Higher type Automata-Logic-Games Correspondences

Luke Ong

(Joint with Matthew Hague, Steven Ramsay, and Takeshi Tsukada)

Shonan Meeting on Higher-Order Model Checking

14-17 March 2016

Model checking higher-type Böhm trees

What is higher order about Higher-Order Model Checking (HOMC)?

Higher-Order Model Checking Problem: Given HORS $\mathcal{G}$ and property φ (APT or mu-calculus formula), does the tree $[\![\mathcal{G}]\!]$ satisfy φ?

Two closely related desiderata:

1. Model check higher-type Böhm trees.
   The HOMC Problem is about computation trees of ground-type programs. To analyse computation trees of higher-type programs, we need to model check Böhm trees, i.e., trees with λ-binders.
2. Make higher-order model checking compositional.
   HOMC is (mostly) whole program analysis. But higher-order is supposed to aid modular structuring of programs.

Unfortunately the elegant theorems of "Rabin's Heaven" fail in the world of Böhm trees. (More on this later.)

Böhm trees are term-trees s, t, etc., defined coinductively $(n \geq 0)$:

$$t^o ::= \perp \mid x^A \, s_1^{A_1} \cdots s_n^{A_n}$$
$$s^A ::= \lambda x_1^{A_1} \cdots x_n^{A_n} . t^o \qquad \text{where } A = A_1 \to \cdots \to A_n \to o$$

- Assume Böhm trees are
  - well-sorted (i.e. simply-typed) and η-long: write $\Gamma \vdash t :: A$
  - have only finitely many free variables.
- Alternative presentation as Σ-binding trees (a version of data trees).
- Every variable occuring in a (closed) Böhm tree of sort A has a sort that is a contravariant subsort of A.
- Böhm trees subsume ordinary (node-labelled, ranked) trees.

Böhm trees of composable sorts can compose. Thus model checking Böhm trees is model checking higher-order functions (on trees).

- For decidability results, we consider $\lambda\mathbf{Y}$-definable Böhm trees, BT(M).

No “Rabin’s Heaven” for Böhm trees

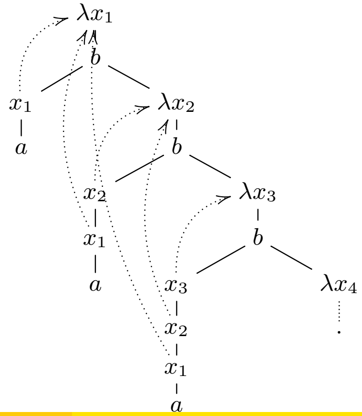
Let $\Gamma = a : o,\ b : o \to ((o \to o) \to o) \to o$ and

$$\Gamma\ \vdash\ \underbrace{\mathbf{Y}\,(\lambda f.\lambda y^{o}.\lambda x^{o\to o}.b\,(x\,y)\,(f\,(x\,y)))\,a}_{M}\ :\ (o \to o) \to o.$$

- $\mathrm{BT}(M)$
uses infinitely many variable names,
and each variable occurs infinitely often.

- $\mathrm{BT}(M)$
has an undecidable MSO theory! (Salvati; Clairambault & Murawski FSTTCS13)

- Emptiness of Stirling’s alternating dependency tree automata—a compelling device for analysing Böhm trees—is undecidable. (O. & Tzevelekos LICS09)



An expressive yet decidable "logic" for higher-type Böhm trees?

Take property $\Phi :=$
"There are only finitely many occurrences of bound variables in each branch."

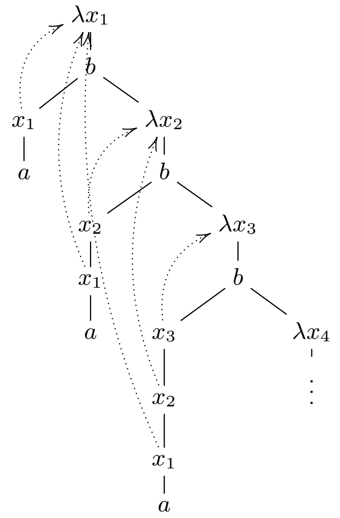
Questions:

1. Is there a "logic" that can describe properties such as Φ?
2. Is the "logic" decidable for Böhm trees?

Ans: (intersection) types.

