Estimation of the length of interactions in arena game semantics

Pierre Clairambault

Department of Computer Science
University of Bath

FoSSaCS 2011
30/03/2011

I. Introduction

The simply-typed λ-calculus

Basic prototypal programming language.

Definition (Types)

$$A ::= o \mid A \to A$$

Definition (Terms)

$$\frac{}{\Gamma, x : A \vdash x : A} \qquad \frac{\Gamma, x : A \vdash M : B}{\Gamma \vdash \lambda x.M : A \to B} \qquad \frac{\Gamma \vdash M : A \to B \quad \Gamma \vdash N : A}{\Gamma \vdash MN : B}$$

Definition (Reductions)

$$\begin{array}{rcl} (\lambda x.M)N & \rightsquigarrow_\beta & M[N/x] \\ \lambda x.Mx & \rightsquigarrow_\eta & M \ (x \notin fv(M)) \end{array}$$

Bounds on reduction for simply typed λ-calculus

Theorem (Beckmann 2001)

$$|M| \leq 2^{h(M)}_{g(M)+1}$$

- *$|M|$ is the longuest $\beta\eta$-reduction sequence on M,*
- *$h(M)$ is the height of M, defined by $h(x) = 0$, $h(\lambda x.M) = 1 + h(M)$ and $h(MN) = \max(h(M), h(N)) + 1$,*
- *$2^p_0 = p$ and $2^p_{n+1} = 2^{2^p_n}$,*
- *$g(M)$ is the highest degree of types of subterms of M, where the degree of a type is $d(o) = 0$ and $d(A \to B) = \max(d(A) + 1, d(B))$.*

Moreover, this bound is optimal.

However, β-reduction is **not** the notion of execution used in practice.

Abstract machines

A more practically meaningful notion of execution on λ-terms.

Definition (Krivine Abstract Machine)

- **Closures.** Pairs M^σ, where M is an open term and for $x \in fv(M)$, $\sigma(x)$ is a closure.
- **States.** Pairs $M^\sigma \star \pi$ where M^σ is a closure and π is a **stack** of closures.
- **Transitions.**

$$\begin{array}{rcll}
(M_1 M_2)^\sigma \star \pi & \to & M_1^\sigma \star M_2^\sigma \cdot \pi & \text{(push)} \\
(\lambda x.M)^\sigma \star N^\tau \cdot \pi & \to & M^{\sigma \cup \{x \mapsto N^\tau\}} \star \pi & \text{(pop)} \\
x_i^\sigma \star \pi & \to & \sigma(x_i) \star \pi & \text{(call)}
\end{array}$$

Computationally sound with respect to β-reduction.

Head linear reduction

The KAM performs **head linear reduction** [Danos-Regnier 2003].

Definition (Head linear reduction)

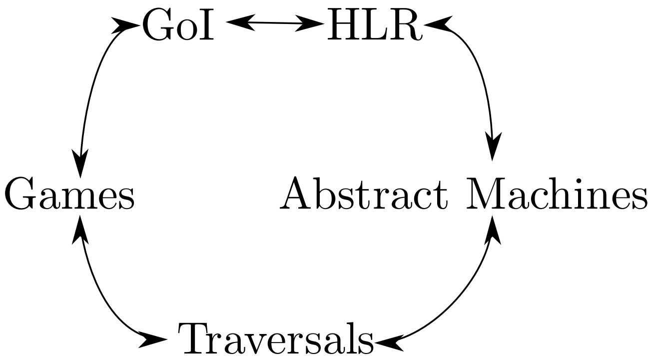
$$C_1[(\lambda x.C_2[x\ M])\ N] \rightsquigarrow C_1[(\lambda x.C_2[N\ M])\ N]$$

When x is the head (leftmost) occurrence of a variable.

- Not canonical on terms but canonical in proof nets,
- No straightforward way to good bounds from those for $\beta\eta$,

De Bruijn has bounds for his similar **minireduction**: iterate of the diagonal of an Ackermann-like function!

