On two notions of higher-order model checking

Naoki Kobayashi
University of Tokyo

Joint work with Etienne Lozes (ENS Cachan) and Florian Bruse (University of Kassel)
What's This Talk About?

- Relationship between two higher-order extensions of model checking:

<table>
<thead>
<tr>
<th>Models</th>
<th>Logic</th>
</tr>
</thead>
<tbody>
<tr>
<td>finite state model checking</td>
<td>finite state systems</td>
</tr>
</tbody>
</table>

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
What's This Talk About?

♦ Relationship between two higher-order extensions of model checking:
  - HORS model checking [Knapik+ 01; Ong 06]

<table>
<thead>
<tr>
<th></th>
<th>Models</th>
<th>Logic</th>
</tr>
</thead>
<tbody>
<tr>
<td>finite state</td>
<td>finite state systems</td>
<td>modal µ-calculus</td>
</tr>
<tr>
<td>model checking</td>
<td></td>
<td></td>
</tr>
<tr>
<td>HORS model checking</td>
<td>higher-order recursion schemes (HORS)</td>
<td>modal µ-calculus</td>
</tr>
</tbody>
</table>
What’s This Talk About?

- Relationship between two higher-order extensions of model checking:
  - HORS model checking [Knapik+ 01; Ong 06]
  - HFL model checking [Viswanathan&Viswanathan 04]

<table>
<thead>
<tr>
<th></th>
<th>Models</th>
<th>Logic</th>
</tr>
</thead>
<tbody>
<tr>
<td>finite state</td>
<td>finite state systems</td>
<td>modal (\mu)-calculus</td>
</tr>
<tr>
<td>model checking</td>
<td>HORS recursion schemes</td>
<td>HFL modal fixpoint logic</td>
</tr>
<tr>
<td>HORS model</td>
<td>higher-order recursion</td>
<td></td>
</tr>
<tr>
<td>checking</td>
<td>schemes (HORS)</td>
<td></td>
</tr>
<tr>
<td>HFL model</td>
<td>finite state systems</td>
<td></td>
</tr>
<tr>
<td>checking</td>
<td></td>
<td>higher-order modal fixpoint logic (HFL)</td>
</tr>
</tbody>
</table>
What's This Talk About?

♦ Relationship between two higher-order extensions of model checking:
  - HORS model checking [Knapik+ 01; Ong 06]
  - HFL model checking [Viswanathan&Viswanathan 04]

♦ Type-based characterization of HFL model checking
  - $L \models \psi$ if and only if $\not\models_L \psi$
Outline

- Reviews of HORS model checking and HFL model checking
  - HORS model checking
  - HFL model checking
- From HORS to HFL model checking
- From HFL to HORS model checking
- Type system for HFL model checking
- Conclusion
Higher-Order Recursion Scheme (HORS)

♦ Grammar for generating an infinite tree

Order-1 HORS

S → A c

A x → a x (A (b x))

S: o, A: o → o
Higher-Order Recursion Scheme (HORS)

Grammar for generating an infinite tree

Order-1 HORS

\[ S \rightarrow A \ c \]

\[ A \ x \rightarrow a \ x \ (A \ (b \ x)) \]

\[ S: \ o, \ A: \ o \rightarrow o \]

Tree whose paths are labeled by \( a^{m+1} b^m c \)
Higher-Order Recursion Scheme (HORS)

Grammar for generating an infinite tree

Order-1 HORS

\[ S \rightarrow A \ c \]
\[ A \ x \rightarrow a \ x \ (A \ (b \ x)) \]

S: o, A: o→ o

HORS \approx \text{Call-by-name simply-typed } \lambda\text{-calculus} + \text{recursion, tree constructors}
HORS Model Checking

Given

- $G$: HORS
- $A$: alternating parity tree automaton (APT) (a formula of modal $\mu$-calculus or MSO),

does $A$ accept $\text{Tree}(G)$?

E.g.

- Does every finite path end with “c”?
- Does “a” occur below “b”?

