Operational Algorithmic Game Semantics

Benedict Bunting    Andrzej Murawski

Department of Computer Science
University of Oxford

LICS, 2023
Overview

- We study contextual equivalence for a call-by-push-value language with first-order state.

- We build a fully abstract model for this language using Operational Game Semantics.

- We refine this model into an automaton, for selected types.

- Using this we derive an algorithm for determining if two terms are contextually equivalent.

- We subsume all decidability results for finitary Idealised Algol and RML with iteration, achieved using finite alphabet automata.
**Call-By-Push-Value (Levy, 1999, TLCA)**

<table>
<thead>
<tr>
<th>Value Type</th>
<th>( \sigma \triangleq U\tau \mid \text{Unit} \mid \text{Int} \mid \text{Ref} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>Computation Type</td>
<td>( \tau \triangleq F\sigma \mid \sigma \to \tau )</td>
</tr>
</tbody>
</table>

\[
\Sigma; \Gamma \vdash^\nu () : \text{Unit} \quad \Sigma; \Gamma \vdash^\nu \hat{n} : \text{Int} \quad (x, \sigma) \in \Gamma \quad \Sigma; \Gamma \vdash^\nu x : \sigma
\]

\[
\Sigma; \Gamma \vdash^c M : \tau \quad \Sigma; \Gamma \vdash^c V : U\tau \quad \Sigma; \Gamma \vdash^c \text{return } V : F\sigma
\]

\[
\Sigma; \Gamma \vdash^c M : F\sigma \quad \Sigma; \Gamma, x : \sigma \vdash^c N : \tau \quad \Sigma; \Gamma \vdash^c \lambda x^\sigma.M : \sigma \to \tau
\]

\[
\Sigma; \Gamma \vdash^c M : \sigma \to \tau \quad \Sigma; \Gamma \vdash^c V : \sigma \quad \Sigma; \Gamma \vdash^c \text{ref } V : F\text{Ref}
\]

\[
\Sigma; \Gamma \vdash^\nu V_{\text{read}} : UF\text{Int} \quad \Sigma; \Gamma \vdash^\nu V_{\text{write}} : U(\text{Int} \to F\text{Unit}) \quad \Sigma; \Gamma \vdash^\nu \text{MkVar } V_{\text{read}} V_{\text{write}} : \text{Ref}
\]
CBPV: Contextual Pre-order and Equivalence

Termination: \((M, h) \Downarrow_{\text{ter}}\)

Value Context

\[ V_C \triangleq \text{thunk } C \mid \text{MkVar } V C \mid \text{MkVar } V V C \]

Context

\[ C \triangleq \bullet \mid \text{force } V C \mid \text{return } V C \mid \lambda x^\sigma . C \mid \text{let } x \text{ be } V.C.M \]

\[ \mid \text{let } x \text{ be } V.C \mid C \text{ to } x.M \mid M \text{ to } x.C \]

\[ \mid CV \mid MV_C \mid !V_C \mid V_C := V \]

\[ \mid \text{while } C \text{ do } M \mid \text{while } M \text{ do } C \]

\[ \mid \text{case } V \text{ of } (M_i)_{i<j} , C , (M_i)_{j<i} \]

**Definition**

Given computations \(\Gamma \vdash^c M_1, M_2 : \tau\), we define \(\Gamma \vdash^c M_1 \preceq_{\text{CBPV}}^{\text{ter}} M_2\) to hold, when for all contexts \(\vdash^k C : \tau \implies F\sigma\), we have \((C[M_1], \emptyset) \Downarrow_{\text{ter}}\) implies \((C[M_2], \emptyset) \Downarrow_{\text{ter}}\).

We write \(\sim_{\text{CBPV}}^{\text{ter}}\) for the equivalence induced by \(\preceq_{\text{CBPV}}^{\text{ter}}\).
Example

Let $\Gamma = \{ f : U(\text{UFInt} \rightarrow \text{FUnit}) \}$

\[
M_1 \triangleq (\text{force } f) \text{ thunk } (\text{return } \hat{1}) \\
M_2 \triangleq \text{ref} \hat{0} \text{ to } x \cdot x := \hat{1}; (\text{force } f) \text{ thunk } (!x \text{ to } z. \text{return } z)
\]

Then $M_1 \equiv_{\text{CBPV}}^\text{ter} M_2$
Game Semantics

- Idea: model a term as a strategy for a game of question and answer between it and its context.

