Logic and Types, Concurrency and Non Determinism

Luís Caires

Universidade Nova de Lisboa
NOVA Laboratory for Computer Science and Informatics

IFIP WG 2.2 Meeting, Lucca
September 2015

Types and Specifications

Typeful programming is a special case of program specification

Types ~ specifications

Type-checking ~ verification

Useful to enforce correctness by construction, at the PL (compiler) level

This view nicely fits with the Curry-Howard (CH) paradigm of propositions as types and of typed programs as proofs of their type

Linear Logic & Process Types

We developed (with Pfenning / Toninho) [CP/Concur10, CTP/MSCS14] a CH interpretation of linear logic propositions as session types.

Key highlights

One gets very natural session type system for π-calculus processes, ensuring deadlock freedom and session fidelity by purely logical means.

Proof conversions express basic laws of observational equivalence.

Proof reduction matches process (full, internal) reduction, and is confluent and terminating.

The normal form (up to some conversions) denotes the (essentially deterministic) behaviour of the given session-typed process / proof.

Can be used to give a deep foundation to existing session-based programming language [Wad/ICFP'14] (translation of GV into CP).

Types and Specifications

Typeful programming is a special case of program specification

Types ~ specifications

Type-checking ~ verification

Useful to enforce correctness by construction, at the PL level

This view nicely fits with the Curry-Howard (CH) “paradigm”

Within CH is often fruitful to investigate to what a concept on one side might correspond on the other side (in both directions), e.g.,

- dependent types ~ interface contracts and certificates [TCP-PPDP11]
- polymorphic types ~ generic behavioural types [CCPT/ESOP13]
- contextual monadic types ~ mobile higher order code [TCP/ESOP13]

What about non-determinism?

Non-determinism seems to resist a reasonable logical analysis in terms of a Curry-Howard correspondence.

Proof reduction must be compatible with behavioural equivalence (an identity) so non-deterministic proof reductions would not make sense

Still, non-deterministic behaviour in the essence of many artificial and natural concurrent systems (e.g., physical / biological ones)

In this talk

I review a basic type system (based on classical linear logic + mix)

I sketch an approach to integrate non-determinism in the framework, extending the system in a principled way

Linear Types as Sessions

A, B	$::=$	$\bot$	$\bullet$	end
	$\mid$	$\mathbf{1}$	$\bullet$	end
	$\mid$	$!A$		shared publish
	$\mid$	$?A$		shared invoke
	$\mid$	$A \otimes B$	$A!.B$	send
	$\mid$	$A \parr B$	$\overline{A}?.B$	receive
	$\mid$	$A \oplus B$		choice select
	$\mid$	$A \,\&\, B$		choice offer

$\bot \equiv \mathbf{1}$

Duality

$$\overline{\mathbf{1}} = \bot$$
$$\overline{\bot} = \mathbf{1}$$
$$\overline{!A} = ?\overline{A}$$
$$\overline{?A} = !\overline{A}$$
$$\overline{A \otimes B} = \overline{A} \parr \overline{B}$$
$$\overline{A \parr B} = \overline{A} \otimes \overline{B}$$
$$\overline{A \oplus B} = \overline{A} \,\&\, \overline{B}$$
$$\overline{A \,\&\, B} = \overline{A} \oplus \overline{B}$$

Typing Judgments

Syntactical form of the Typing Judgment - based on [And92]

$P \vdash \Delta \; ; \; \Theta$

P process (this is a standard π-calculus program)

Δ linear context (declares types of "currently open" sessions)

Θ cartesian context (types of shared servers, implicitely ?'ed)

The judgment "morally" states:

Process P is lock free under reduction (compositionally).

All interactions in P conform to the prescribed protocols.

NB. Typing is closed under composition (using the cut rule).

