199?

LOGO

Phil Wadler

FPCA

Functional Programming Languages and computer Architectures
199?

Logo

Phil Wadler

FPSCA

Functional Programming Languages and Computer Architectures
Jean-Jacque Levy

Andy Gordon

Luc Maranget
Session Types and Linear Logic

WadlerFest

Bernardo Toninho, Nobuko Yoshida

11 April 2016
Session Types and Linear Logic

Context

Session Types [Honda et al93]:

- Typing discipline for $\pi$-calculus
Session Types and Linear Logic

Context

Session Types [Honda et al93]:

- Typing discipline for $\pi$-calculus
- Structure channel-based comm. with the notion of a session.
  - Sequence of interactive behaviours between two agents.

Linear Logic [Girard98]:

- A substructural logic of resources.
- Marriage of the dualities of classical logic with constructive aspects of intuitionistic logic.
- Far reaching applications in CS (linear $\pi$-calculus, implicit comp. complexity, linear types, etc.)
Session Types and Linear Logic

Context

Session Types [Honda et al93]:

- Typing discipline for $\pi$-calculus
- Structure channel-based comm. with the notion of a session.
  - Sequence of interactive behaviours between two agents.
  - Intrinsic notion of duality (send/receive, branch/select).

Linear Logic [Girard98]:

- A substructural logic of resources.
- Marriage of the dualities of classical logic with constructive aspects of intuitionistic logic.
- Far reaching applications in CS (linear $\pi$-calculus, implicit comp. complexity, linear types, etc.)
Session Types and Linear Logic

Context

Session Types [Honda et al.'93]:

- Typing discipline for $\pi$-calculus
- Structure channel-based comm. with the notion of a session.
  - Sequence of interactive behaviours between two agents.
  - Intrinsic notion of duality (send/receive, branch/select).
- Ensure communication safety (liveness, fidelity, ...)

Linear Logic [Girard '98]:

- A substructural logic of resources.
- Marriage of the dualities of classical logic with constructive aspects of intuitionistic logic.
- Far reaching applications in CS (linear $\pi$-calculus, implicit comp. complexity, linear types, etc.)
Session Types and Linear Logic

Context

Session Types [Honda et al.93]:

- Typing discipline for $\pi$-calculus
- Structure channel-based comm. with the notion of a session.
  - Sequence of interactive behaviours between two agents.
  - Intrinsic notion of duality (send/receive, branch/select).
- Ensure communication safety (liveness, fidelity, . . . )

Linear Logic [Girard98]:

- A substructural logic of resources.
Session Types and Linear Logic

Context

Session Types [Honda et al93]:
- Typing discipline for $\pi$-calculus
- Structure channel-based comm. with the notion of a session.
  - Sequence of interactive behaviours between two agents.
  - Intrinsic notion of duality (send/receive, branch/select).
- Ensure communication safety (liveness, fidelity, ...)

Linear Logic [Girard98]:
- A substructural logic of resources.
- Marriage of the dualities of classical logic with constructive aspects of intuitionistic logic.
- Far reaching applications in CS (linear $\lambda$-calculus, implicit comp. complexity, linear types, etc.)
Session Types and Linear Logic

A bit of history

Propositions as Types:
- A (deep) connection between prop. logic and $\lambda$-calculus.
Session Types and Linear Logic

A bit of history

Propositions as Types:

- A (deep) connection between propositional logic and $\lambda$-calculus.
- Proofs as programs, computation as proof simplification.
Session Types and Linear Logic
A bit of history

Propositions as Types:

- A (deep) connection between prop. logic and $\lambda$-calculus.
- Proofs as programs, computation as proof simplification.
- Not just prop. logic...
Session Types and Linear Logic
A bit of history

Propositions as Types:
- A (deep) connection between prop. logic and $\lambda$-calculus.
- Proofs as programs, computation as proof simplification.
- Not just prop. logic...

Concurrency Theory:
- Process Algebra (CSP [Hoare78], CCS, $\pi$-calculus [Milner80,89])
- Language-based models of message-passing concurrency.
- A plethora of typing systems (I/O types, Usage types, Linear types, . . . , Session types)
Session Types and Linear Logic
A bit of history

Linear Logic and Concurrency:

- A logic of interacting resources?
Session Types and Linear Logic

A bit of history

Linear Logic and Concurrency:

- A logic of interacting resources?
- Initial efforts explored connections to concurrency:
  - Abramsky’s computational interpretation [Abramsky93]
  - Bellin and Scott’s refinement to a $\pi$-calculus [BellinScott94]
  - Specification structures / Interaction cat. [Abramsky et al.95]
Session Types and Linear Logic
A bit of history

Linear Logic and Concurrency:

- A logic of interacting resources?
- Initial efforts explored connections to concurrency:
  - Abramsky’s computational interpretation [Abramsky93]
  - Bellin and Scott’s refinement to a π-calculus [BellinScott94]
  - Specification structures / Interaction cat. [Abramsky et al.95]
- No real “Curry-Howard isomorphism”…
Session Types and Linear Logic
A bit of history

Linear Logic and Concurrency:
- A logic of **interacting** resources?
- Initial efforts explored connections to concurrency:
  - Abramsky’s computational interpretation [Abramsky93]
  - Bellin and Scott’s refinement to a $\pi$-calculus [BellinScott94]
  - Specification structures / Interaction cat. [Abramsky et al.95]
- No real “Curry-Howard isomorphism”…

Why does it matter?
- New means of reasoning about concurrent phenomena.
- Good metalogical properties map to good program properties.
- …
Session Types and Linear Logic
What is old is new again

ILL and Session Types – SILL [CairesPfenning10]

- Interpret Session Types as ILL propositions.
- Proofs as typing derivations for π-calculus.
- Process reduction as cut reduction/elimination.
### ILL and Session Types – SILL [CairesPfenning10]
- Interpret Session Types as ILL propositions.
- Proofs as typing derivations for $\pi$-calculus.
- Process reduction as cut reduction/elimination.

### CLL and Session Types – CP [Wadler12,14]
- Full linear logic.
- Further from $\pi$-calculus, but matching LL precisely.
- Embeds a session-typed functional language (GV) into CP.
Session Types and Linear Logic

Meanings of Propositions

CLL Propositions as Sessions

\[ A, B ::= \]

- \( A \rightarrow B \): output
- \( A \leftarrow B \): input
- \( A \rightarrow B \): select from \( A \) or \( B \)
- \( A \& B \): choice of \( A \) or \( B \)
- \( A \leftarrowoffice? B \): server accept
- \( A \rightarrowoffice? B \): client request
Session Types and Linear Logic
Meanings of Propositions

CLL Propositions as Sessions

\[ A, B ::= A \otimes B \text{ output } A \text{ then behave as } B \]
\[ A \bowtie B \text{ Input } A \text{ then behave as } B \]
Session Types and Linear Logic

Meanings of Propositions

**CLL Propositions as Sessions**

\[ A, B ::= A \otimes B \quad \text{output } A \text{ then behave as } B \]
\[ A \curlyvee B \quad \text{Input } A \text{ then behave as } B \]
\[ A \oplus B \quad \text{Select from } A \text{ or } B \]
\[ A \& B \quad \text{Offer choice of } A \text{ or } B \]
Session Types and Linear Logic
Meanings of Propositions

CLL Propositions as Sessions

\[ A, B ::= A \otimes B \quad \text{output } A \text{ then behave as } B \]
\[ A \bowtie B \quad \text{Input } A \text{ then behave as } B \]
\[ A \oplus B \quad \text{Select from } A \text{ or } B \]
\[ A \& B \quad \text{Offer choice of } A \text{ or } B \]
\[ !A \quad \text{Server Accept} \]
\[ ?A \quad \text{Client Request} \]
## CLL Propositions as Sessions

\[
A, B ::= A \otimes B \quad \text{output } A \text{ then behave as } B \\
A \triangleright B \quad \text{Input } A \text{ then behave as } B \\
A \oplus B \quad \text{Select from } A \text{ or } B \\
A \& B \quad \text{Offer choice of } A \text{ or } B \\
!A \quad \text{Server Accept} \\
?A \quad \text{Client Request} \\
1 \quad \text{Unit for } \otimes \\
\ldots
\]
Session Types and Linear Logic
Meanings of Propositions

CLL Propositions as Sessions

\[ A, B ::= A \otimes B \quad \text{output } A \text{ then behave as } B \]
\[ A \ltimes B \quad \text{Input } A \text{ then behave as } B \]
\[ A \oplus B \quad \text{Select from } A \text{ or } B \]
\[ A \& B \quad \text{Offer choice of } A \text{ or } B \]
\[ !A \quad \text{Server Accept} \]
\[ ?A \quad \text{Client Request} \]
\[ 1 \quad \text{Unit for } \otimes \]
\[ \ldots \]

Cut as Composition

\[
\frac{P \vdash \Delta, x:A \quad Q \vdash \Delta', x:A^\perp}{\nu x.(P \mid Q) \vdash \Delta, \Delta'}
\]
Session Types and Linear Logic
Meanings of Propositions