- Originally due to Abramsky, Jagadeesan and Malacaria, Hyland and Ong, and Nickau (all early 90’s).

\[ \Gamma \vdash M \simeq_{ctx} N : \Theta \iff [\Gamma \vdash M : \Theta] = [\Gamma \vdash N : \Theta] \]

\[ [\Gamma \vdash MN : \Theta_2] = \langle [\Gamma \vdash M : \Theta_1 \rightarrow \Theta_2], [\Gamma \vdash N : \Theta_1] \rangle; \text{ev} \]
Algorithmic Game Semantics

▶ Use automata to represent strategies (and so terms).
▶ $M_1 \simeq_{ctx} M_2$ iff $\mathcal{L}(\mathcal{A}_{M_1}) = \mathcal{L}(\mathcal{A}_{M_2})$
▶ Early results by (Ghica and McCusker, 2000, ICALP), (Ong, 2002, LICS), (Murawski and Walukiewicz, 2005, FoSSaCS), (Hopkins et al., 2011, ICALP).

$\mathcal{A}_{if \, x^\beta \, then \, M_1 \, else \, M_2}$

$\mathcal{A}_{let \, x^\beta = N \, in \, M}$
Operational Game Semantics

- Strategies represented as sets of traces (concrete!)
- ...generated by a labelled transition system (LTS) derived from the operational semantics.
- Most relevant results by (Laird, 2007, ICALP), (Jaber, 2015, FoSSaCS), we follow (Jaber and Murawski, 2021, ESOP).
- Other related work: (Jeffrey and Rathke, 2005, ESOP), (Lassen and Levy, 2007, CSL).
- We claim it is ‘easier’ to obtain an automaton from OGS.

**Example**

\[
\bar{f}(rw4, c_1) \quad w(\hat{1}, c_2) \quad \bar{c}_2((())) \quad c_1((())) \quad \bar{c}(g) \quad g(h, c_3) \quad \bar{h}(\epsilon, c_4) \quad c_4(2) \quad \bar{c}_3(3)
\]
OGS: Transition System $\mathcal{L}_{CBPV}$

$$C^\rho_c \triangleq (\langle M\{\rho\}, c, \emptyset, NO, \emptyset, [NO \mapsto \emptyset] \rangle, \bot)$$

(PT) $\langle M, c, \gamma, \phi, h, H \rangle \xrightarrow{\tau} \langle N, c, \gamma, \phi, h', H \rangle$ when $(M, h) \rightarrow (N, h')$

(PA) $\langle \text{return } V, c, \gamma, \phi, h, H \rangle \xrightarrow{\text{c}(A)} \langle \gamma \cdot \gamma', \phi \uplus \nu(A), h, H, H(c) \uplus \nu(A) \rangle$ when $c : \sigma$, $(A, \gamma') \in \text{AVal}_\sigma(V)$

(PQ) $\langle K[(\text{force } f) \xrightarrow{V}], c, \gamma, \phi, h, H \rangle \xrightarrow{\text{f}(\overline{A}, c')/(c', (K,c))} \langle \gamma \cdot \gamma', \phi \uplus \phi', h, H, H(f) \uplus \nu(\overline{A}) \rangle$ when $f : U_{\tau}$, $(\overline{A}, \gamma') \in \text{AVal}(\overline{V})$, $\sigma = \text{RType}(\tau)$, $c' : \sigma$ and $\phi' = \nu(\overline{A}) \uplus \{c'\}$

(OA) $\langle \gamma, \phi, h, H, Fn \rangle \xrightarrow{\text{c}(A),(c,(K,c'))} \langle K[\text{return } A], c', \gamma, \phi \uplus \nu(A), h, H \cdot [\nu(A) \mapsto Fn] \rangle$ when $c : \sigma$, $A : \sigma$

