Game semantics and friends

Pierre-Louis Curien (CNRS – Université Paris Cité – Inria)

OPLSS 2022, Eugene, OR, 20-23/6/2022

Main source for the course: my

Notes on game semantics (2006), available from `curien.galene.org/papers`

and bibliography therein

Some dual pairs in the world of programming

• memory *cell* (or location or register) versus its actual contents or *value*; in object-oriented style, record field names versus their values, method names versus their actual definition.

• *input* and *output*, or (in the language of proof theory) hypotheses and conclusions;

• *sending* and *receiving* messages (in process calculi);

• a *program* and its *context* (the libraries of your program environment – or the larger program of which the program under focus is a subpart, or a module); the programmer and the computer; two programs that call each other;

• call-by-name (CBN) and call-by-value (CBV).

Proofs as strategies

Proofs can be given a **dialogue-game** interpretation:a formula is tested through a dialogue between an opponent who doubts some formulas, and the player who justifies his proof step-by-step by exhibiting the rules he has used.

$$\dfrac{\dfrac{}{t_1+SSt_2 = S(t_1+St_2)} \quad \dfrac{\dfrac{}{t_1+St_2 = S(t_1+t_2)}}{S(t_1+St_2) = SS(t_1+t_2)}}{t_1+SSt_2 = SS(t_1+t_2)}$$

Untyped λ-calculus

The syntax of the untyped λ-calculus (λ-calculus for short) is given by:

$$M ::= x \mid MM \mid \lambda x.M,$$

where x is called a variable, $M_1 M_2$ is called an application, and $\lambda x.M$ is called an abstraction.

β-reduction:

$$\frac{}{(\lambda x.M)N \to M[x \leftarrow N]}$$

$$\frac{M \to M'}{MN \to M'N} \qquad \frac{N \to N'}{MN \to MN'} \qquad \frac{M \to M'}{\lambda x.M \to \lambda x.M'}$$

Normal forms in the λ-calculus

Any λ-term has exactly one of the following two forms:

- head normal form ($n \geq 1$, $p \geq 1$):

  $\lambda x_1 \cdots x_n . x M_1 \cdots M_p$

- head redex ($n \geq 0$, $p \geq 1$):

  $\lambda x_1 \cdots x_n . (\lambda x . M) M_1 \cdots M_p$

Note that a normal form can be only of the first form, and recursively so.

Böhm trees

$$\omega(M) = \begin{cases} \Omega & \text{if } M = \lambda x_1 \cdots x_n.(\lambda x.P) M_1 \cdots M_p \\ \lambda x_1 \cdots x_n.x\omega(M_1) \cdots \omega(M_p) & \text{if } M = \lambda x_1 \cdots x_n.x M_1 \cdots M_p \ , \end{cases}$$

$$\frac{}{\Omega \leq M} \qquad \frac{M_1 \leq N_1 \cdots M_p \leq N_p}{\lambda x_1 \cdots x_n.x M_1 \cdots M_p \leq \lambda x_1 \cdots x_n.x N_1 \cdots N_p}$$

Then the Böhm tree of a term is the (possibly infinite) least upper bound of all $\omega(N)$, for all N such that $M \to^* N$.

The Böhm tree of a normalisable term is its normal form.

Böhm trees as strategies

$$\dfrac{M_1 \quad \ldots \quad \dfrac{N_1 \ \ldots \ N_p}{\lambda y_1 \cdots y_p.y} \quad \ldots \quad M_n}{\lambda x_1 \cdots x_n.x}$$

$$\lambda x_1 \cdots x_n.x M_1 \ldots (\lambda y_1 \cdots y_p.y N_1 \ldots N_p) \ldots M_n$$

Simply typed λ-calculus

$$\sigma ::= C \mid \sigma \to \sigma \qquad (C \text{ base type})$$

Thus every type write s uniquely as $\boldsymbol{\sigma_1 \to \cdots \to \sigma_n \to C}$.

$$\frac{x : \sigma \in \Gamma}{\Gamma \vdash x : \sigma}$$

$$\frac{\Gamma, x : \sigma \vdash M : \tau}{\Gamma \vdash \lambda x : \sigma . M : \sigma \to \tau}$$

$$\frac{\Gamma \vdash M : \sigma \to \tau \quad \Gamma \vdash N : \sigma}{\Gamma \vdash MN : \tau}$$

Böhm trees associated to simply typable terms are finite (see Silvia's lectures!).

η-long Böhm trees

We restrict ourselves to the simply typed setting, and impose that

• each occurrence of a variable must appear in a context where it is applied to all its arguments,

• and in each sequence of abstractions $\lambda\vec{x}.M$ the number of parameters $x_1, \ldots, x_n$ is exactly the number of arguments $N_1, \ldots, N_n$ which the term $\lambda\vec{x}.M$ can accept according to its type.

Executing Böhm trees (the abstract machine)

Terms	$M ::= (\lambda\vec{x}.W)$	Environments	$e ::= nil \mid B\bullet e$
Code	$W ::= y\vec{M}$	Frames	$B ::= \langle \vec{z} \leftarrow \vec{M} \rangle [e].$

We split M in this way even if $\vec{x}$ is empty: in this case we write $M = (W)$ and $W = y\vec{N}$. We also call W a body.

$$(K) \quad (x\vec{M})[e] \to W[\langle \vec{z} \leftarrow \vec{M} \rangle [e] \bullet e_i]$$

$$\text{where} \begin{cases} e = B_0 \bullet \cdots \bullet B_n & B_i = \langle \vec{x_i} \leftarrow \vec{N_i} \rangle [e_i] \\ x = x_{ij} & N_{ij} = (\lambda \vec{z}.W) \end{cases}$$