$k$-EXPTIME-complete [Ong, LICS06] (for order-$k$ HORS)
Outline

♦ Reviews of HORS model checking and HFL model checking
  - HORS model checking
  - HFL model checking
♦ From HORS to HFL model checking
♦ From HFL to HORS model checking
♦ Type system for HFL model checking
♦ Conclusion
Higher-Order Modal Fixpoint Logic (HFL) [Viswanathan&Viswanathan 04]

Higher-order extension of the modal μ-calculus

φ ::= true

\[ \varphi_1 \land \varphi_2 \]

\[ \varphi_1 \lor \varphi_2 \]

\[ [a] \varphi \quad \text{φ must hold after a} \]

\[ <a> \varphi \quad \text{φ may hold after a} \]

\[ X \quad \text{propositional variable} \]

\[ \mu X. \varphi \quad \text{least fixpoint} \]

\[ \nu X. \varphi \quad \text{greatest fixpoint} \]
Higher-Order Modal Fixpoint Logic (HFL) [Viswanathan&Viswanathan 04]

♦ Higher-order extension of the modal $\mu$-calculus

$\varphi ::= \text{true}$

$\varphi_1 \land \varphi_2$

$\varphi_1 \lor \varphi_2$

$[a] \varphi$ \quad $\varphi$ must hold after $a$

$\langle a \rangle \varphi$ \quad $\varphi$ may hold after $a$

$X$ \quad predicate variable

$\mu X^\kappa. \varphi$ \quad least fixpoint

$\nu X^\kappa. \varphi$ \quad greatest fixpoint

$\lambda X^\kappa. \varphi$ \quad (higher-order) predicate application

$\varphi_1 \varphi_2$

$\kappa ::= \bullet \mid \kappa_1 \rightarrow \kappa_2$
Selected Typing Rules for HFL

\[
\begin{align*}
\Gamma & \vdash \text{true}: \bullet \\
\Gamma & \vdash \varphi: \bullet \quad \Gamma \vdash \psi: \bullet \\
\Gamma & \vdash \varphi \land \psi: \bullet \\
\Gamma & \vdash \lambda X. \varphi: \kappa_1 \rightarrow \kappa_2 \\
\Gamma, X: \kappa_1 & \vdash \varphi: \kappa_2 \\
\Gamma & \vdash \mu X. \varphi: \kappa \\
\end{align*}
\]
Semantics

$[\varphi]_I$: the set of states that satisfy $\varphi$

$L \models \varphi \iff s_{\text{init}} \in [\varphi]_{\emptyset}$ ($s_{\text{init}}$: initial state of $L$)

$[\text{true}]_I = \text{States}$

$[\varphi \land \psi]_I = [\varphi]_I \cap [\psi]_I$

$[\langle \alpha \rangle \varphi]_I = \{s \mid \forall t.(s \rightarrow t \in [\varphi]_I)\}$

$[X]_I = I(X)$

$[\mu X^\kappa.\varphi]_I = \text{lfp}(\lambda x \in [\kappa].[\varphi]_{I\{X=x\}})$

$[\nu X^\kappa.\varphi]_I = \text{gfp}(\lambda x \in [\kappa].[\varphi]_{I\{X=x\}})$

(Note: $\lambda x \in [\kappa].[\varphi]_{I\{X=x\}}$ is monotonic)

$[\lambda X^\kappa.\varphi]_I = \lambda x \in [\kappa].[\varphi]_{I\{X=x\}}$

$[\varphi \psi]_I = [\varphi]_I [\psi]_I$

$[\bullet]_I = 2^{\text{States}}$

$[\kappa_1 \rightarrow \kappa_2] = \{f \in [\kappa_1] \rightarrow [\kappa_2] \mid f: \text{monotonic}\}$
Example

\[(\mu F \cdots \cdots \cdot \lambda X. \lambda Y. (X \land Y) \lor F (<a>X) (<b>Y)) \ A \ B\]

= \((A \land B) \lor (\mu F \cdots \cdots \cdot \lambda X. \lambda Y. (X \land Y) \lor F(<a>X)(<b>Y)) \ (<a>A)(<b>B)\)

= \((A \land B) \lor (<a>A \land <b>B) \lor (<a><a>A \land <b><b>B) \lor \ldots\)

For some \(n\), \(<a>^n A\) and \(<b>^n B\) hold.
**HFL Model Checking**

Given

L: (finite-state) labeled transition system

φ: HFL formula,

does L satisfy φ?

e.g. \( L \models \varphi \) for:

L: 

\[ \varphi: (\mu F. \lambda X. \lambda Y. (X \land Y) \lor F(<a>X)(<b>Y)) \land (<c>\text{true}) \land (<d>\text{true}) \]
## HORS vs HFL model checking

<table>
<thead>
<tr>
<th></th>
<th>Model</th>
<th>Spec.</th>
<th>complexity</th>
<th>Applications</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>HORS model checking</strong></td>
<td>HORS</td>
<td>APT</td>
<td>k-EXPTIME complete (for order-k HORS)</td>
<td>Automated verification of functional programs [K 09][K+11]...</td>
</tr>
<tr>
<td><strong>HFL model checking</strong></td>
<td>LTS</td>
<td>HFL</td>
<td>k-EXPTIME complete (for order-k HFL)</td>
<td>Assume-guarantee reasoning [VV 04] Process equivalence checking [Lange+ 14]</td>
</tr>
</tbody>
</table>

APT: alternating parity tree automaton  
LTS: finite-state labeled transition system
Hierarchical Equation Systems (HES)

\[ X_1 = _{\alpha_1} \varphi_1; \ldots; X_n = _{\alpha_n} \varphi_n \]

(\(\alpha_i \in \{\mu, \nu\}\) )

\[ \text{toHFL}(X = _{\alpha} \varphi) = _{\alpha} X.\varphi \]

\[ \text{toHFL}(H; X = _{\alpha} \varphi) = \text{toHFL}([_\alpha X.\varphi / X]H) \]

Example:

**HFL:** \( \nu X.\mu Y.(<a>X \lor <b>Y) \)

(there exists a path \((b*a)^{\omega})\)

**HES:** \( X = _{\nu} Y; Y = _{\mu} <a>X \lor <b>Y \)
Outline

♦ Reviews of HORS model checking and HFL model checking
  - HORS model checking
  - HFL model checking
♦ From HORS to HFL model checking
♦ From HFL to HORS model checking
♦ Type system for HFL model checking
♦ Conclusion
From HORS to HFL model checking

♦ **Input:**
  - HORS $G$
  - APT $A$ (with largest priority $p$)

♦ **Output:**
  - LTS $L_A$
  - HFL formula $\varphi_{G,p}$

such that $G \models A$ iff $L_A \models \varphi_{G,p}$

**Intuition:**
- $L_A$ simulates the transitions of $A$
- $\varphi_{G,p}$ describes “$L_A$ has transitions corresponding to an accepting run of $A$ over Tree($G$)”
Construction of $L_A$

$A$: 
\[ \delta(q_0,a) = (1,q_0) \land (2,q_0) \quad \delta(q_1,a) = \text{false} \]
\[ \delta(q_0,b) = \delta(q_1,b) = (1,q_1) \quad \delta(q_0,c) = \delta(q_1,c) = \text{true} \]
\[ \Omega(q_0) = 0 \quad \Omega(q_1) = 1 \]

$L_A$: 

![Diagram of $L_A$]
Construction of $L_A$

$A$: 
- $\delta(q_0,a) = (1,q_0) \land (2,q_0)$
- $\delta(q_0,b) = \delta(q_1,b) = (1,q_1)$
- $\Omega(q_0) = 0$
- $\Omega(q_1) = 1$

$L_A$: 
- The states of $L_A$ consist of:
  - states of $A$ and
  - subformulas of $\delta(q,a)$
Construction of $L_A$

$A$:  
\[ \delta(q_0, a) = (1, q_0) \land (2, q_0) \]
\[ \delta(q_0, b) = \delta(q_1, b) = (1, q_1) \]
\[ \Omega(q_0) = 0 \quad \Omega(q_1) = 1 \]
\[ \delta(q_1, a) = \text{false} \]
\[ \delta(q_0, c) = \delta(q_1, c) = \text{true} \]

$L_A$:  
A transition label is an input symbol annotated with a priority; or ...