(OQ) $\langle \gamma, \phi, h, H, Fn \rangle \xrightarrow{\text{f}(\overline{A}, c)} \langle (\text{force } V)\overline{A}, c, \gamma, \phi \uplus \phi', h, H \cdot [\phi' \mapsto Fn] \rangle$ when $f \in Fn$, $f : U_{\tau}$, $\overline{A} \in \text{ASeq}(\tau)$, $\sigma = \text{RType}(\tau)$, $c : \sigma$, $\gamma(f) = V$ and $\phi' = \nu(\overline{A}) \uplus \{c\}$

Given $N \subseteq \text{Names}$, $[N \mapsto V]$ stands for the map $[n \mapsto V \mid n \in N]$. 
Full Abstraction

Theorem (Full Abstraction)

For any CBPV computations $\Gamma \vdash^c M_1, M_2 : F\sigma$, then
$\Gamma \vdash^c M_1 \sim_{\text{ter}}^{\text{CBPV}} M_2$ iff $\text{Tr}_{\text{CBPV}}(\Gamma \vdash^c M_1) \subseteq \text{Tr}_{\text{CBPV}}(\Gamma \vdash^c M_2)$.

Example

$M_i \triangleq \text{force } f \text{ to } g_1.\text{force } f \text{ to } g_2.\text{force } g_i$

$\bar{f}(\epsilon, c_1) c_1(g_1) \bar{f}(\epsilon, c_2) c_2(g_2) \bar{g}_1(\epsilon, c_3) c_3(() \bar{c}_0(()))$ is in $\text{Tr}(C_{M_1}^{\rho,c_0})$
but not $\text{Tr}(C_{M_2}^{\rho,c_0})$

$\bar{f}(\epsilon, c_1) c_1(g_1) \bar{f}(\epsilon, c_2) c_2(g_2) \bar{g}_2(\epsilon, c_3) c_3(() \bar{c}_0(()))$ is in $\text{Tr}(C_{M_2}^{\rho,c_0})$
but not $\text{Tr}(C_{M_1}^{\rho,c_0})$
Finite Alphabet Traces

For $M_i = \text{force } f \text{ to } g_1.\text{force } f \text{ to } g_2.\text{force } g_i$, traces have the form

$$t_i = \bar{f}(\epsilon, c_1) c_1(g_1) \bar{f}(\epsilon, c_2) c_2(g_2) \bar{g}_i(\epsilon, c_3) c_3() \bar{c}_0()$$

**Base**

$$\Delta^\Delta(\text{Tr}(C_{M_i}^{\rho, c_0})) = \{ t = \bar{f}(\epsilon, d) d(g) \bar{f}(\epsilon, d) d(g) \bar{g}(\epsilon, e) e() \bar{c}_0() \}$$

**Rename**

$$\Delta^\Delta(\text{Tr}(C_{M_1}^{\rho, c_0})) = \{ \bar{f}(\epsilon, d) d(g) \bar{f}(\epsilon, d) d(g) \bar{g}(\epsilon, e) e() \bar{c}_0(), \bar{f}(\epsilon, d) d(\hat{g}) \bar{f}(\epsilon, d) d(\hat{g}) \bar{g}(\epsilon, e) e() \bar{c}_0(), \bar{f}(\epsilon, d) d(g) \bar{f}(\epsilon, d) d(\hat{g}) \bar{g}(\epsilon, e) e() \bar{c}_0() \}$$

$$\Delta^\Delta(\text{Tr}(C_{M_2}^{\rho, c_0})) = \{ \bar{f}(\epsilon, d) d(g) \bar{f}(\epsilon, d) d(g) \bar{g}(\epsilon, e) e() \bar{c}_0(), \bar{f}(\epsilon, d) d(\hat{g}) \bar{f}(\epsilon, d) d(\hat{g}) \bar{g}(\epsilon, e) e() \bar{c}_0(), \bar{f}(\epsilon, d) d(g) \bar{f}(\epsilon, d) d(\hat{g}) \bar{g}(\epsilon, e) e() \bar{c}_0() \}$$

