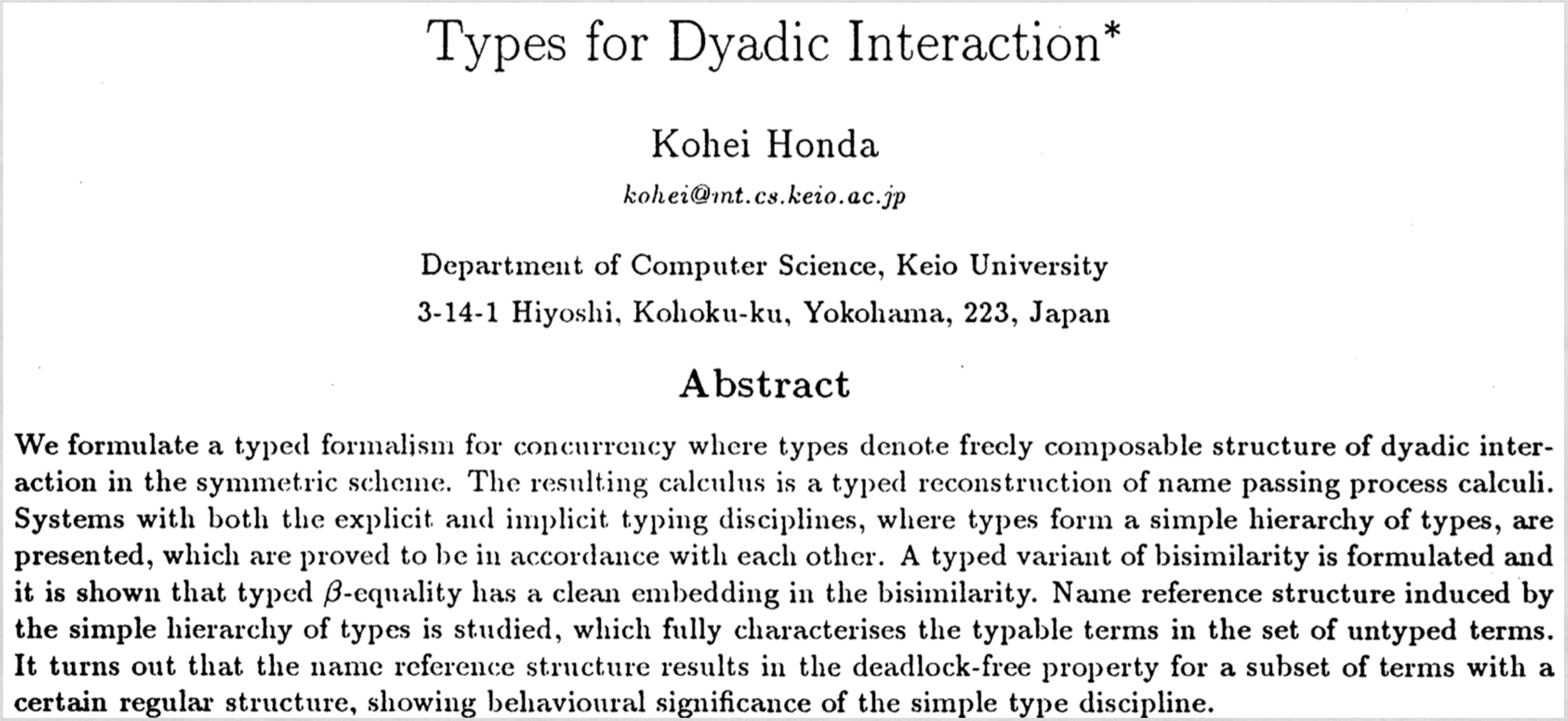
LOGICAL FOUNDATIONS OF SESSION-BASED INTERACTION

BETTY MEETING 2013 Rome Italy

CONCUR 1993



Types for Dyadic Interaction*

Kohei Honda

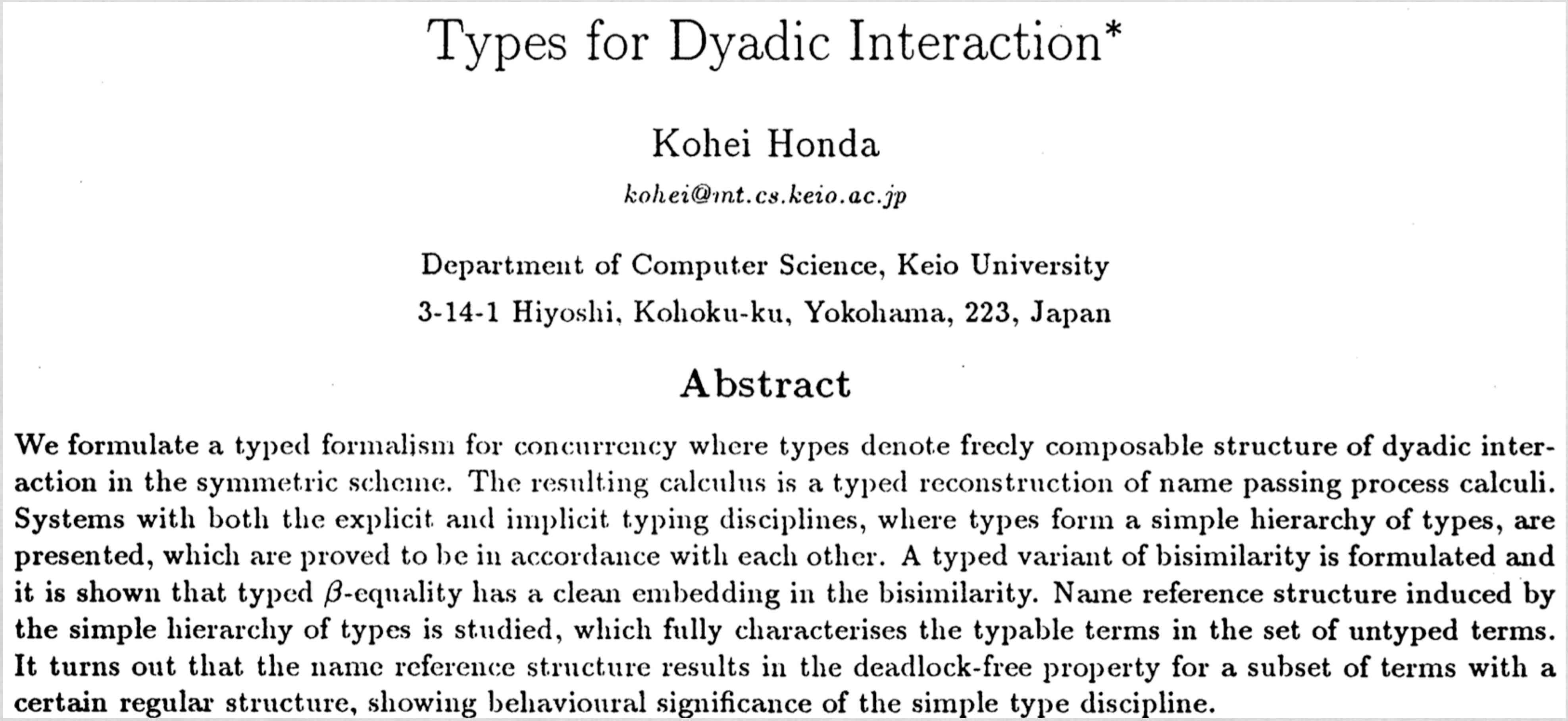
kohei@mt.cs.keio.ac.jp

Department of Computer Science, Keio University
3-14-1 Hiyoshi, Kohoku-ku, Yokohama, 223, Japan

Abstract

We formulate a typed formalism for concurrency where types denote freely composable structure of dyadic interaction in the symmetric scheme. The resulting calculus is a typed reconstruction of name passing process calculi. Systems with both the explicit and implicit typing disciplines, where types form a simple hierarchy of types, are presented, which are proved to be in accordance with each other. A typed variant of bisimilarity is formulated and it is shown that typed β-equality has a clean embedding in the bisimilarity. Name reference structure induced by the simple hierarchy of types is studied, which fully characterises the typable terms in the set of untyped terms. It turns out that the name reference structure results in the deadlock-free property for a subset of terms with a certain regular structure, showing behavioural significance of the simple type discipline.

