Probabilistic extension of higher-order model-checking

Charles Grellois Ugo dal Lago

FOCUS Team – INRIA & University of Bologna

Institute for Mathematical Sciences, Singapore
September 9, 2016

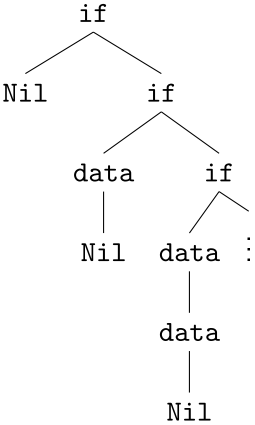
Roadmap

1. A quick introduction to higher-order model-checking (HOMC) and intersection types for HOMC
2. Automata for probabilistic properties, comparison with quantitative μ-calculus
3. Towards probabilistic HOMC: first steps and main challenges

Higher-order model-checking

Model-checking

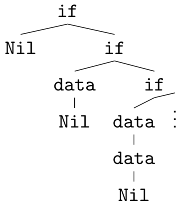
$\mathcal{T} \quad = $



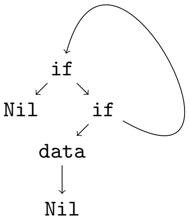
ϕ a logical property on trees, e.g. "all executions are finite".

Model-checking: does $\mathcal{T} \models \phi$?

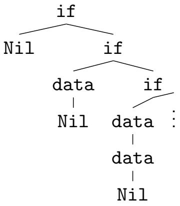
Finite representations of infinite trees



is not regular: it is not the unfolding of a finite graph as



Finite representations of infinite trees



but it is represented by a higher-order recursion scheme (HORS).

$$\mathcal{G} \;=\; \begin{cases} \mathtt{S} & = \quad \mathtt{L}\ \mathtt{Nil} \\ \mathtt{L}\ x & = \quad \mathtt{if}\ x\,(\mathtt{L}\ (\mathtt{data}\ x\ )\ ) \end{cases}$$

Higher-order recursion schemes

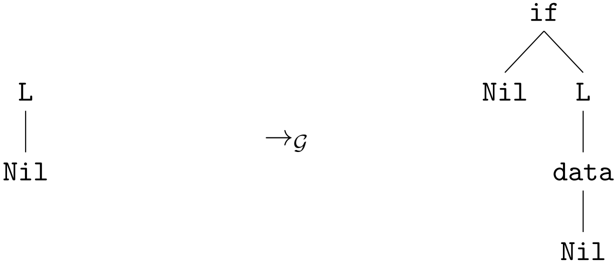
$$\mathcal{G} \;=\; \begin{cases} \mathtt{S} & = \quad \mathtt{L}\ \mathtt{Nil} \\ \mathtt{L}\ x & = \quad \mathtt{if}\ x\ (\mathtt{L}\ (\mathtt{data}\ x\ )\ ) \end{cases}$$

Rewriting starts from the start symbol S:

$$\mathtt{S} \quad \rightarrow_{\mathcal{G}} \quad \begin{array}{c} \mathtt{L} \\ | \\ \mathtt{Nil} \end{array}$$

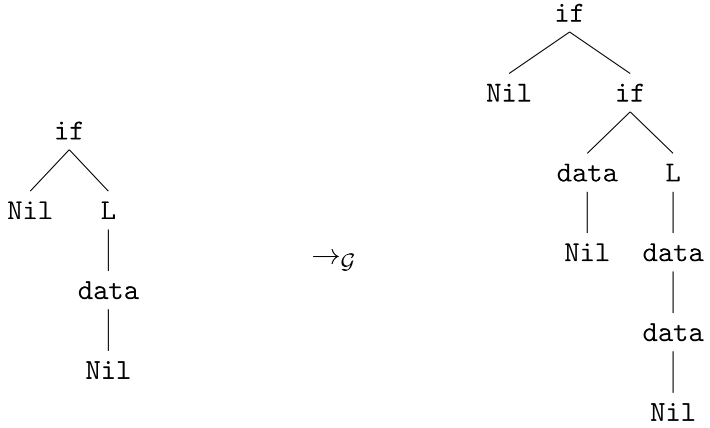
Higher-order recursion schemes

$$\mathcal{G} \;=\; \left\{ \begin{array}{lcl} \mathtt{S} & = & \mathtt{L}\ \mathtt{Nil} \\ \mathtt{L}\ x & = & \mathtt{if}\ x\ (\mathtt{L}\ (\mathtt{data}\ x\ )\ ) \end{array} \right.$$