1. Fix a semantics – type-checking game, $\Gamma \models t : \tau$, for Böhm trees t and types τ
2. Develop decidable, complete proof system, $\Gamma \vdash M : \tau$, for definable Böhm trees:

$$\Gamma \vdash M : \tau \iff \Gamma \models \mathrm{BT}(M) : \tau$$



Outline of Talk:

Higher type Automata-Logic-Games Correspondences

0. Motivations: compositional higher-order model checking / model checking Böhm trees

1. Type-checking Games
2. Alternating Dependency Tree Automata
3. Higher-type Mu-Calculus
4. Conclusions and Further Directions

Type-Checking Game (Tsukada & O. LICS14)

$\Gamma \models t : \tau$ means "Verifier has a winning strategy in the game that checks Böhm tree t has type τ in environment Γ".

Types τ (Kobayashi & O. LICS09)

prime types	σ, τ	$::=$	$q \mid \alpha \to \tau$
intersection types	α	$::=$	$\bigwedge_{i \in I} (\tau_i, e_i)$

where $q \in Q$ (base types), I a finite set, and $e_i \in \mathbb{E}$ of a max-parity winning condition $\langle \mathbb{E}, \circ, 0, \preceq \rangle$.

We only consider types τ, α that are refinements of sorts A, written $\tau, \alpha :: A$.

Max-Parity Winning Condition $\langle \mathbb{E}, \circ, 0, \preceq \rangle$

- $\mathbb{E} = \{0, 1, 2, \ldots, 2N\}$ where $N \geq 1$ is the set of priorities (or effects)
- $\preceq$ is sub-priority order (Fujima et al. APLAS13)

$$2N \prec \cdots \prec 4 \prec 2 \prec 0 \prec 1 \prec 3 \prec \cdots \prec 2N - 1;$$

  compares priorities from P's (Verifier's) perspective, whose goal is max-parity ($2N$ being best)
- monoidal product, $e_1 \circ e_2 := \max_{\leq}(e_1, e_2)$ with 0 identity

The ordered monoid, $\langle \mathbb{E}, \circ, 0, \preceq \rangle$, has left-residuals. I.e. -\\- : $\mathbb{E}^2 \to \mathbb{E}$, satisfying for all $d, e, e' \in \mathbb{E}$

$$e \circ d \preceq e' \iff d \preceq e \backslash e'.$$

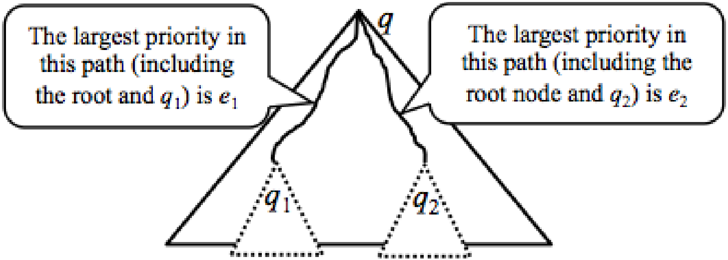
An infinite sequence of priorities satisfies the max-parity winning condition if the $\leq$-maximum infinitely-occurring priority is even.

Intuition

Regard automaton states as the base types (of trees) (Kobayashi POPL09)

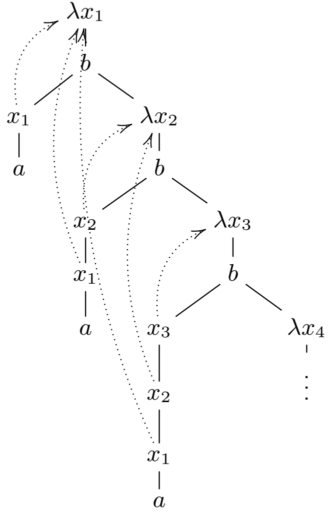
- q is the type of trees accepted by the automaton from state q
- $q_1 \wedge q_2$ is the type of trees accepted from both q_1 and q_2
- $\tau \to q$ is the type of functions that take a tree of type τ and return a tree of type q

Example. A tree function described by $(q_1, e_1) \wedge (q_2, e_2) \to q$.



(The above is a tree context of a run-tree, not the generated tree.)

- Verifier is responsible for typing variable nodes, and Refuter lambda nodes.
- Whenenver it is Verifier's turn, he picks a ground type (in the type environment) for the variable node in question, and labels each of the child node (all lambdas) with a set of types.
- Refuter then picks a labelled node, and a type from the set of types, and labels the transition with a priority from the type environment.
- A play traces out a path in the Böhm tree, whose edges are labelled with priorities.



