Fair asynchronous session subtyping

Gianluigi Zavattaro

Department of Computer Science
University of Bologna

INRIA research team FOCUS

Structure of the presentation

- Binary Session Subtyping
  - Gay-Hole
- Asynchronous Session Subtyping
  - Mostrous-Yoshida
- Fair asynchronous session subtyping
  - Bravetti-Lange-Zavattaro
- A couple of open-problems

Structure of the presentation

- **Binary Session Subtyping**
  - Gay-Hole
- Asynchronous Session Subtyping
  - Mostrous-Yoshida
- Fair asynchronous session subtyping
  - Bravetti-Lange-Zavattaro
- A couple of open-problems

Session types [ESOP98]

Language Primitives and Type Discipline for Structured Communication-Based Programming

KOHEI HONDA*, VASCO T. VASCONCELOS†, AND MAKOTO KUBO‡

- A type discipline for communication-based **concurrent** programming

$$S ::= \mathbf{nat} \mid \mathbf{bool} \mid \langle \alpha, \overline{\alpha} \rangle \mid s \mid \mu s.S$$

$$\alpha ::= \downarrow[\tilde{S}];\alpha \mid \downarrow[\alpha];\beta \mid \&\{l_1 : \alpha_1, \ldots, l_n : \alpha_n\} \mid \mathbf{1} \mid \bot$$
$$\mid \uparrow[\tilde{S}];\alpha \mid \uparrow[\alpha];\beta \mid \oplus\{l_1 : \alpha_1, \ldots, l_n : \alpha_n\} \mid t \mid \mu t.\alpha$$

(Simple) session types

- A **minimal** syntax for session types:

$$T ::= \oplus\{l_i : T_i\}_{i \in I} \mid \&\{l_i : T_i\}_{i \in I} \mid \mu \mathbf{t}.T \mid \mathbf{t} \mid \mathbf{end}$$

(Simple) session types

- A **minimal** syntax for session types:

$$T ::= \oplus\{l_i : T_i\}_{i \in I} \mid \&\{l_i : T_i\}_{i \in I} \mid \mu \mathbf{t}.T \mid \mathbf{t} \mid \mathbf{end}$$

Selection among **outputs**

(Simple) session types

- A **minimal** syntax for session types:

$$T ::= \oplus\{l_i : T_i\}_{i \in I} \mid \&\{l_i : T_i\}_{i \in I} \mid \mu \mathbf{t}.T \mid \mathbf{t} \mid \mathbf{end}$$

Selection among **outputs**

Branching among **inputs**

(Simple) session types

- A **minimal** syntax for session types:

$$T ::= \oplus\{l_i : T_i\}_{i \in I} \mid \&\{l_i : T_i\}_{i \in I} \mid \mu \mathbf{t}.T \mid \mathbf{t} \mid \mathbf{end}$$

Selection among **outputs**

Recursion

Branching among **inputs**

(Simple) session types

- A **minimal** syntax for session types:

$$T ::= \oplus\{l_i : T_i\}_{i \in I} \mid \&\{l_i : T_i\}_{i \in I} \mid \mu \mathbf{t}.T \mid \mathbf{t} \mid \mathbf{end}$$

Selection among **outputs**

Recursion

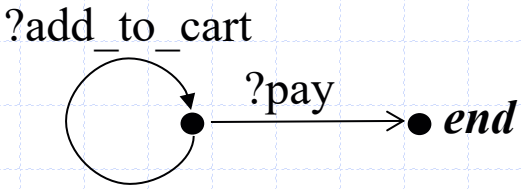
Termination

Branching among **inputs**

(Simple) example

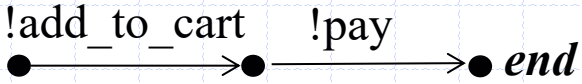
- A session type for a service endpoint:

$$S_{service} = \mu \mathbf{t}.\&\{\text{add\_to\_cart} : \mathbf{t}\ ,\ \ \text{pay} : \mathbf{end}\}$$



- A session type for a client endpoint:

$$T_{client} = \oplus\{\text{add\_to\_cart} : \oplus\{\text{pay} : \mathbf{end}\}\}$$



Liskov substitution principle [ToPLaS94]

A Behavioral Notion of Subtyping

BARBARA H. LISKOV
MIT Laboratory for Computer Science
and
JEANNETTE M. WING
Carnegie Mellon University

Subtype Requirement: Let $\phi(x)$ be a property provable about objects x of type T. Then $\phi(y)$ should be true for objects y of type S where S is a subtype of T.

- An endpoint can be safely replaced by another endpoint with a session **subtype**

Session subtyping [ActaInf05]

Simon Gay · Malcolm Hole

Subtyping for session types in the pi calculus

Definition (Synchronous Subtyping, $\leq$). *$\mathcal{R}$ is a synchronous subtyping relation whenever $(T, S) \in \mathcal{R}$ implies that:*

1. *if* $T = \mathbf{end}$ *then* $\exists n \geq 0$ *such that* $\mathsf{unfold}^n(S) = \mathbf{end}$;
2. *if* $T = \oplus\{l_i : T_i\}_{i \in I}$ *then* $\exists n \geq 0$ *such that* $\mathsf{unfold}^n(S) = \oplus\{l_j : S_j\}_{j \in J}$, $I \subseteq J$ *and* $\forall i \in I.\, (T_i, S_i) \in \mathcal{R}$;
3. *if* $T = \&\{l_i : T_i\}_{i \in I}$ *then* $\exists n \geq 0$ *such that* $\mathsf{unfold}^n(S) = \&\{l_j : S_j\}_{j \in J}$, $J \subseteq I$ *and* $\forall j \in J.\, (T_j, S_j) \in \mathcal{R}$;
4. *if* $T = \mu \mathbf{t}.T'$ *then* $(T'\{T/\mathbf{t}\}, S) \in \mathcal{R}$.