Executing Böhm trees (illustration)

$2'$	$u(\lambda x.u(\lambda y.x))$	$\langle u \overset{2}{\leftarrow} (\lambda r.r(r(z)))\rangle[\ ] = \rho_0$
$3'$	$r(r(z))$	$\langle r \overset{3}{\leftarrow} (\lambda x.u(\lambda y.x))\rangle[\rho_0] = \rho_1$
$4'$	$u(\lambda y.x)$	$\langle x \leftarrow (r(z))\rangle[\rho_1]\bullet(\rho_0 = \langle u \overset{4}{\leftarrow} (\lambda r.r(r(z)))\rangle[\ ]) = \rho_2$
$5'$	$r(r(z))$	$\langle r \overset{5}{\leftarrow} (\lambda y.x)\rangle[\rho_2] = \rho_3$
$6'$	x	$\langle y \leftarrow (r(z))\rangle[\rho_3]\bullet(\rho_2 = \langle x \overset{6}{\leftarrow} (r(z))\rangle[\rho_1]\bullet\rho_0) = \rho_4$
$7'$	$r(z)$	$\langle\rangle[\rho_4]\bullet(\rho_1 = \langle r \overset{7}{\leftarrow} (\lambda x.u(\lambda y.x))\rangle[\rho_0]) = \rho_5$
$8'$	$u(\lambda y.x)$	$\langle x \leftarrow (z)\rangle[\rho_5]\bullet(\rho_0 = \langle u \overset{8}{\leftarrow} (\lambda r.r(r(z)))\rangle[\ ]) = \rho_6$
$9'$	$r(r(z))$	$\langle r \overset{9}{\leftarrow} (\lambda y.x)\rangle[\rho_6] = \rho_7$
$10'$	x	$\langle y \leftarrow (r(z))\rangle[\rho_7]\bullet(\rho_6 = \langle x \overset{10}{\leftarrow} (z)\rangle[\rho_5]\bullet\rho_0) = \rho_8$
$11'$	z	$\langle\rangle[\rho_8]\bullet\rho_5$

The language PCF

PCF is simply typed λ-calculus, with the following constants

n	$: Nat$	$(n \in \omega)$
T, F	$: Bool$	
$succ, pred$	$: Nat \to Nat$	
$zero?$	$: Nat \to Bool$	
if then else	$: Bool \to Nat \to Nat \to Nat$	
if then else	$: Bool \to Bool \to Bool \to Bool$	
Ω	$: \sigma$	for all σ
Y	$: (\sigma \to \sigma) \to \sigma$	for all σ

Operational semantics of PCF

$(\lambda x.M)N \to M[x \leftarrow N]$ $\qquad$ $YM \to M(YM)$

$zero?(0) \to T$ $\qquad$ $zero?(n+1) \to F$

$succ(n) \to n+1$ $\qquad$ $pred(n+1) \to n$

if T *then* N *else* $P \to N$ $\qquad$ *if* F *then* N *else* $P \to P$

$$\frac{M \to M'}{MN \to M'N} \qquad \frac{M \to M'}{\textit{if } M \textit{ then } N \textit{ else } P \to \textit{if } M' \textit{ then } N \textit{ else } P}$$

$$\frac{M \to M'}{f(M) \to f(M')} \quad \text{(for } f \in \{succ, pred, zero?\}\text{)}$$

PCF Böhm trees

$$M ::= \lambda \vec{x} : \vec{\sigma}.W \qquad W ::= v \parallel \mathsf{case}\ yM_1 \cdots M_n\ [v_1 \to W_1, \ldots, v_k \to W_k]$$

$$\frac{v : C}{\Gamma \vdash v : C}$$

$$\frac{\Gamma \bullet y = \sigma_1 \to \cdots \to \sigma_n \to C \quad v_1, \ldots, v_k : C \qquad \Gamma \vdash M_1 : \sigma_1 \ \cdots\ \Gamma \vdash M_n : \sigma_n \quad \Gamma \vdash W_1 : C_1 \ \cdots\ \Gamma \vdash W_k : C_1}{\Gamma \vdash \mathsf{case}\ yM_1 \cdots M_n\ [v_1 \to W_1, \ldots, v_k \to W_k] : C_1}$$

$$\frac{\Gamma, \vec{x} : \vec{\sigma} \vdash W : C}{\Gamma \vdash (\lambda \vec{x} : \vec{\sigma}.W) : \vec{\sigma} \to C}$$

PCF Böhm trees as strategies

$$\frac{M_1 \quad \cdots \quad M_p \quad v_1 \to W_1 \quad \cdots \quad v_k \to W_k}{\lambda \vec{x}.\mathsf{case}\ y}$$

HO games

Game semantics arose as such in the early 1990's from parallel works of

- Abramsky, Jagadeesan, Malacaria (AJM)

- Hyland and Ong (HO)

A fore-runner was **sequential algorithms** (Berry-Curien) (late 1970's).

In this course, we focus on HO.

Entering the arena!

We start with the simplest kind of data type. The arena **nat** for natural numbers has the following moves: one (initial) O move denoted q, and a P move n for each natural number n.

$$\left\{\begin{array}{l}\{\epsilon\}\\ \{\epsilon, qn\}\end{array}\right. \qquad \left\{\begin{array}{l}\bot\\ n\end{array}\right.$$

Product of arenas

The arena **nat** $\times$ **nat** is made of two disjoints copies of nat.

$$\begin{cases} \{\epsilon\} \\ \{\epsilon, q_1 m_1\} \\ \{\epsilon, q_2 n_2\} \\ \{\epsilon, q_1 m_1, q_2 n_2\} \end{cases} \qquad \begin{cases} (\bot, \bot) \\ (m, \bot) \\ (\bot, n) \\ (m, n) \end{cases}$$

Function types ($\mathsf{nat}_1 \to \mathsf{nat}_\epsilon$)

$$q_\epsilon q_1 \left\{ \begin{array}{l} 0_1 3_\epsilon \\ 3_1 6_\epsilon \end{array} \right. \qquad \lambda x.\, \mathsf{case}\, x \left\{ \begin{array}{l} 0 \to 3 \\ 3 \to 6 \end{array} \right.$$

Note the inversion of polarity for $\mathbf{nat}_1$. Also, $q_\epsilon \vdash q_1$.