CONCUR 1993



Types for Dyadic Interaction*

Kohei Honda

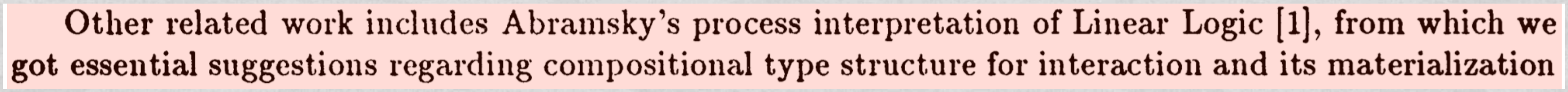
kohei@mt.cs.keio.ac.jp

Department of Computer Science, Keio University

3-14-1 Hiyoshi, Kohoku-ku, Yokohama, 223, Japan

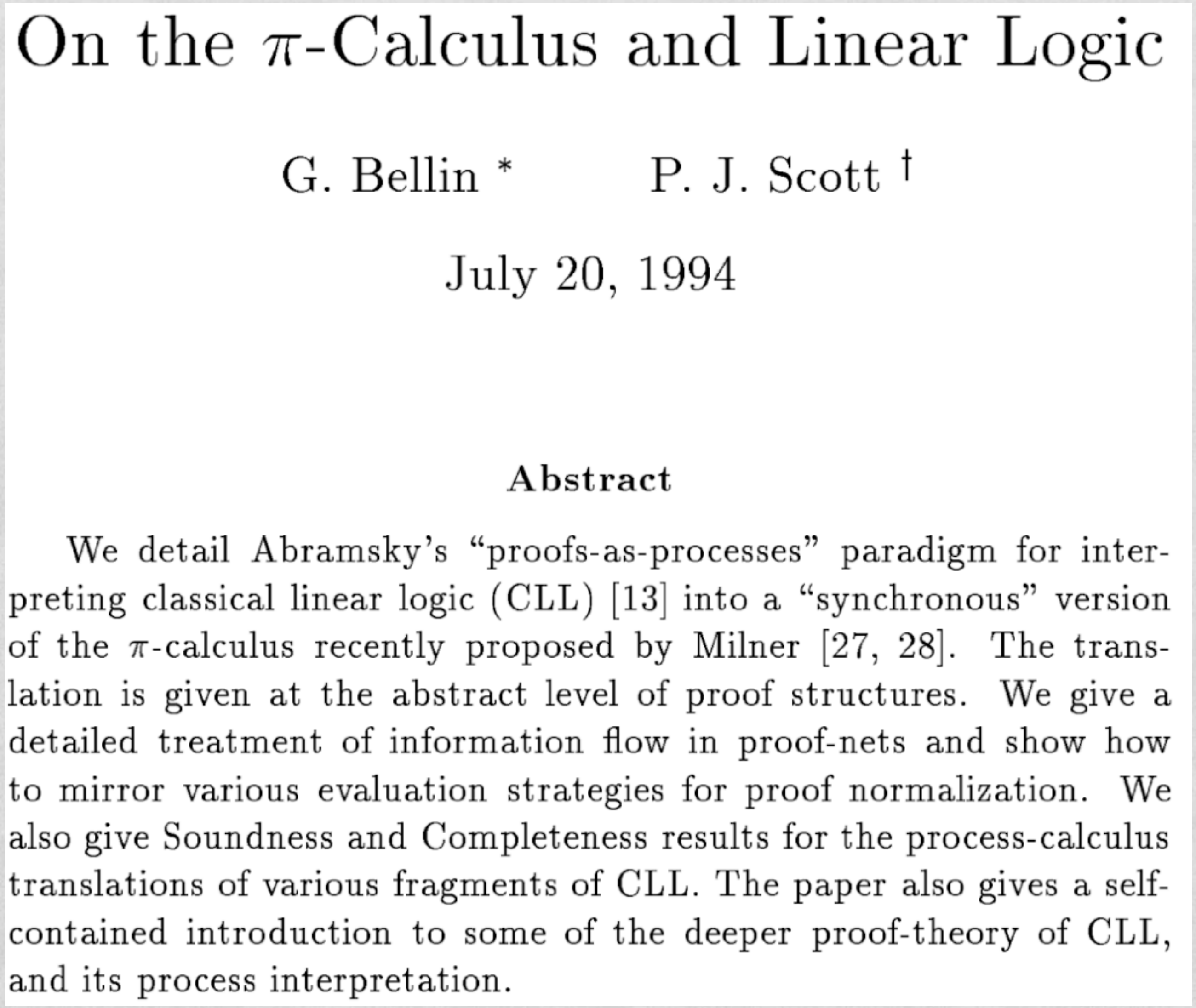
Abstract

We formulate a typed formalism for concurrency where types denote freely composable structure of dyadic interaction in the symmetric scheme. The resulting calculus is a typed reconstruction of name passing process calculi. Systems with both the explicit and implicit typing disciplines, where types form a simple hierarchy of types, are presented, which are proved to be in accordance with each other. A typed variant of bisimilarity is formulated and it is shown that typed β-equality has a clean embedding in the bisimilarity. Name reference structure induced by the simple hierarchy of types is studied, which fully characterises the typable terms in the set of untyped terms. It turns out that the name reference structure results in the deadlock-free property for a subset of terms with a certain regular structure, showing behavioural significance of the simple type discipline.



Other related work includes Abramsky's process interpretation of Linear Logic [1], from which we got essential suggestions regarding compositional type structure for interaction and its materialization