![Diagram of $L_A$ with states $q_0$, $q_1$, transitions annotated with priorities, and labels $a_0$, $b_0$, $c_0$, $a_1$, $b_1$, $c_1$.]
Construction of $L_A$

$A$:  
\[ \begin{align*} 
\delta(q_0, a) &= (1, q_0) \land (2, q_0) \\
\delta(q_0, b) &= \delta(q_1, b) = (1, q_1) \\
\Omega(q_0) &= 0 \\
\Omega(q_1) &= 1 \\
\delta(q_1, a) &= \text{false} \\
\delta(q_0, c) &= \delta(q_1, c) = \text{true} \\
\delta(q_1, b) &= (1, q_1) \\
\Omega(q_0) &= 0 \\
\Omega(q_1) &= 1 
\end{align*} \]

$L_A$:  
A transition label is ...; or a constructor of transition formulas (i, and, or, tt)
Outline

- Reviews of HORS model checking and HFL model checking
- From HORS to HFL model checking
  - construction of $L_A$
  - construction of $\varphi_{G,p}$
    - case $p=0$
    - general case
- From HFL to HORS model checking
- Type system for HFL model checking
- Conclusion
From trees to HFL formulas

\( \varphi_T: \) “the current state has transitions corresponding to an accepting run for \( T \)”

\( \varphi_{a \ c \ (b \ c)} = \)
\(<a_0> \) “can visit 1\textsuperscript{st} and 2\textsuperscript{nd} children with states satisfying \( \varphi_c \) and \( \varphi_{b \ c} \) respectively”

\( = <a_0>( \)
\(<1>\varphi_c \quad /* \text{case (1,q)} */ \)
\(\lor<2>\varphi_{b \ c} \quad /* \text{case (2,q)} */ \)
\(\lor<\text{true} > \quad /* \text{case true} */ \)
From trees to HFL formulas

\( \varphi_T \): “the current state has transitions corresponding to an accepting run for \( T \)”

\( \varphi_{a \ c \ (b \ c)} = \)

\(<a_0> \) “can visit 1\textsuperscript{st} and 2\textsuperscript{nd} children with states satisfying \( \varphi_c \) and \( \varphi_{b \ c} \) respectively”

= \( <a_0>( \)

\(<1>\varphi_c \) /* case (1,q) */

\( \lor <2>\varphi_{b \ c} \) /* case (2,q) */

\( \lor <\texttt{true}> \) /* case true */

\( \lor (\text{and}<\texttt{true}> /* case f\land g */

\( \land [\text{and}] \)

```
(2,q_0)
```

```
(1,q_0) \land (2,q_0)
```

```
q_0
```

```
q_1
```

```
tt
```

```
true
```

```
false
```

```
a_0
```

```
b_0
```

```
c_0
```

```
1
```

```
2
```

```
and
```

```
(1,q_0)
```

```
(1,q_1)
```

```
1
```

\( c_0 \)

\( b_0 \)

\( a_0 \)

\( a_0 \)

\( b_0 \)

\( c_0 \)
\( \varphi_T: \) “the current state has transitions corresponding to an accepting run for T”

\( \varphi_{a \, c \, (b \, c)} = \)
\(<a_0> \) “can visit 1\(^{st}\) and 2\(^{nd}\) children with states satisfying \( \varphi_c \) and \( \varphi_{b \, c} \) respectively”

= \(<a_0>(\nu X.\)
\(<1>\varphi_c \) /* case (1,q) */
\(\lor<2>\varphi_{b \, c} \) /* case (2,q) */
\(\lor<tt>\text{true} \) /* case true */
\(\lor(\langle\text{and}\rangle\text{true} /*case f \land g */ \land [\text{and}] X)\)
From trees to HFL formulas

\( \varphi_T: \) “the current state has transitions corresponding to an accepting run for \( T \)”

\( \varphi_{a \ c \ (b \ c)} = \)

\(<a_0> \) “can visit 1\textsuperscript{st} and 2\textsuperscript{nd} children with states satisfying \( \varphi_c \) and \( \varphi_{b \ c} \) respectively”

= \(<a_0>(\forall X.

\(<1>\varphi_c \) /* case \((1,q)\) */

\(\lor <2>\varphi_{b \ c} \) /* case \((2,q)\) */

\(\lor <\text{true} >\text{true} \) /* case true */

\(\lor (<\text{and}>\text{true} /*case f \land g */

\(\land [\text{and}]X)\)

\(\lor <\text{or}>X) \) /*case f \lor g */
From trees to HFL formulas

\( \varphi_T: \) “the current state has transitions corresponding to an accepting run for T”

\[ \varphi_{a \, c \, (b \, c)} = \langle a_0 \rangle (H_2 \, \varphi_c \, \varphi_{b \, c}) \]

where \( H_2 = \lambda Y_1. \lambda Y_2. \forall X. \)