Interaction with $\{q_1 0\}$ converges, interaction with $\{q_1 2\}$ diverges.

Plays and (deterministic) strategies

The tree $q_\epsilon q_1 \left\{ \begin{array}{l} 0_1 3_\epsilon \\ 1_1 4_\epsilon \end{array} \right.$ can equivalently be described by the set of its even-length branches, which are called plays:

$$\{\epsilon, q_\epsilon q_1, q_\epsilon q_1 0_1 3_\epsilon, q_\epsilon q_1 1_1 4_\epsilon\}$$

A strategy is a set of even-length plays. Note that plays start with O and alternate between O and P.

Here we deal with deterministic strategies σ:

$$\text{if } pv, pw \in \sigma, \text{ then } v = w.$$

The player knows how to react to each opponent's move: "my head variable is".

Just a paraphrase of syntax?

Mathematics is full of useful equivalent presentations: descriptive / analytical geometry, matrices / linear maps, etc...

Sign of **good** syntax: Böhm trees!

In fact, the correspondence is an **injection**: games offer a *wider* picture (cf. IA).

Stuttering

$\lambda x.\,\mathsf{case}\,x\,[4 \to \mathsf{case}\,x\,[3 \to 2]]$ as strategy contains $q_\epsilon q_1 4_1 q_1 3_1 2_\epsilon$.

Stuttering strategies *do not* exist in the earlier model of sequential algorithms of PCF (Berry-Curien, late 1970).

A higher-order type

$(\mathsf{nat}_{11} \to \mathsf{nat}_1) \to \mathsf{nat}_\epsilon$

$$h = \lambda f.\, \mathsf{case}\, f(3)[4 \to 7, 6 \to 9]$$

As a strategy:

$$\lambda f.\, \mathsf{case}\, f \begin{cases} (3) \\ 4 \to 7 \\ 6 \to 9 \end{cases} \qquad q_\epsilon q_1 \begin{cases} q_{11} 3_{11} \\ 4_1 7_\epsilon \\ 6_1 9_\epsilon \end{cases}$$

Pointers

$$Kierstead_1 = \lambda f.\mathsf{case}\ f(\lambda x.\mathsf{case}\ f(\lambda y.\mathsf{case}\ x))$$

$$Kierstead_2 = \lambda f.\mathsf{case}\ f(\lambda x.\mathsf{case}\ f(\lambda y.\mathsf{case}\ y))$$

$$q_\epsilon q_1 \begin{cases} q_{11}q_1 \begin{cases} q_{11}q_{111} \begin{cases} T_{111}T_{11} \\ F_{111}F_{11} \end{cases} \\ T_1T_{11} \\ F_1F_{11} \end{cases} \\ T_1T_\epsilon \\ F_1F_\epsilon \end{cases}$$

(type $((\mathsf{bool}_{111} \to \mathsf{bool}_{11}) \to \mathsf{bool}_1) \to \mathsf{bool}_\epsilon$)

$$Kierstead_1 = q_\epsilon[q_1, \overset{0}{\hookleftarrow}] \begin{cases} q_{11}[q_1, \overset{1}{\hookleftarrow}] \begin{cases} q_{11}\underline{[q_{111}, \overset{1}{\hookleftarrow}]} \begin{cases} T_{111}[T_{11}, \overset{1}{\hookleftarrow}] \\ F_{111}[F_{11}, \overset{1}{\hookleftarrow}] \end{cases} \\ T_1[T_{11}, \overset{1}{\hookleftarrow}] \\ F_1[F_{11}, \overset{1}{\hookleftarrow}] \end{cases} \\ T_1[T_\epsilon, \overset{1}{\hookleftarrow}] \\ F_1[F_\epsilon, \overset{1}{\hookleftarrow}] \end{cases}$$

$$Kierstead_2 = q_\epsilon[q_1, \overset{0}{\hookleftarrow}] \begin{cases} q_{11}[q_1, \overset{1}{\hookleftarrow}] \begin{cases} q_{11}\underline{[q_{111}, \overset{0}{\hookleftarrow}]} \begin{cases} T_{111}[T_{11}, \overset{1}{\hookleftarrow}] \\ F_{111}[F_{11}, \overset{1}{\hookleftarrow}] \end{cases} \\ T_1[T_{11}, \overset{1}{\hookleftarrow}] \\ F_1[F_{11}, \overset{1}{\hookleftarrow}] \end{cases} \\ T_1[T_\epsilon, \overset{1}{\hookleftarrow}] \\ F_1[F_\epsilon, \overset{1}{\hookleftarrow}] \end{cases}$$

