Higher-Order Matching, Games and Automata

Colin Stirling
School of Informatics
University of Edinburgh

System = finite/infinite state transition graph

Model checking: $\text{so} \not= \phi$?

Apply automata-theoretic/game-theoretic techniques to solve it. Mostly computing monadic fixed points by traversing graph (possibly repeatedly).

Equivalence checking: $s \equiv t$?

Mostly computing dyadic fixed points e.g., bisimulation equivalence. May need some algebraic/combinatorial properties of generators of transition graph.
Active research goal: transfer these techniques to finite/infinite state systems with binding

- Deciding observational equivalence for fragments of idealized Algol (w.r.t finite value sets)  
  [Ghica+ McCusker 2000, Ong LICS 2002, ....]

- Model checking higher order trees  
  [Knapik, Niwinski, Urzyczyn 2002, Cavcal 2002,  
  Ong LICS 2006, ....]

- Application of tree automata to matching  
  [Comon+ Jurski 1997]
Higher order schemes

Base type 0 - finite/infinite trees with nodes labelled by elements of \( f_1, \ldots, f_n \):
- each \( f_i \) has arity \( > 0 \)

**Scheme**

\[ F_i(x_1, \ldots, x_{n_i}) = t_i, \quad 1 \leq i \leq m \]
- each \( F_i \) is typed

**Typed variables**

\[ t_i : O \]
- built from \( x_j \)'s, \( F_j \)'s and \( f_j \)'s + application

**Start closed expression**

\[ S : O \]

(Avoiding \( \lambda \)-terms)

Interpretation of a scheme: finite/infinite tree generated by \( S \).
Example: first-order

\[ F(x_1, x_2) = f(F(x, h(x_2)), x_2) \]

\[ F(b, b) \] start

\[ F(b, b) \rightarrow f \]

\[ F(b, h(b)) \]

\[ F(b, h(h(b))) \rightarrow f \]

\[ F(b, h(h(h(b)))) \rightarrow f \]

\[ \ldots \]

\[ \text{infinite tree} \]
2nd-order example

\[ F(x_1, x_2, x_3) = f(F(Gx_1, Hx_2, x_3), x_1(x_2(x_3))) \]

\[ G(x_1, x_2) = g(x_1(x_2)) \]

\[ H(x_1, x_2) = h(x_1(x_2)) \]

\[ F(g,h,a) \] _start_

\[ F(g,h,a) \rightarrow f \]

\[ F(Gg, Hh, a) \]

\[ g \quad h \quad a \]

infinite tree
Model checking problem

- Given $S$, does its tree have a decidable monadic 2nd-order theory?

- Solved positively for a subset of schemes, safe schemes
  - Knapik, Niwinski, Urzyczyn
  - Fossacs 2002
  - Uses geometry of interaction on infinite $\lambda$-terms + higher order push down automata

- Extended to all schemes
  - Ong LICS 2006
  - Uses game semantics on infinite $\lambda$-terms
\[ F_{x_1 x_2 x_3} = f ( F (G x_1) (H x_2) x_3, x_1 (x_2 (x_3))) \]
\[ G x_1 x_2 = g (x_1 (x_2)) \]
\[ H x_1 x_2 = h (x_1 (x_2)) \]

Follow Ong's transformation into normal form

\[ F = \lambda x_1 x_2 x_3. f (@ \ldots \ldots, \lambda. x_1 (\lambda. x_2 (\lambda. x_3))) \]
\[ \vdots \]

Infinite \( \lambda \)-term (can be folded into a finite tree with back edges)
"capture" infinite tree

using game semantics

transform parity tree automata on

into tree automata on "finite" unfolding of λ-term
Equivalence problem

- Given two schemes $S_1, S_2$ do they generate the same tree? $S_1 \sim S_2$?

- For order 1:
  
  $S_1 \sim S_2$?

  Courcelle 1978

DPDA equivalence problem

do 2 configurations of a deterministic pushdown automaton generate the same language?