\( \langle 1 \rangle Y_1 /* \text{case (1, q)} */ \)
\( \lor \langle 2 \rangle Y_2 /* \text{case (2, q)} */ \)
\( \lor \langle \text{true} \rangle /* \text{case true} */ \)
\( \lor (\langle \text{and} \rangle \text{true} /* \text{case f} \land \text{g} */ \)
\( \land [\text{and}]X) \)
\( \lor \langle \text{or} \rangle X) /* \text{case f} \lor \text{g} */ \)
From trees to HFL formulas

\[ \varphi_T: \ "the \ current \ state \ has \ transitions \ corresponding \ to \ an \ accepting \ run \ for \ T" \]

\[ \varphi_{a \ c \ (b \ c)} = \langle a_0 \rangle (H_2 \ \varphi_c \ \varphi_{b \ c}) \]
\[ = \langle a_0 \rangle (H_2 (\langle c_0 \rangle H_0) (\langle b_0 \rangle H_1 (\langle c_0 \rangle H_0))) \]

where \( H_2 = \lambda Y_1. \lambda Y_2. \nu X. \)
\[ \langle 1 \rangle Y_1 \quad /* \ case \ (1,q) */ \]
\[ \lor \langle 2 \rangle Y_2 \quad /* \ case \ (2,q) */ \]
\[ \lor \langle \text{true} \rangle \quad /* \ case \ true */ \]
\[ \lor (\langle \text{and} \rangle \text{true} \quad /* \ case \ f \land g */ \]
\[ \land [\text{and}]X) \]
\[ \lor \langle \text{or} \rangle X) \quad /* \ case \ f \lor g */ \]

\[ \varphi_T: \ "the \ current \ state \ has \ transitions \ corresponding \ to \ an \ accepting \ run \ for \ T" \]
From trees to HFL formulas

\( \varphi_T: \) “the current state has transitions corresponding to an accepting run for T”

\[
\varphi_{ac \ (b \ c)} = \langle a_0 \rangle (H_2 \varphi_c \varphi_{b \ c}) \\
= \langle a_0 \rangle (H_2 (\langle c_0 \rangle H_0) (\langle b_0 \rangle H_1 (\langle c_0 \rangle H_0)))
\]

where \( H_2 = \lambda Y_1. \lambda Y_2. \nu X. \)

\( <1>Y_1 \) /* case (1,q) */

\( \lor <2>Y_2 \) /* case (2,q) */

\( \lor<tt>true \) /* case true */

\( \lor(\langle and \rangle true \) /*case f \wedge g */

\( \land [\text{and}]X) \)

\( \lor <\text{or}>X) \) /*case f \lor g */

\[
\begin{align*}
q_0 &\quad \rightarrow \quad q_1 \\
(1,q_0) &\quad \rightarrow \quad (2,q_0) \\
(1,q_0) &\quad \rightarrow \quad (1,q_0) \land (2,q_0) \\
(1,q_1) &\quad \rightarrow \quad (2,q_0) \\
true &\quad \rightarrow \quad (1,q_0) \\
false &\quad \rightarrow \quad (1,q_1)
\end{align*}
\]
From trees to HFL formulas

\( \varphi_T: \) “the current state has transitions corresponding to an accepting run for \( T \)”

\[ \varphi_{a_{T1} \ldots T_k} = \langle a_0 \rangle (H_k \varphi_{T1} \ldots \varphi_{Tk}) \]  
(co-inductively defined)
From HORS to HFL

\[ F \rightarrow \tau \]
\[ \Rightarrow F = _\nu \tau^# \]

where:

\[ F^# = F \quad x^# = x \]
\[ (\tau_1 \tau_2)^# = (\tau_1)^#(\tau_2)^# \]
\[ (\lambda x. \tau)^# = \lambda x. (\tau)^# \]
\[ a^# = \lambda x_1 \ldots \lambda x_k. <a_0>(H_k x_1 \ldots x_k) \]
Example

**HORS G**

\[ S \rightarrow F \ c \]

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

\( \varphi_{G,0} \)

\[ S = \nu \ F \ (\langle c_0 \rangle H_0) \]

\[ F \ x = \nu \langle a_0 \rangle (H_2 \times (F (\langle b_0 \rangle (H_1 \times)))) \]