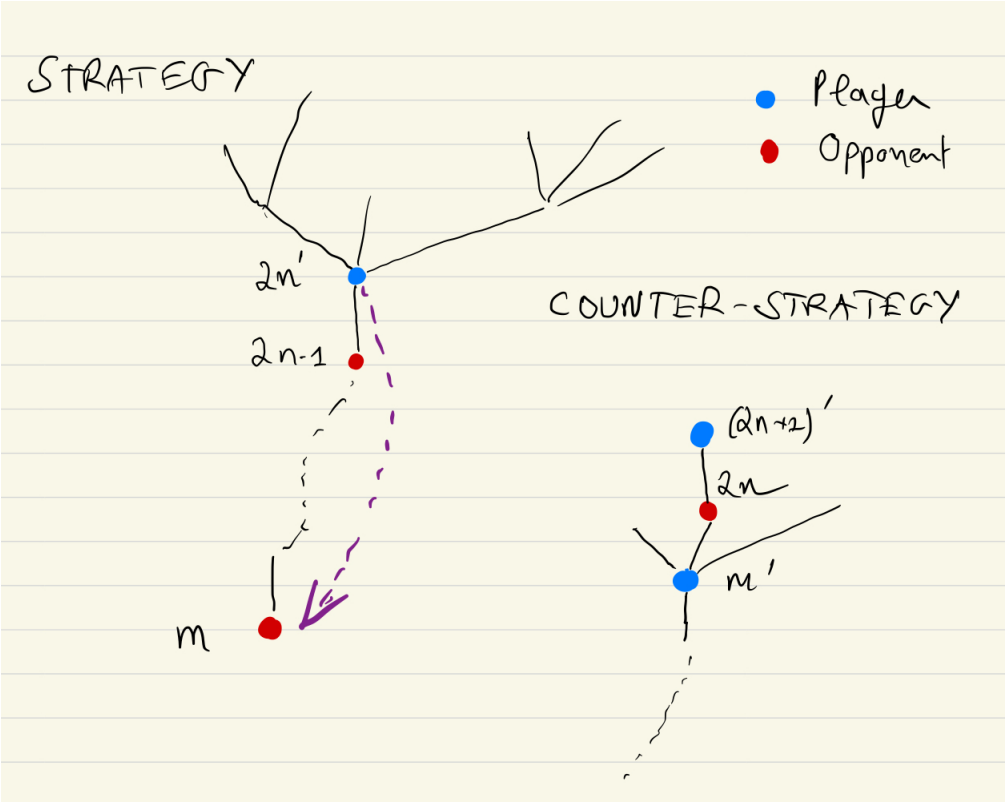
Inserting the implicit pointers in our previous examples

$h = \lambda f.\,\mathsf{case}\, f(3)[4 \to 7, 6 \to 9]$ $\quad$ $\lambda x.\,\mathsf{case}\, x[0 \to 3, 3 \to 6]$

$$q_\epsilon q_1 \begin{cases} q_{11}3_{11} \\ 4_1 7_\epsilon \\ 6_1 9_\epsilon \end{cases} \qquad q_1 q_{11} \begin{cases} 0_{11}3_1 \\ 3_{11}6_1 \end{cases}$$

$$q_\epsilon[q_1, \overset{0}{\hookleftarrow}] \begin{cases} q_{11}[3_{11}, \overset{0}{\hookleftarrow}] \\ 4_1[7_\epsilon, \overset{1}{\hookleftarrow}] \\ 6_1[9_\epsilon, \overset{1}{\hookleftarrow}] \end{cases} \qquad q_1[q_{11}, \overset{0}{\hookleftarrow}] \begin{cases} 0_{11}[3_1, \overset{1}{\hookleftarrow}] \\ 3_{11}[6_1, \overset{1}{\hookleftarrow}] \end{cases}$$

Game abstract machine for PCF Böhm trees



- p' determined from $(p-1)$ by $\begin{cases} \text{the strategy} & \text{for } n \text{ even} \\ \text{the counter-strategy} & \text{for } n \text{ odd} \end{cases}$
- If p' points to **m**, then play **p** over m'

Game abstract machine for PCF Böhm trees

$$q_\epsilon[q_1, \overset{0}{\hookleftarrow}] \begin{cases} q_{11}[3_{11}, \overset{0}{\hookleftarrow}] \\ 4_1[7_\epsilon, \overset{1}{\hookleftarrow}] \\ 6_1[9_\epsilon, \overset{1}{\hookleftarrow}] \end{cases} \qquad q_1[q_1, \overset{0}{\hookleftarrow}] \begin{cases} 0_{11}[3_1, \overset{1}{\hookleftarrow}] \\ 3_{11}[6_1, \overset{1}{\hookleftarrow}] \end{cases}$$

interaction

$$\langle q_\epsilon, \mathbf{1}\rangle[q_1, \overset{0}{\hookleftarrow}] \overset{\boxed{2'}}{} \begin{cases} \langle q_{11}, \mathbf{3}\rangle[3_{11}, \overset{0}{\hookleftarrow}] \overset{\boxed{3'}}{} \\ \langle 6_1, \mathbf{5}\rangle[9_\epsilon, \overset{1}{\hookleftarrow}] \overset{\boxed{6'}}{} \end{cases}$$

$$\langle q_1, \mathbf{2}\rangle[q_{11}, \overset{0}{\hookleftarrow}] \overset{\boxed{3'}}{} \begin{cases} \langle 3_{11}, \mathbf{4}\rangle[6_1, \overset{1}{\hookleftarrow}] \overset{\boxed{5'}}{} \end{cases}$$

