

# Duality of Session Types: The Final Cut

Simon J. Gay

**Peter Thiemann**

Vasco T. Vasconcelos

University of Glasgow

University of Freiburg

University of Lisbon

VEST  
online, June 2020

## Session Types — Types for Structured Communication

$S$	$::=$	$!T.S'$	send
		$?T.S'$	receive

## Session Types — Types for Structured Communication

$S$	$::=$	$!T.S'$	send
		$?T.S'$	receive
		$\oplus\{\ell_i : S_i\}$	select
		$\&\{\ell_i : S_i\}$	choice

## Session Types — Types for Structured Communication

$S$	$::=$	$!T.S'$	send
		$?T.S'$	receive
		$\oplus\{\ell_i : S_i\}$	select
		$\&\{\ell_i : S_i\}$	choice
		end	

# The good old math server

## Session type of the server

```
type Server = &{  
  Neg: ?Int. !Int. end ,  
  Add: ?Int. ?Int. !Int. end}
```

# The good old math server

## Session type of the server

```
type Server = &{  
  Neg: ?Int. !Int. end ,  
  Add: ?Int. ?Int. !Int. end}
```

## Session type of the client

```
type Client =  $\oplus$ {  
  Neg: !Int. ?Int. end ,  
  Add: !Int. !Int. ?Int. end}
```

# The good old math server

## Session type of the server

```
type Server = &{  
  Neg: ?Int. !Int. end ,  
  Add: ?Int. ?Int. !Int. end }
```

## Session type of the client

```
type Client =  $\oplus$ {  
  Neg: !Int. ?Int. end ,  
  Add: !Int. !Int. ?Int. end }
```

Client type is the dual of server type

# Duality

## Definition 1

$$\overline{\text{end}} = \text{end}$$

$$\overline{!T.S} = ?T.\overline{S}$$

$$\overline{?T.S} = !T.\overline{S}$$

$$\overline{\oplus\{\ell_i : S_i\}} = \&\{\ell_i : \overline{S_i}\}$$

$$\overline{\&\{\ell_i : S_i\}} = \oplus\{\ell_i : \overline{S_i}\}$$



# Duality

## Definition 1

$$\begin{array}{lll} \overline{\text{end}} = \text{end} & \overline{!T.S} = ?T.\overline{S} & \overline{\oplus\{\ell_i : S_i\}} = \&\{\ell_i : \overline{S_i}\} \\ \overline{?T.S} = !T.\overline{S} & & \overline{\&\{\ell_i : S_i\}} = \oplus\{\ell_i : \overline{S_i}\} \end{array}$$

## Undebatably correct!

- ▶ Kohei Honda (CONCUR 1993): Types for Dyadic Interaction.
- ▶ Kaku Takeuchi, Kohei Honda, Makoto Kubo (PARLE1994): An Interaction-based Language and its Typing System.
- ▶ Kohei Honda, Vasco Thudichum Vasconcelos, Makoto Kubo (ESOP 1998): Language Primitives and Type Discipline for Structured Communication-Based Programming.

## Adding Recursion

$S ::= \dots$   
 $\mu X.S$   
 $X$

recursive session

type variable

# A more interesting math server

## Session type of the server

```
type Server =  $\mu$  X. &{  
  Neg: ?Int. !Int. X,  
  Add: ?Int. ?Int. !Int. X,  
  Quit: end}
```

# A more interesting math server

## Session type of the server

```
type Server =  $\mu$  X. &{  
  Neg: ?Int. !Int. X,  
  Add: ?Int. ?Int. !Int. X,  
  Quit: end}
```

## Session type of the client

```
type Client =  $\mu$  X.  $\oplus$ {  
  Neg: !Int. ?Int. X,  
  Add: !Int. !Int. ?Int. X,  
  Quit: end}
```

# Naive Duality

Definition (extends Definition 1)

$$\overline{X} = X$$

$$\overline{\mu X.S} = \mu X.\overline{S}$$

## Drawback: Naive Duality is not always correct

Consider

$$S = \mu X. !X.X \quad = \quad !(\mu X. !X.X).(\mu X. !X.X)$$

## Drawback: Naive Duality is not always correct

Consider

$$S = \mu X. !X.X \quad = !(\mu X. !X.X).(\mu X. !X.X) \quad = !S.S$$

Unfolding shows that this server wants to send a channel of type  $S$ .

