Ghica 1.1

Denotational Semantics vs. Operational semantics

$\uparrow$

translational

$\uparrow$

need meta-language.

$t, c \longmapsto t', c'$

term context

dynamics self-contained, Syntactic here

SD := semantic domain

PL := programming language

$[\![\ ]\!] ; PL \to SD$

an SD should be a well-understood mathematical system

$$[\![K; t_0, \ldots, t_n]\!] = f([\![t_0]\!], [\![t_1]\!], \ldots, [\![t_n]\!])$$

Compositional

E regular languages — Kleene algebras (syntax); FSA (semantic)

Equivalence in PLs

Operational: $t \mapsto t'$, $t \equiv t'$ BUT $\forall C[\ ]$ $C[t] \stackrel{?}{\equiv} C[t']$ in general, no

a context is a program w a hole in it.

Denotational: $t \mapsto t'$ $t \equiv t'$ your PL equivalence is just SD equality

$$[\![C(t)]\!] = [\![t]\!] \circ [\![C]\!] \qquad [\![t]\!] = [\![t']\!] \Leftrightarrow \forall C, [\![C(t)]\!] = [\![C(t')]\!]$$

Relating OS w DS

$=, \bot, T \in SD$

① termination $t\Uparrow := [\![t]\!] = \bot$

$t\Downarrow := [\![t]\!] = T$

② observational equivalence $t_0 \equiv t_1$

$\Leftrightarrow \forall C[\ ], C[t_0]\Downarrow$ iff $C[t_1]\Downarrow$

prop1 Soundness := $[\![t_0]\!] = [\![t_1]\!] \Rightarrow t_0 \equiv t_1$

prop2 Adequacy := $[\![t]\!] = T \Leftrightarrow t\Downarrow$

Prop3 Definability := $\forall \tau \in SD\ \exists t \in PL\ [\![t]\!] = \tau$

Prop4

Scanned by CamScanner

Ghica 12

Full Abstraction

$[\![t_0]\!] = [\![t_1]\!] \Leftrightarrow t_0 \equiv t_1$

R we're in a CBN PCF

T $1+2+3 \Rightarrow FA$

PS assume $[\![t_0]\!] \neq [\![t_1]\!]$

$\exists^* \tau \in SD \quad [\![t_0]\!] \circ \tau = \top \neq \bot$

(from 3) $\Rightarrow \exists C,\ [\![C]\!] = \tau$

$[\![t_0]\!] \circ \tau = [\![t_0]\!] \circ [\![C]\!] = [\![C(t_0)]\!] = \top$

? $\Rightarrow$ (from 2) $C[t_0]\Downarrow \qquad C[t_1]\Uparrow$?

Game Semantics (is a denotational semantics)

interaction

two protagonists
- P (roponent) (player)
- O (pponent) (lincoln)

"moves" (actions)

	O	P
Q		
A		

Q(uestions) correspond to function calls

A(nswers) correspond to function returns

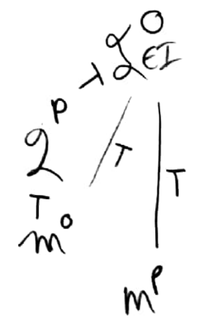
E1 program 3 is the constant 0

$0 : \mathbb{N}$

q^O ← opponent asks "what are you" $\in I$

0^P (answer) "I'm zero"

E2 $\lambda x. x : Nat \to Nat$



D

play := interactions

all plays := strategies
↳ SD

arena := $\langle M, Q, I, \vdash \rangle$

where M is a set of moves

$Q \subseteq M \quad A = M \setminus Q$

$O \subseteq M \quad P \subseteq M \setminus O$

$I \subseteq O \cap Q$ initial moves

$\vdash \subseteq Q \times M$ enabling

$m \vdash n \Rightarrow m \in O$ iff $n \in P$

$m \notin I$

Ghica 1.3

- where "+" is set sum/coproduct (disjoint union)
- set coproduct is only associative up to isomorphism