Equivalent formulations (Danos - Herbelin - Regnier)



Common combinatorics: **pure pointer structures**

II. Complexity and pure pointer structures

II.1 Arena games

Arenas, plays

Two players: Player (P) and Opponent (O)

Definition

An **arena** is a tree $A = (M_A, \lambda_A, \vdash_A, i_A)$

Definition

A **legal play** on A is a **pointing string** on A which is:

- Alternating,
- Respects $\vdash_A$ and i_A.

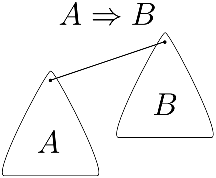
The set of legal plays on A is denoted by $\mathcal{L}_A$

Definition

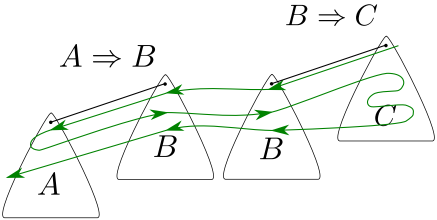
A **strategy** $\sigma : A$ is a non-empty set of even-length legal plays closed by even-length prefix.

Arrow, composition

Given arenas A and B, we define $A \Rightarrow B$:



Given $\sigma : A \Rightarrow B$ and $\tau : B \Rightarrow C$, we form $\sigma; \tau : A \Rightarrow C$



This builds a category of arenas and strategies.

Notions of view

Terms of the λ-calculus only access to part of the history.

Definition (P-view/O-view)

The **Player view**, or P-view, is defined as follows:

$$
\begin{aligned}
\ulcorner sop \urcorner &= \ulcorner so \urcorner p \\
\ulcorner s_1 \; p \, \overset{\frown}{s_2} \; o \urcorner &= \ulcorner s_1 p \urcorner o \\
\ulcorner si \urcorner &= i
\end{aligned}
$$

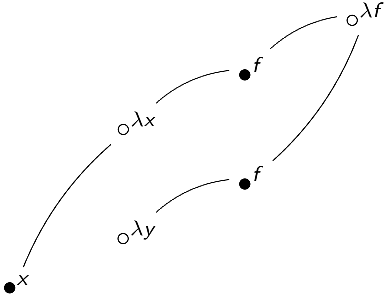
The (long) **Opponent view**, or O-view, is defined as follows:

$$
\begin{aligned}
\llcorner spo \lrcorner &= \llcorner sp \lrcorner o \\
\llcorner s_1 \; o \, \overset{\frown}{s_2} \; p \lrcorner &= \llcorner s_1 o \lrcorner p
\end{aligned}
$$

P-views

P-views are abstract representations of branches of **Böhm trees**.

$$\lambda f.f(\lambda x.f(\lambda y.x)) : ((o \longrightarrow o) \longrightarrow o) \longrightarrow o$$



So the length of P-views correspond roughly to the size of terms.

Innocence/Visibility

Definition

A strategy is **innocent** if it only depends on its P-view.

Definition

A strategy is P-**visible** if it always point inside its P-view.

- Innocent strategies correspond to terms of PCF
- P-visible strategies correspond to terms of PCF with non-determinism and ground type references

Our results will hold as long as visibility holds.

II.2 Finiteness of interactions

Size of strategies, termination

Definition

If $\sigma : A$ is P-visible, its **size** is

$$size(\sigma) = sup\{\frac{|\ulcorner s \urcorner|}{2} \mid s \in \sigma\}$$

If the size of $\sigma : A$ is finite, we say that σ is **bounded**.

Theorem (APAL09)

If $\sigma : A \Rightarrow B$ and $\tau : B \Rightarrow C$ are bounded, then any **passive** *interaction (only one move by the external Opponent) $u \in \sigma || \tau$ is finite.*

Corollary

If $\sigma : A \Rightarrow B$ and $\tau : B \Rightarrow C$ are total and bounded, then so is $\sigma ; \tau : A \Rightarrow C$.