## Drawback: Naive Duality is not always correct

Consider

$$S = \mu X. !X.X \quad = !(\mu X. !X.X).(\mu X. !X.X) \quad = !S.S$$

Unfolding shows that this server wants to send a channel of type  $S$ .  
But its naive dual is

$$\bar{S} = \overline{\mu X. !X.X} \quad = \mu X. \overline{!X.X} \quad = \mu X. ?X.X \quad = ?\bar{S}.\bar{S}$$

so the client wrongly expects to receive a channel of type  $\bar{S} \neq S!$



## Goal

Find a satisfactory definition of duality for recursive session types in  $\mu$  notation.

# Outline

A Coinductive Definition

Bernardi and Hennessy

Lindley and Morris

Mechanization

## Recursive Session Types, Coinductively

A recursive type is a potentially infinite tree, labeled by the type constructors.  
The sets  $\text{Type}$  of type trees and  $\text{SType}$  of session type trees are given by the greatest fixpoint of

$$F(\mathcal{S}, \mathcal{T}) = (\{\text{end}\} \cup \{?T.S, !T.S \mid T \in \mathcal{T}, S \in \mathcal{S}\}) \\ \times (\{\text{int}\} \cup \mathcal{S})$$

Duality is a binary relation on  $\text{SType}$  defined as the greatest fixpoint of

$$F(\mathcal{D}) = \{(\text{end}, \text{end})\} \\ \cup \{(?T.S_1, !T.S_2) \mid T \in \text{Type}, (S_1, S_2) \in \mathcal{D}\} \\ \cup \{(!T.S_1, ?T.S_2) \mid T \in \text{Type}, (S_1, S_2) \in \mathcal{D}\}$$

This gets more involved with the  $\mu$  . notation...

## Example

$$\begin{aligned}\mu X. ?T.X &\approx \mu X. ?T.?T.X \\ \mu X. ?T.X &\perp \mu X. !T.!T.X\end{aligned}$$

# Ground Truth

A coinductive definition

## Definition (Syntactic Duality of Session Types)

If  $\mathcal{D}$  is a relation on  $\text{SType}$  then  $F_{\perp}(\mathcal{D})$  is the relation on  $\text{SType}$  defined by:

$$\begin{aligned} F_{\perp}(\mathcal{D}) = & \{(\text{end}, \text{end})\} \\ & \cup \{(?T_1.S_1, !T_2.S_2) \mid T_1 \approx T_2 \text{ and } (S_1, S_2) \in \mathcal{D}\} \\ & \cup \{(!T_1.S_1, ?T_2.S_2) \mid T_1 \approx T_2 \text{ and } (S_1, S_2) \in \mathcal{D}\} \\ & \cup \{(S_1, \mu X.S_2) \mid (S_1, S_2[\mu X.S_2/X]) \in \mathcal{D}\} \\ & \cup \{(\mu X.S_1, S_2) \mid (S_1[\mu X.S_1/X], S_2) \in \mathcal{D} \text{ and } S_2 \neq \mu Y.S_3\} \end{aligned}$$

A relation  $\mathcal{D}$  on  $\text{SType}$  is a *session duality* if  $\mathcal{D} \subseteq F_{\perp}(\mathcal{D})$ .

Duality of session types,  $\cdot \perp \cdot$ , is the largest session duality.

# Outline

A Coinductive Definition

Bernardi and Hennessy

Lindley and Morris

Mechanization

## Bernardi and Hennessey's Discovery (CONCUR 2014)

- ▶ Given  $S = \mu X. ?X.X$

## Bernardi and Hennessey's Discovery (CONCUR 2014)

- ▶ Given  $S = \mu X. ?X.X$
- ▶ Then  $\bar{S} = \mu X. !X.X$  is its naive dual (with  $S \not\approx \bar{S}$ )



## Bernardi and Hennessy's Discovery (CONCUR 2014)

- ▶ Given  $S = \mu X. ?X.X$
- ▶ Then  $\bar{S} = \mu X. !X.X$  is its naive dual (with  $S \not\approx \bar{S}$ )
- ▶ But is this correct?