Executing Kierstead against

$$\lambda g.\mathsf{case}\ g(\mathsf{case}\ gT\ [T \to T, F \to F])\ [T \to F, F \to T]$$

$$q_1[q_{11}, \overset{0}{\hookleftarrow}] \begin{cases} q_{111}[q_{11}, \overset{1}{\hookleftarrow}] \begin{cases} q_{111}[F_{111}, \overset{0}{\hookleftarrow}] \\ T_{11}[T_{111}, \overset{1}{\hookleftarrow}] \\ F_{11}[F_{111}, \overset{1}{\hookleftarrow}] \end{cases} \\ T_{11}[F_1, \overset{1}{\hookleftarrow}] \\ F_{11}[T_1, \overset{1}{\hookleftarrow}] \end{cases}$$

$$Kierstead_1 = q_\epsilon[q_1, \overset{0}{\hookleftarrow}] \begin{cases} q_{11}[q_1, \overset{1}{\hookleftarrow}] \begin{cases} q_{11}\underline{[q_{111}, \overset{1}{\hookleftarrow}]} \begin{cases} T_{111}[T_{11}, \overset{1}{\hookleftarrow}] \\ F_{111}[F_{11}, \overset{1}{\hookleftarrow}] \end{cases} \\ T_1[T_{11}, \overset{1}{\hookleftarrow}] \\ F_1[F_{11}, \overset{1}{\hookleftarrow}] \end{cases} \\ T_1[T_\epsilon, \overset{1}{\hookleftarrow}] \\ F_1[F_\epsilon, \overset{1}{\hookleftarrow}] \end{cases}$$

$$\langle q_\epsilon, \mathbf{1}\rangle[q_1, \overset{0}{\hookleftarrow}] \begin{cases} \langle q_{11}, \mathbf{3}\rangle[q_1, \overset{1}{\hookleftarrow}] \begin{cases} \langle q_{11}, \mathbf{5}\rangle[q_{111}, \overset{1}{\hookleftarrow}] \begin{cases} \langle T_{111}, \mathbf{15}\rangle[T_{11}, \overset{1}{\hookleftarrow}] \\ \langle F_1, \mathbf{17}\rangle[F_{11}, \overset{1}{\hookleftarrow}] \end{cases} \\ \langle q_{11}, \mathbf{9}\rangle[q_{111}, \overset{1}{\hookleftarrow}] \begin{cases} \langle F_{111}, \mathbf{11}\rangle[F_{11}, \overset{1}{\hookleftarrow}] \\ \langle T_1, \mathbf{13}\rangle[T_{11}, \overset{1}{\hookleftarrow}] \end{cases} \end{cases} \\ \langle T_1, \mathbf{19}\rangle[T_\epsilon, \overset{1}{\hookleftarrow}] \end{cases}$$

$$\begin{cases} \langle q_1, \mathbf{2}\rangle[q_{11}, \overset{0}{\hookleftarrow}] \begin{cases} \langle q_{111}, \mathbf{6}\rangle[q_{11}, \overset{1}{\hookleftarrow}] \begin{cases} \langle q_{111}, \mathbf{10}\rangle[F_{111}, \overset{0}{\hookleftarrow}] \\ \langle T_{11}, \mathbf{14}\rangle[T_{111}, \overset{1}{\hookleftarrow}] \end{cases} \\ \langle F_{11}, \mathbf{18}\rangle[T_1, \overset{1}{\hookleftarrow}] \end{cases} \\ \langle q_1, \mathbf{4}\rangle[q_{11}, \overset{0}{\hookleftarrow}] \begin{cases} \langle T_{11}, \mathbf{16}\rangle[F_1, \overset{1}{\hookleftarrow}] \end{cases} \\ \langle q_1, \mathbf{8}\rangle[q_{11}, \overset{0}{\hookleftarrow}] \begin{cases} \langle F_{11}, \mathbf{12}\rangle[T_1, \overset{1}{\hookleftarrow}] \end{cases} \end{cases}$$

Well-bracketing

This strategy of $(\mathsf{nat}_{11} \times \mathsf{nat}_{12} \to \mathsf{nat}_1) \to \mathsf{nat}_\epsilon$ is not the interpretation of a PCF term:

$$q_\epsilon q_1 \left\{ \begin{array}{l} q_{11} 0_\epsilon \\ q_{12} 1_\epsilon \\ n_1 (n+2)_\epsilon \end{array} \right.$$

The initial **question** q_ϵ may be answered while q_1 and q_{11} are still open.

Strong (or partial) evaluation

$$q_\epsilon q_2 \left\{ \begin{array}{l} 6_2 q_1 \left\{ \begin{array}{l} 4_1 8_\epsilon \\ 2_1 1_\epsilon \end{array} \right. \\ 3_2 q_1 \left\{ \begin{array}{l} 4_1 5_\epsilon \\ 1_1 6_\epsilon \end{array} \right. \end{array} \right. \quad \text{against} \quad q_1 4_1 \quad \text{yields} \quad q_\epsilon q_2 \left\{ \begin{array}{l} 6_2 8_\epsilon \\ 3_2 5_\epsilon \end{array} \right.$$

$$(\lambda xy.\mathsf{case}\ y\ [6 \to \mathsf{case}\ x\ [\ldots], \ldots])4 \quad \to \quad \lambda y.\mathsf{case}\ y\ [\ldots]$$

Lazy, stream-like loop of evaluation.

On the form of the plays involved in PCF Böhm trees

They are alternating sequences of moves OPO. . . , equipped with backward pointers from P moves to some previous O move.

Such plays are called views.

General plays (that will feature in a more synthetic definition of composition of strategies) will also have pointers from the O moves (to some previous P move).

Arenas

An arena A is given by a set of moves M, which have a polarity O or P (formally, there is function $\lambda_A : M \to \{O, P\}$)

and by an enabling relation $\vdash$, which is the disjoint union of a subset of $M \times M$ (one writes $m \vdash n$) and of M (one writes $\vdash m$).

If $m \vdash n$, then m and n have opposite polarities, and $\not\vdash n$. If $\vdash m$, then m is an opponent move.

