X )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \emptyset )</td>
<td>( \emptyset )</td>
</tr>
<tr>
<td>( {s_0} )</td>
<td>( \emptyset )</td>
</tr>
<tr>
<td>( {s_1} )</td>
<td>( \emptyset )</td>
</tr>
<tr>
<td>( {s_0, s_1} )</td>
<td>---</td>
</tr>
</tbody>
</table>

N. Kobayashi, E. Lozes, F. Bruse
On two Notions of Higher-Order Model-Checking
November 29th, 2016 15 / 31
Global Model-Checking

- represent functions in extension
- compute fixpoints by their approximants

\[ \mu F. \lambda X. X \land [a]F \langle b \rangle X \]

<table>
<thead>
<tr>
<th>X</th>
<th>( F^0 X )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \emptyset )</td>
<td>( \emptyset )</td>
</tr>
<tr>
<td>( { s_0 } )</td>
<td>( \emptyset )</td>
</tr>
<tr>
<td>( { s_1 } )</td>
<td>( { s_1 } )</td>
</tr>
<tr>
<td>( { s_0, s_1 } )</td>
<td>( { s_0, s_1 } )</td>
</tr>
</tbody>
</table>
Global Model-Checking

- represent functions in extension
- compute fixpoints by their approximants

\[ \mu F. \lambda X. X \land [a]F \langle b \rangle X \]

<table>
<thead>
<tr>
<th>X</th>
<th>F^0 X</th>
<th>F^1 X</th>
</tr>
</thead>
<tbody>
<tr>
<td>\emptyset</td>
<td>\emptyset</td>
<td>\emptyset</td>
</tr>
<tr>
<td>{s_0}</td>
<td>\emptyset</td>
<td>{s_1}</td>
</tr>
<tr>
<td>{s_1}</td>
<td>{s_1}</td>
<td>{s_1}</td>
</tr>
<tr>
<td>{s_0, s_1}</td>
<td>{s_1}</td>
<td>{s_1}</td>
</tr>
</tbody>
</table>
Global Model-Checking

- represent functions in extension
- compute fixpoints by their approximants

\[
\mu F. \lambda X. X \land [a]F \langle b \rangle X
\]

<table>
<thead>
<tr>
<th></th>
<th>$F^0 X$</th>
<th>$F^1 X$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\emptyset$</td>
<td>$\emptyset$</td>
<td>$\emptyset$</td>
</tr>
<tr>
<td>{s_0}</td>
<td>$\emptyset$</td>
<td>{s_1}</td>
</tr>
<tr>
<td>{s_1}</td>
<td>{s_1}</td>
<td>{s_1}</td>
</tr>
<tr>
<td>{s_0, s_1}</td>
<td>{s_1}</td>
<td>{s_1}</td>
</tr>
</tbody>
</table>
Global Model-Checking

- represent functions in extension
- compute fixpoints by their approximants

\[
\begin{align*}
\mu F. \lambda X. X \land [a]F \langle b \rangle X
\end{align*}
\]

<table>
<thead>
<tr>
<th>$X$</th>
<th>$F^0 X$</th>
<th>$F^1 X$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\emptyset$</td>
<td>$\emptyset$</td>
<td>$\emptyset$</td>
</tr>
<tr>
<td>${s_0}$</td>
<td>$\emptyset$</td>
<td>${s_0}$</td>
</tr>
<tr>
<td>${s_1}$</td>
<td>${s_1}$</td>
<td>${s_0}$</td>
</tr>
<tr>
<td>${s_0, s_1}$</td>
<td>${s_1}$</td>
<td>${s_0}$</td>
</tr>
</tbody>
</table>
Global Model-Checking

- represent functions in extension
- compute fixpoints by their approximants

\[ \mu F. \lambda X. X \land [a]F \langle b \rangle X \]

<table>
<thead>
<tr>
<th>X</th>
<th>( F^0 X )</th>
<th>( F^1 X )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \emptyset )</td>
<td>( \emptyset )</td>
<td>( \emptyset )</td>
</tr>
<tr>
<td>( { s_0 } )</td>
<td>( \emptyset )</td>
<td>( { s_0 } )</td>
</tr>
<tr>
<td>( { s_1 } )</td>
<td>( { s_1 } )</td>
<td>( { s_1 } )</td>
</tr>
<tr>
<td>( { s_0, s_1 } )</td>
<td>( { s_1 } )</td>
<td></td>
</tr>
</tbody>
</table>
Global Model-Checking

- represent functions in extension
- compute fixpoints by their approximants

\[
\mu F. \lambda X. X \land [a]F \langle b \rangle X
\]

<table>
<thead>
<tr>
<th></th>
<th>$F^0 X$</th>
<th>$F^1 X$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\emptyset$</td>
<td>$\emptyset$</td>
<td>$\emptyset$</td>
</tr>
<tr>
<td>${s_0}$</td>
<td>$\emptyset$</td>
<td>${s_0}$</td>
</tr>
<tr>
<td>${s_1}$</td>
<td>${s_1}$</td>
<td>${s_1}$</td>
</tr>
<tr>
<td>${s_0, s_1}$</td>
<td>${s_1}$</td>
<td>${s_0, s_1}$</td>
</tr>
</tbody>
</table>
Global Model-Checking