TCS 1994



On the π-Calculus and Linear Logic

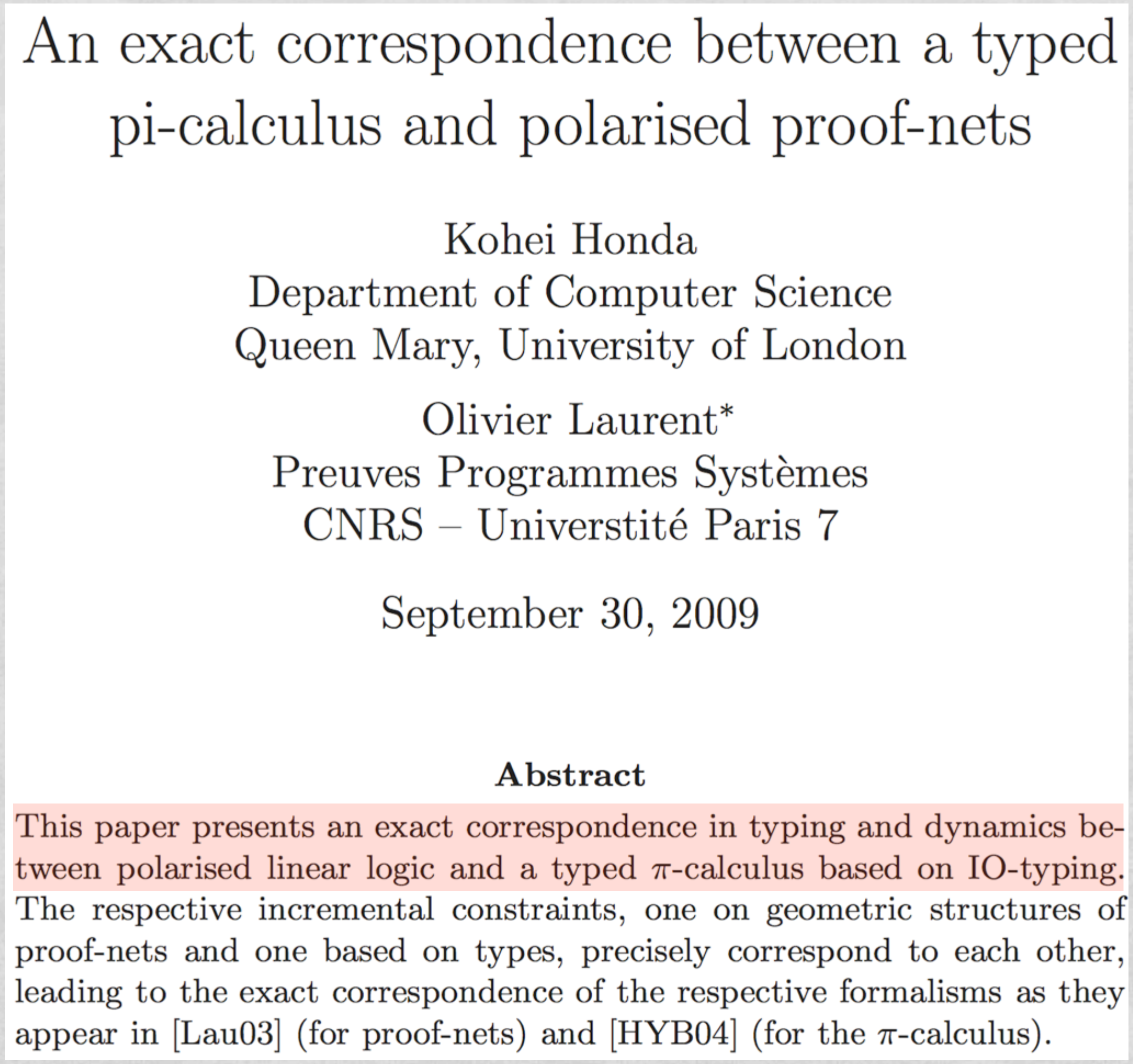
G. Bellin [*] P. J. Scott [†]

July 20, 1994

Abstract

We detail Abramsky's "proofs-as-processes" paradigm for interpreting classical linear logic (CLL) [13] into a "synchronous" version of the π-calculus recently proposed by Milner [27, 28]. The translation is given at the abstract level of proof structures. We give a detailed treatment of information flow in proof-nets and show how to mirror various evaluation strategies for proof normalization. We also give Soundness and Completeness results for the process-calculus translations of various fragments of CLL. The paper also gives a self-contained introduction to some of the deeper proof-theory of CLL, and its process interpretation.

TCS 2009



An exact correspondence between a typed pi-calculus and polarised proof-nets

Kohei Honda
Department of Computer Science
Queen Mary, University of London

Olivier Laurent*
Preuves Programmes Systèmes
CNRS – Universtité Paris 7

September 30, 2009

Abstract

This paper presents an exact correspondence in typing and dynamics between polarised linear logic and a typed π-calculus based on IO-typing. The respective incremental constraints, one on geometric structures of proof-nets and one based on types, precisely correspond to each other, leading to the exact correspondence of the respective formalisms as they appear in [Lau03] (for proof-nets) and [HYB04] (for the π-calculus).