Product of two arenas

Let A and B be arenas. The arena $A \times B$ has as moves all m_1 such that m is a move of A and all moves n_2 such that n is a move of B. Polarities of these moves are as in A and B. Enabling is defined as follows:

$$\frac{\vdash_A m}{\vdash m_1} \qquad \frac{\vdash_B n}{\vdash n_2} \qquad \frac{m \vdash_A a}{m_1 \vdash a_1} \qquad \frac{n \vdash_B b}{n_2 \vdash b_2}$$

Function space of two arenas

Let A and B be arenas. The arena $A \to B$ has as moves all m_1 such that m is a move of A, with polarity opposite to that in A, and all moves n_2 such that n is a move of B, with the same polarity as in B. Enabling is defined as follows:

$$\frac{\vdash_B n}{\vdash n_2} \qquad \frac{\vdash_A m \qquad \vdash_B n}{n_2 \vdash m_1} \qquad \frac{m \vdash_A a}{m_1 \vdash a_1} \qquad \frac{n \vdash_B b}{n_2 \vdash b_2}$$

Plays and strategies

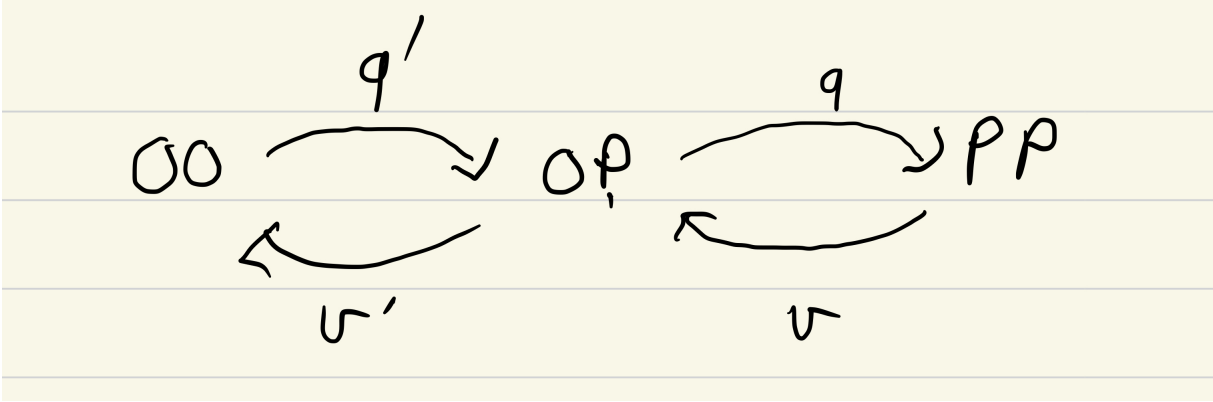
A legal play, or play for short, is a (possibly empty) sequence of moves of alternating polarity which is such that every occurrence of non-initial move is equipped with a pointer to a previous occurrence of a move justifying it. The set of legal plays over an arena A is written L_A. (Clearly, a play starts with Opponent, since only opponent moves can be initial.)

A strategy on an arena A is a non-empty set of even-length legal plays, which is closed under even-length prefixes.

Other conditions, like determinism and innocence, will be added later.

An automaton recognising $L_{A \to B}$

Convention: O and P moves of A (B) are written q, v (q', v'). Legal plays are all even-length words read by the following automaton (initial state OO):



Legal interactions

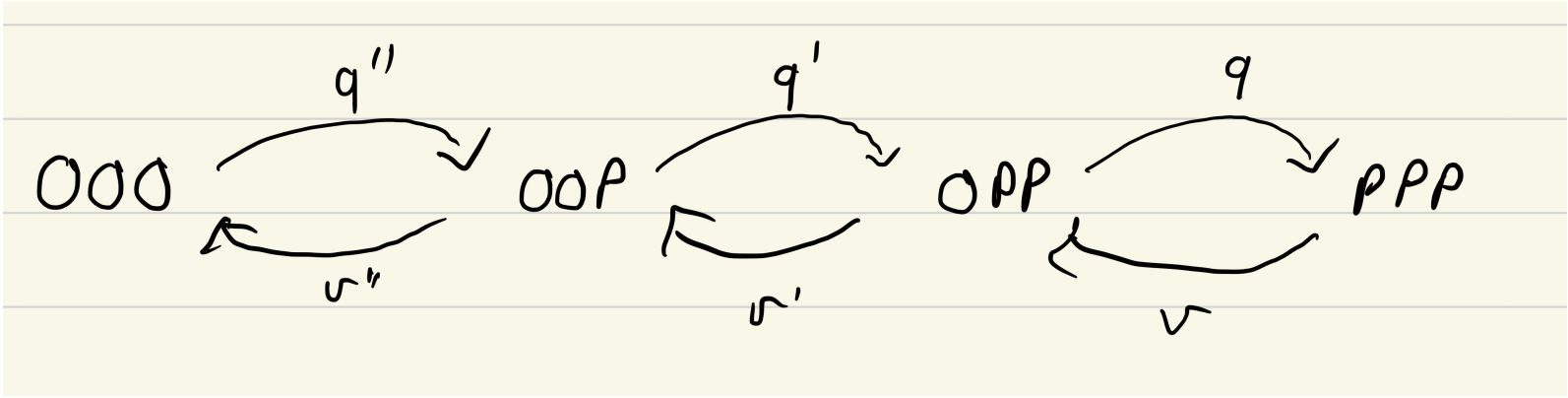
Let A, B, C be three arenas. A legal interaction, or interaction for short, over these arenas is a sequence u of moves from the three arenas such that

$$u \upharpoonright_{A,B} \in L_{A \to B} \quad , \quad u \upharpoonright_{B,C} \in L_{B \to C} \quad , \quad u \upharpoonright_{A,C} \in L_{A \to C}$$

We write $int(A, B, C)$ for the set of legal interactions over A, B, C.