Higher-order recursion schemes

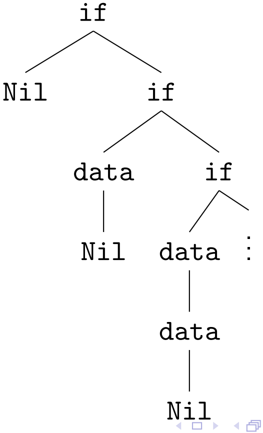
$$\mathcal{G} \;=\; \begin{cases} \mathtt{S} & = \quad \mathtt{L}\ \mathtt{Nil} \\ \mathtt{L}\ x & = \quad \mathtt{if}\ x\ (\mathtt{L}\ (\mathtt{data}\ x\ )\ ) \end{cases}$$



Higher-order recursion schemes

$$\mathcal{G} \;=\; \begin{cases} \mathtt{S} & = \quad \mathtt{L}\ \mathtt{Nil} \\ \mathtt{L}\ x & = \quad \mathtt{if}\ x\ (\mathtt{L}\ (\mathtt{data}\ x\ )\ ) \end{cases}$$

$\langle \mathcal{G} \rangle \quad =$



Higher-order recursion schemes

$$\mathcal{G} \;=\; \left\{ \begin{array}{lcl} \mathtt{S} & = & \mathtt{L}\ \mathtt{Nil} \\ \mathtt{L}\ x & = & \mathtt{if}\ x\,(\mathtt{L}\ (\mathtt{data}\ x\,)\,) \end{array} \right.$$

HORS can alternatively be seen as simply-typed λ-terms with

simply-typed recursion operators $Y_\sigma \;:\; (\sigma \rightarrow \sigma) \rightarrow \sigma$.

Modal μ-calculus

Equivalent to MSO over trees.

$$\phi,\ \psi\ ::=\ X \mid a \mid \phi \vee \psi \mid \phi \wedge \psi \mid \Box\,\phi \mid \Diamond_i\,\phi \mid \mu X.\,\phi \mid \nu X.\,\phi$$

$\Diamond_i\,\phi$: ϕ holds on a successor in direction i

$\Diamond\,\phi$: ϕ holds on a successor

$\Box\,\phi$: ϕ holds on all successors

Modal μ-calculus

Equivalent to MSO over trees.

$$\phi,\,\psi \;::=\; X \mid a \mid \phi \vee \psi \mid \phi \wedge \psi \mid \Box\,\phi \mid \Diamond_i\,\phi \mid \mu X.\,\phi \mid \nu X.\,\phi$$

$\mu X.\,\phi$ is the least fixpoint of $\phi(X)$. It is computed by expanding finitely the formula:

$$\mu X.\,\phi(X) \quad\longrightarrow\quad \phi(\mu X.\,\phi(X)) \quad\longrightarrow\quad \phi(\phi(\mu X.\,\phi(X)))$$

$\nu X.\,\phi$ is the greatest fixpoint of $\phi(X)$. It is computed by expanding infinitely the formula:

$$\nu X.\,\phi(X) \quad\longrightarrow\quad \phi(\nu X.\,\phi(X)) \quad\longrightarrow\quad \phi(\phi(\nu X.\,\phi(X)))$$

Modal μ-calculus

Equivalent to MSO over trees.

$$\phi,\,\psi \ ::= \ X \mid a \mid \phi \vee \psi \mid \phi \wedge \psi \mid \Box\,\phi \mid \ \Diamond_i\,\phi \mid \mu X.\,\phi \mid \nu X.\,\phi$$

Example formula:

$$\nu X.\,(\ \texttt{if} \wedge \Diamond_1\ (\ \mu Y.\,(\ \texttt{Nil} \vee \Box Y\ )\ ) \wedge \Diamond_2\ X\ )$$

Companion automata model: APT = ATA + parity condition.