**A:**

\[ \delta(q_{0},a) = (1,q_{0}) \land (2,q_{0}) \]

\[ \delta(q_{1},a) = \text{false} \]

\[ \delta(q_{0},b) = \delta(q_{1},b) = (1,q_{1}) \]

\[ \delta(q_{0},c) = \delta(q_{1},c) = \text{true} \]

\( L_A \)
Outline

♦ Reviews of HORS model checking and HFL model checking

♦ From HORS to HFL model checking
  - construction of $L_A$
  - construction of $\varphi_{G,p}$
    • case $p=0$
    • general case

♦ From HFL to HORS model checking

♦ Type system for HFL model checking

♦ Conclusion
Challenge

• How to translate the parity condition of APT:
  “for every path of a run-tree, the largest priority visited infinitely often is even”

to a proper nesting of least/greatest fixpoint formulas?

e.g. A: \( \delta(q_a,a) = (1,q_a) \quad \delta(q_a,b) = (1,q_b) \quad \Omega(q_a) = 0, \quad \Omega(q_b) = 1 \)

\( G: S \rightarrow a \ (b \ F) \quad F \rightarrow a \ S \)

\( G \not\models A \quad \text{but} \)

\[
S = \nu \ <a_0>(H_1 \ <b_0>(H_1 \ F)))
\]

\[
F = \nu \ <a_1>(H_1 \ S)
\]
Ideas

- Duplicate each non-terminal for each priority

\[ S \rightarrow a \ (b \ F) \quad F \rightarrow a \ S \]

\[ S^1 = _\mu <a_0>(H_1 (H_1 \ <b_0>(H_1 F^0))); \]
\[ F^1 = _\mu <a_1>(H_1 S^1); \]
\[ S^0 = _\nu <a_0>(H_1 (H_1 \ <b_0>(H_1 F^0))); \]
\[ F^0 = _\nu <a_1>(H_1 S^1); \]

An appropriate copy is chosen, depending on the largest priority seen since the last unfolding.

The largest priority seen since the previous unfolding of a non-terminal.
Ideas

- Duplicate each non-terminal for each priority
- Duplicate also each argument, so that a function can choose an appropriate copy

\[ S \rightarrow F \ G \quad F \ x \rightarrow b \ (x \ S) \quad G \ y \rightarrow a \ y \]

We cannot locally decide the priority annotation for \( G \); only \( F \) knows when \( G \) is unfolded.

\[ S^1 = \mu \ F^0 \ G^0 \ G^1 \]
\[ F^1 \ x^0 \ x^1 = \mu \ <b_0>(H_1 (x^0 \ S^0 \ S^1)) \cup <b_1>(H_1 (x^1 \ S^1 \ S^1)) \]

...
General construction of $\varphi_{G,p}$

$G: F_1 \times_1 \ldots \times_{k_1} \to t_1$, $\ldots$, $F_n \times_1 \ldots \times_{k_n} \to t_n$

\[
\begin{align*}
F_1^p \times_1^0 \ldots \times_1^p \ldots \times_{k_1}^0 \ldots \times_{k_1}^p &= \alpha(p) t_1 \#^0; \ldots; \\
F_n^p \times_1^0 \ldots \times_1^p \ldots \times_{k_1}^0 \ldots \times_{k_1}^p &= \alpha(p) t_n \#^0; \\
\ldots;
F_1^0 \times_1^0 \ldots \times_1^p \ldots \times_{k_1}^0 \ldots \times_{k_1}^p &= \alpha(0) t_1 \#^0; \ldots; \\
F_n^0 \times_1^0 \ldots \times_1^p \ldots \times_{k_1}^0 \ldots \times_{k_1}^p &= \alpha(0) t_n \#^0
\end{align*}
\]

where $\alpha(i) = \nu$ if $i$ is even and $\mu$ otherwise
General construction of $\varphi_{G,p}$

Given $F_1 x_1 \ldots x_{k_1} \rightarrow t_1, \ldots, F_n x_1 \ldots x_{k_n} \rightarrow t_n$

$F_1^p x_1^0 \ldots x_1^p \ldots x_{k_1}^0 \ldots x_{k_1}^p = \alpha(p) t_1^{#0}$;

$\ldots$;