- Order > 1
  
  Open + hard

Simply typed $\lambda$-calculus

**Church style**

**Base type**

**Types**

\[ A ::= \text{O} | A \rightarrow A \]

A type always has form \( A_1 \rightarrow (...) (A_n \rightarrow \text{O}) (...) \) or \( (A_1, \ldots, A_n, \text{O}) \)

**Order**

\( \text{order} (\text{O}) = 1 \)

\( \text{order} ((A_1, \ldots, A_n, \text{O})) = \max \{ \text{order}(A_i) \} + 1 \)

**Terms**

Variables, constants have unique type

\( x : A, f : A \Rightarrow x : A, f : A \in T \)

\( x : A, t : B \in T \Rightarrow \lambda x.t : A \rightarrow B \in T \)

\( t : A \rightarrow B, u : A \in T \Rightarrow t u : B \)

**Closed**

\( t : A \) contains no free variables

**$\alpha$-equivalence**

\( t, t' \) are renamings of each other w.r.t bound variables
Reduction et al

\[(\beta)\quad (\lambda x. t)v \rightarrow_\beta t\{v/x\}\]

\[(\eta)\quad \lambda x. (tx) \rightarrow_\eta t \quad x \text{ not free in } t\]

Fact

Each term has a unique $\beta\eta$-normal form (up to $\alpha$-equivalence)

$\equiv_{\beta\eta}$ same normal form

$\eta$-long normal form

- $t : O$
- $u : O$ or $u \, t_1 \ldots t_k$

$\lambda x_1 \ldots x_n. \, t'$

Abbreviates $\lambda x_1 \ldots \lambda x_n. \, t'$

$t' : O$ in long normal form

Fact

When $\eta$-long normal forms $\equiv_\beta$ is $\equiv_{\beta\eta}$
Higher-order unification

\[ \nu = \upsilon \]
contains free variables \( x_1, \ldots, x_n \)

Solution: terms \( t_1, \ldots, t_n \) (\( t_i \) same type as \( x_i \))
such that
\[ \nu \Theta = \beta \eta \upsilon \Theta \]
\[ \Theta = \{ t_1 / x_1, \ldots, t_n / x_n \} \]

Decision question: given \( \nu = \upsilon \) does it have a solution?

order is largest order of variable \( x_i \)

Undecidable (even at order 2)

\[ \text{can encode } +, \times \text{ over Church numbers} \]
\[ \therefore \text{Hilbert's 10th Problem as a unification problem} \]

[Huet 1976]

[Huet '72]

[Goldfarb '81]

Post's correspondence
Higher order matching

\[ v = u \quad \text{BUT} \quad u \text{ closed} \]

Solution: terms \( t_1, \ldots, t_n \) such that

\[ v \{ t_1/x_1, \ldots, t_n/x_n \} =_{\beta \eta} u \]

Decision question: given \( v = u \) does it have a solution?

Order: largest order of \( x_i \) in \( v \).

- Decidable order 2 \((\text{Huet '76})\)
- Order 3 \((\text{Dowek '93})\)
- Order 4 \((\text{Padovani 2000})\)
- Special cases \((\text{Padovani, Schubert, \ldots})\)
- Undecidable for \( =_{\beta} \) \([\text{Loader 2003}]\)
Matching essentially monadic

Given $v = u$ with $x_1 : A_1, \ldots, x_n : A_n$ free in $v$.

\[ x \lambda x_1 \ldots x_n. v = u \]

one free variable

"conceptually" simpler problem

interpolation problem

\[ x \cdot v_1 \ldots v_n = u \]

\[ x : (A_1, \ldots, A_n, 0) \]

Solution: closed $t$ in normal form

\[ t \cdot v_1 \ldots v_n =_{\beta\eta} u \]
Padovani Proof at Order 4

- \( t, t': A \)

- \( t \equiv^u_A t' \) if \( t, t' \) solve same interpolation problems with right term \( u \)

\[ \equiv^u_A : \text{"observational equivalence"} \]

- Set of normal forms: \( A / \equiv^u_A \) is finite

- Up to order 4, representatives for each equivalence class can be decided.

- [Comon+Jurski 1997]

Fact: Set of solutions to 4th order interpolation is regular (modulo...) = recognizable by a tree automaton.

States of automaton use Padovani representatives \( \equiv^u_A \) for subterms \( u' \) of \( u \), subtypes \( A' \) of \( A \).
Model checking vs Satisfiability checking

- Given $M$ and $\phi$, $M \models \phi$?