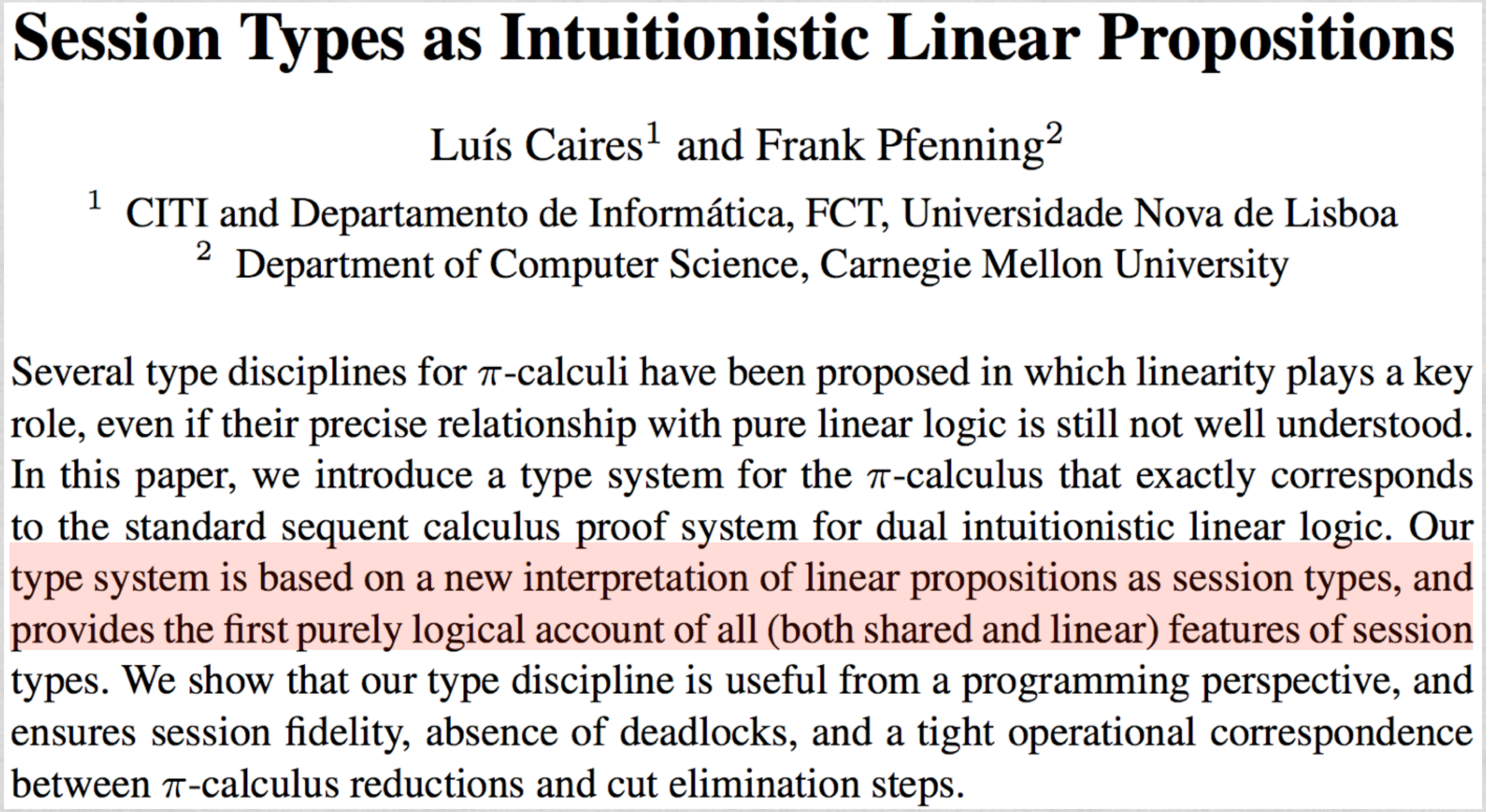
CURRY-HOWARD FOUNDATIONS FOR SESSIONS

- functional-typed sequential programming
  - intuitionistic logic
  - the λ calculus