- represent functions in extension
- compute fixpoints by their approximants

\[ \mu F. \lambda X. X \land [a]F \langle b \rangle X \]

<table>
<thead>
<tr>
<th></th>
<th>( F^0 X )</th>
<th>( F^1 X )</th>
<th>( F^2 X )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \emptyset )</td>
<td>( \emptyset )</td>
<td>( \emptyset )</td>
<td>( \emptyset )</td>
</tr>
<tr>
<td>( {s_0} )</td>
<td>( \emptyset )</td>
<td>( {s_0} )</td>
<td>( {s_0} )</td>
</tr>
<tr>
<td>( {s_1} )</td>
<td>( {s_1} )</td>
<td>( {s_1} )</td>
<td>( {s_1} )</td>
</tr>
<tr>
<td>( {s_0, s_1} )</td>
<td>( {s_1} )</td>
<td>( {s_0, s_1} )</td>
<td>( {s_0, s_1} )</td>
</tr>
</tbody>
</table>

\[ s_0 \quad a \quad s_1 \]

\[ b \]

N. Kobayashi, E. Lozes, F. Bruse

On two Notions of Higher-Order Model-Checking

November 29th, 2016 15 / 31
From HORS Model-Checking to HFL Model-Checking

recursion scheme

tree(\mathcal{G})
is accepted by

altern. parity tree autom.

\mathcal{A}

\models \varphi

\mathcal{S}

HFL formula

Its
From HORS Model-Checking to HFL Model-Checking

recursion scheme

\text{tree}(G)

is accepted by

\mathcal{A}

altern. parity tree autom.

\mathcal{S} \models \varphi

HFL formula
Encoding an automaton as a LTS

N. Kobayashi, E. Lozes, F. Bruse
On two Notions of Higher-Order Model-Checking
November 29th, 2016 17 / 31
Recursion Schemes as HFL Formulas

A recursion scheme $\text{tree}(G)$ is accepted by an alternating parity tree automaton $A$.

$S \models \varphi$

$\text{Its}$

HFL formula
Notation

The sequence $\mathcal{E} := X_1^{\eta_1} =_{\alpha_1} \varphi_1; \ldots; X_n^{\eta_n} =_{\alpha_n} \varphi_n$ stands for the formula $\text{toHFL}(\mathcal{E})$ defined as

$$\text{toHFL}(X^{\eta} =_{\alpha} \varphi) = \alpha X^{\eta}.\varphi$$
$$\text{toHFL}(\mathcal{E}; X^{\eta} =_{\alpha} \varphi) = \text{toHFL}([\alpha X^{\eta}.\varphi/X]\mathcal{E}).$$

example:

$$A =_{\mu} \langle a \rangle (B A); \quad \text{stands for} \quad \mu A. \langle a \rangle ((\nu B.\lambda X. \ A \lor \langle b \rangle X) \ A).$$

$$B =_{\nu} \lambda X. \ A \lor \langle b \rangle X$$

Note: in general, the order of the equations matters.
From HORS to HFL

naive idea

- for every rule
  \[ F \ x_1 \ldots x_n \rightarrow t \]

  introduce an equation

  \[ F =_\nu \ \lambda x_1 \ldots x_n. \ (t)^\dagger \]

  the formula \((t)^\dagger\) mimicks the term \(t\)

  - a non-terminal \(F\) becomes a recursive variable
  - a parameter \(x\) becomes a \(\lambda\)-bound variable
  - a terminal \(a\) becomes a formula that forces to move along the transition of the LTS that encodes the transitions \(\delta(\_, a)\) of the automaton.
Example

assume a is of arity 2

\[
S \rightarrow F a
\]
\[
F x \rightarrow x \ (F x) \ (F x)
\]

with

\[
\delta(q_i, a) = (1, q_1) \land (2, q_2)
\]
\[
\Omega(q_i) = 0
\]

becomes

\[
S \ = \_\nu \ F \ (\lambda x. \lambda y. \langle a_0 \rangle (\langle 1 \rangle x \land \langle 2 \rangle y))
\]
\[
F \ = \_\nu \ \lambda x. \ x \ (F x) \ (F x)
\]
Trivial Automata

An alternating parity tree automaton is *trivial* if $\Omega(q) = 0$ for all states $q$.