E $\mathbb{N}at = \langle 1+\mathbb{N}, 1, 1, 1, 1\times\mathbb{N}\rangle$

$I = \langle \emptyset, \emptyset, \emptyset, \emptyset, \emptyset \rangle$

Composite arenas

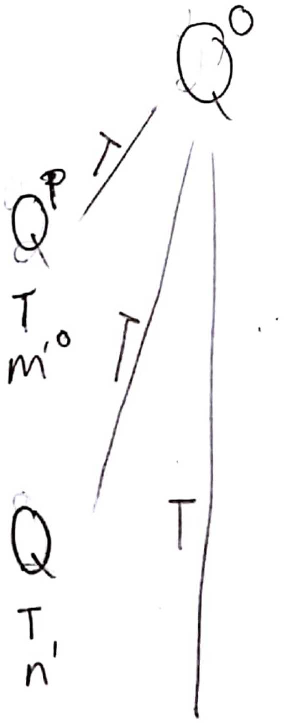
$A\times B = \langle M_A + M_B, \overset{O_A + O_B}{\cancel{Q_A + Q_B}}, I_A + I_B, \vdash_A + \vdash_B \rangle$

$A \to B = \langle M_A + M_B, Q_A + Q, P_A + O_B, inr(I_B), (\vdash_A + \vdash_B) \cup (inr(I_B) \times inl(I_A)) \rangle$

I $(A\times B) \to C \cong A \to (B \to C)$

$I \times A \cong A \cong A \times I$

E_2 $\lambda x.x : Nat \to Nat$



$p = m + n$

a play is sequence + pointers

$Q\,a\langle b\rangle \cdot g' b\langle c\rangle \cdot m' e \cdot g' b$

Ghica 1.4

D a **Play** is a pointer seq in arena A s.t. $p' \cdot m_a \sqsubseteq p$ then $\exists q \in Q_A$ s.t. $q\, b\langle a\rangle \in p'$ and $q \vdash_A m$

if $q\, a\langle b\rangle \sqsubseteq p$ then $q \in I_A$ where $\sqsubseteq$ is <u>prefix</u> relation.

D (Strategy) A strat $\sigma : A$ is a set of plays s.t. $\forall p \in \sigma$

- if $p' \sqsubseteq p$ then $p' \in \sigma$
- $p \cdot m \in Play_A$, $m \in O_A$ then $p \cdot m \in \sigma$
- $\forall$ permutation $\pi : A \to A$ $\quad \pi \cdot p \in \sigma$

$\pi \cdot \varepsilon = \varepsilon$

$\pi \cdot (p \cdot m\, a\langle b\rangle) \cancel{\begin{cases} (\pi \cdot p) \cdot m, \\ \pi(a) \end{cases}}$

$= (\pi \cdot p) \cdot m\ \pi(a)\langle \pi(b)\rangle\ \pi \cdot (p \cdot m a)$

$= (\pi \cdot p) \cdot m\, \pi(a)$

Ghica 2.1

$\sigma_* : Nat' \to Nat'' - Nat$

env asks "what is val?" q^O

op. asks "I need input" q^P

n^O

q^P

m^O

$p^P = m * n$

"time flows downward"

σ is a strategy for some operator $*$

arena is type sig.

$\to$ is arena-constructor

subtypes are also arenas

$Nat \to Nat$

$q \quad \dashv \quad q$

$\top \quad \top$

$n \quad n$

these pointer arcs are backwards-avoiding

$q^O \cdot q^{P'} \cdot n' \cdot q^{P''} \cdot m^{O''} \cdot p$ and $p = m * n$

R interaction between operator and environment

R env. is always first move

R notice columns follow type signature

R enabling is causal but not dataflow dependency

dangling pointer

$qa\langle b \rangle \cdot q' b \langle c \rangle$

introduce a fresh name.

b is an address of q.

$\sigma_{\div} : Nat' \to Nat'' \to Nat$

q^O

q^P

n^O

q^P

divide by zero $\to$ 0^O

game over, abort

$\sigma(p^O) = m^P$

"if I see a game end in an O move, I'll give you the next P move"

$\sigma_{\div} : Nat' \to Nat'' \to Nat$

q^O

q^P

0^O

$p = 0$

Ghica did

N M, is a subterm

Local State

int x in $M \overset{def}{=}$ new(λx:var. M) where M : nat, new : (var → command) → command

read^{QO}, write(n)^{QO} ⊣ q' ⊣ q

n^{PA} ok^{PA} a' a

write := p · wr(n) ↦ ok

rd :=

previous action is either read or write.

