

# Contracts for Mobile Processes

Giuseppe Castagna    Luca Padovani

Laboratoire PPS, CNRS, Université Paris Diderot

Istituto di Scienze e Tecnologie dell'Informazione, Università di Urbino

Parsec meeting, January 26th, 2009

# Outline

## ① Motivation

Protocols and processes

Contracts and mobile systems

## ② Contracts

Syntax

Semantics

## ③ Results

## ④ Concluding remarks

# Protocols and processes

## Session types

- prescriptions on the use of channels

$$u : \sigma, v : \tau, \dots \vdash P$$

## Contracts

- overall process behavior

$$u : \text{Ch}, v : \text{Ch}, \dots \vdash P : T$$

## Summary

- both are behavioral types
- $\sigma$  = projection of  $T$  on  $u$

# What session types and contracts are for

## Characterizing **well-formed** systems

- the system eventually terminates
- the system never deadlocks

## Characterizing **well-typed** processes

- sent messages have the correct/expected type
- messages sent/delivered in the right order

## **Reasoning** about processes by means of their type

- refactoring processes
- searching for services

# A problem of abstraction

Session types

?Int.?Int.(!Real  $\oplus$  !Error)

?(!Bool.!Bool)

Contracts

$a.a.(\bar{b} \oplus \bar{c})$

$a$

?

A natural candidate

Contracts without channel passing  $\Rightarrow$  **CCS**

Contracts **with** channel passing  $\Rightarrow$   $\pi$ -calculus

# A problem of abstraction

Session types

$?Int.?Int.(!Real \oplus !Error)$

$?(!Bool.!Bool)$

Contracts

$a.a.(\bar{b} \oplus \bar{c})$

$a$

?

A natural candidate

Contracts without channel passing  $\Rightarrow$  **CCS**

Contracts **with** channel passing  $\Rightarrow$   $\pi$ -calculus

# An example

```
process    store?(x).x?(y : Item).  
            if y is in stock  
              then bank!(x)  
              else x!(available(y))
```



```
contract  store?(x).x?Item.(bank!x.1⊕x!Date.1)
```

# An example

**process**

`store?(x).x?(y : Item).`

`if y is in stock`

`then bank!(x)`

`else x!(available(y))`

**contract**

`store?(x).x?Item.(bank!x.1⊕x!Date.1)`

# An example

**process**

```
store?(x).x?(y : Item).
```

```
  if y is in stock
```

```
    then bank!(x)
```

```
    else x!⟨available(y)⟩
```

**contract**

```
store?(x).x?Item.(bank!x.1⊕x!Date.1)
```




# An example

**process**     `store?(x).x?(y : Item).`  
                  `if y is in stock`  
                  `then bank!(x)`  
                  `else x!(available(y))`


**contract**     `store?(x).x?Item.(bank!x.1⊕x!Date.1)`

# Some typing rules

V-SEND

$$\frac{\Gamma \vdash e : t \quad \Gamma \vdash P : T}{\Gamma \vdash \alpha!e.P : \alpha!t.T}$$


V-RECV

$$\frac{\Gamma, x : t \vdash P : T}{\Gamma \vdash \alpha?(x : t).P : \alpha?t.T}$$


C-SEND

$$\frac{\Gamma \vdash P : T}{\Gamma \vdash \alpha!(\beta).P : \alpha!\beta.T}$$

C-RECV

$$\frac{\Gamma, x : \text{Ch} \vdash P : T}{\Gamma \vdash \alpha?(x).P : \alpha?(x).T}$$

# Some typing rules

V-SEND

$$\frac{\Gamma \vdash e : t \quad \Gamma \vdash P : T}{\Gamma \vdash \alpha!e.P : \alpha!t.T}$$



V-RECV

$$\frac{\Gamma, x : t \vdash P : T}{\Gamma \vdash \alpha?(x : t).P : \alpha?t.T}$$