**Corollary**

For PTR-computations $\Gamma \vdash^c M_1, M_2 : F\sigma$, continuation name $c : \sigma$, a $\Gamma$-assignment $\rho$ and $\Delta = (TB, CB, \rho, c, \text{Suc}_T, \text{Suc}_C)$, we have $\text{Tr}(C_{M_1}^{\rho, c}) = \text{Tr}(C_{M_2}^{\rho, c})$ iff $\text{Rename}^\Delta(\text{Tr}(C_{M_1}^{\rho, c})) = \text{Rename}^\Delta(\text{Tr}(C_{M_2}^{\rho, c}))$. 
Recycling Scheme I

\[ N_i \triangleq (\text{force } f) (\text{thunk return } \widehat{1}) \text{ to } g_1. \]
\[(\text{force } f) (\text{thunk return } \widehat{2}) \text{ to } g_2. \text{force } g_i \]

Need to distinguish occurrences of the same name during reduction; use \textbf{indexing}.

- Consider \( t \ f(\overrightarrow{A}, c) \ t' \ \overline{c}(A') \ t'' \), when \( A' \) has a non-thunk type
- Names introduced in \( f(\overrightarrow{A}, c) \ t' \) can never appear in \( t'' \)

\[ \langle \text{return } V, c^i, \gamma, h, H, i_h, \eta, \mu, l \rangle \overline{c}(V) \]
\[ \xrightarrow{\overline{c}(V)} \langle \gamma<\eta', h<i'_h, H<\eta', H(c^i), i'_h, \eta', \mu<\eta', l \rangle \]
when \( c \neq c_0 \) and \( (i'_h, \eta') = \mu(c^i) \)
Recycling Scheme II

- Consider \( t \Rightarrow f(\overrightarrow{A}, c) t' \Rightarrow g(\overrightarrow{A'}, d) t'' d(A) \) when \( \overrightarrow{A'} \) contains no thunk names, \( g \) originated in \( \overrightarrow{A} \)

- Then names introduced in \( f(\overrightarrow{A}, c) t' \Rightarrow g(\overrightarrow{A'}, d) \) cannot be used in \( t'' \)

\[
(PQ) \quad \langle K[(\text{force } f^i) \overrightarrow{V}], \gamma, h, H, i_h, \eta, \mu, l \rangle \xrightarrow{f(\overrightarrow{V}, c)/(c^0, (K, c^i), P)}
\langle \gamma < \eta', h < i^l_h, H < \eta', H(f^i), i^l_h, \eta', \mu < \eta', l \rangle
\]
when \( f \) is a level 2 name, and \( (i^l_h, \eta') = \mu(f^i) \), \( \text{Suc}_C(f) = c \), and \( P = (i_h, \eta, \gamma \geq \eta', h \geq i^l_h, H \geq \eta', \mu \geq \eta') \)

\[
(OA) \quad \langle \gamma, h, H, F_n, i_h, \eta, \mu, l \rangle \xrightarrow{c(\beta(A)), (c^0, (K, c^l), P)}
\langle K[\text{return } A], \gamma \cdot \gamma', h, H', i^l_h, \eta', \mu', l' \rangle
\]
when \( c : \sigma, P = (i^l_h, \eta'', \gamma', h', H'', \mu'') \), \( (A', \eta') \in \text{IVals}_\sigma^\Delta (c, \eta'') \)
and if \( l = 1 \) then \( A = A', l' = 1 \)
else \( A \in \text{Select}(A') \), and \( l' = \text{IsMark}(A) \); and
\( H' = H \cdot H'' \cdot [\nu(A) \mapsto F_n] \), and \( \mu' = \mu \cdot \mu'' \cdot [\nu(A) \mapsto (i_h, \eta)] \)
Refined transition system $\mathcal{L}^\Delta_{\text{PTR}}$