Parallel composition

$$\frac{P \vdash \Delta, x{:}\overline{A}; \Theta \quad Q \vdash \Delta', x{:}A; \Theta}{(\nu x)(P \mid Q) \vdash \Delta, \Delta'; \Theta}\ (\text{Tcut})$$

$$\frac{}{\mathbf{0} \vdash ; \Theta}\ (\text{T}\cdot) \qquad \frac{P \vdash \Delta; \Theta \quad Q \vdash \Delta'; \Theta}{P \mid Q \vdash \Delta, \Delta'; \Theta}\ (\text{T} \mid)$$

Send and Receive

$$\frac{P \vdash \Delta, y{:}A; \Theta \quad Q \vdash \Delta', x{:}B; \Theta}{\overline{x}(y).(P \mid Q) \vdash \Delta, \Delta', x{:}A \otimes B; \Theta}\ (\mathrm{T}\otimes)$$

$$\frac{R \vdash \Gamma, y{:}C, x{:}D; \Theta}{x(y).R \vdash \Gamma, x{:}C \parr D; \Theta}\ (\mathrm{T}\parr)$$

Principal cut reduction corresponds to communication reduction

$$(\nu x)(\overline{x}(y).M \mid x(y).R) \to (\nu x)(\nu y)(M \mid R) \quad (M \equiv P \mid Q)$$

Termination

$$\frac{}{x.\mathsf{close} \vdash x{:}\mathbf{1}; \Theta}\ (\mathbf{T1})$$

$$\frac{P \vdash \Delta; \Theta}{x.\mathsf{close}; P \vdash x{:}\bot, \Delta; \Theta}\ (\mathbf{T}\bot)$$

Principal cut reduction corresponds to session termination

$$(\nu x)(x.\mathsf{close} \mid x.\mathsf{close}; P) \to P$$

Offer and Choice

$$\frac{R \vdash \Delta, x{:}A; \Theta}{x.\mathtt{inl}; R \vdash \Delta, x{:}A \oplus B; \Theta}\ (\mathrm{T}\oplus_1)$$

$$\frac{R \vdash \Delta, x{:}B; \Theta}{x.\mathtt{inr}; R \vdash \Delta, x{:}A \oplus B; \Theta}\ (\mathrm{T}\oplus_2)$$

$$\frac{P \vdash \Delta, x{:}A; \Theta \quad Q \vdash \Delta, x{:}B; \Theta}{x.\mathtt{case}(P, Q) \vdash \Delta, x{:}A \,\&\, B; \Theta}\ (\mathrm{T}\&)$$

Principal cut reductions corresponds to case selection reductions

$$(\nu x)(x.\mathtt{case}(P, Q) \mid x.\mathtt{inl}; R) \to (\nu x)(P \mid R)$$
$$(\nu x)(x.\mathtt{case}(P, Q) \mid x.\mathtt{inr}; R) \to (\nu x)(Q \mid R)$$

Example: Movie Server

System

$$SBody(s) \triangleq s.\mathtt{case}(s(title).s(card).\overline{s}(movie).\mathsf{close},\ s(title).\overline{s}(trailer).\mathsf{close})$$

$$Alice(s) \triangleq s.\mathtt{inr}; \overline{s}(\text{“}solaris\text{”}).s(preview).\mathsf{close}; \mathbf{0}$$

$$System \triangleq (\boldsymbol{\nu} s)(SBody(s) \mid Alice(s))$$

Types Assigned

$$SBT \triangleq (T \multimap C \multimap M \otimes \mathbf{1}) \ \& \ (T \multimap M \otimes \mathbf{1})$$

$$SBody(s) \vdash s : SBT$$

$$Alice(s) \vdash s : \overline{SBT}$$

$$System \vdash \cdot$$

Sharing and Replication

$$\frac{P \vdash \Delta; x{:}A, \Theta}{P \vdash \Delta, x{:}?A; \Theta}\ (\mathrm{T?}) \qquad \frac{Q \vdash y{:}A; \Theta}{!x(y).Q \vdash x{:}!A; \Theta}\ (\mathrm{T!})$$

$$\frac{P \vdash \Delta, y{:}A; x{:}A, \Theta}{\overline{x}_?(y).P \vdash \Delta; x{:}A, \Theta}\ (\mathsf{Tcopy})$$

Principal cut reduction corresponds to shared server invocation

$$(\nu x)(!x(y).Q \mid \overline{x}_?(y).P) \to (\nu x)(!x(y).Q \mid (\nu y)(P \mid Q))$$