CURRY-HOWARD FOUNDATIONS FOR SESSIONS

- session-typed concurrent programming
  - linear logic
  - the π calculus

CONCUR 2010



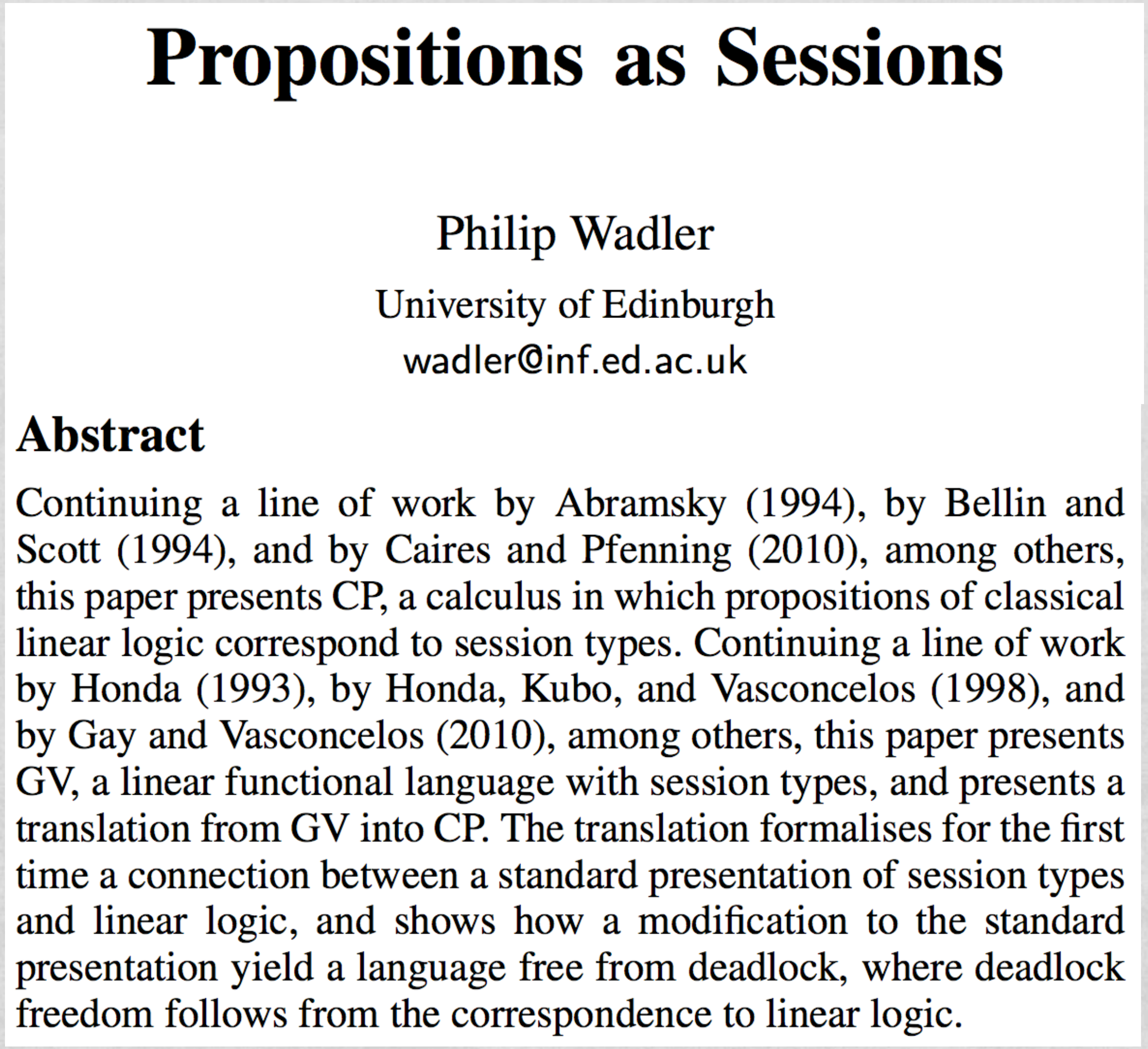
Session Types as Intuitionistic Linear Propositions

Luís Caires[1] and Frank Pfenning[2]

[1] CITI and Departamento de Informática, FCT, Universidade Nova de Lisboa
[2] Department of Computer Science, Carnegie Mellon University

Several type disciplines for π-calculi have been proposed in which linearity plays a key role, even if their precise relationship with pure linear logic is still not well understood. In this paper, we introduce a type system for the π-calculus that exactly corresponds to the standard sequent calculus proof system for dual intuitionistic linear logic. Our type system is based on a new interpretation of linear propositions as session types, and provides the first purely logical account of all (both shared and linear) features of session types. We show that our type discipline is useful from a programming perspective, and ensures session fidelity, absence of deadlocks, and a tight operational correspondence between π-calculus reductions and cut elimination steps.