Alternating tree automata (ATA)

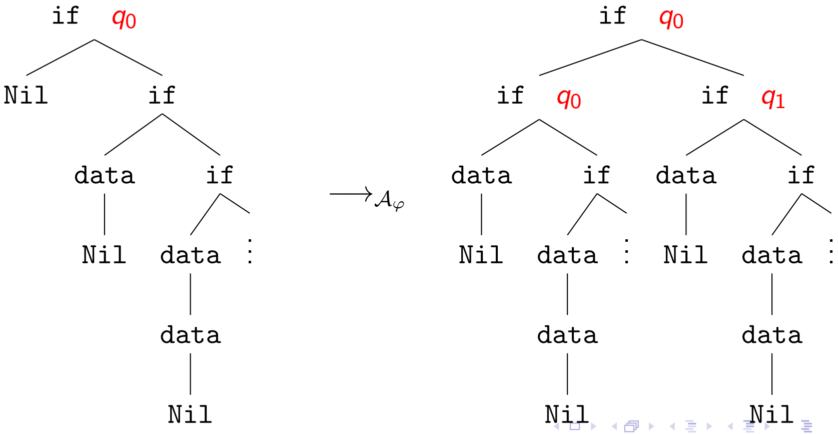
ATA: non-deterministic tree automata whose transitions may duplicate or drop a subtree.

Typically: $\delta(q_0, \texttt{if}) \;=\; (2, q_0) \wedge (2, q_1)$.

Alternating tree automata (ATA)

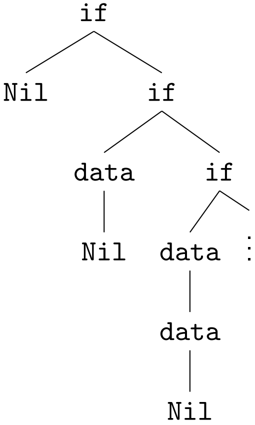
ATA: non-deterministic tree automata whose transitions may duplicate or drop a subtree.

Typically: $\delta(q_0, \mathtt{if}) \ = \ (2, q_0) \wedge (2, q_1)$.



Alternating parity tree automata

Express reachability with ATA: does every branch ends by `Nil`?



Problem: ATA execute coinductively.

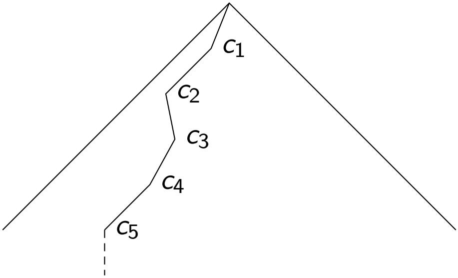
Solution: parity condition.

Alternating parity tree automata

Each state of an APT is attributed a color

$$\Omega(q) \in Col \subseteq \mathbb{N}$$

An infinite branch of a run-tree is winning iff the maximal color among the ones occuring infinitely often along it is even.



Alternating parity tree automata

Each state of an APT is attributed a color

$$\Omega(q) \in Col \subseteq \mathbb{N}$$

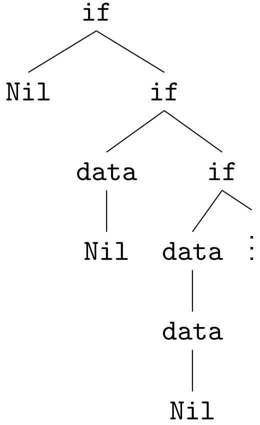
An infinite branch of a run-tree is winning iff the maximal color among the ones occuring infinitely often along it is even.

A run-tree is winning iff all its infinite branches are.