NB. Rule (T?) is silent on the process term (just bookkeeping)

Ex: Shared Movie Server

System

$$MOVIES(srv) \triangleq !srv(s).SBody(s)$$

$$SAlice(srv) \triangleq \overline{srv}_?(s).Alice(s)$$

$$SBob(srv) \triangleq \overline{srv}_?(s).s.\mathtt{inl}; \overline{s}(\text{“}inception\text{”}).$$
$$\overline{s}(bobscard).s(mpeg).s.\mathsf{close}; \mathbf{0}$$

$$System \triangleq (\boldsymbol{\nu} srv)(MOVIES(srv) \mid Alice(srv) \mid Bob(srv))$$

Types Assigned

$$MOVIES(srv) \vdash srv : !SBT;$$

$$Alice(srv) \vdash \cdot; srv : \overline{SBT}$$

$$Bob(srv) \vdash \cdot; srv : \overline{SBT}$$

$$Alice(srv) \mid Bob(srv) \vdash srv : ?\overline{SBT};$$

$$System \vdash \cdot; \cdot$$

Non Determinism

Non Determinism

We rely on two ingredients

We introduce a "sum" process construct

$P \oplus Q$ represents "superposed" (alternative) states

Alternative states do not "collapse" during cut elim / reduction, but either proliferate or simplify (e.g., $P \oplus \mathsf{none} \equiv P$)

At the level of types, we consider two new modalities

$\&A$ the type of sessions that may produce some session of type A

$\oplus A$ the type of sessions that may consume any session of type A

We have $\overline{\&A} = \oplus\overline{A}$

NonDet Operators (rules)

$$\frac{P \vdash \Delta, x{:}A; \Theta}{x.\overline{\mathsf{some}}; P \vdash \Delta, x{:}\&A; \Theta}\ (\mathrm{T}\&_d)$$

$$\frac{P \vdash \Delta; \Theta}{\overline{x.\mathsf{none}}; P \vdash \Delta, x{:}\&A; \Theta}\ (\mathrm{T}\&_w) \qquad \frac{P \vdash \&\Delta; \Theta \quad Q \vdash \&\Delta; \Theta}{P \oplus Q \vdash \&\Delta; \Theta}\ (\mathrm{T}\&_c)$$

$$\frac{P \vdash w{:}\&\Delta, x{:}A; \Theta}{x; \mathsf{some}_{\overline{w}}.P \vdash w{:}\&\Delta, x{:}\oplus A; \Theta}\ (\mathrm{T}\oplus)$$

$\&A$ additive monad, $\oplus A$ co-monad (cf. ?,!)

NonDet Operators (reduction)

$$\frac{P \vdash \overline{w}{:}\&\Delta, x{:}\overline{A}; \Theta \qquad Q \vdash \Delta', x{:}A; \Theta}{x.\mathsf{some}; P \vdash \overline{w}{:}\&\Delta, x{:}\oplus\overline{A}; \Theta \qquad x.\overline{\mathsf{some}}; Q \vdash \Delta', x{:}\&A; \Theta}$$

$$(\boldsymbol{\nu} x)(x.\mathsf{some}_{\overline{w}}; P \mid x.\overline{\mathsf{some}}; Q) \vdash \&\Delta, \Delta'; \Theta$$

$$\rightarrow$$

$$\frac{P \vdash \&\Delta, x{:}\overline{A}; \Theta \quad Q \vdash \Delta', x{:}A; \Theta}{(\boldsymbol{\nu} x)(P \mid Q) \vdash \&\Delta, \Delta'; \Theta}$$

N.B. Cf. promotion - ?d reduction in CLL, but ≠ interpretation

NonDet Operators (reduction)

$$\dfrac{\dfrac{P \vdash \overline{w}{:}\&\Delta, x{:}\overline{A}; \Theta}{x.\mathsf{some}; P \vdash \overline{w}{:}\&\Delta, x{:}\oplus\overline{A}; \Theta} \quad \dfrac{Q \vdash \Delta'; \Theta}{\overline{x.\mathsf{none}}; Q \vdash \Delta', x{:}\&A; \Theta}}{(\boldsymbol{\nu} x)(x.\mathsf{some}_{\overline{w}}; P \mid x.\overline{\mathsf{none}}; Q) \vdash \overline{w}{:}\&\Delta, \Delta'; \Theta} \rightarrow \dfrac{Q \vdash \Delta'; \Theta}{(\boldsymbol{\nu} x)(\overline{w}.\overline{\mathsf{none}} \mid Q) \vdash \overline{w}{:}\&\Delta, \Delta'; \Theta}$$