## Bernardi and Hennessey's Discovery (CONCUR 2014)

- ▶ Given  $S = \mu X. ?X.X$
- ▶ Then  $\bar{S} = \mu X. !X.X$  is its naive dual (with  $S \not\approx \bar{S}$ )
- ▶ But is this correct?
- ▶ Let's rewrite  $S$  by unrolling one occurrence of  $X$

## Bernardi and Hennessey's Discovery (CONCUR 2014)

- ▶ Given  $S = \mu X. ?X.X$
- ▶ Then  $\bar{S} = \mu X. !X.X$  is its naive dual (with  $S \not\approx \bar{S}$ )
- ▶ But is this correct?
- ▶ Let's rewrite  $S$  by unrolling one occurrence of  $X$
- ▶  $S' = \mu X. ?S.X = \mu X. ?(\mu X. ?X.X).X$

## Bernardi and Hennessey's Discovery (CONCUR 2014)

- ▶ Given  $S = \mu X. ?X.X$
- ▶ Then  $\bar{S} = \mu X. !X.X$  is its naive dual (with  $S \not\approx \bar{S}$ )
- ▶ But is this correct?
- ▶ Let's rewrite  $S$  by unrolling one occurrence of  $X$
- ▶  $S' = \mu X. ?S.X = \mu X. ?(\mu X. ?X.X).X$
- ▶  $\bar{S}' = \mu X. !S.X$

## Bernardi and Hennessey's Discovery (CONCUR 2014)

- ▶ Given  $S = \mu X. ?X.X$
- ▶ Then  $\bar{S} = \mu X. !X.X$  is its naive dual (with  $S \not\approx \bar{S}$ )
- ▶ But is this correct?
- ▶ Let's rewrite  $S$  by unrolling one occurrence of  $X$
- ▶  $S' = \mu X. ?S.X = \mu X. ?(\mu X. ?X.X).X$
- ▶  $\bar{S}' = \mu X. !S.X$
- ▶ But unrolling yields  $\bar{S} \approx \mu X. !\bar{S}.X$

## Bernardi and Hennessey's Discovery (CONCUR 2014)

- ▶ Given  $S = \mu X. ?X.X$
- ▶ Then  $\bar{S} = \mu X. !X.X$  is its naive dual (with  $S \not\approx \bar{S}$ )
- ▶ But is this correct?
- ▶ Let's rewrite  $S$  by unrolling one occurrence of  $X$
- ▶  $S' = \mu X. ?S.X = \mu X. ?(\mu X. ?X.X).X$
- ▶  $\bar{S}' = \mu X. !S.X$
- ▶ But unrolling yields  $\bar{S} \approx \mu X. !\bar{S}.X$
- ▶ Now  $S \approx S'$  but  $\bar{S} \not\approx \bar{S}'$ !

# Bernardi and Hennessy's Solution

## BH Duality

- ▶ Compute the *message closure* of a session type.
- ▶ Apply naive duality to the result.

# Bernardi and Hennessy's Solution

## BH Duality

- ▶ Compute the *message closure* of a session type.
- ▶ Apply naive duality to the result.

## Definition

A session type is *message-closed* if all message types are closed.



# Bernardi and Hennessy's Solution

## BH Duality

- ▶ Compute the *message closure* of a session type.
- ▶ Apply naive duality to the result.

## Definition

A session type is *message-closed* if all message types are closed.

## For example

- ▶  $S = \mu X. ?X.X$  is *not* message-closed
- ▶  $S' = \mu X. ?S.X$  is message-closed

# Bernardi and Hennessey's Results

- ▶ BH duality is sound wrt  $\cdot \perp \cdot$
- ▶ but the definition of message closure is quite involved and may increase the size of a type substantially

## Definition (Message Closure [BH2014])

For any type  $T$  and substitution  $\sigma$  closing for  $T$ , the type  $\text{mclo}(T, \sigma)$  is defined inductively by the following rules.

$$\text{mclo}(\text{end}, \sigma) = \text{end}$$

$$\text{mclo}(\text{int}, \sigma) = \text{int}$$

$$\text{mclo}(X, \sigma) = X$$

$$\text{mclo}(\text{?}T.S, \sigma) = \text{?}(T\sigma). \text{mclo}(S, \sigma)$$

$$\text{mclo}(\text{!}T.S, \sigma) = \text{!}(T\sigma). \text{mclo}(S, \sigma)$$

$$\text{mclo}(\mu X.S, \sigma) = \mu X. \text{mclo}(S, [(\mu X.S)/X]; \sigma)$$

Define  $\text{mclo}(S)$  as  $\text{mclo}(S, \varepsilon)$ .