Length of interactions

Theorem

If P-visible $\sigma : A \Rightarrow B$ and $\tau : B \Rightarrow C$ both have finite size, then any passive $u \in \sigma || \tau$ is finite.

Hence by König's lemma (or the fan theorem):

Corollary

For all $n, p \in \mathbb{N}$, there is $N(n, p) \in \mathbb{N}$ such that for all P-visible $\sigma : A \Rightarrow B$ and $\tau : B \Rightarrow C$ such that $size(\sigma) \leq n$ and $size(\tau) \leq p$, for all passive $u \in \sigma || \tau$,

$$|u| \leq N(n, p)$$

Our goal is to estimate $N(n, p)$.

Strategy-free formulation

Definition

A play $s \in \mathcal{L}_A$ is **visible** if, whenever s_j point to s_i,

$$\begin{aligned} \lambda_A(s_j) = P \quad &\Leftrightarrow \quad s_i \in \ulcorner s_j \urcorner \\ \lambda_A(s_j) = O \quad &\Leftrightarrow \quad s_i \in \llcorner s_j \lrcorner \end{aligned}$$

Definition

A visible play $s \in \mathcal{L}_A$ has:

- P-size n iff for all $s' \sqsubseteq s$, $|\ulcorner s' \urcorner| \leq 2n$
- O-size p iff for all $s' \sqsubseteq s$, $|\llcorner s' \lrcorner| \leq 2p + 1$

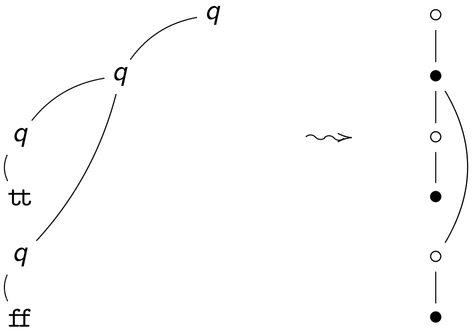
Proposition

$N(n, p)$ is the maximal length of a visible play of P-size n and O-size p.

Pure pointer structures

Here, only pointers matter, not the identity of moves.

$$(\mathbb{B} \Longrightarrow \mathbb{B}) \Longrightarrow o$$



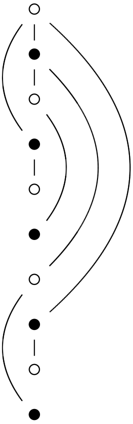
We get the notion of **pure pointer structures**.

III. Agents and rewriting

III.1 Agents, intuitively

Abstract machines

The PAM connects pure pointer structures with HLR.

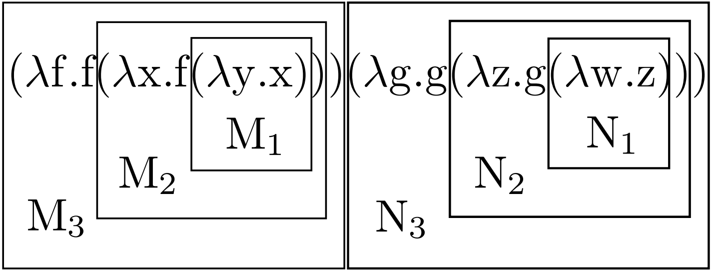


0. $(\lambda f.f(\lambda x.f(\lambda y.x)))(\lambda g.g(\lambda z.g(\lambda w.z)))$
1. $(\lambda f.f(\lambda x.f(\lambda y.x)))(\lambda g.g(\lambda z.g(\lambda w.z)))$
2. $(\lambda f.f(\lambda x.f(\lambda y.x)))(\lambda g.g(\lambda z.g(\lambda w.z)))$
3. $(\lambda f.f(\lambda x.f(\lambda y.x)))(\lambda g.g(\lambda z.g(\lambda w.z)))$
4. $(\lambda f.f(\lambda x.f(\lambda y.x)))(\lambda g.g(\lambda z.g(\lambda w.z)))$
5. $(\lambda f.f(\lambda x.f(\lambda y.x)))(\lambda g.g(\lambda z.g(\lambda w.z)))$
6. $(\lambda f.f(\lambda x.f(\lambda y.x)))(\lambda g.g(\lambda z.g(\lambda w.z)))$
7. $(\lambda f.f(\lambda x.f(\lambda y.x)))(\lambda g.g(\lambda z.g(\lambda w.z)))$
8. $(\lambda f.f(\lambda x.f(\lambda y.x)))(\lambda g.g(\lambda z.g(\lambda w.z)))$
9. $(\lambda f.f(\lambda x.f(\lambda y.x)))(\lambda g.g(\lambda z.g(\lambda w.z)))$