Given $\phi$, is there an $M$ s.t. $M \models \phi$?

Similar for matching:

given $x_1 \ldots x_n = u$
and $t$ is $t_1 \ldots t_n = \beta_n u$

given $x_1 \ldots x_n = u$
is there a non-trivial $t$
s.t. $t_1 \ldots t_n = \beta_n u$
Interpolation game

- Given $x \, v_1 \ldots v_n = u$ with $x : A$, and $t : A$ in normal form.

- Define game whose board is:
Like game semantics

Model β-reduction without destroying terms by substituting into them
\[(\lambda y_1. y_2. y_1) (\lambda y_3. f y_3 y_3) = f a a\]
Given $x v_1..v_n = u$, $x : A$ and $t : A$ there is the game $G(t, v_1..v_n, u)$.

**Fact** $A$ loses $G(t, v_1..v_n, u)$ iff $tv_1..v_n = \beta \eta u$.

- Abstract from game playing (jumping) to a tree automaton
- Variable assumptions

$z : O \quad (z, r, \phi)$

$z : (A_1, ..., A_n, O) \quad (z, r, \Gamma_1 u ... u \Gamma_n)$

$r$ subterm of $u$

$\Gamma_i$ is a set of assumptions of variables of type $A_i$

- State of automaton: $\{(r_1, \Gamma_1), ..., (r_k, \Gamma_k)\}$

$\Gamma_i$'s are variable assumptions

- Restrict variables + constants allowed in $t$ based on/borrowed from [Ong 2006]
Transitions of form \( z \Lambda_i, \ldots \Lambda_k \Rightarrow \Lambda \)

\[
\begin{tikzpicture}
  \node (z) {z};
  \node (x1) at (z -| 2mm) {$\lambda x_1, \ldots \lambda x_k$};
  \draw (z) -- (x1);
\end{tikzpicture}
\]

if \((r, r') \in \Lambda\) then \((z, r, r') \in \Gamma\)

and \(\Gamma'\) is

\[
\left\{ (y_{i1}, r_{i1}, r'_{i1}), \ldots, (y_{im_i}, r_{im_i}, r'_{im_i}), \ldots \right\}
\]

and \((r_{ij}, r'_{ij} \cup \Sigma_{ij}) \in \Lambda_i\)

and \(\ldots\)

\[
\vdots
\]
Fact

Given \( x v_1 \ldots v_n = u \) with \( x : A \) order(\( A \)) > 0, the set of solutions built out of a fixed finite alphabet of variables/constants is regular (recognized by a tree automaton).

[ LICS 2007 ]

generalizes Comon + Jurski’s result

- proof technique more elementary (games instead of observational equivalence classes)

- Not enough to prove decidability for order(\( A \)) > 4 of matching
Problem

When $A$ has order $> 4$, set of closed terms may require infinitely many different variables.

$$A = (((((0,0),0),0),0),0,0)$$

$\lambda x, x_2$

$x_1 : (((0,0),0),0)$

$\lambda z_1$

$x_1 : (((0,0),0),0)$

$\lambda z_n$

$z_n : (0,0)$

$\lambda z_1$

$z_1 : (0,0)$

$\lambda z_2$

$z_2 : 0$

Each occurrence of $x_i$ introduces new variables $z_i$ that have to be distinct. $z_i$'s are embedded.
Satisfiability checking

Model checking games can solve satisfiability checking.

Prove small model property

if there is a model for \( \phi \),
then there is one of size at most \( f(\phi) \)

Techniques

- unravel model into tree
- no play enters a subtree: delete it
- two different plays have positions \((s, EX\phi), (s, EX\psi)\)
  and only one state \(t\). \(s \rightarrow t\). Transform model

Play separation: \(\exists\) chooses left \(t\) in \((s, EX\phi)\)
\(\exists\) chooses right \(t\) in \((s, EX\psi)\)
For matching analogous

- unravel = unfold w.r.t playing which incorporates binding

Approximate tree model property

“reduces game play jumping” (localizes it)

- Very complicated.

Non-elementary: small model property grows exponentially with order

- Find a simpler proof!