For a MSO formula φ:

$$\mathcal{A}_\varphi \text{ has a winning run-tree over } \langle \mathcal{G} \rangle \text{ iff } \langle \mathcal{G} \rangle \vDash \phi.$$

Alternating parity tree automata



$$Q = \{q\}$$

$$\Omega(q) = 1$$

$$\delta(\texttt{if}, q) = (1, q) \wedge (2, q)$$

$$\delta(\texttt{data}, q) = (1, q)$$

$$\delta(\texttt{Nil}, q) = \top$$

HOMC and intersection types

Alternating tree automata and intersection types

A key remark (Kobayashi 2009):

$$\delta(q_0, \mathtt{if}) \ = \ (2, q_0) \wedge (2, q_1)$$

can be seen as the intersection typing

$$\mathtt{if} \ : \ \emptyset \rightarrow (q_0 \wedge q_1) \rightarrow q_0$$

refining the simple typing

$$\mathtt{if} \ : \ o \rightarrow o \rightarrow o$$

Alternating tree automata and intersection types

A run-tree over $\mathtt{if} \quad T_1 \quad T_2$ is a derivation of $\emptyset \vdash \mathtt{if} \quad T_1 \quad T_2$:

$$\dfrac{\dfrac{\dfrac{}{\emptyset \vdash \mathtt{if} : \emptyset \to (q_0 \wedge q_1) \to q_0}\,\delta \qquad \color{red}{\emptyset}}{\emptyset \vdash \mathtt{if}\ T_1 : (q_0 \wedge q_1) \to q_0}\,\mathrm{App} \qquad \dfrac{\vdots}{\color{red}{\emptyset \vdash T_2 : q_0}} \qquad \dfrac{\vdots}{\color{red}{\emptyset \vdash T_2 : q_1}}}{\emptyset \vdash \mathtt{if}\ T_1\ T_2 : q_0}\,\mathrm{App}$$

Intersection types naturally lift to higher-order – and thus to $\mathcal{G}$, which finitely represents $\langle \mathcal{G} \rangle$.

Theorem (Kobayashi)

$S : q_0 \vdash S : q_0$ *iff* *the ATA $\mathcal{A}_\varphi$ has a run-tree over $\langle \mathcal{G} \rangle$.*

A type-system for verification: without parity conditions

(G.-Melliès 2014, from Kobayashi 2009 and Kobayashi-Ong 2009)

Axiom $$\frac{}{x : \bigwedge_{\{i\}} \ \theta_i :: \kappa \ \vdash \ x : \theta_i :: \kappa}$$

δ $$\frac{\{ (i, q_{ij}) \mid 1 \leq i \leq n, 1 \leq j \leq k_i \} \quad \text{satisfies} \quad \delta_A(q, a)}{\emptyset \vdash a : \bigwedge_{j=1}^{k_1} \ q_{1j} \to \ldots \to \bigwedge_{j=1}^{k_n} \ q_{nj} \to q :: o \to \cdots \to o}$$

App $$\frac{\Delta \vdash t : (\ \theta_1 \ \wedge \cdots \wedge \ \theta_k) \to \theta :: \kappa \to \kappa' \qquad \Delta_i \vdash u : \theta_i :: \kappa}{\Delta + \Delta_1 + \ldots + \Delta_k \ \vdash \ t\, u : \theta :: \kappa'}$$

λ $$\frac{\Delta, x : \bigwedge_{i \in I} \ \theta_i :: \kappa \ \vdash \ t : \theta :: \kappa'}{\Delta \ \vdash \ \lambda x \, . \, t : \left(\bigwedge_{i \in I} \ \theta_i\right) \to \theta :: \kappa \to \kappa'}$$

fix $$\frac{\Gamma \vdash \mathcal{R}(F) : \theta :: \kappa}{\overline{F : \theta :: \kappa \vdash F : \theta :: \kappa}}$$

Colored intersection types

A type-system for verification

(G.-Melliès 2014, from Kobayashi-Ong 2009)

$$\text{App} \quad \frac{\Delta \vdash t : (\Box_{c_1}\ \theta_1\ \wedge \cdots \wedge \Box_{c_k}\ \theta_k) \rightarrow \theta\ ::\ \kappa \rightarrow \kappa' \qquad \Delta_i \vdash u\ :\ \theta_i\ ::\ \kappa}{\Delta\ +\ \Box_{c_1}\Delta_1\ +\ \ldots\ +\ \Box_{c_k}\Delta_k\ \ \vdash\ \ t\,u\ :\ \theta :: \ \kappa'}$$