Head occurrence – argument subterm

Abstract machines

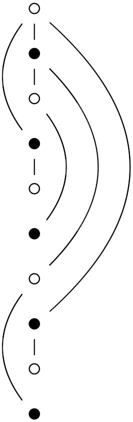
Let us give names to the subterms of this λ-term:



Where indices correspond to the **size** of the subterm.

The Krivine Abstract Machine

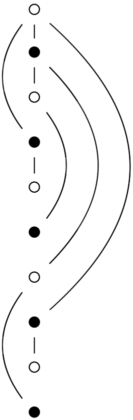
The PAM also relates to some states of the KAM.



0. $(\lambda f.f(\lambda x.f(\lambda y.x)))(\lambda g.g(\lambda z.g(\lambda w.z)))$
1. $(\lambda f.f(\lambda x.f(\lambda y.x)))(\lambda g.g(\lambda z.g(\lambda w.z)))$
2. $(\lambda f.f(\lambda x.f(\lambda y.x)))(\lambda g.g(\lambda z.g(\lambda w.z)))$
3. $(\lambda f.f(\lambda x.f(\lambda y.x)))(\lambda g.g(\lambda z.g(\lambda w.z)))$
4. $(\lambda f.f(\lambda x.f(\lambda y.x)))(\lambda g.g(\lambda z.g(\lambda w.z)))$
5. $(\lambda f.f(\lambda x.f(\lambda y.x)))(\lambda g.g(\lambda z.g(\lambda w.z)))$
6. $(\lambda f.f(\lambda x.f(\lambda y.x)))(\lambda g.g(\lambda z.g(\lambda w.z)))$
7. $(\lambda f.f(\lambda x.f(\lambda y.x)))(\lambda g.g(\lambda z.g(\lambda w.z)))$
8. $(\lambda f.f(\lambda x.f(\lambda y.x)))(\lambda g.g(\lambda z.g(\lambda w.z)))$
9. $(\lambda f.f(\lambda x.f(\lambda y.x)))(\lambda g.g(\lambda z.g(\lambda w.z)))$

The Krivine Abstract Machine

The PAM also relates to some states of the KAM.



$M_3 \star N_3$

0\. $M_3 \star N_3$

1\. $(\lambda f.f(\lambda x.f(\lambda y.x)))(\lambda g.g(\lambda z.g(\lambda w.z)))$

2\. $(\lambda f.f(\lambda x.f(\lambda y.x)))(\lambda g.g(\lambda z.g(\lambda w.z)))$

3\. $(\lambda f.f(\lambda x.f(\lambda y.x)))(\lambda g.g(\lambda z.g(\lambda w.z)))$

4\. $(\lambda f.f(\lambda x.f(\lambda y.x)))(\lambda g.g(\lambda z.g(\lambda w.z)))$

5\. $(\lambda f.f(\lambda x.f(\lambda y.x)))(\lambda g.g(\lambda z.g(\lambda w.z)))$

6\. $(\lambda f.f(\lambda x.f(\lambda y.x)))(\lambda g.g(\lambda z.g(\lambda w.z)))$

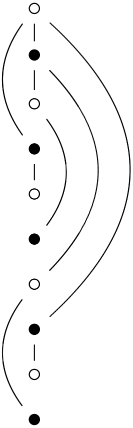
7\. $(\lambda f.f(\lambda x.f(\lambda y.x)))(\lambda g.g(\lambda z.g(\lambda w.z)))$