In this definition, say, $u \upharpoonright_{A,B}$ denotes the subsequence of u consisting only of the moves of A, B. One takes care of maintaining the moves of A, B, C all distinct by tagging them if needed.

An automaton recognising legal interactions



Composition of strategies

Let A, B, C be three arenas, and let σ (resp. τ) be a strategy of $A \to B$ (resp. $B \to C$). The following defines a strategy of $A \to C$, called the composition of σ and τ:

$$\tau \circ \sigma = \{v \mid \exists\, u \in int(A, B, C)\ \ v = u \restriction_{A,C}, u \restriction_{A,B} \in \sigma, v \restriction_{B,C} \in \tau\}\ .$$

(we say that u is a witness of v).

In the vocabulary of concurrency theory:

(|| composition) + hiding

Associativity of composition

If $u \in int(A, C, D)$ and $v \in int(A, B, C)$ are such that $u \restriction_{A,C} = v \restriction_{A,C}$, then there is a unique $w \in int(A, B, C, D)$ such that $w \restriction_{A,C,D} = u$ and $w \restriction_{A,B,C} = v$.

If σ, τ, υ are strategies of $A \to B, B \to C$, and $C \to D$, respectively, then $\upsilon \circ (\tau \circ \sigma) = (\upsilon \circ \tau) \circ \sigma$.

Identity strategy

We define

$$id' = \{u \in L_{A \to A} \mid v \restriction_1 = v \restriction_2 \text{ for all even prefixes } v \text{ of } u\}$$

We define id'' as the smallest set of plays closed under the following rules:

$$\frac{}{\epsilon \in id''} \qquad \frac{v \in id'', a\ O \text{ move}}{va_2a_1 \in id''} \qquad \frac{v \in id'', a\ P \text{ move}}{va_1a_2 \in id''}$$

We have $id' = id''$ (*id* for short), and *id* is a strategy.

The identity strategy is an identity

Let A, B, C be three arenas and let $u \in int(A, B, C)$. Then u is a sequence of blocks of the form $mb_1 \dots b_k n$ where m is an O $(A \to C)$ move, the b_i's are B moves, and n is a P $(A \to C)$ move. Moreover, if u is a witness for $\tau \circ \sigma$ (i.e., $u \upharpoonright_{A,B} \in \sigma$, $u \upharpoonright_{B,C} \in \tau$), and if $u = u' mb_1 \dots b_k n$ where $mb_1 \dots b_k n$ is as above, then u' is also a witness.

We have always $id \circ \sigma = \sigma$ and $\sigma \circ id = \sigma$.

Determinism, innocence

A strategy σ is called deterministic when

$$smn_1, smn_2 \in \sigma \;\Rightarrow\; n_1 = n_2$$

The P view $\ulcorner s \urcorner$ of a play s is defined as follows:

$$\begin{array}{ll}
\ulcorner \epsilon \urcorner = \epsilon & \\
\ulcorner sn \urcorner = \ulcorner s \urcorner\, n & (n \text{ P move}) \\
\ulcorner sm \urcorner = m & (m \text{ initial}) \\
\ulcorner sns'm \urcorner = \ulcorner sn \urcorner\, m & (m \text{ O move}, m \text{ points to } n)
\end{array}$$

A deterministic strategy is called innocent if

$$s \in \sigma \Leftrightarrow \ulcorner s \urcorner \in \sigma$$

Arenas and strategies form a category. It contains as subcategories the categories of arenas and deterministic strategies, and of arenas and innocent strategies (idem for well-bracketed).

Innocence, logically

In a view, O, unlike P, has to play in the immediate subtype:

$$\lambda x. \ldots . \lambda y. x M_1 M_2$$

- P = head variable $x : A_1 \to A_2 \to B$ can be bound far away
- O = initial move of M_1 (or M_2) has to be in type A_1 (or A_2)

Fat versus meager

In the official HO semantics, the meaning of a PCF Böhm tree (or term) is made of all plays whose view is in (the transcription of the) tree (as strategy), as stressed in this course. We call the resulting set of plays the fat version. In contrast, we call our preferred version, consisting of views only, the meager version.

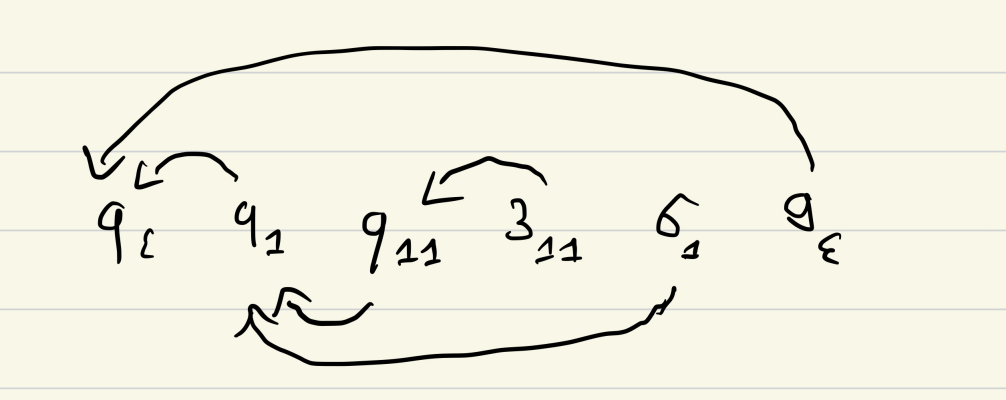
Theorem. The composition defined on the fat versions through (|| composition) + hiding is the fat version of the composition of the meager versions via the game abstract machine.