C-SEND

$$\frac{\Gamma \vdash P : T}{\Gamma \vdash \alpha!(\beta).P : \alpha!\beta.T}$$

C-RECV

$$\frac{\Gamma, x : \text{Ch} \vdash P : T}{\Gamma \vdash \alpha?(x).P : \alpha?(x).T}$$

undecidable  $\rightarrow$  decidable

# Outline

## ① Motivation

Protocols and processes

Contracts and mobile systems

## ② Contracts

Syntax

Semantics

## ③ Results

## ④ Concluding remarks

# Syntax

failure, success


$$T ::= \mathbf{0} \mid \mathbf{1} \quad \pi.T \mid T + T \mid T \oplus T \quad T|T \mid (\nu a)T$$
$$\pi ::= \alpha?f \mid \alpha!f \mid \alpha!(a)$$
$$f ::= x \mid (x) \mid a \mid \text{Int} \mid \text{Bool} \mid \dots$$

Infinite behaviors = infinite terms

- regularity
- boundedness

$$X = c?\text{Int}.X$$
$$X = a?(x).(c!x.\mathbf{1} \mid X)$$

# Syntax

dynamic operators


$$T ::= \mathbf{0} \mid \mathbf{1} \quad \pi.T \mid T + T \mid T \oplus T \quad T|T \mid (\nu a)T$$
$$\pi ::= \alpha?f \mid \alpha!f \mid \alpha!(a)$$
$$f ::= x \mid (x) \mid a \mid \text{Int} \mid \text{Bool} \mid \dots$$

Infinite behaviors = infinite terms

- regularity
- boundedness

$$X = c?\text{Int}.X$$
$$X = a?(x).(c!x.\mathbf{1} \mid X)$$

# Syntax

systems



$T ::= \mathbf{0} \mid \mathbf{1} \quad \pi.T \mid T + T \mid T \oplus T \quad T|T \mid (\nu a)T$

$\pi ::= \alpha?f \mid \alpha!f \mid \alpha!(a)$

$f ::= x \mid (x) \mid a \mid \text{Int} \mid \text{Bool} \mid \dots$

Infinite behaviors = infinite terms

- regularity
- boundedness

$X = c?\text{Int}.X$

$X = a?(x).(c!x.\mathbf{1} \mid X)$

# Syntax

$$T ::= \mathbf{0} \mid \mathbf{1} \quad \pi.T \mid T + T \mid T \oplus T \quad T|T \mid (\nu a)T$$
$$\pi ::= \alpha?f \mid \alpha!f \mid \alpha!(a) \leftarrow \text{--- prefixes}$$
$$f ::= x \mid (x) \mid a \mid \text{Int} \mid \text{Bool} \mid \dots$$

Infinite behaviors = infinite terms

- regularity
- boundedness

$$X = c?\text{Int}.X$$
$$X = a?(x).(c!x.\mathbf{1} \mid X)$$

# Syntax

$$T ::= \mathbf{0} \mid \mathbf{1} \quad \pi.T \mid T + T \mid T \oplus T \quad T|T \mid (\nu a)T$$
$$\pi ::= \alpha?f \mid \alpha!f \mid \alpha!(a)$$
$$f ::= x \mid (x) \mid a \mid \text{Int} \mid \text{Bool} \mid \dots$$

Info **patterns = sets of values and names + binders**

- regularity
- boundedness

$$X = c?\text{Int}.X$$
$$X = a?(x).(c!x.\mathbf{1} \mid X)$$

# Syntax

$$T ::= \mathbf{0} \mid \mathbf{1} \quad \pi.T \mid T + T \mid T \oplus T \quad T|T \mid (\nu a)T$$

$$\pi ::= \alpha?f \mid \alpha!f \mid \alpha!(a)$$

$$f ::= x \mid (x) \mid a \mid \text{Int} \mid \text{Bool} \mid \dots$$

## Infinite behaviors = infinite terms

- regularity
- boundedness

$$X = c?\text{Int}.X$$

$$X = a?(x).(c!x.\mathbf{1} \mid X)$$

# Labeled operational semantics

$$\mathbf{1} \xrightarrow{\checkmark} \mathbf{1}$$

$$\frac{m \in f \rightsquigarrow \sigma}{c?f.T \xrightarrow{c?m} T\sigma}$$

$$\frac{m \in f}{c!f.T \longrightarrow c!m.T}$$

$$c!m.T \xrightarrow{c!m} T$$

Example

$$\begin{array}{c}
 \boxed{20 \in \text{Int}} \\
 \downarrow \\
 c!\text{Int}.\mathbf{1} \mid c?\text{Real}.\mathbf{1} \longrightarrow c!20.\mathbf{1} \mid c?\text{Real}.\mathbf{1} \longrightarrow \mathbf{1} \mid \mathbf{1} \xrightarrow{\checkmark}
 \end{array}
 \quad
 \begin{array}{c}
 \boxed{20 \in \text{Real} \rightsquigarrow \emptyset} \\
 \downarrow \\
 \longrightarrow \mathbf{1} \mid \mathbf{1} \xrightarrow{\checkmark}
 \end{array}$$