N.B. Cf. !-promotion - ?w reduction in CLL, but ≠ interpretation

NonDet Operators (reduction)

$$\dfrac{P \vdash \&\Delta, x{:}\oplus\overline{A}; \Theta \qquad \dfrac{Q \vdash \&\Delta', x{:}\&A; \Theta \quad R \vdash \&\Delta', x{:}\&A; \Theta}{Q \oplus R \vdash \&\Delta', x{:}\&A; \Theta}}{(\boldsymbol{\nu} x)(P \mid (Q \oplus R)) \vdash \&\Delta, \&\Delta'; \Theta}$$

$$\rightarrow$$

$$\dfrac{\dfrac{P \vdash \&\Delta, x{:}\oplus\overline{A}; \Theta \quad Q \vdash \&\Delta', x{:}\&A; \Theta}{(\boldsymbol{\nu} x)(P \mid Q) \vdash \&\Delta, \&\Delta'; \Theta} \qquad \dfrac{P \vdash \&\Delta, x{:}\oplus\overline{A}; \Theta \quad R \vdash \&\Delta', x{:}\&A; \Theta}{(\boldsymbol{\nu} x)(P \mid R) \vdash \&\Delta, \&\Delta'; \Theta}}{(\boldsymbol{\nu} x)(P \mid Q) \oplus (\boldsymbol{\nu} x)(P \mid R) \vdash \&\Delta, \&\Delta'; \Theta}$$

cut commutation rule captures distribution of | over internal choice

$$(\boldsymbol{\nu} x)(P \mid (Q \oplus R)) \equiv (\boldsymbol{\nu} x)(P \mid Q) \oplus (\boldsymbol{\nu} x)(P \mid R)$$

Cf. familiar behavioural equivalence equation in PA

Some basic laws

$\vdash x{:}\oplus\oplus\overline{A}, y{:}\&A \;;\; -$ (cf. $\&\&A \multimap \&A$)

$\vdash x{:}\overline{A}, y{:}\&A \;;\; -$ (cf. $A \multimap \&A$)

$\vdash y{:}\&\overline{A}, x{:}A \;;\; -$ (cf. $\oplus A \multimap A$)

$\vdash x{:}\oplus\bullet, y{:}\bullet \;;\; -$ (cf. $\&\bullet \multimap \bullet$)

Simple Example

System

$$Alice(s) \triangleq s.\mathtt{inr}; \overline{s}(\text{“}solaris\text{”}).s(preview).\mathsf{close}; \mathbf{0}$$
$$Bob(s) \triangleq s.\mathtt{inl}; \overline{s}(\text{“}inception\text{”}).\overline{s}(bobscard).s(movfile).\mathsf{close}; \mathbf{0}$$
$$Schizo(s) \triangleq s.\overline{\mathsf{some}}; Alice(s) \oplus s.\overline{\mathsf{some}}; Bob(s)$$
$$NSystem \triangleq (\boldsymbol{\nu} s)(s.\mathsf{some}_{\emptyset}; SBody(s) \mid Schizo(s))$$

Types Assigned

$$SBT \triangleq (T \multimap C \multimap M \otimes \mathbf{1}) \;\&\; (T \multimap M \otimes \mathbf{1})$$
$$\mathsf{some}_{\emptyset}; SBody(s) \vdash s : \oplus SBT$$
$$Schizo(s) \vdash s : \&\overline{SBT}$$
$$NSystem \vdash \cdot$$

“Interruptible” Sessions

$$SBuyer(srv) \triangleq \overline{srv}_?(s).\overline{s}(item).\\ s(info).\\ \mathsf{if}\ G(info)\ \mathsf{then}\ Proceed(s)\\ \mathsf{else}\ s.\overline{\mathsf{none}};\ Retry$$

If *info* is not *G* then *SBuyer* “kills” session and tries something else.