**Theorem**

Let $E(G)$ be the HES obtained by the naive translation of the HORS $G$. Let $A$ be a trivial APTA and let $S(A)$ be its associated LTS. Then

$$\text{tree}(G) \in L(A) \iff S(A) \models E(G)$$

**Issues:**

- how to deal with non-trivial automata?
- how to prove this theorem *simply*?
Main Technical Tool: HFL Typing Games

similar to Kobayashi-Ong typing games [Kobayashi,Ong, 2009] but a bit simpler

- no priorities in the intersection types
- simpler parity condition: the outermost recursive variable that is unfolded infinitely often determines the winner

\[
\tau ::= s \mid \tau_1 \land \cdots \land \tau_n \rightarrow \tau'
\]

- the type \( s \) refines the type \( \bullet \) of formulas that denote predicates
  \( \vdash \varphi : s \) if \( \varphi : \bullet \) and \( s \models \varphi \)
- \( \vdash \varphi : \tau_1 \land \cdots \land \tau_n \rightarrow \tau' \) if for all \( \psi \) such that \( \vdash \psi : \tau_i \) for all \( i = 1, \ldots, n \), it holds that \( \vdash \varphi \psi : \tau' \)
Example

\[
\begin{align*}
S &= \mu X; \\
Y &= \nu \lambda Z. \langle a \rangle (Z \land X); \\
X &= \mu \langle a \rangle (Y \land X).
\end{align*}
\]

with

\[
\begin{align*}
E(S) &: s \\
\frac{Y : s \rightarrow s \quad \langle a \rangle (Y \land X) : s}{\langle a \rangle (Y \land X) : s} \\
\frac{Z : s \vdash Z \land X : s}{Z : s \vdash \langle a \rangle (Z \land X) : s} \\
\frac{E(Y) : s \rightarrow s}{E(Y) : s \rightarrow s}
\end{align*}
\]
Ingredients of the Proof

**Theorem**

HFL typing games capture HFL semantics: \( \vdash \varphi : s \) is derivable (i.e. Prover has a winning strategy in the typing game) if and only if \( s \models \varphi \).

**Theorem**

The translation \( (\cdot)^{\dagger} \) preserves typability: for trivial automata \( \mathcal{A} \), \( \vdash (t)^{\dagger} : q \) in the HFL typing game iff \( \vdash t : q \) in the KO typing game for trivial automata.
A Taste of the Case of Non-Trivial Automata

Same idea, but in order to account for priorities
- non-terminals get duplicated
- arguments get duplicated

example

\[
S \rightarrow F \ b \\
F \ x \rightarrow x \ (F \ x)
\]

becomes

\[
S^{\#1} = _\mu F^{\#1} \ b^{\#1} \ b^{\#1};
\]
\[
F^{\#1} = _\mu \lambda X^{\#1}.\lambda X^{\#0}. \ X^{\#1} \ (F^{\#1} \ X^{\#1} \ X^{\#1}) \ (F^{\#0} \ X^{\#1} \ X^{\#0});
\]
\[
S^{\#0} = _\nu F^{\#0} \ b^{\#1} \ b^{\#0};
\]
\[
F^{\#0} = _\nu \lambda X^{\#1}.\lambda X^{\#0}. \ X^{\#0} \ (F^{\#1} \ X^{\#1} \ X^{\#1}) \ (F^{\#0} \ X^{\#1} \ X^{\#0})
\]
A Taste of the Case of Non-Trivial Automata (2)

Why argument duplication is needed can be illustrated at the level of types. Remember KO types [Kobayashi, Ong, 2009] are

\[ \theta ::= q \mid (\theta_1, m_1) \land \cdots \land (\theta_n, m_n) \rightarrow \theta \]

where \( m_i \) are priorities.

The translation relies on

- KO type \( q \) being mapped to HFL type \( q \)
- KO type

\[
\bigwedge_{j \in J_0} (\theta_j, 0) \land \cdots \land \bigwedge_{j \in J_p} (\theta_j, p) \rightarrow \theta
\]

being mapped to

\[
\bigwedge_{j \in J_0} \theta_j \rightarrow \cdots \rightarrow \bigwedge_{j \in J_p} \theta_j \rightarrow \theta
\]
From HFL Model-Checking to HORS Model-Checking

Recursion scheme \( \text{tree}(G) \) is accepted by altern. parity tree autom.

\( S \models \varphi \)

HFL formula
Main Ideas

- on LTS with $n$ states, a HFL formula $\varphi$ of order $k$ is equivalent to a non-recursive formula $\varphi^{(\alpha)}$ obtained by $\alpha = 2^n_k$ unfoldings
- we create a HORS that generates the syntax tree of $\varphi^{(\alpha)}$
- the APTA evaluates the syntax tree of the formula over the LTS.

- challenge: generate $\varphi^{(\alpha)}$ at order $k$:
- we used Jones encoding of large numbers [Jones, JFP 2001]
Conclusion

no free lunch today

- new proof of HORS MC decidability, but not really simpler (unless perhaps for trivial automata)
- not clear that HORS model-checkers can be used for HFL model-checking, because of our use of large numbers encoding
- not clear that we cannot do better for HFL→HORS

but

- interesting type system for HFL
- answers the question of local model-checking in HFL
- possibly more intuitive than original KO types
Related Work?