Illustrating ($\|$ composition) + hiding

The (unique) legal interaction witnessing the interaction

$$q_\epsilon[q_1, \overset{0}{\hookleftarrow}] \begin{cases} q_{11}[3_{11}, \overset{0}{\hookleftarrow}] \\ 4_1[7_\epsilon, \overset{1}{\hookleftarrow}] \\ 6_1[9_\epsilon, \overset{1}{\hookleftarrow}] \end{cases} \qquad q_1[q_{11}, \overset{0}{\hookleftarrow}] \begin{cases} 0_{11}[3_1, \overset{1}{\hookleftarrow}] \\ 3_{11}[6_1, \overset{1}{\hookleftarrow}] \end{cases}$$

is



(which belongs to the fat version of the strategy on the left)

The key result of Hyland and Ong

The properties characterizing (the injective interpretation of) PCF *Böhm trees* are

determinism, innocence and well-bracketing

(For AJM, characterization via history-freeness)

A cartesian closed category

We get the canonical currying isos "for free" (the moves are the same, up to retagging):

$$\frac{(A \times B) \to C}{A \to (B \to C)}$$

Moreover, one can interpret fixpoints (since we were able to read infinite Böhm trees as strategies).

Therefore, we have a model of PCF.

Full abstraction

We write $M =_{obs} N$ iff for all C s.t. $C[M]$ and $C[N]$ are closed and of base type, we have: $C[M] \longrightarrow^\star c$ iff $C[N] \longrightarrow^\star c$.

A model is **fully abstract** when

$$[\![M]\!] = [\![N]\!] \quad \text{iff} \quad M =_{obs} N$$

The "if" direction is the difficult one

Definable separability

If the model is such that for every distinct f, g of the same type A (interpreting some syntactic type) there exists a *definable* h of type $A \to \mathsf{bool}$ such that $hf \neq hg$, then it is fully abstract.

Compact definability + *extensionality* imply definable separability.

Some results and some non-results

Language	Model	Def.	FA
PCF + **por**	$Cont$	**Yes**	**Yes**
PCF + **catch**	$SA \approx (\mathbf{G}_{inn}/=_{op})$	**Yes**	**Yes**
PCF	$PCFBT/=_{op}$	**Yes**	**Yes**
PCF	$\mathbf{G}_{AJM}, \mathbf{G}_{HO}, PCFBT$	**Yes**	*No*
PCF + *control*	$\mathbf{G}_{inn}$	**Yes**	*No*
Idealized Algol	$\mathbf{G}_{wb}$	**Yes**	**Yes**

HO games vs sequential algorithms

Interpret exponential differently:

- HO games are *repetitive*, bool→bool infinite
- Sequential algorithms have *memory*, bool→bool infinite

Cf. two exponentials of coherence spaces (multiset and set)

Around full abstraction

• Stable model (Berry, reinvented by Girard) $\mapsto$ **linear logic**.
• Sequential algorithms led ... to the categorical abstract machine, and then to explicit substitutions.
• The full abstraction problem boosted also the study of logical relations (Sieber, O'Hearn and Ricky, Bucciarelli), and motivated Bucciarelli-Ehrhard's and Longley's extensional accounts of sequentiality.

The game semantics program

- references (Abramsky, McCusker, Honda)
- control (Laird)
- subtyping (Chroboczek)
- nondeterminism (Harmer), probabilistic choice (Danos and Harmer)
- call-by-value (Honda and Yoshida)
- concurrency (Ghica and Murawski, Laird)

Imperative arenas

- The arena **comm**: run and done
- The arena **var**:

read write(n) ($n \in \omega$)
OK n ($n \in \omega$)

The strategy cell :

write(0) OK read 0
write(0) OK write(2) OK read 2

reads last written value (not innocent)

Cell discipline enforced by interaction

$$[\![(x := 0); (x := x + 1)]\!] =$$

$$\mathsf{run}_\epsilon \; \mathsf{write}(0)_1 \mathsf{OK}_1 \mathsf{read}_1 \left\{ \begin{array}{l} \vdots \\ n_1 \; \mathsf{write}(n+1)_1 \; \mathsf{OK}_1 \; \mathsf{done}_\epsilon \\ \vdots \end{array} \right.$$

Interaction play with cell:

$$\mathsf{run}_\epsilon \; \mathsf{write}(0)_1 \; \mathsf{OK}_1 \; \mathsf{read}_1 \; 0_1 \; \mathsf{write}(1)_1 \; \mathsf{OK}_1 \; \mathsf{done}_\epsilon$$

Definability through factorization

Every strategy $\sigma : A$ can be written as $(\tau\ \mathsf{cell})$ for some *innocent* strategy $\tau : \mathsf{var} \to A$

Other such results:

- catch for non well-bracketing (Laird)
- a dice strategy for nondeterminism / probabilistic games (Danos, Harmer)
- a form of case for non-rigidity (a condition dual to well-bracketing) (Danos, Harmer, Laurent)

Definable separability

Essentially the same argument for sequential algorithms,and for the IA (non innocent) model

Let $\sigma_1, \sigma_2 : A$, let $m_1 \ldots m_{2n} \in \sigma_1 \setminus \sigma_2$. We define $\tau : A \to \mathsf{nat}$ as the minimal strategy that contains

$$q\, m_1 \ldots m_{2n}\, 0$$

Game semantics for verification

Up to **second-order**:

- pointers are useless, i.e., can be reconstructed uniquely (Ghica and McCusker)
- the strategies interpreting second-order IA terms (as sets of words) are *regular languages*.

Applications to decidability results. More results have been obtained for larger fragments (third order: Ong and others)