# GTV's optimization

- ▶ BH duality can be simplified by symbolic composition of message closure and naive duality (and deforestation)

## Definition (Duality with On-the-fly Message Closure)

For any session type  $S$  and substitution  $\sigma$  closing for  $S$ , the session type  $\text{dualof}(S, \sigma)$  is defined inductively by the following rules.

$$\begin{array}{ll} \text{dualof}(\text{end}, \sigma) = \text{end} & \text{dualof}(!T.S, \sigma) = !(T\sigma). \text{dualof}(S, \sigma) \\ & \text{dualof}(!T.S, \sigma) = ?(T\sigma). \text{dualof}(S, \sigma) \\ \text{dualof}(X, \sigma) = X & \text{dualof}(\mu X.S, \sigma) = \mu X. \text{dualof}(S, [\mu X.S/X]; \sigma) \end{array}$$

Define  $\text{dualof}(S)$  as  $\text{dualof}(S, \varepsilon)$ .

# Outline

A Coinductive Definition

Bernardi and Hennessy

Lindley and Morris

Mechanization

# Lindley and Morris's Approach

- ▶ Lindley and Morris [ICFP 2016] give another definition of duality

# Lindley and Morris's Approach

- ▶ Lindley and Morris [ICFP 2016] give another definition of duality
- ▶ It relies on a technical twist, negative type variables, ...

# Lindley and Morris's Approach

- ▶ Lindley and Morris [ICFP 2016] give another definition of duality
- ▶ It relies on a technical twist, negative type variables, ...
- ▶ Each type variable  $X$  comes with its companion *negative type variable*  $\overline{X}$

# Lindley and Morris's Approach

- ▶ Lindley and Morris [ICFP 2016] give another definition of duality
- ▶ It relies on a technical twist, negative type variables, ...
- ▶ Each type variable  $X$  comes with its companion *negative type variable*  $\overline{X}$
- ▶ A negative variable  $\overline{X}$  behaves like a suspended application of duality, which gets triggered by substitution for  $X$ .



# Lindley and Morris's Solution

Definition (Lindley-Morris Duality, Original Version [ICFP2016])

$$\text{Imd}(\text{end}) = \text{end}$$

$$\text{Imd}(X) = \overline{X}$$

$$\text{Imd}(!T.S) = !T.\text{Imd}(S)$$

$$\text{Imd}(\overline{X}) = X$$

$$\text{Imd}(!T.S) = ?T.\text{Imd}(S)$$

$$\text{Imd}(\mu X.S) = \mu X.(\text{Imd}(S)\{\overline{X}/X\})$$

# Lindley and Morris's Solution

Definition (Lindley-Morris Duality, Original Version [ICFP2016])

$$\begin{array}{ll} \text{lmd}(\text{end}) = \text{end} & \text{lmd}(X) = \overline{X} \\ \text{lmd}(!T.S) = !T.\text{lmd}(S) & \text{lmd}(\overline{X}) = X \\ \text{lmd}(!T.S) = ?T.\text{lmd}(S) & \text{lmd}(\mu X.S) = \mu X.(\text{lmd}(S)\{\overline{X}/X\}) \end{array}$$

## Caveat

- ▶ The operation  $T\{\overline{X}/X\}$  is *not* standard substitution.
- ▶ It rather swaps  $X$  and  $\overline{X}$ .

## Example

$$\begin{aligned} \text{lmd}(\mu X. ?X.X) &= \mu X. \text{lmd} (?X.X)\{\overline{X}/X\} \\ &= \mu X. (!X.\overline{X})\{\overline{X}/X\} \\ &= \mu X. (!\overline{X}.X) \end{aligned}$$

## GTV's Results on LM Duality

- ▶ Unfortunately, Lindley and Morris just state this definition without proof.

## GTV's Results on LM Duality

- ▶ Unfortunately, Lindley and Morris just state this definition without proof.
- ▶ We prove its soundness in several ways.

## GTV's Results on LM Duality

- ▶ Unfortunately, Lindley and Morris just state this definition without proof.
- ▶ We prove its soundness in several ways.
  - ▶ manually

## GTV's Results on LM Duality

- ▶ Unfortunately, Lindley and Morris just state this definition without proof.
- ▶ We prove its soundness in several ways.
  - ▶ manually
  - ▶ mechanized in Agda  
<https://github.com/peterthiemann/dual-session>

# GTV's Results on LM Duality

- ▶ Unfortunately, Lindley and Morris just state this definition without proof.
- ▶ We prove its soundness in several ways.
  - ▶ manually
  - ▶ mechanized in Agda  
`https://github.com/peterthiemann/dual-session`
- ▶ We observe that it is size-preserving.