\[
\begin{align*}
\text{(P)} & \quad \langle M, \sigma, \gamma, h, H, i_h, \eta, \mu, l \rangle \xrightarrow{\tau} \langle N, \sigma, \gamma, h', H, i'_h, \eta', \mu, l' \rangle \\
\text{when } (M, h, i_h, \eta) \rightarrow_e (N, h', i'_h, \eta') \\
\text{(PA)} & \quad \langle \text{return } V, c_0 \rangle, \gamma, h, H, i_h, \eta, \mu, l \xrightarrow{\epsilon_0(\beta(A))} \langle \gamma \cdot \gamma', h, H(c_0) \uplus \nu(A), i_h, \eta', \mu, l \rangle \\
\text{when } c_0 : \sigma, (A, \gamma', \eta') = \text{IVal}^\Delta_\sigma(c_0, V, \eta) \\
\text{(PA)} & \quad \langle \text{return } V, c, \gamma, h, H, i_h, \eta, \mu, l \rangle \xrightarrow{\epsilon(V)} \langle \gamma \cdot \gamma', h, H(c_0) \uplus \nu(A), i_h, \eta', \mu, l \rangle \\
\text{when } c \neq c_0 \text{ and } (i_h, \eta') = \mu(c) \\
\text{(PQ)} & \quad \langle K[(\text{force } f^i) \overrightarrow{V}], c, \gamma, h, H, i_h, \eta, \mu, l \rangle \xrightarrow{f(\overrightarrow{V}, \epsilon) / (\epsilon_0, (K,c^0), p)} \langle \gamma \cdot \gamma', h, H(f^i) \uplus \nu(A), i_h, \eta', \mu, l \rangle \\
\text{when } f \text{ is not a level 2 name}, (A, \gamma', \eta') \in \text{IVal}^\Delta_\sigma(f, \overrightarrow{V}, \eta), \text{ and Succ}(f) = c \\
\text{(PQ)} & \quad \langle K[(\text{force } f^i) \overrightarrow{V}], c, \gamma, h, H, i_h, \eta, \mu, l \rangle \xrightarrow{f(\overrightarrow{V}, \epsilon) / (\epsilon_0, (K,c^0), p)} \langle \gamma \cdot \gamma', h, H(f^i) \uplus \nu(A), i_h, \eta', \mu, l \rangle \\
\text{when } f \text{ is a level 2 name}, (i_h, \eta') = \mu(f^i), \text{ Succ}(f) = c, \text{ and } P = (i_h, \eta, \gamma \geq \gamma', h \geq h', H \geq \eta', \mu \geq \mu') \\
\text{(OA)} & \quad \langle \gamma, h, H, F_n, i_h, \eta, \mu, l \rangle \xrightarrow{c(\beta(A)),(c,\epsilon)} \langle K[\text{return } A], c, \gamma, h, H \cdot [\nu(A) \mapsto F_n], i_h, \eta', \mu, l' \rangle \\
\text{when } c : \sigma, (A', \eta') \in \text{IVals}^\Delta_\sigma(c, \eta) \text{ and if } l = 1 \text{ then } A = A', l' = 1 \text{ else } A \in \textbf{Select}(A') \text{ and } l' = \textbf{IsMark}(A) \\
\text{(OA)} & \quad \langle \gamma, h, H, F_n, i_h, \eta, \mu, l \rangle \xrightarrow{c(\beta(A)),(c,\epsilon)} \langle K[\text{return } A], c, \gamma, h, H', i_h, \eta', \mu, l' \rangle \\
\text{when } c : \sigma, P = (i_h, \eta'', \gamma', h', H'', \mu''), (A', \eta') \in \text{IVals}^\Delta_\sigma(c, \eta'') \text{ and if } l = 1 \text{ then } A = A', l' = 1 \text{ else } A \in \textbf{Select}(A') \text{ and } l' = \textbf{IsMark}(A); \text{ and } H' = H \cdot H'' \cdot [\nu(A) \mapsto F_n] \text{, and } \mu' = \mu \cdot \mu'' \cdot [\nu(A) \mapsto (i_h, \eta)] \\
\text{(OQ)} & \quad \langle \gamma, h, H, F_n, i_h, \eta, \mu, l \rangle \xrightarrow{f(\overrightarrow{A}, \epsilon)} \langle \text{force } V\overrightarrow{A}, c, \gamma, h, H \cdot [\nu(A'), c \mapsto F_n], i_h, \eta', \mu', l \rangle \\
\text{when } f^i \in F_n \text{, } (\overrightarrow{A}, \eta'') \in \text{IValseq}^\Delta(f, \eta), \text{ Succ}(f) = c, \eta(c) = j, \eta' = \eta''[c \mapsto j + 1], \gamma(f^i) = V, \text{ and } \text{ if } l = 1 \text{ then } A = A', l' = 1 \text{ else } A \in \textbf{Select}(A') \text{ and } l' = \textbf{IsMark}(A) \text{; and } \mu' = \mu \cdot [\nu(A'), c \mapsto (i_h, \eta)]
\end{align*}
\]