**CLL Propositions as Sessions**

- $A, B ::= A \& B$  \quad \text{output } A \text{ then behave as } B
- $A \Rightarrow B$  \quad \text{Input } A \text{ then behave as } B
- $A \oplus B$  \quad \text{Select from } A \text{ or } B
- $A \& B$  \quad \text{Offer choice of } A \text{ or } B
- $!A$  \quad \text{Server Accept}
- $?A$  \quad \text{Client Request}
- $1$  \quad \text{Unit for } \&
- \ldots

**Identity as Forwarding**

\[
(id) \quad \frac{x \leftrightarrow y \vdash x : A, y : A^\perp}{x \leftrightarrow y \vdash x : A, y : A^\perp}
\]
Input and Output:

\[
\begin{align*}
\& \quad (\otimes) \quad P \vdash \Delta_1, y: A \quad Q \vdash \Delta_2, x: B \\
\longmapsto \quad x[y].(P | Q) \vdash \Delta_1, \Delta_2, x: A \otimes B 
\end{align*}
\]
Input and Output:

\[
\begin{align*}
(P \triangleright \Delta_1, y : A) &\quad (\otimes) \quad \frac{Q \triangleright \Delta_2, x : B}{x[y].(P \parallel Q) \triangleright \Delta_1, \Delta_2, x : A \otimes B} \\
(P \triangleright \Delta_3, y : A, x : B) &\quad (\triangleright) \quad \frac{Q \triangleright \Delta_2, x : A \otimes B}{x(y).P \triangleright \Delta_3, x : A \triangleright B}
\end{align*}
\]
### Session Types and Linear Logic

#### Reductions

**Input and Output:**

\[
\begin{align*}
(\otimes) & \quad P \vdash \Delta_1, y : A \quad Q \vdash \Delta_2, x : B \\
& \quad x[y].(P \mid Q) \vdash \Delta_1, \Delta_2, x : A \otimes B
\end{align*}
\]

\[
(\otimes) & \quad P \vdash \Delta_3, y : A, x : B \\
& \quad x(y).P \vdash \Delta_3, x : A \otimes B
\]

**Communication as principal cut reductions:**

\[
\begin{align*}
\frac{P \vdash \Delta_1, y : A \quad Q \vdash \Delta_2, x : B}{x[y].(P \mid Q) \vdash \Delta_1, \Delta_2, x : A \otimes B}
\end{align*}
\]

\[
\frac{R \vdash y : A \perp, x : B \perp}{x(y).R \vdash \Delta_3, x : A \perp \otimes B \perp}
\]

\[
\nu x. (x[y].(P \mid Q) \mid x(y).R) \vdash \Delta_1, \Delta_2, \Delta_3
\]
Session Types and Linear Logic

Reductions

Input and Output:

\[
\begin{align*}
(P \otimes Q) & \vdash \Delta_1, \Delta_2, x:A \otimes B \\
\nu_x.(x[y].(P \mid Q) \mid x(y).R) & \vdash \Delta_1, \Delta_2, \Delta_3
\end{align*}
\]

Communication as principal cut reductions:

\[
\begin{align*}
(P \mid Q) & \vdash \Delta_1, y:A, x:B \\
\nu_x.(Q \mid R) & \vdash \Delta_2, \Delta_3, y:A \perp
\end{align*}
\]
What about the other proof conversions?

\[
\begin{align*}
\otimes &\quad (\otimes)
\quad P \vdash \Delta_1, y:A, z:C &\quad Q \vdash \Delta_2, x:B \\
&\quad x[y].(P \mid Q) \vdash \Delta_1, \Delta_2, x:A \otimes B, z:C &\quad R \vdash \Delta_3, z:C^\perp \\
&\quad \nu z.(x[y].(P \mid Q) \mid R) \vdash \Delta_1, x:A \otimes B, \Delta_2, \Delta_3
\end{align*}
\]
What about the other proof conversions?

$$\begin{align*}
(\otimes) & \\
& P \vdash \Delta_1, y:A, z:C \quad Q \vdash \Delta_2, x:B \\
& x[y].(P \mid Q) \vdash \Delta_1, \Delta_2, x:A \otimes B, z:C \\
& R \vdash \Delta_3, z:C^\perp \\
& \nu z.(x[y].(P \mid Q) \mid R) \vdash \Delta_1, x:A \otimes B, \Delta_2, \Delta_3 \\
& (cut) \quad P \vdash \Delta_1, y:A, z:C \quad R \vdash \Delta_3, z:C^\perp \\
& \quad \nu z.(P \mid R) \vdash \Delta_1, \Delta_3, y:A \\
& \quad Q \vdash \Delta_2, x:B \\
& \quad \implies (\otimes) \\
& \quad x[y].(\nu z.(P \mid R) \mid Q) \vdash \Delta_1, x:A \otimes B, \Delta_2, \Delta_3
\end{align*}$$
Session Types and Linear Logic

What about the other proof conversions?