$F_n^0 x_1^0 \ldots x_1^p \ldots x_{k_n}^0 \ldots x_{k_n}^p = \alpha(0) t_n^{#0}$

$(a)^{#i} = \lambda x_{1,0} \ldots \lambda x_{1,p} \ldots \lambda x_{k,0} \ldots \lambda x_{k,p} \cdot$

$\langle a_0 \rangle (H_k x_{1,0} \ldots x_{k,0}) \lor \ldots \lor \langle a_p \rangle (H_k x_{1,p} \ldots x_{k,p})$

$(x)^{#i} = x^i$

$(F)^{#i} = F^i$

$(s \top)^{#i} = (s)^{#i} (\top)^{#\max(0,i)} \ldots (\top)^{#\max(p,i)}$
Correctness of Translation

♦ Theorem:

\[ G \models A \iff L_A \models \varphi_{G,p} \]

Follows from the type-based characterizations of HORS and HFL model checking:

\[ G \models A \iff \neg A G \iff \neg L_A \varphi_{G,p} \iff L_A \models \varphi_{G,p} \]

[K&Ong 09] (new)
Correctness of Translation

♦ Theorem:

\[ G \models A \text{ if and only if } L_A \models \varphi_{G,p} \]

\[ |L_A| \text{ is polynomial in } |A| \]

\[ |\varphi_{G,p}| \text{ is polynomial in } |G|, \ p \]
Outline

♦ Reviews of HORS model checking and HFL model checking
♦ From HORS to HFL model checking
♦ From HFL to HORS model checking
♦ Type system for HFL model checking
♦ Conclusion
From HFL to HORS model checking

♦ Input:
  - LTS $L$
  - HFL formula $\phi$

♦ Output:
  - HORS $G_{\phi,c}$
  - APT $A_L$

such that $L \models \phi$ iff $G_{\phi,c} \models A_L$ for sufficiently large $c$

Intuition:

- $G_{\phi,c}$ generates tree representation of the formula obtained from $\phi$ by unfolding fixedpoint operators sufficiently many times
- $A_L$ accepts trees representing valid formulas
HFL-to-HORS Translation: Overview

$F X = \nu \varphi$

Remove fixpoint operators by finite unfoldings

$F(c) X = [F(c-1)/F] \varphi ; \ldots ; F(1) X = [F(0)/F] \varphi ; F(0) X = \text{true}$

Convert it to HORS, which generates the tree representation of the formula

$F(c) X \rightarrow [F(c-1)/F] \varphi' ; \ldots ; F(1) X \rightarrow [F(0)/F] \varphi' ; F(0) X \rightarrow \text{true}$

Parameterize $F$ by a number, and implement numbers (up to $k$) as functions (cf. [Jones01])

$F m X \rightarrow \text{if (Zero? m) true ([F (m-1)/F] \varphi')}$
Correctness of Translation

♦ Theorem:

\[ L \models \varphi \]
if and only if

\[ G_{\varphi,|L|} \models A_L \]

\(|G_{\varphi,|L|}| \text{ is polynomial in } |\varphi| \text{ and } |L| \]
\(|A_L| \text{ is polynomial in } |L| \)
Outline

♦ Reviews of HORS model checking and HFL model checking
♦ From HORS to HFL model checking
♦ From HFL to HORS model checking
♦ Type system for HFL model checking
♦ Conclusion
Goal

Design a type system $\vdash_L$ such that:

$L \vdash \varphi$

if and only if

$\vdash_L \varphi$

(cf. K-Ong type system for HORS model checking [K&Ong, LICS09])

Applications:

- correctness proof of HORS-to-HFL translation
- practical model checkers for HFL
  (cf. practical HORS model checkers based on intersection types)
Types

\( \tau ::= s \) type of propositions that hold at state \( s \)

(i.e. \( s \models \phi \iff \vdash \lnot \phi : s \))

\( \sigma \rightarrow \tau \) type of functions from \( \sigma \) to \( \tau \)

\( \sigma ::= \tau_1 \land \ldots \land \tau_n \) intersection types

\( \land : \ s \rightarrow s \rightarrow s \) for every \( s \)