\[\text{Tr}_{\text{PTR}}(\Gamma \vdash c M : F \sigma) \triangleq \{ (\Delta, t) \mid \Delta \text{ is a } (\Gamma, F \sigma)\text{-name scheme, } t \in \text{Tr}_{\text{PTR}}(C_{\text{M}}^\Delta), t \text{ is complete} \} \]
Correctness and Decidability

Lemma
\[ \text{Tr}_{\text{PTR}}(\mathcal{C}_M^{\text{PTR},\Delta}) = \text{Rename}^\Delta(\text{Tr}_{\text{CBPV}}(\mathcal{C}_M^{\rho,c_0})). \]

Theorem (PTR Full Abstraction)
For any PTR computations \( \Gamma \vdash M_1, M_2 : F\sigma \), then \( \Gamma \vdash M_1 \sim_{\text{ter}}^{\text{CBPV}} M_2 \) iff \( \text{Tr}_{\text{PTR}}(\Gamma \vdash M_1) \subseteq \text{Tr}_{\text{PTR}}(\Gamma \vdash M_2) \).

Lemma
For PTR-computation \( \Gamma \vdash^c M : F\sigma \) and \( (\Gamma, \sigma) \)-name scheme \( \Delta \), one can effectively construct a deterministic Visibly Pushdown Automaton (Alur and Madhusudan (2004)) accepting \( \text{Tr}_{\text{PTR}}^\Delta(\mathcal{C}_M^{\text{PTR},\Delta}) \). If \( M \) is in canonical form, the construction can be carried out in exponential time.

Theorem
Contextual approximation for the PTR-fragment of CBPV is decidable. For computations in canonical form, it is decidable in exponential time.
PTR Fragment

- Polarise occurrences of $U$ (corresponding to question actions)
- $U^+$ for occurrences of $U$ that produce O-questions, and $U^-$ for P-questions
- Problematic types: $FU^+FU^+F\text{Int}$ and $FU^+(U^-(U^+F\text{Int} \rightarrow F\text{Unit}) \rightarrow F\text{Unit})$
- PTR-fragment: forbids nested occurrences of $U^+$, while allowing nested occurrences of $U^-$

Definition

A CBPV computation $\Gamma \vdash^c M : F\sigma^P$ is in the **P-thunk-restricted** (PTR) fragment when all types in $\Gamma$ can be generated by $\sigma^2$ in the grammar below.

\[
\begin{align*}
\sigma^2 & \triangleq \sigma^1 \mid U\tau^2 \\
\sigma^P & \triangleq \sigma^0 \mid \text{Ref} \mid U\tau^P \\
\tau^2 & \triangleq F\sigma^2 \mid \sigma^P \rightarrow \tau^2 \\
\tau^P & \triangleq F\sigma^0 \mid \sigma^1 \rightarrow \tau^P \\
\tau^1 & \triangleq F\sigma^1 \mid \sigma^0 \rightarrow \tau^1 \\
\sigma^0 & \triangleq \text{Int} \mid \text{Unit}
\end{align*}
\]
### IA and RML