p·rd·n·rd ↦ n

p·wr(m)·ok·rd ↦ m

q ↦ q'

p·a' ↦ a

q·q'·rd ↦ :(

like a crash

1+nat ≅ option nat

Control

... escape x ...

label x in M : 1+nat

||| defn

catch (λx.M)

Catch : (Command → Nat) → (option Nat)

Q

Q

n

inr(n)

Q^O

Q^P

Q^O

inl[*]

composing strategies

$A \xrightarrow{\sigma} B \xrightarrow{\tau} C$, $\sigma;\tau : A \to C$

1. Sync on B
2. restart
3. hiding B

two operations on traces

1. Hiding take a pointer sequence

defined inductively on sequences

suppose we want to hide a move

p L x

sequence, hide operator, set of moves

p L x = (p', π)

new seq, bij A→A name reassigner

ε L x = ε (empty case)

m∉x p · m a⟨b⟩ L x = (p' ; π(a)⟨b⟩, π)

m∈x p · m a⟨b⟩ L x = (p' ; (π ∪ (b ↦ π(a))))

Ghica 2.3

② selecting/threading

$$p \upharpoonright X = (p', X') \qquad X \leq A$$

$$\varepsilon \upharpoonright X = \varepsilon$$

$$p \cdot ma\langle b\rangle = (p' \cdot ma\langle b\rangle, X \cup \{b\}) \qquad a \in X$$

$$p \cdot ma\langle b\rangle \upharpoonright X = (p', X) \qquad a \notin X$$

justified sequences, not necessarily strategy

<u>D</u> Synchronization ("interaction") $\sigma \subseteq J_M, \ \tau \subseteq J_N$

$$\sigma \| \tau = \{ p \in J_{M \cup N} \mid p \restriction (M \setminus N) \in \tau, \ p \restriction (N \setminus M) \in \sigma \}$$

<u>D</u> Iteration $\sigma \subseteq J_M \qquad N \subseteq M$

$$!_N \sigma = \{ p \in J_M \mid \forall\ ma\langle b\rangle \in p\ m \in N \Rightarrow p \upharpoonright \{b\} \in \sigma$$

<u>D</u> Composition $\sigma : A \to B, \ \tau : B \to C$

$$\tau \circ \sigma = \sigma ; \tau = (!_B\, \sigma \| \tau) \restriction M_B$$

$M_{A \to B},$

$M_{B \to C}$

Ghica 2.4

<u>Q</u> Is composition sensible?

- is it well formed
  - equivariant
  - well justified? yes,
  - prefix-closed
- is it associative
- monotonicity $\dfrac{\sigma \subseteq \sigma'}{\tau;\sigma \subseteq \tau;\sigma' \quad \sigma;\tau \subseteq \sigma';\tau}$

<u>Exercises/Lemma</u>

- $p \in P_{A \cancel{\times} B \to C}$ then $p \upharpoonright M_A \in P_{B \to C}$, $p \upharpoonright M_B \in P_{A \to C}$
- $\forall$ - $ma\langle b\rangle \in P$, $m \in I_A$ then $p \upharpoonright \{b\} \in P_A$
- $p \upharpoonright M_A \upharpoonright M_B = p \upharpoonright M_B \upharpoonright M_A$
- $p \upharpoonright M_c \upharpoonright \{b\} = p \upharpoonright \{b\} \upharpoonright M_c$

there is an identity K

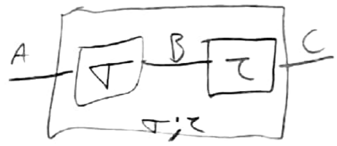
$K_A;\sigma = \sigma$

$\sigma;K_A = \sigma$

Scanned by CamScanner

Ghica 3.1

Composition = synch + replication + hiding



· associative

· identity?