No further use of session s is possible after s.none

But *Proceed(s)* can continue with some s behaviour

Inside the server instance, no “garbage” pending (hereditarily)

Encoding "Exceptions"

$$
\begin{aligned}
[\![\mathsf{raise}\ v]\!]_{y,x} = \quad & y.\mathsf{none}; \overline{x}(v) \\
[\![\mathsf{try}]\!]_{y,x} = \quad & (\boldsymbol{\nu} pq)([\![M]\!]_{p,q} \mid \\
& \qquad p.\mathsf{some}; p(v).p(c).\overline{c}(v) \mid \\
& \qquad q(v).\overline{y.\mathsf{some}}.\overline{y}(v).\overline{y}(x))
\end{aligned}
$$

Types of y: $\&(!V.!!V.\mathsf{end})$ $\quad x$: $!V$

y used as continuation, gets the value and exception handler loc

x is the context exception handler location

Encoding "Exceptions"

$$[\![\mathsf{raise}\ v]\!]_{y,x} = \quad y.\mathsf{none}; \overline{x}(v)$$
$$[\![\mathsf{try}]\!]_d = \quad (\boldsymbol{\nu} pq)([\![M]\!]_{p,q} \mid p.\mathsf{some}; p(v).p(c).\overline{c}(v) \mid q(v).\overline{d}(v))$$

Types of y: $\&(!V.!!V.\mathsf{end})$ $\quad x, d$: $!V$

N.B.: Isolated "killed" computations inside deterministic one

One may also represent more usual try - catch constructs (with explicit exception handlers)

Main Results

$$P \oplus Q \equiv Q \oplus P \qquad a\langle y\rangle.b\langle z\rangle.P \equiv b\langle y\rangle.a\langle z\rangle.P$$

Main Results

1. Cut elimination holds
2. Well-typed processes are (compositionally):

   deadlock free

   confluent

   terminating

   Technique: linear logical relations [PCPT/ESOP'12,I&C'14]
3. Reduction / conversion preserves observational equivalence

   cut normal form is up to "structural" conversions, e.g.,

$$P \oplus Q \equiv Q \oplus P \qquad a\langle y\rangle.b\langle z\rangle.P \equiv b\langle y\rangle.a\langle z\rangle.P$$

Main Results

4. Let $\rightarrow_c$ be the extension of $\rightarrow$ with "non-det collapse rules"

$$P \oplus Q \rightarrow_c P \qquad\qquad P \oplus Q \rightarrow_c Q$$

Then:

THEOREM 5 (POSTPONING). *Let $P \vdash \Delta; \Theta$. We have*

1. If $P \Rightarrow P_1 \oplus \ldots \oplus P_n \not\rightarrow$ with P_i prime for all i, then $P \Rightarrow_c P_i$ for all $0 < i \leq n$.

2. Let $\mathcal{C} = \{ P_i \mid P \Rightarrow_c P_i \not\rightarrow_c$ and P_i is prime $\}$. Then $\mathcal{C}$ is finite up to $\equiv$, with $\#\mathcal{C} = n$, and for all $0 < i \leq n$, $P \Rightarrow P_1 \oplus \ldots \oplus P_n \rightarrow_c P_i$.

Summary

A CH model of non-determinism for session typed processes

Based on CLL+mix

Non-determinism encapsulated by a dual pair of (co)/monads $\&A$ $\oplus A$

These operators obey the basic rules of LL !?-exponentials" with contraction replaced by sum, and a different operational interpretation

(Typed) processes satisfy session fidelity, progress and confluence

Conversions express known laws of behavioural equivalence, but full behavioural theory still needs to be developed

Compatible with standard non-deterministic "collapse" rules, which reflect the local view of the contextual observer, and maybe used to guide implementation of the model in concrete programming languages

Logic and Types, Concurrency and Non Determinism

Luis Caires

Universidade Nova de Lisboa
NOVA Laboratory for Computer Science and Informatics

IFIP WG 2.2 Meeting, Lucca
September 2015