<table>
<thead>
<tr>
<th>RML type</th>
<th>CBPV value types</th>
<th>IA type</th>
<th>CBPV computation types</th>
</tr>
</thead>
<tbody>
<tr>
<td>Int, Unit, Ref</td>
<td>Int, Unit, Ref</td>
<td>expr, com</td>
<td>FInt, FUnit</td>
</tr>
<tr>
<td>$\sigma_1 \to \sigma_2$</td>
<td>$U(\sigma_1^{\text{RML}} \to F\sigma_2^{\text{RML}})$</td>
<td>var</td>
<td>Int $\to$ Int $\to$ FInt</td>
</tr>
<tr>
<td></td>
<td></td>
<td>$\tau_1 \to \tau_2$</td>
<td>$U\tau_1^{\text{IA}} \to \tau_2^{\text{IA}}$</td>
</tr>
</tbody>
</table>

- We provide translations from Idealised Algol and RML into CBPV
- PTR-fragment subsumes
  - Third-Order IA (Murawski and Walukiewicz, 2005, FoSSaCS)
  - O-Strict RML (Hopkins et al., 2011, ICALP)
DFA Fragment

- DFA: $\Gamma \vdash^c M : F\sigma^1$, where each type in $\Gamma$ is a $\sigma^2$

\[
\begin{align*}
\sigma^2 & \triangleq \sigma^1 \mid U_{\tau^1} \\
\sigma^1 & \triangleq \sigma^0 \mid \text{Ref} \mid U_{\tau^0} \\
\sigma^0 & \triangleq \text{Int} \mid \text{Unit}
\end{align*}
\]

- DFA fragment subsume

  - Second-Order IA (Ghica and McCusker, 2000, ICALP)

  - Bi-Strict $\text{RML}_{f}^-$ (Murawski, 2005, Theor. Comput. Sci.)
Conclusions

▸ Provide full-abstract trace model for CBPV.

▸ A new decidability result; first for CBPV.

▸ Subsume 4 results from the literature.

▸ A new approach: **Operational Algorithmic Game Semantics**

▸ OGS models are (generally) simpler to construct

▸ More intuitive

▸ Retain operational character suitable for further program analysis tasks

▸ We believe that it can be easily adapted for other frameworks

▸ Next steps: more powerful automata
Conclusions

▶ Provide full-abstract trace model for CBPV.

▶ A new decidability result; first for CBPV.

▶ Subsume 4 results from the literature.

▶ A new approach: **Operational Algorithmic Game Semantics**

  ▶ OGS models are (generally) simpler to construct

  ▶ More intuitive

  ▶ Retain operational character suitable for further program analysis tasks

  ▶ We believe that it can be easily adapted for other frameworks

  ▶ Next steps: more powerful automata
Conclusions

▶ Provide full-abstract trace model for CBPV.

▶ A new decidability result; first for CBPV.

▶ Subsume 4 results from the literature.

▶ A new approach: **Operational Algorithmic Game Semantics**

▶ OGS models are (generally) simpler to construct

▶ More intuitive

▶ Retain operational character suitable for further program analysis tasks

▶ We believe that it can be easily adapted for other frameworks

▶ Next steps: more powerful automata
Conclusions

▶ Provide full-abstract trace model for CBPV.

▶ A new decidability result; first for CBPV.

▶ Subsume 4 results from the literature.

▶ A new approach: **Operational Algorithmic Game Semantics**

▶ OGS models are (generally) simpler to construct

▶ More intuitive

▶ Retain operational character suitable for further program analysis tasks

▶ We believe that it can be easily adapted for other frameworks

▶ Next steps: more powerful automata
Conclusions

- Provide full-abstract trace model for CBPV.
- A new decidability result; first for CBPV.
- Subsume 4 results from the literature.
- A new approach: **Operational Algorithmic Game Semantics**
  - OGS models are (generally) simpler to construct
  - More intuitive
  - Retain operational character suitable for further program analysis tasks
  - We believe that it can be easily adapted for other frameworks
  - Next steps: more powerful automata
References I