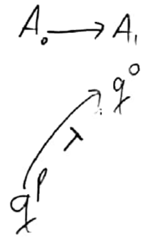
Kappa for copycat,

$\kappa_A : A_0 \longrightarrow A_1$

$inl(m) = m_0$ $inr(m) = m_1$

· define via a next move function $\hat{\kappa}_A$

$q_1\, a\langle b\rangle \mapsto q_0\, b\langle c\rangle$



Copycat strategy

$p \cdot m_{1-j}\, a\langle b\rangle \cdot m_j\, c\langle\langle d\rangle \cdot p' \cdot n^o_i\, d\langle e\rangle \longmapsto n^p_{1-j}\, b\langle f\rangle$

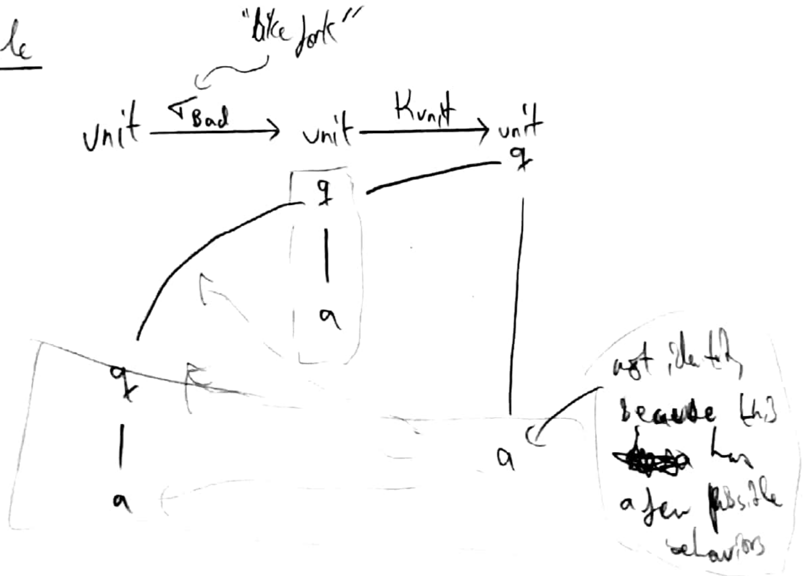
L① for $\kappa_A : A_1 \longrightarrow A_2$

$\kappa_A \restriction A_1 = \kappa_A \restriction A_2$

L② κ is a strategy

nice T $\boxed{\kappa_A ; \sigma = \sigma ; \kappa_B = \sigma}$ → but it's not true

Counterexample



Ghica 3,2

How to fix identity

A. discipline plays w extra conditions (rule out σ_{BAD})

B. Add closure conditions on strategies

Careful - language specific

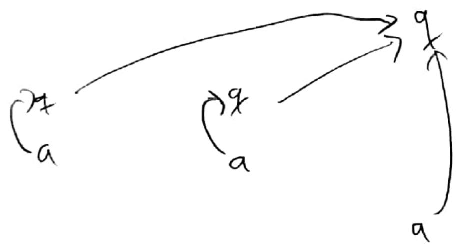
Concurrency - async games

R: enabling is in arena
a play can instantiate a move multiple times
justification is the per-instance version [of] en

Extra Conditions b.c. "static concurrency" $C_1 \| C_2$

Fork/Join : If a $\underbrace{\text{thread finishes}}_{\text{question is answered}}$, then all its $\underbrace{\text{subthreads}}_{\text{justified questions}}$ must have $\underbrace{\text{finished}}_{\text{been answered}}$

typical strategy σ_{par} : $Com_1 \longrightarrow Com_2 \longrightarrow Com$



D strict scoping

$p \cdot m\,a\langle\_\rangle \cdot p' \in P$

and $m \in A$ then $a \notin p'$

D strict nesting

$p \cdot m_1 a\langle b\rangle \cdot m_2 b\langle c\rangle p_3 \cdot m\, b\langle d\rangle \in P$

and $m_1 \in A$ then $\exists n_2 \in A \;\; n_2 c\langle\_\rangle \in P_3$

T $A \xrightarrow{\sigma} B \xrightarrow{\tau}$ have Strongly Nested (SN), Strongly Scoped (SS) plays

$\sigma;\tau$ has SN, SS plays

when can two moves synch in a play?

- .... m_1 m_2 ....
- .... O_{move} P_{move} ....