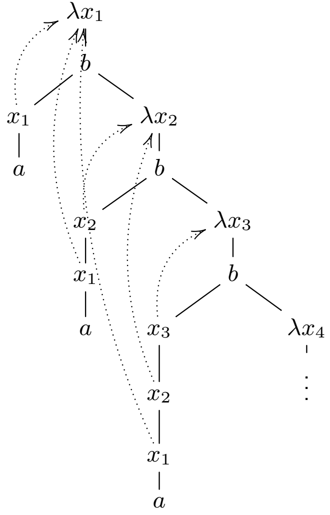
Example Revisited

Property $\Phi :=$
"There are only finitely many occurrences of bound variables in each branch."

Take parity winning condition, with effects (priorities) $2 \prec 0 \prec 1$, and base type q.

Encoding Φ: Böhm tree t satisfies Φ just if $\Gamma \models t : (((q,1) \to q,1) \to q,0)$ where $\Gamma = a : (q,1), b : ((q,0) \to (((q,1) \to q,1) \to q,0) \to q,1)$.

It is decidable whether $\Gamma \models t : (((q,1) \to q,1) \to q,0)$, for a given definable Böhm tree t.



Compositionality and Effective Selection

1. $\models$ conservatively extends the MSO properties of trees: games over a tree given by an APT are type-checking games over the tree.
   - Type-checking games is a significant generalistion of the situation of games over tree.
2. Two-Level Compositionality:
   - If Böhm trees s and t are composable, then the set of properties (i.e. types) of $s \circ t$ is completely determined by those of s and of t.
   - Furthermore if $\models s : \tau$ and $\models t : \sigma$ imply $\models s \circ t : \delta$, then the winning strategies $\mathfrak{s}_s^\tau$ of $\models s : \tau$ and $\mathfrak{s}_t^\sigma$ of $\models t : \sigma$ are composable, and yield a winning strategy $\mathfrak{s}_s^\tau \circ \mathfrak{s}_t^\sigma$ of $\models s \circ t : \delta$.
3. Effective Selection: If $\models \mathrm{BT}(M) : \tau$ then there exists, constructively, a $\lambda\mathbf{Y}$-definable winning strategy of $\models \mathrm{BT}(M) : \tau$.

Underpining the above is a cartesian closed category of 2-level effect arena games. They give a strategy-aware model—the basis of compositional model check higher-type Böhm trees.

Theorem (Transfer)

For all $\lambda\mathbf{Y}$-terms M and types τ, have $\Gamma \vdash M : \tau \iff \Gamma \models \mathrm{BT}(M) : \tau$

$$\frac{\Gamma(x) = \bigwedge_{i \in I} (\theta_i, e_i) \qquad 0 \preceq e_i \text{ for some } i \in I}{\Gamma \vdash x : \theta_i}$$

$$\frac{\Gamma \vdash M : \tau \to \theta \quad \Gamma \vdash N : \tau}{\Gamma \vdash M\ N : \theta} \qquad \frac{\Gamma, x : \tau \vdash M : \theta}{\Gamma \vdash \lambda x.M : \tau \to \theta}$$

$$\frac{\Gamma' \preceq \Gamma \quad \Gamma \vdash M : \theta \quad \theta \preceq \theta'}{\Gamma' \vdash M : \theta'} \qquad \frac{\models \mathrm{BT}(\mathbf{Y}) : \theta}{\Gamma \vdash \mathbf{Y} : \theta}$$

$$\frac{\forall i \in I.\ \Gamma \vdash M : (\theta_i, e_i)}{\Gamma \vdash M : \bigwedge_{i \in I} (\theta_i, e_i)} \qquad \frac{\Gamma \vdash M : \theta}{e \circ \Gamma \vdash M : (\theta, e)}$$

$\Gamma \vdash M : \tau$ is decidable: syntax-directed rules reduce the problem to solving parity games, $\models \mathrm{BT}(\mathbf{Y}) : \tau$, which is decidable.

Positive and negative actions of effects (priorities) on types

Careful effect handling is crucial for compositional model checking: in contrast to HORS model checking, $\mathbf{Y}$ can be applied to open terms.
N.B. Whereas $\vdash M : \tau \iff \vdash M : (\tau, e)$, we have
$\Gamma \vdash M : \tau \not\Longleftrightarrow \Gamma \vdash M : (\tau, e)$.