\( \lor : (s \rightarrow T \rightarrow s) \land (T \rightarrow s \rightarrow s) \) for every \( s \)
Typing Rules

\[\Gamma \vdash \text{true}: s\]

\[\Gamma \vdash \phi: s \quad \Gamma \vdash \psi: s \]
\[\Gamma \vdash \phi \land \psi: s\]

\[\Gamma, X: \tau \vdash X: \tau\]

\[\Gamma \vdash \phi: \tau_1 \land \ldots \land \tau_k \rightarrow \tau\]
\[\Gamma \vdash \psi: \tau_i \text{ for each } i\]
\[\Gamma \vdash \phi \land \psi: \tau\]

\[\Gamma \vdash \phi: s' \quad s \rightarrow_a s'\]
\[\Gamma \vdash <a>\phi: s\]

\[\Gamma \vdash \phi: s'\]
\[\text{for every } s' \text{ such that } s \rightarrow_a s'\]
\[\Gamma \vdash [a]\phi: s\]

\[\Gamma, x: \tau_1, \ldots, x: \tau_k \vdash \phi: \tau\]
\[\Gamma \vdash \lambda x.\phi: \tau_1 \land \ldots \land \tau_k \rightarrow \tau\]
Typing Fixpoint Formulas

\[
\emptyset \vdash \phi : \tau \quad X = \alpha \phi \\
\hline
\Gamma \vdash X : \tau
\]

Definition:

\[\vdash_{L} X_{1} = \alpha_{1} \varphi_{1} ; \ldots ; X_{n} = \alpha_{n} \varphi_{n}\]

if there is a possibly infinite derivation for

\[\emptyset \vdash X_{1} : s_{\text{init}}\]

such that, for each infinite derivation path,

\(\alpha_{j} = \nu\) for the least \(j\) such that \(X_{j}\) is unfolded infinitely often.
Example

\[ X: s_0 \vdash X: s_0 \]
\[ X: s_0 \vdash [b]X: s_1 \]
\[ \emptyset \vdash \lambda X. [b]X: s_0 \rightarrow s_1 \]
\[ \emptyset \vdash F: s_0 \rightarrow s_1 \]
\[ \emptyset \vdash F A: s_1 \]
\[ \emptyset \vdash <a>(F A): s_0 \]
\[ \emptyset \vdash A: s_0 \]

LTS:

\[ \begin{align*}
\text{LTS:} & \\
\begin{tikzpicture}
  \node (s0) at (0,0) {$s_0$};
  \node (s1) at (2,0) {$s_1$};
  \draw[->] (s0) edge[bend left=45] node[left] {$a$} (s1);
  \draw[->] (s1) edge[bend left=45] node[right] {$b$} (s0);
\end{tikzpicture}
\end{align*} \]

HES

\[ A = \nu a <a>(F A); \]
\[ F = \mu X. [b]X \]
Correctness of Type System

♦ Theorem:

\[ L \models \varphi \text{ if and only if } |-_L \varphi \]

♦ Corollary:

\[ L \models \varphi \text{ can be decided in time polynomial in the size of } \varphi, \text{ if the following parameters are fixed:} \]
- \( L \)
- the largest size of types in \( \varphi \)
- alternation depth of \( \varphi \)
Outline

♦ Reviews of HORS model checking and HFL model checking
♦ From HORS to HFL model checking
♦ From HFL to HORS model checking
♦ Type system for HFL model checking
♦ Related work and Conclusion
Related Work

♢ HORS model checking
  - decidability [Knapik+02][Ong06]...
  - type-based characterization [K09][K&Ong09]
  - algorithms [K09][K11][Ramsay+14]...
  - applications [K09][K+11][Ong+11]...

♢ HFL model checking
  - decidability [Viswanathan² 04]
  - complexity [Axelsson+ 07]
  - applications [Viswanathan² 04][Lange+ 12]
## Related Work

- **Type-based characterization of HORS model checking** [K 09][K&Ong 09]
  - translation from HORS to HFL model checking
  - type-based characterization

- **Encoding of big numbers as functions** [Jones 01][Tsukada&K 14]

- **Reduction from HORS model checking to nested least/greatest fixedpoint computation** [Salvati&Walukiewicz, CSL15]
Conclusion

♦ Revealed close relationships between HORS/HFL model checking through:
  - order-preserving mutual reductions
  - type-based characterization of HFL model checking similar to that of HORS model checking

♦ Future work: mutual transfer of results (e.g. practical model checking algorithms)