$+$ coloring of typing tree nodes and parity condition on derivations

A type system for verification

Theorem (G.-Melliès 2014, from Kobayashi-Ong 2009)

$S : q_0 \vdash S : q_0$ admits a winning typing derivation

iff

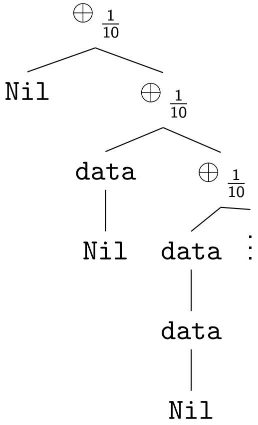
the alternating parity automaton $\mathcal{A}$ has a winning run-tree over $\langle \mathcal{G} \rangle$.

Static analysis: directly on the finite HORS $\mathcal{G}$.

Probabilistic automata

Probabilistic HOMC

```
IntList random_list() {
  IntList list = Nil;
  while(rand() > 0.1) {
    list := rand_int()::list;
  }
  return l;
}
```

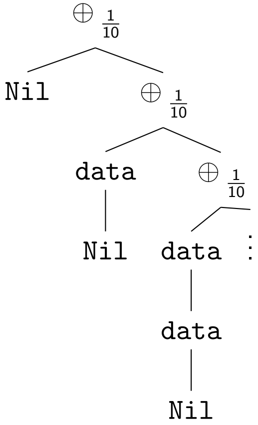


Probabilistic HOMC

Allows to represent probabilistic programs.

And to define higher-order regular MDP: those bisimilar to their encoding represented by a HORS.

(encoding of probabilities + payoffs in symbols)



Probabilistic automata

Idea: no longer verify ϕ but $Pr_{\geq p}\ \phi$.

- Step one: quantitative ATA.
- Step two: deal with colors and parity condition.

Probabilistic automata (PATA):

- ATA on non-probabilistic symbols
- $+$ probabilistic behavior on choice symbol $\oplus_p$

Run-tree: labels $(q,\ p_b,\ p_f)$.

The root of a run-tree of probability p is labeled $(q_0, 1, p)$, where p is the probability with which we want the tree to satisfy the formula.

Probabilistic alternating tree automata

Probabilistic behavior:

$$\oplus_p \quad (q,\, p_b,\, p_f)$$
$$b \qquad\qquad c$$

is labeled as

$$\oplus_p \quad (q,\, p_b,\, p_f)$$
$$b \quad (q,\, p \times p_b,\, p'_f) \qquad\qquad c \quad (q,\, (1-p) \times p_b,\, p_f - p'_f)$$

for some $p'_f \in [0, p_f]$ such that $p'_f \leq p \times p_b$ and $p_f - p'_f \leq (1-p) \times p_b$.

Example of PATA run

$\phi \quad = \quad$ "all the branches of the tree contain $\texttt{data}$"

is modeled by the PATA:

- $\delta_1(q_0, \texttt{data}) \;=\; (1, q_1),$
- $\delta_1(q_1, \texttt{data}) \;=\; (1, q_1),$
- $\delta_1(q_0, \texttt{Nil}) \;=\; \perp,$
- $\delta_1(q_1, \texttt{Nil}) \;=\; \top.$

Example of PATA run

- $\oplus_{\frac{1}{10}}$ $(q_0,\ 1,\ \frac{9}{10})$
  - Nil $(q_0,\ \frac{1}{10},\ 0)$
  - $\oplus_{\frac{1}{10}}$ $(q_0,\ \frac{9}{10},\ \frac{9}{10})$
    - data $(q_0,\ \frac{9}{100},\ \frac{9}{100})$
      - Nil $(q_1,\ \frac{9}{100},\ \frac{9}{100})$
    - $\oplus_{\frac{1}{10}}$ $(q_0,\ \frac{81}{100},\ \frac{81}{100})$
      - data $(q_0,\ \frac{81}{1000},\ \frac{81}{1000})$
        - data $(q_1,\ \frac{81}{1000},\ \frac{81}{1000})$
          - Nil $(q_1,\ \frac{81}{1000},\ \frac{81}{1000})$
      - $\vdots$