The positive action (Melliès 2012) of e to intersection types and type environments:

$$e \circ \big(\bigwedge_{i \in I} (\tau_i, d_i) \big) := \bigwedge_{i \in I} (\tau_i, e \circ d_i) \quad (e \circ \Gamma)(x) := e \circ (\Gamma(x)).$$

Similarly for negative action $e \backslash -$, i.e., $e \backslash \big(\bigwedge_{i \in I} (\tau_i, d_i) \big) := \bigwedge_{i \in I} (\tau_i, e \backslash d_i)$.

The key observation is: $\Gamma \vdash M : (\tau, e) \iff e \backslash \Gamma \vdash M : \tau$.

By defining $\alpha \Rightarrow (\tau, e) := ((e \backslash \alpha) \to \tau, e)$, we have:

$$\frac{\Gamma, x : \alpha \vdash M : (\tau, e)}{\Gamma \vdash \lambda x.M : \alpha \Rightarrow (\tau, e)}$$

Parity Games, Mu-Calculus and APT are Equivalent

Mu-Calculus: modal logic extended with least and greatest fixpoint operators. (Scott, de Bakker; Kozen 82)

- Mu-calculus and MSOL are equi-expressive for tree languages (Niwinski).
- Mu-calculus is the bisimulation-invariant fragment of MSOL (JW 96).

Mu-calculus Model Checking Problem and PARITY are inter-reducible

$\Rightarrow$: Essentially: Fundamental Semantic Theorem [Streett and Emerson Info & Comp 1989]

$\Leftarrow$: E.g. [Walukiewicz Info & Comp 2001]

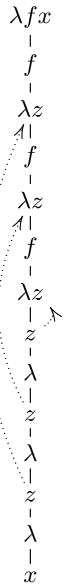
Mu-calculus and APT are effectively equi-expressive for tree languages [Emerson & Jutla FoCS 91]

$\Rightarrow$: E.g. [Kupferman & Vardi JACM 2000]

$\Leftarrow$: E.g. [Walukiewicz Info & Comp 2001]

Alternating Dependency Tree Automata (Stirling FoSSaCS09)

- ADTA are automata over Böhm trees (in general, over Σ-binding trees with $\Sigma = \Sigma_{\mathrm{var}} \uplus \Sigma_\lambda \uplus \Sigma_{\mathrm{aux}}$, a kind of (ranked) data trees).
- Stirling used ADTA to characterise solution sets of the Higher-Order Matching Problem.
- We extend ADTA to infinite trees with ω-regular conditions.



Example: Böhm trees as Σ-binding trees

The order-4 sort

$$\mathfrak{M} = (((o \to o) \to o) \to o) \to o \to o$$

called monster, is inhabited by the terms

$$M_n := \lambda f\, x. f(\lambda z_1.(f(\lambda z_2.(\cdots f(\lambda z_n.z_n(\cdots z_2(z_1\, x)))))))$$

for all $n \geq 1$.

We can represent M_3 as a $\Sigma_{\mathfrak{M}}$-binding tree (see LHS) where

$$\Sigma_{\mathfrak{M}} = \underbrace{\{f^{((o \to o) \to o) \to o},\ x^o,\ z^{o \to o}\}}_{\Sigma_{\mathrm{var}}} \cup \underbrace{\{\lambda f\, x,\ \lambda z,\ \lambda\}}_{\Sigma_{\lambda}} \cup \underbrace{\{\perp\}}_{\Sigma_{\mathrm{aux}}}$$

Alternating dependency tree automata (ADTA)

An ADTA of sort A is a tuple $\mathcal{A} = (Q, \Sigma_A, Q_I, \Delta, \Omega)$

- Q is a finite state-set; $Q_I \subseteq Q$ are initial states
- $\Sigma_A = \Sigma_\lambda \cup \Sigma_{\mathrm{var}} \cup \{\perp\}$ where

$$\Sigma_{\mathrm{var}} := \{x^B \mid B \text{ contravariant subsort of } A\}$$

$$\Sigma_\lambda := \{\lambda x_1^{B_1} \cdots x_m^{B_m} \mid (B_1 \to \cdots \to B_m \to o) \text{ covariant subsort of } A\}$$