# Contracts as behavioral types

## Systems

$$S \stackrel{\text{def}}{=} T_1 \mid T_2 \mid \cdots \mid T_n$$

- ① when is a **system well-formed**?
- ② when is a **process well-typed**?
- ③ when are two **types equal**?

# Participant satisfaction

## Definition

$T \triangleleft S$  if  $T \mid S \implies T' \mid S'$  and  $T' \not\rightarrow$  implies

- $T' \xrightarrow{\mu_1}$  and  $S' \xRightarrow{\mu_2}$
- $\mu_1 \# \mu_2$

$(c!m \# c?m, \checkmark \# \checkmark)$

for some  $\mu_1$  and  $\mu_2$

## Examples

- $c!\text{Int}.1 \triangleleft c?\text{Real}.1$

- $c!\text{Real}.1 \not\triangleleft c?\text{Int}.1$

$c!\text{Real}.1 \mid c?\text{Int}.1 \longrightarrow c!\sqrt{2}.1 \mid c?\text{Int}.1$

stuck

# Well-formed systems

$$S \stackrel{\text{def}}{=} T_1 \mid T_2 \mid \cdots \mid T_n$$

## Definition

$S$  is *well formed* if  $T_k \triangleleft \prod_{i \in \{1, \dots, n\} \setminus \{k\}} T_i$  for every  $1 \leq k \leq n$

## Examples

- $c!Int.1 \mid c?Real.1$  is **well formed**
- $c!Real.1 \mid c?Int.1$  is **ill formed**

# Well-typed participant

## Definition

$T$  is *viable* if  $T \mid S$  is well formed for some  $S$

## Example

$$\begin{aligned} T &\stackrel{\text{def}}{=} c?\text{Int.}\mathbf{1} + c?\text{Bool.}\mathbf{0} \\ S &\stackrel{\text{def}}{=} c?\text{Int.}\mathbf{0} + c?\text{Bool.}\mathbf{1} \end{aligned}$$

- $T$  is viable
- $S$  is viable
- $T \oplus S$  is **not** viable

# Example: global order on channels

$P \stackrel{\text{def}}{=} a?(x).b?(y).x!3.x?(z : \text{Int}).y!\text{true}.0$

$P' \stackrel{\text{def}}{=} a?(x).b?(y).x!3.y!\text{true}.x?(z : \text{Int}).0$

$Q \stackrel{\text{def}}{=} a!(c).b!(d).c?(z : \text{Int}).d?(z : \text{Bool}).c!5.0$

$Q' \stackrel{\text{def}}{=} a!(c).b!(d).c?(z : \text{Int}).c!5.d?(z' : \text{Bool}).0$

- deadlock because of cyclic dependency
- $T_P \mid T_Q$  **ill-formed (not viable!)**

# Example: global order on channels

$P \stackrel{\text{def}}{=} a?(x).b?(y).x!3.x?(z : \text{Int}).y!\text{true}.0$

$P' \stackrel{\text{def}}{=} a?(x).b?(y).x!3.y!\text{true}.x?(z : \text{Int}).0$

$Q \stackrel{\text{def}}{=} a!(c).b!(d).c?(z : \text{Int}).d?(z : \text{Bool}).c!5.0$

$Q' \stackrel{\text{def}}{=} a!(c).b!(d).c?(z : \text{Int}).c!5.d?(z' : \text{Bool}).0$


- imposing global order
- $T_P \mid T_{Q'}$  well-formed

# Example: global order on channels

$P \stackrel{\text{def}}{=} a?(x).b?(y).x!3.x?(z : \text{Int}).y!\text{true}.0$

$P' \stackrel{\text{def}}{=} a?(x).b?(y).x!3.y!\text{true}.x?(z : \text{Int}).0$

$Q \stackrel{\text{def}}{=} a!(c).b!(d).c?(z : \text{Int}).d?(z : \text{Bool}).c!5.0$