# Outline

A Coinductive Definition

Bernardi and Hennessy

Lindley and Morris

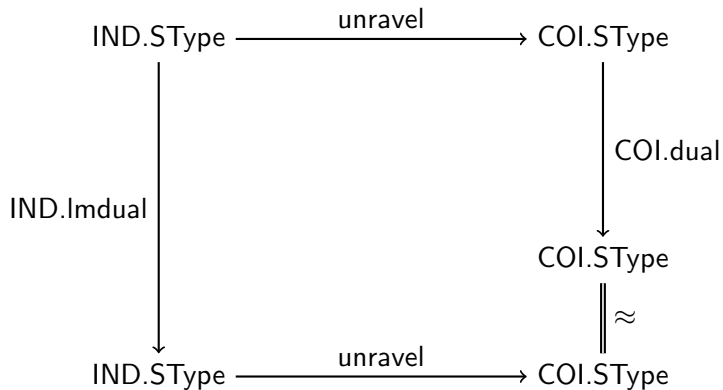
Mechanization



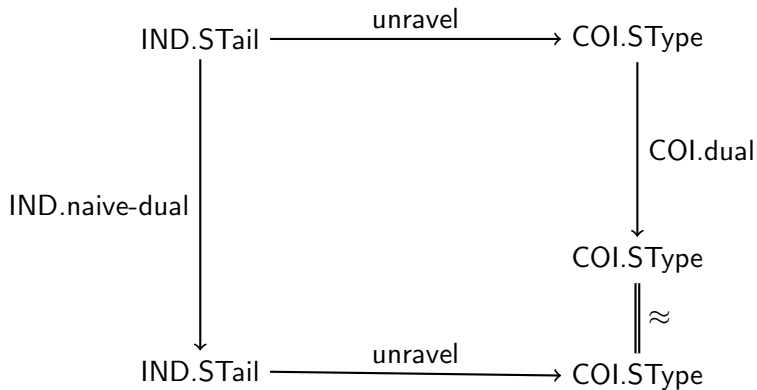
## Some Glimpses at the Agda Code

- ▶ Baseline: coinductive definitions of
  - ▶ session types with recursion
  - ▶ functional and relational duality
- ▶ inductive definition of session types with recursion
- ▶ definition of LM duality
- ▶ correspondence of LM duality with functional duality (new result)
- ▶ Not shown:
  - ▶ soundness of naive duality for tail recursive session types (new result)
  - ▶ definition of BH duality and its soundness
  - ▶ what if recursive types are not normalized? contractiveness ...
- ▶ Details in paper at the PLACES 2020 workshop  
<https://arxiv.org/abs/2004.01322v1>

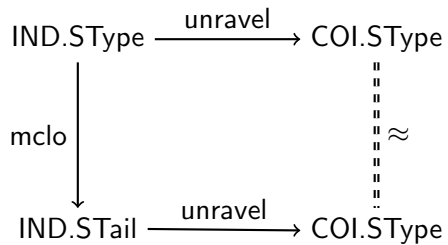
## Plan of proof: soundness of lmd



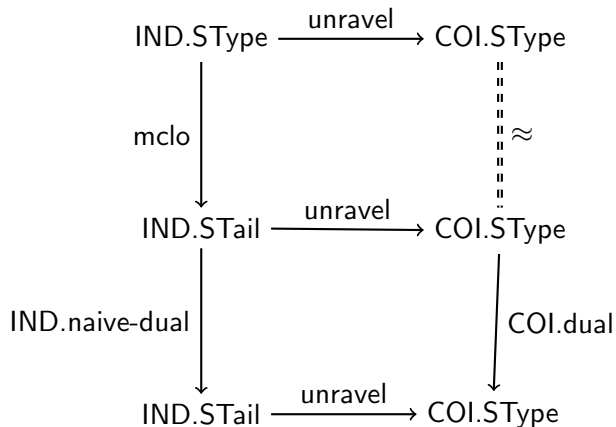
## Plan of proof: soundness of naive dual for tail-recursive session types



## Plan of proof: soundness of message closure



## Plan of proof: soundness of message closure



Thank you!