ICFP 2013



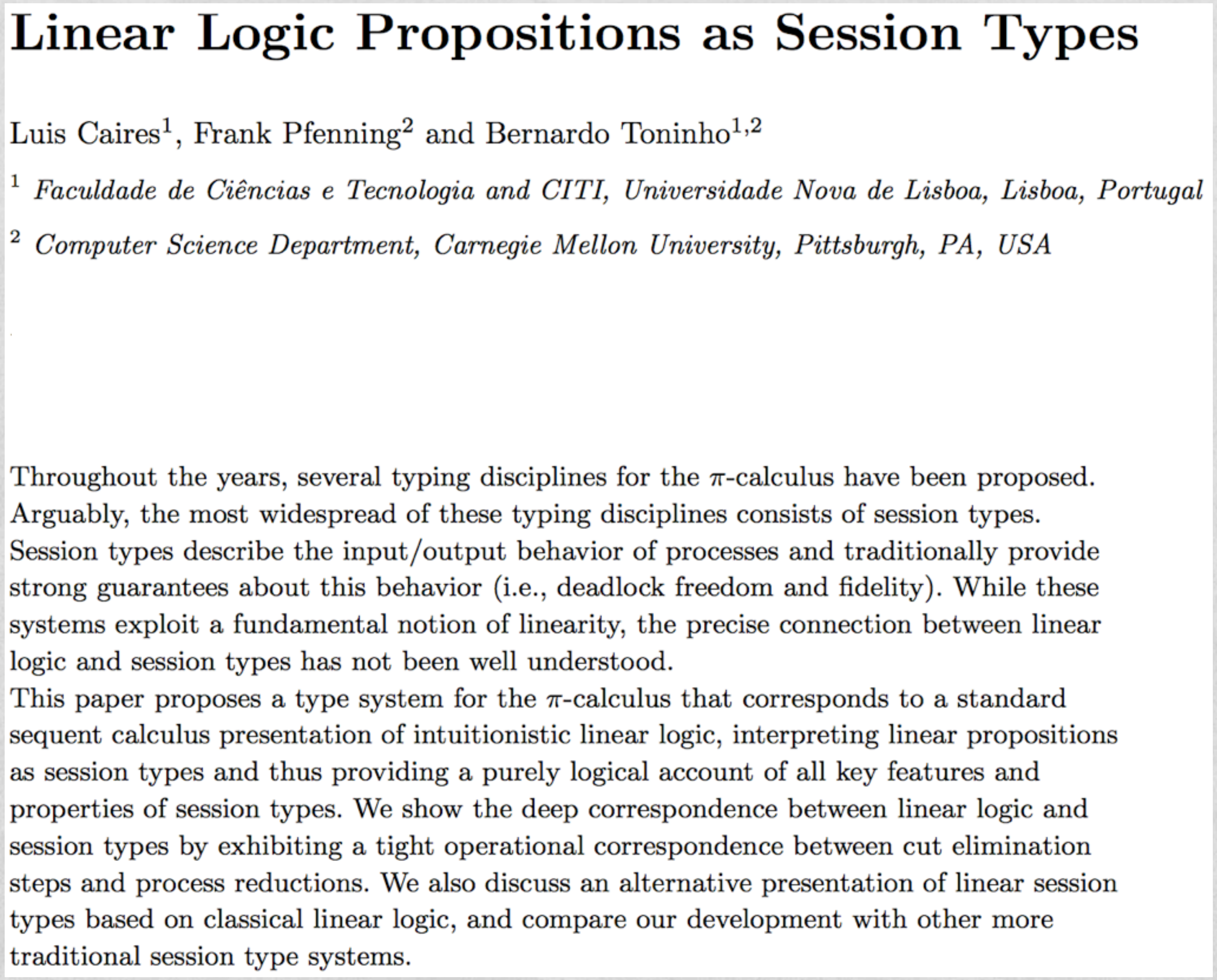
Propositions as Sessions

Philip Wadler
University of Edinburgh
wadler@inf.ed.ac.uk

Abstract

Continuing a line of work by Abramsky (1994), by Bellin and Scott (1994), and by Caires and Pfenning (2010), among others, this paper presents CP, a calculus in which propositions of classical linear logic correspond to session types. Continuing a line of work by Honda (1993), by Honda, Kubo, and Vasconcelos (1998), and by Gay and Vasconcelos (2010), among others, this paper presents GV, a linear functional language with session types, and presents a translation from GV into CP. The translation formalises for the first time a connection between a standard presentation of session types and linear logic, and shows how a modification to the standard presentation yield a language free from deadlock, where deadlock freedom follows from the correspondence to linear logic.

MSCS (TO APPEAR)



Linear Logic Propositions as Session Types

Luis Caires[1], Frank Pfenning[2] and Bernardo Toninho[1,2]

[1] *Faculdade de Ciências e Tecnologia and CITI, Universidade Nova de Lisboa, Lisboa, Portugal*

[2] *Computer Science Department, Carnegie Mellon University, Pittsburgh, PA, USA*

Throughout the years, several typing disciplines for the π-calculus have been proposed. Arguably, the most widespread of these typing disciplines consists of session types. Session types describe the input/output behavior of processes and traditionally provide strong guarantees about this behavior (i.e., deadlock freedom and fidelity). While these systems exploit a fundamental notion of linearity, the precise connection between linear logic and session types has not been well understood.
This paper proposes a type system for the π-calculus that corresponds to a standard sequent calculus presentation of intuitionistic linear logic, interpreting linear propositions as session types and thus providing a purely logical account of all key features and properties of session types. We show the deep correspondence between linear logic and session types by exhibiting a tight operational correspondence between cut elimination steps and process reductions. We also discuss an alternative presentation of linear session types based on classical linear logic, and compare our development with other more traditional session type systems.