$Q' \stackrel{\text{def}}{=} a!(c).b!(d).c?(z : \text{Int}).c!5.d?(z' : \text{Bool}).0$

- global order is not necessary
- $T_{P'} \mid T_Q$  well-formed

# Example: linearity

$a?(x).b?(y).x!(y).x?(z : \text{Int}).y!\text{true}.0$   
 $a!(c).b!(d).c?(z).c!5.z?(z' : \text{Bool}).0$

# Subcontract

## Definition

$T \preceq S$  if  $T \mid R$  well formed implies  $S \mid R$  well formed for every  $R$

## Examples

- $T \oplus S \preceq T$
- $\pi.T + \pi.S \approx \pi.(T \oplus S)$   
... very much like the *must* preorder ...
- $\mathbf{0} \preceq T$

$\preceq$  is **not** a precongruence

$$\mathbf{0} \preceq T$$

## Definition (strong subcontract)

Let  $\sqsubseteq$  be the largest precongruence included in  $\preceq$

## Theorem

*If  $T$  is viable, then  $T \preceq S$  iff  $T \sqsubseteq S$*

- $T \sqsubseteq \mathbf{0}$  iff  $T$  is not viable
- if  $\mathbf{1} + T \sqsubseteq T$ , then  $T$  is well formed
- $\pi.\mathbf{0} \sqsubseteq \pi.T$

# Outline

## ① Motivation

Protocols and processes

Contracts and mobile systems

## ② Contracts

Syntax

Semantics

## ③ Results

## ④ Concluding remarks

# On progress

## Theorem

*If  $\vdash P : T$  and  $T$  w.f. and  $P \xRightarrow{\tau} Q \not\xrightarrow{\tau}$ , then  $Q$  has succeeded*

- success = “no pending actions”

# On decidability

## Proposition

- *well-formedness*
- *viability*
- *subcontract*

are decidable provided that  $c!f$  matches *finitely many* names

If a name is sent:

- either it is **fresh**
- or it is a **public** name
- or it was **received earlier**

$c!(a)$   
 $c!a$   
 $c?(x) \cdots d!x$

# Outline

## ① Motivation

Protocols and processes

Contracts and mobile systems

## ② Contracts

Syntax

Semantics

## ③ Results

## ④ Concluding remarks

# Session types and contracts: a comparison

- optimistic vs conservative
- global vs compositional

	Session types	Contracts
structuring	++	--
analysis	--	++

# Concluding remarks

## Contributions

- ① contracts for processes with channel mobility
- ② straightforward solution to global progress  
(of bounded systems)

## Our wish list

- algorithms (almost done)
- choreographic specifications
- expressiveness

# Concluding remarks

## Contributions

- ① contracts for processes with channel mobility
- ② straightforward solution to global progress (of bounded systems)

## Our wish list

- algorithms (almost done)
- choreographic specifications
- expressiveness

**Thank you.**

# Regular does not mean finite-state

- unbounded participants
- unbounded buffers
- state encoded within processes

$$P(x : \text{Int}) = \text{deposit?}(y : \text{Int}).P(x + y) \\ + \text{withdraw?}(y : \text{Int}).P(\max\{0, x - y\})$$

$$P(0)$$

$$P = c?(x : \text{Int}).(\text{deposit?}(y : \text{Int}).c!\langle x + y \rangle.P \\ + \text{withdraw?}(y : \text{Int}).c!\langle \max\{0, x - y\} \rangle.P)$$

$$Q = c?(x : \text{Int}).c!\langle x \rangle.Q$$

$$(\nu c)(P \mid c!\langle 0 \rangle.Q)$$

# Simulating asynchrony

INPUT

$$\frac{\Gamma \vdash \alpha : \text{Ch} \quad \Gamma, x : t \vdash P : T}{\Gamma \vdash \alpha?(x : t).P : \alpha?t.T + \alpha?\neg t.\mathbf{0}}$$

C-RECV

$$\frac{\Gamma \vdash \alpha : \text{Ch} \quad \Gamma, x : \text{Ch} \vdash P : T}{\Gamma \vdash \alpha?(x).P : \alpha?(x).T + \alpha?\neg \text{Ch}.\mathbf{0}}$$