Intersection types for PATA

As for ATA, except for tree constructors:

$$\frac{\{\,(i, q_{ij}) \mid 1 \leq i \leq n, 1 \leq j \leq k_i\} \quad \text{satisfies} \quad \delta_A(q, a)}{\emptyset \vdash a \,:\, \bigwedge_{j=1}^{k_1} (q_{1j}, p_b, p_f) \to \ldots \to \bigwedge_{j=1}^{k_n} (q_{nj}, p_b, p_f) \to (q, p_b, p_f)}$$

$$\frac{p'_f \in]0, p_f[\quad \text{and} \quad p'_f \leq p \times p_b \quad \text{and} \quad p_f - p'_f \leq (1-p) \times p_b}{\emptyset \vdash \oplus_p \,:\, (q, p \times p_b, p'_f) \to (q, (1-p) \times p_b, p_f - p'_f) \to (q, p_b, p_f)}$$

$$\frac{q \in Q \quad \text{and} \quad p \times p_b \geq p_f}{\emptyset \vdash \oplus_p \,:\, (q, p \times p_b, p_f) \to \emptyset \to (q, p_b, p_f)}$$

$$\frac{q \in Q \quad \text{and} \quad (1-p) \times p_b \geq p_f}{\emptyset \vdash \oplus_p \,:\, \emptyset \to (q, (1-p) \times p_b,\ p_f) \to (q, p_b, p_f)}$$

Intersection types for PATA

Theorem

$$\emptyset \vdash S : (q_0, 1, p)$$

iff

the PATA $\mathcal{A}$ has a run-tree of probability p over the tree $\langle \mathcal{G} \rangle$ generated by $\mathcal{G}$.

Under connection Rel/non-idempotent types, we obtain a similar denotational theorem.

Note that $[\![o]\!] \;=\; Q \times [0,1] \times [0,1]$.

PATA and quantitative μ-calculus

Quantitative μ-calculus (McIver-Morgan): interpret ϕ not in $\mathbb{B}$ but in $[0, 1]$.

When all payoffs are 1, semantics = size of the set of branches satisfying ϕ:

$$\|\phi\|_{\mathcal{V}}^{\underline{\sigma},\bar{\sigma}} \cdot s \;=\; \int_{[\![\phi]\!]_{\mathcal{V}}^{\underline{\sigma},\bar{\sigma}} \cdot s} Val$$

Result holding for regular (= finite) Markov chains.

PATA and quantitative μ-calculus

Deal with infinite branches? PATA accept them all...

For trivial formulas (only ν, never μ/only color is 0 = safety properties) and all payoffs set to 1 (for commodity, can be patched):

$$\|\phi\|_{\mathcal{V}} \quad = \quad \sup\{p \in [0,1] \mid \text{there is a run-tree of probability } p\}$$

PATA acts similarly to the game interpretation, resolving non-determinism but playing all alternating choices in parallel.

The type system approach captures these safety properties.

How to capture the general parity condition?

Towards the parity condition

How to capture the general parity condition?

Idea: a colored run-tree of probability

$$p - p_{bad}$$

is

- a run-tree of probability p,
- where p_{bad} is the measure of the set of rejecting ($=$ odd-colored) branches in the run-tree.

Problem: relate the size of rejecting branches set throughout infinite β-reduction?

Thank you for your attention!

Towards the parity condition

How to capture the general parity condition?

Idea: a colored run-tree of probability

$$p - p_{bad}$$

is

- a run-tree of probability p,
- where p_{bad} is the measure of the set of rejecting ($=$ odd-colored) branches in the run-tree.

Problem: relate the size of rejecting branches set throughout infinite β-reduction?

Thank you for your attention!