- Δ is a set of transition rules: 3 kinds
  - Acceptor transitions
  - Rejecter transitions
  - $\perp$ transitions
- Ω assigns a priority to transitions

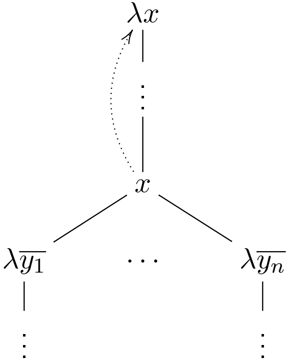
A Böhm tree t is accepted by $\mathcal{A}$ if acceptor has a winning strategy in the acceptance game $\mathcal{G}_{\mathrm{Acc}}(t, \mathcal{A})$ $\mathcal{G}_{\mathrm{Acc}}(t, \mathcal{A})$.

We first present a constrained version, Parity Permissive (PP) ADTA.

Acceptor chooses transitions to read variables:

$$(p, q)\, x \Rightarrow (Q_1, \cdots, Q_n)$$

Suppose we have a run to x:

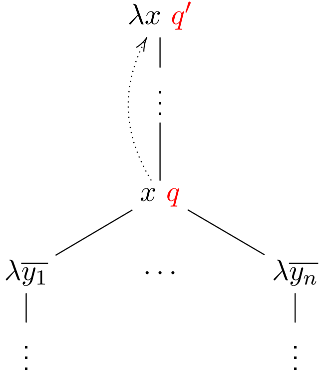


Acceptor transitions of a PP ADTA: $(q', q)\, x \Rightarrow (Q_1, \cdots, Q_n)$

Acceptor chooses transitions to read variables:

$$(q', q)\, x \Rightarrow (Q_1, \cdots, Q_n)$$

Suppose we have a run to x:

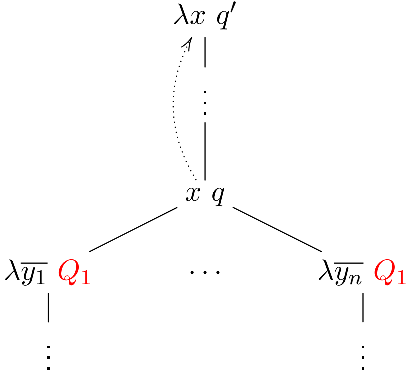


Acceptor transitions of a PP ADTA: $(q', q)\, x \Rightarrow (Q_1, \cdots, Q_n)$

Acceptor chooses transitions to read variables, where each $Q_i \subseteq Q$:

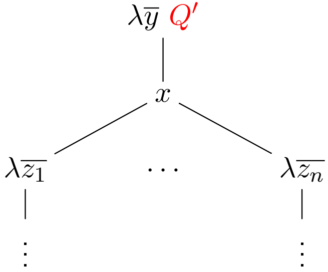
$$(q', q)\, x \Rightarrow (Q_1, \cdots, Q_n)$$

Suppose we have a run to x:



Rejecter transitions of a PP ADTA: $q\,\lambda\overline{y} \overset{e}{\Rightarrow} q'$

Suppose Acceptor has just labelled λ-nodes with $Q' \subseteq Q$:



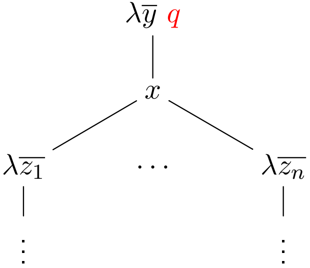
Rejecter chooses some $q \in Q'$ and a transition

$$q\,\lambda\overline{y} \overset{e}{\Rightarrow} q'$$

where e is the priority assigned by Ω

Rejecter transitions of a PP ADTA: $q\,\lambda\overline{y} \stackrel{e}{\Rightarrow} q'$

Acceptor labels λ-nodes with a set of states $Q' \subseteq Q$:

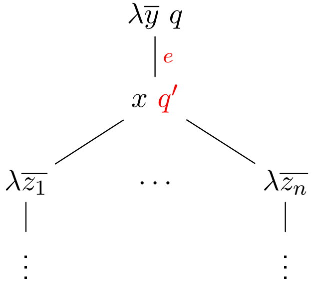


Rejecter chooses some $q \in Q'$ and a transition

$$q\,\lambda\overline{y} \stackrel{e}{\Rightarrow} q'$$

where e is the priority assigned by Ω

Acceptor labels λ-nodes with a set of states $Q' \subseteq Q$:



Rejecter chooses some $q \in Q'$ and a transition

$$q\,\lambda\overline{y} \stackrel{e}{\Rightarrow} q'$$

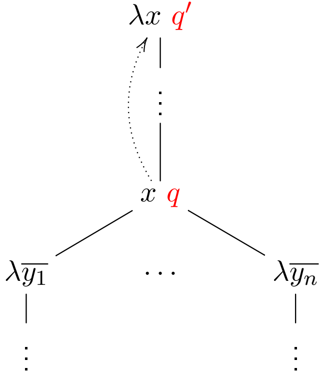
where e is the priority assigned by Ω. Acceptor wins if the maximum infinitely-occurring priorities in the sequence labelling the transitions is even.

Acceptor transitions of (unconstrained) ADTA: $(q', q)\, x \stackrel{e}{\Rightarrow} (Q_1, \cdots, Q_n)$

We augment acceptor transitions with a priority e

$$(q', q)\, x \stackrel{e}{\Rightarrow} (Q_1, \cdots, Q_n)$$

subject to a sub-priority condition: $e' \preceq e$, where e' is the maximum priority labelling the transitions between nodes λx and x



Properties of ADTA

Previously known:

1. ADTA are closed under union and intersection (Stirling FoSSaCS09).
2. Emptiness of nondeterministic DTA is decidable (Stirling FoSSaCS09).
3. Emptiness of ADTA is undecidable (O. & Tzvelekos LICS09).

New results:

1. ADTA are closed under complementation.
2. ADTA Acceptance of $\lambda\mathbf{Y}$-definable Böhm trees is decidable.

Correspondence between ADTA and Types

Let $e_M \in \mathbb{E}$ be the $\preceq$-maximum priority.

A type τ is called parity permissive (PP) if every contravariant prime subtype of τ has priority e_M.

Given type $\alpha :: A$, define $[\![\alpha]\!] := \{u \in \mathsf{BT}^A \mid \ \models u : \alpha\}$.

Theorem (Effective Equi-expressivity)

1. *Types and ADTA are effectively equi-expressive for defining languages of Böhm trees.*
2. *PP types and PP ADTA are effectively equi-expressive for defining languages of Böhm trees.*

I.e. Take a sort A. There is an algorithm mapping ADTA $\mathcal{A}$ to types $\alpha_{\mathcal{A}}$ such that $L(\mathcal{A}) = [\![\alpha_{\mathcal{A}}]\!]$; and vice versa.

Fix a sort A. Formulas are indexed by subsorts of A. Assume:
- a finite set of sorted abstract λ-variables Σ_{var}, ranged over by x^B, y^C, etc., where B and C are contravariant subsorts of A
- for each $\mathsf{x}^B \in \Sigma_{\mathrm{var}}$, a denumerable set $\mathcal{V}^B$ of concrete variables of sort B

Higher-type Mu-calculus

$$\varphi^o ::= \langle \perp \rangle \mid \mathbf{t} \mid \mathbf{f} \mid \varphi^o \wedge \varphi^o \mid \varphi^o \vee \varphi^o \mid \neg \varphi^o \mid \mathbb{v}_{\mathsf{x}^B} \mid [i]\varphi^{B \to C} \mid \alpha \mid \mu\alpha.\varphi^o \mid \nu\alpha.\varphi^o$$

$$\varphi^{B \to C} ::= \varphi^{B \to C} \wedge \varphi^{B \to C} \mid \varphi^{B \to C} \vee \varphi^{B \to C} \mid \neg \varphi^{B \to C} \mid \mathbb{\lambda}_{\mathsf{x}^B}.\varphi^C.$$

Two new constructs:

- abstract-variable predicate, $\mathbb{v}_{\mathsf{x}^B}$
- abstract-lambda formula, $\mathbb{\lambda}_{\mathsf{x}^B}.\varphi^C$

which are detectors of (concrete) variables and λ-abstractions respectively.

Semantics of higher-type mu-calculus

A concretisation function is a sort-respecting map $\zeta : \Sigma_{\mathrm{var}} \to \mathcal{P}_{\mathrm{fin}}(\mathcal{V})$, giving the possible concrete names $\zeta(\mathsf{x}^B)$ (in a Böhm tree) that an abstract variable $(x)^B$ (in the formula) can represent.