\[ (\otimes) \]
\[
\frac{
\begin{align*}
P \vdash \Delta_1, y : A, z : C \\
Q \vdash \Delta_2, x : B \\
R \vdash \Delta_3, z : C' \n\end{align*}
}{
\begin{align*}
x[y].(P \mid Q) \vdash \Delta_1, \Delta_2, x : A \otimes B, z : C \\
\nu z.(x[y].(P \mid Q) \mid R) \vdash \Delta_1, x : A \otimes B, \Delta_2, \Delta_3 \n\end{align*}
}\]

\[ \text{cut} \]
\[
\frac{
\begin{align*}
P \vdash \Delta_1, y : A, z : C \\
R \vdash \Delta_3, z : C' \n\end{align*}
}{
\begin{align*}
\nu z.(P \mid R) \vdash \Delta_1, \Delta_3, y : A \\
x[y].(\nu z.(P \mid R) \mid Q) \vdash \Delta_1, x : A \otimes B, \Delta_2, \Delta_3 \n\end{align*}
}\]

\[ \text{cut} \]
\[ (\otimes) \]
\[
\frac{
\begin{align*}
\nu z.(x[y].(P \mid Q) \mid R) \Rightarrow x[y].(\nu z.(P \mid R) \mid Q) \n\end{align*}
}{
\text{if } z \in \text{fn}(P) \}
Session Types and Linear Logic
Putting it all together

CP Reduction

- One CP reduction for every principal cut reduction (one per dual prop. pair).
- One CP reduction for each commutting conversion (2 for $\otimes$, 2 for $\oslash$, none for 0, one for the rest).
**Session Types and Linear Logic**

**Putting it all together**

**CP Reduction**
- One CP reduction for every principal cut reduction (one per dual prop. pair).
- One CP reduction for each commutting conversion (2 for $\otimes$, 2 for $\oplus$, none for 0, one for the rest).

**Metatheorems**
- If $P \vdash \Delta$ and $P \Rightarrow Q$ then $Q \vdash \Delta$.
- If $P \vdash \Delta$ there exists $Q$ such that $P \Rightarrow^* Q$ and $Q$ is not a cut.
Session Types and Linear Logic

Why does it matter?

- Safety/liveness properties “for free”.
- A solid foundation to build on:
  - Encodings of $\lambda$-calculi into $\pi$-calculus / CP [Toninho et al.12, Wadler12, LindleyMorris15].
  - Value-dependent / refinement session types [Toninho11].
  - Multiparty sessions [Carbone et al.,15].
  - Dynamic monitoring [Jia et al.16]
  - “Better” designed languages [Wadler12, Toninho et al13]
  - ...
Session Types and Linear Logic

Why does it *still* matter?

Loads still to do (fortunately)!

- Curry-Howard iso. gave us Haskell, ML, …
- Need a “real” language that puts this all together!
Session Types and Linear Logic

Why does it still matter?

Loads still to do (fortunately)!

- Curry-Howard iso. gave us Haskell, ML, ... 
- Need a “real” language that puts this all together!
- A better approach to logic and multiparty sessions.
Session Types and Linear Logic
Why does it still matter?

Loads still to do (fortunately)!

- Curry-Howard iso. gave us Haskell, ML, ... 
- Need a “real” language that puts this all together!
- A better approach to logic and multiparty sessions.
- True non-determinism?
- More $\pi$-calculus-like behaviours?
Session Types and Linear Logic
Why does it still matter?

Loads still to do (fortunately):
- Curry-Howard iso. gave us Haskell, ML, . . .
- Need a “real” language that puts this all together!
- A better approach to logic and multiparty sessions.
- True non-determinism?
- More $\pi$-calculus-like behaviours?
- Dependent types?
- etc...
The connections of linear logic and session types:
- Linear propositions as session types.
- Proofs as processes.
- Communication and proof conversion.
Session Types and Linear Logic

Conclusion

- The connections of linear logic and session types:
  - Linear propositions as session types.
  - Proofs as processes.
  - Communication and proof conversion.
- Logic gives us comm. safety / liveness for free.
Conclusion

The connections of linear logic and session types:
- Linear propositions as session types.
- Proofs as processes.
- Communication and proof conversion.

Logic gives us comm. safety / liveness for free.

... but also a general and powerful framework to reason about concurrency!
Session Types and Linear Logic

Conclusion

- The connections of linear logic and session types:
  - Linear propositions as session types.
  - Proofs as processes.
  - Communication and proof conversion.
- Logic gives us comm. safety / liveness for free.
- ...but also a general and powerful framework to reason about concurrency!
- Only scratched the surface!
HAPPY BIRTHDAY

PHIL

from MRG

DO NN RH JL BT AS AA RN WT