Scanned by CamScanner

Ghica 3.3

D asynchronous strategy

Asynch strat $\sigma : A$ of SS/SN plays

i) $p \cdot \underbrace{m_0 a_0 \langle b_0 \rangle}_{p_0} \cdot \underbrace{m_1 a_1 \langle b_1 \rangle}_{p_1} \in \sigma$ and $(m_0 \in P_A$ or $m_1 \in O_A)$ and $p \cdot p_1 \cdot p_0 \in P_A$

then $p \cdot p_1 \cdot p_0 \in \sigma$

Concurrent Idealized Algol

PCF + local state + concurrency + binary sems

newvar, asg, der (local state); par (concurrency); newsem, grab, release (binary sems)

$\Gamma \vdash M : \theta$

$\parallel$

$x_0 : \theta_0, \ldots, x_k : \theta_k$

Challenge

- $\lambda c.\ \text{new}\ x = 0\ \text{in}\ c; !x \equiv \lambda c.\ c; 0$

$\lambda f.\ \text{new}\ x = 0\ \text{in}\ f(x := !x+1)(!x) \equiv \lambda f.\ \text{new}\ y = 0\ \text{in}\ f(x := !x-1)(-!x)$

general recipe for interpreting into CCC

$[\![Nat]\!] = Nat \quad [\![var]\!] = Var$

$[\![\theta \to \theta']\!] = [\![\theta]\!] \to [\![\theta']\!]$

$[\![\Gamma]\!] = [\![\theta_0]\!] \times \cdots \times [\![\theta_n]\!]$

$[\![\Gamma, x_k : \theta_k, \Gamma' \vdash x_k : \theta_k]\!] = \pi_k = \text{strat}(K_{[\![\theta_k]\!]})$

$[\![\Gamma \vdash \lambda x. M : \theta \to \theta']\!] = \lambda [\![\Gamma, x : \theta \vdash M : \theta']\!]$

$[\![\Gamma \vdash MN : \theta]\!] = \langle [\![M : \theta' \to \theta]\!], [\![N : \theta']\!] \rangle ; ev$

$= ([\![M]\!] \cup [\![N]\!]) ; \text{strat}(K_\theta \cup K_{\theta'})$

$[\![\cdot \vdash M_0 : \theta_0]\!] = \sigma_{M_0}$

abstraction and application

a model of λ-calc (Cartesian Closed Category (CCC))

- product $A \times B$
- unit s.t. $1 \times A \cong A \cong A \times 1$ — the empty arena
- projections $\pi_k : A_1 \times \cdots \times A_n \longrightarrow A_k$ (copycat strategy) $\pi_k = \text{strat}(K_{A_k})$
- pairing $\langle \sigma, \tau \rangle : A_1 \times A_2 \longrightarrow B$

  $\sigma : A_1 \to B$
  $\tau : A_2 \to B$ $\qquad \langle \sigma, \tau \rangle = \sigma \cup \tau$

- exponential $A \to B$
- transpose $\dfrac{\sigma : A \times B \to C}{\lambda\sigma : A \to B \to C}$
- evaluation strategy

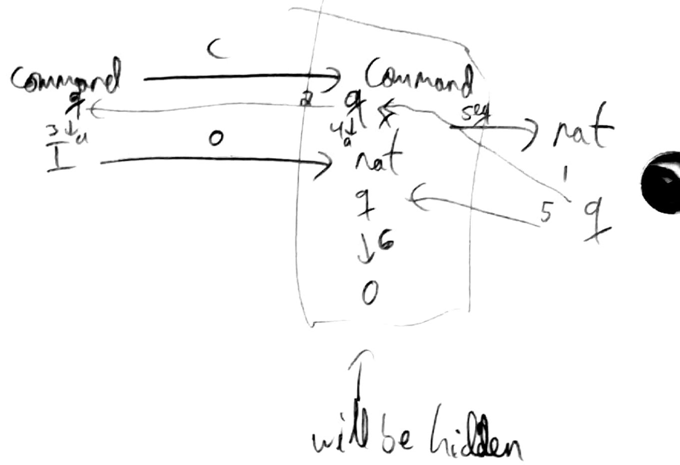
  $ev : (A \to B) \times A \longrightarrow B \qquad ev = K_A \cup K_B$

Ghica 3.4

$\boxed{!x \text{ is Dereferencing}}$

$$[\![\lambda c.\, c;0]\!] = \lambda[\![c;0]\!]$$

$$= \lambda\,\mathrm{strat}(q\, q_c\, a_c\, 0)$$



$$\lambda[\![\mathrm{new}\ x=0\ \mathrm{in}\ c;\ !x]\!] = \mathrm{strat}(q \cdot q_c \cdot a_c \cdot 0)$$

$\lambda x.$ —