Given a concretisation function ζ, $[\![\varphi^B]\!]_\rho(\zeta)$ returns a set of Böhm trees u of sort B, such that $\mathrm{FV}(u) \subseteq \mathrm{Im}(\zeta)$.

Fix ζ, and take a Böhm tree t. Then

- $t \in [\![\mathbb{v}_{\mathsf{x}^B}]\!]_\rho(\zeta)$ just if the root node of t is labelled by a (concrete) variable $y^B \in \zeta(\mathsf{x}^B)$
- $t \in [\![\mathbb{\lambda}_{\mathsf{x}^B}.\varphi^C]\!]_\rho(\zeta)$ just if $t = \lambda y^B.s$ and $s \in [\![\varphi^C]\!]_\rho(\zeta[\mathsf{x}^B \mapsto \zeta(\mathsf{x}) \cup \{y^B\}])$.

Example: Monster sort $\mathfrak{M}$

Recall: $M_3 := \lambda f\, x. f(\lambda z_1.(f(\lambda z_2.(f(\lambda z_3.z_3(z_2(z_1\, x)))))))$.

Let $\varphi^{\mathfrak{M}} = \mathbb{\lambda}_{\mathsf{f}\,\mathsf{x}}.\mu\alpha.\Big(\big(\mathbb{v}_{\mathsf{f}} \wedge [1]\mathbb{\lambda}_{\mathsf{z}}.\alpha\big) \vee \mu\beta.\big(\mathbb{v}_{\mathsf{x}} \vee (\mathbb{v}_{\mathsf{z}} \wedge [1]\mathbb{\lambda}.\beta)\big)\Big)$.

Then $M_3 \in [\![\varphi^{\mathfrak{M}}]\!]_{\emptyset}$.

Let $t_0 = \lambda f x. f\,(\lambda z_1.z_1\,(f\,(\lambda z_2.x))) :: \mathfrak{M}$; then $t_0 \notin [\![\varphi^{\mathfrak{M}}]\!]_{\emptyset}$.

N.B. The abstract-lambda predicate, $\mathbb{\lambda}_{\mathsf{x}^{\mathsf{B}}}$.-, is not a binder of x^B or of $\mathbb{v}_{\mathsf{x}^{\mathsf{B}}}$.

Properties of Higher-type Mu-calculus

Characterisation by an intuitive notion of model checking game $\mathcal{G}_{\mathrm{MC}}(t, \varphi, \zeta)$:
- $t \in [\![\varphi]\!]_\emptyset(\zeta)$ if and only if Verifier has a (memoryless) winning strategy for $\mathcal{G}_{\mathrm{MC}}(t, \varphi, \zeta)$.

Theorem (Effective Equi-expressivity)

There is an algorithm that, given a higher-type mu-calculus formula φ^A, returns a parity permissive ADTA $\mathcal{A}_{\varphi^A}$, such that $[\![\varphi^A]\!] = L(\mathcal{A}_{\varphi^A})$; and vice versa.
Hence, "PP ADTA $\equiv$ PP Types $\equiv$ Higher-type Mu-Calculus".

Parity permissiveness is not a restriction on universal dependency tree automata. However:

Conjecture (Parity Permissive)

There is a type τ such that for all PP types τ', $[\![\tau]\!] \neq [\![\tau']\!]$.

Conclusions

The automata-logic-games correspondences for Böhm trees are:

- PP ADTA $\equiv$ Higher-type Mu-Calculus $\equiv$ PP Type-checking Games
- ADTA $\equiv$ Effectful Mu-Calculus $\equiv$ Type-checking Games

Further Directions

- Develop a method of composing ADTA.
  Type checking of Böhm trees is compositional: $\Gamma \models s \,@\, t : \tau$ iff $\Gamma \models s : \alpha \to \tau$ and $\Gamma \models t : \alpha$ for some α.
- Effectful Mu-Calculus – rephrases sub-priority condition in terms of fixpoint variables, and then (i) extends the abstract-variable predicate with a named fixpoint variable α to $\mathbb{v}_{\mathsf{x}}^{\alpha}$, (ii) extends the concretisation function so that it records, for each concrete variable $y \in \zeta(\mathsf{x})$, the most subsuming fixpoint variable β seen in a play from the binder of y to its occurrence.
- Parity Permissive Conjecture. Consider intermediate results for (classes of) order-3 Böhm trees.