CURRY-HOWARD FOUNDATIONS FOR SESSIONS

- CH provides a canonical design space to session types
  - streamlines several ad-hoc features of other systems
  - naturally amenable to extensions and generalizations
  - “applied” languages will diverge from the pure model
  - but, the foundational value of the interpretation remains

SESSION TYPES

$S,T ::=$	$\mathbf{1}$	stop		
$\mid$	$T \otimes S$	output	$T!.S$	
$\mid$	$T \multimap S$	input	$T?.S$	$(T^{\perp} ⅋ S)$
$\mid$	$S \& T$	branch		
$\mid$	$S \oplus T$	choice		
$\mid$	$!S$	“shared” channel type (server side)		
$\mid$	$?S$	“shared” channel type (client side)		

TYPING JUDGEMENT

Intuitionistic

$$n_1{:}S_1, \ldots, n_k{:}S_k \vdash P :: m{:}S$$

Process P safely provides session S at (name) m, whenever composed with any system providing session S_j at each n_j

No ?S type. Enforces locality on shared channels.

Immediately compatible with dependent type theory.

Classical

$$\vdash P :: n_1{:}S_1, \ldots, n_k{:}S_k$$

Process P safely provides session S_j at each n_j

Full duality ($S^\perp$)

SAMPLE TYPE SYSTEM (CLL)

$$\frac{}{\mathbf{0} \vdash x{:}\mathbf{1};\Theta}\ (\mathrm{T}\mathbf{1}) \qquad \frac{P \vdash \Delta;\Theta}{P \vdash x{:}\bot,\Delta;\Theta}\ (\mathrm{T}\bot)$$

$$\frac{P \vdash \Delta, y{:}A, x{:}B;\Theta}{x(y).P \vdash \Delta, x{:}A?.B;\Theta}\ (\mathrm{T}?) \qquad \frac{P \vdash \Delta, y{:}A;\Theta \quad Q \vdash \Delta', x{:}B;\Theta}{(\nu y)x\langle y\rangle.(P \mid Q) \vdash \Delta,\Delta', x{:}A!.B;\Theta}\ (\mathrm{T}!)$$

$$\frac{P \vdash \Delta, x{:}A_i;\Theta}{x.\mathtt{l}_i; P \vdash \Delta, x{:} \oplus_j A_j;\Theta}\ (\mathrm{T}\oplus) \qquad \frac{P_1 \vdash \Delta, x{:}A_1;\Theta \ \cdots \ P_j \vdash \Delta, x{:}A_j;\Theta}{x.\mathtt{case}(l_j : P_j) \vdash \Delta, x{:} \&_j A_j;\Theta}\ (\mathrm{T}\&)$$

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

$$\frac{P \vdash \Delta, y{:}A; u{:}A,\Theta}{(\nu y)u\langle y\rangle.P \vdash \Delta; u{:}A,\Theta}\ (\mathrm{Tcopy})$$

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

HIGHLIGHTS

* Logical interpretation ensures global progress, compositionally

* Logical foundation promotes extensions and generalizations:

  - polymorphism (*e.g.*, behavioral genericity, parametricity)

  - dependent types (*e.g.*, contracts, certificates, signatures)

  - modalities (*e.g.*, authorization, classification)

  - higher-order (*e.g.*, functional encodings, integration)

  - logical relations (*e.g.*, termination, behavioral equivalences)

* Challenges:

  - describing interactions between several parties ?

  - accommodating non-determinism ? concurrency ? state?

  - expressing security properties ? information flow?

SOME REFERENCES

Luis Caires, Frank Pfenning, Bernardo Toninho: Linear Logic Propositions as Session Types. MSCS, to appear.

Luís Caires, Jorge A. Pérez, Frank Pfenning, Bernardo Toninho: Behavioral Polymorphism and Parametricity in Session-Based Communication. ESOP 2013: 330-349

Bernardo Toninho, Luís Caires, Frank Pfenning: Higher-Order Processes, Functions, and Sessions: A Monadic Integration. ESOP 2013: 350-369

Philip Wadler: Propositions as Sessions. ICFP 2012: 273-286

Jorge A. Pérez, Luís Caires, Frank Pfenning, Bernardo Toninho: Linear Logical Relations for Session-Based Concurrency. ESOP 2012: 539-558

Bernardo Toninho, Luís Caires, Frank Pfenning: Functions as Session-Typed Processes. FoSSaCS 2012: 346-360

Frank Pfenning, Luís Caires, Bernardo Toninho: Proof-Carrying Code in a Session-Typed Process Calculus. CPP 2011: 21-36

Bernardo Toninho, Luís Caires, Frank Pfenning: Dependent session types via intuitionistic linear type theory. PPDP 2011: 161-172

Luís Caires, Frank Pfenning: Session Types as Intuitionistic Linear Propositions. CONCUR 2010: 222-236