8\. $(\lambda f.f(\lambda x.f(\lambda y.x)))(\lambda g.g(\lambda z.g(\lambda w.z)))$

9\. $(\lambda f.f(\lambda x.f(\lambda y.x)))(\lambda g.g(\lambda z.g(\lambda w.z)))$

The Krivine Abstract Machine

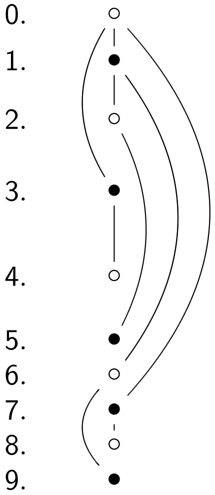
The PAM also relates to some states of the KAM.



0. $M_3 \star N_3$
1. $N_3 \star M_2^{f \mapsto N_3}$
2. $(\lambda f.f(\lambda x.f(\lambda y.x)))(\lambda g.g(\lambda z.g(\lambda w.z)))$
3. $(\lambda f.f(\lambda x.f(\lambda y.x)))(\lambda g.g(\lambda z.g(\lambda w.z)))$
4. $(\lambda f.f(\lambda x.f(\lambda y.x)))(\lambda g.g(\lambda z.g(\lambda w.z)))$
5. $(\lambda f.f(\lambda x.f(\lambda y.x)))(\lambda g.g(\lambda z.g(\lambda w.z)))$
6. $(\lambda f.f(\lambda x.f(\lambda y.x)))(\lambda g.g(\lambda z.g(\lambda w.z)))$
7. $(\lambda f.f(\lambda x.f(\lambda y.x)))(\lambda g.g(\lambda z.g(\lambda w.z)))$
8. $(\lambda f.f(\lambda x.f(\lambda y.x)))(\lambda g.g(\lambda z.g(\lambda w.z)))$
9. $(\lambda f.f(\lambda x.f(\lambda y.x)))(\lambda g.g(\lambda z.g(\lambda w.z)))$

The Krivine Abstract Machine

The PAM also relates to some states of the KAM.



$$M_3 \star N_3$$

$$N_3 \star M_2^{f \mapsto N_3}$$

$$M_2^{f \mapsto N_3} \star N_2^{g \mapsto M_2^{f \mapsto N_3}}$$

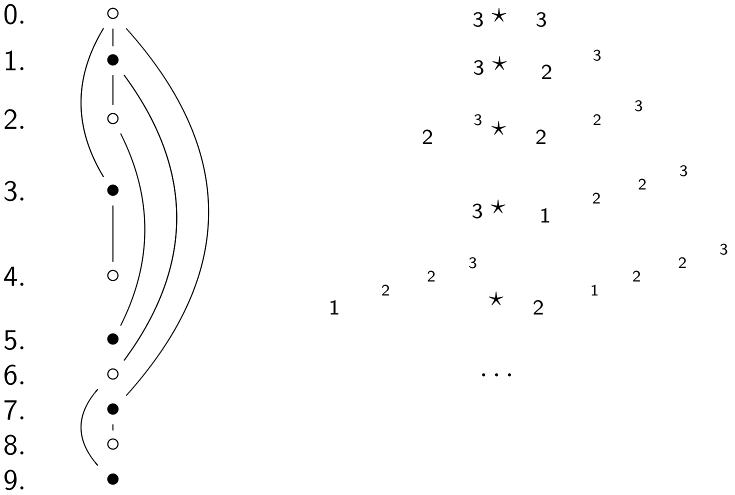
$$N_3 \star M_1^{x \mapsto N_2^{g \mapsto M_2^{f \mapsto N_3}}}$$

$$M_1^{x \mapsto N_2^{g \mapsto M_2^{f \mapsto N_3}}} \star N_2^{g \mapsto M_1^{x \mapsto N_2^{g \mapsto M_2^{f \mapsto N_3}}}}$$

$$\dots$$

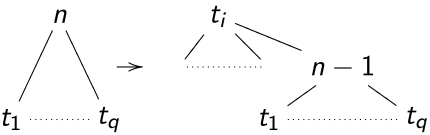
Agents

. . . which collapses to an operation on integers.



Rewriting on agents

All steps are instances of the following reduction



$(n \geq 1)$

Definition

An **agent** is a finite tree of natural numbers.

Simulation

Proposition (Simulation)

For all pure pointer structure s, there is a labelling of moves $s_0, s_1, \ldots$ by agents:

$$a_0 \rightsquigarrow a_1 \rightsquigarrow a_2 \rightsquigarrow a_3 \rightsquigarrow \ldots$$

Remark

If s has P-size n and O-size p, we can choose $a_0 = \begin{array}{c} n \\ | \\ p \end{array}$.

Corollary

$$N(n,p) \leq N\left(\begin{array}{c} n \\ | \\ p \end{array}\right) + 1$$

where $N(a)$ is the length of the longuest reduction sequence of a.

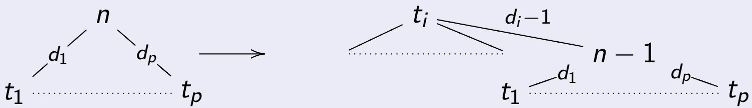
Typed variant

Definition

$N_d(n,p)$ is the maximal size of a passive interaction between $\sigma : A \Rightarrow B$ of size n and $\tau : B \Rightarrow C$ of size p where B has depth $d-1$.

Definition

A **typed agent** is a tree of natural numbers, whose edges are labelled be natural numbers. Their reduction is, for $n, d_i \geq 1$:



Proposition

$$N_d(n,p) \leq N\left(\begin{array}{c} n \\ \vdots d \\ p \end{array}\right) + 1$$

III.2 Bounds for reduction of agents

Beckmann's method

For $\alpha, \rho \in \mathbb{N}$, define an inductive predicate $\vdash^{\alpha}_{\rho} a$.

"a has size α with cuts of degree less than ρ"

The derivation tree for $\vdash^{\alpha}_{\rho} a$ has aspects of both syntax tree and explicit reduction tree.

Lemma (Syntax)

For any typed agent a of maximum degree d, maximum label m and size s, we have $\vdash^{m \cdot s}_{d} a$.

Lemma (Cut elimination)

If $\vdash^{\alpha}_{\rho+1} a$ with $\alpha \geq 1$, then $\vdash^{2^{\alpha-1}}_{\rho} a$.

Lemma (Bound lemma)

If $\vdash^{\alpha}_{0} a$, then $N(a) \leq \alpha$.

Results

Theorem

For any typed agent a of maximum degree d, maximum label m and size s, we have:

$$N(a) \leq 2_{d-1}^{m \cdot s - 1}$$

Theorem

$$2_{d-2}^{2} \leq N_d(n, p) \leq 2_{d-2}^{n(p+1)}$$

The construction of the lower bound is standard (it only makes use of innocent behaviour).

Application: head linear reduction

Theorem (Game situation)

Let $\Gamma \vdash M : A \to B$ and $\Gamma \vdash N : A$ be β-normal η-long λ-terms, then:

$$|MN| \leq 2_{d(A)-1}^{h(M)\cdot(h(N)+1)}$$

Theorem (General case)

Let $\Gamma \vdash M : A$, then:

$$|M| \leq 2_{g(M)}^{(h(M)+g(M)+1)\cdot(g(M)+1)}$$

Bounds for β-reduction on closed terms: $2_{g(M)}^{h(M)+g(M)}$

The price of linearity is not as high as expected!

IV. Conclusions

Conclusions

Achievements

- *Bounds for the length of plays in game semantics*
- *Also holds for HLR, abstract machines, traversals ...*
- *Holds for models of non determinism, ground type references, but also call-by-value, restricted concurrent languages...*

Questions & further work

- *Could agents be used to study languages with restricted complexity (e.g. light linear logics)?*
- *Could we optimise these tools (especially on small types), to statically generate useful bounds for programming languages?*