T is a synchronous subtype of S, written $T \leq S$, if there is a synchronous subtyping relation $\mathcal{R}$ such that $(T, S) \in \mathcal{R}$.

Output Covariance and Input Contravariance

- Output Covariance
  - subtype may have a **subset of outputs**
  - example:

$$\oplus\{l_1 : T_1\} \leq \oplus\{l_1 : T_1, l_2 : T_2\}$$

- Input Contravariance
  - subtype may have a **superset of inputs**
  - example:

$$\&\{l_1 : T_1, l_2 : T_2\} \leq \&\{l_1 : T_1\}$$

Structure of the presentation

- Binary Session Subtyping
  - Gay-Hole
- **Asynchronous Session Subtyping**
  - Mostrous-Yoshida
- Fair asynchronous session subtyping
  - Bravetti-Lange-Zavattaro
- A couple of open-problems

Asynchronous Subtyping [ESOP09]

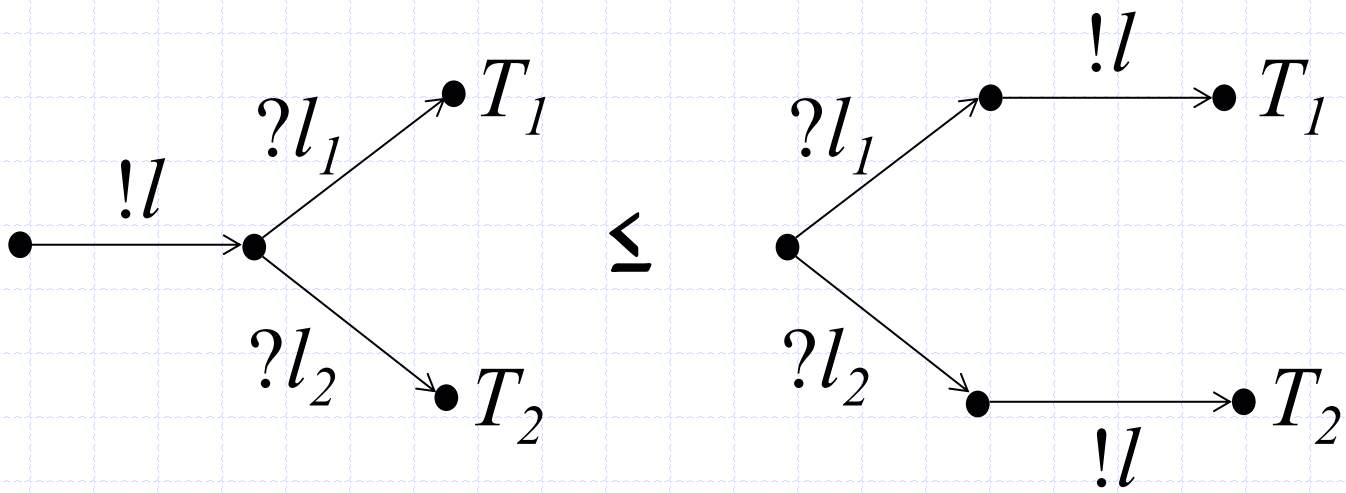
Global Principal Typing in Partially Commutative Asynchronous Sessions

Dimitris Mostrous[1], Nobuko Yoshida[1], and Kohei Honda[2]

- Besides output covariance / input contravariance a subtype may have outputs **anticipated** w.r.t. inputs

Asynchronous Subtyping [ESOP09]

$$\oplus\{l : \&\{l_1 : T_1, l_2 : T_2\}\} \leq \&\{l_1 : \oplus\{l : T_1\}, l_2 : \oplus\{l : T_2\}\}$$



- Besides output covariance / input contravariance a subtype may have outputs **anticipated** w.r.t. inputs

Synchronous Session Subtyping

Definition (Synchronous Subtyping, $\leq$). *$\mathcal{R}$ is a synchronous subtyping relation whenever $(T, S) \in \mathcal{R}$ implies that:*

1. *if $T = \mathbf{end}$ then $\exists n \geq 0$ such that $\mathsf{unfold}^n(S) = \mathbf{end}$;*
2. *if $T = \oplus\{l_i : T_i\}_{i \in I}$ then $\exists n \geq 0$ such that $\mathsf{unfold}^n(S) = \oplus\{l_j : S_j\}_{j \in J}$, $I \subseteq J$ and $\forall i \in I. (T_i, S_i) \in \mathcal{R}$;*
3. *if $T = \&\{l_i : T_i\}_{i \in I}$ then $\exists n \geq 0$ such that $\mathsf{unfold}^n(S) = \&\{l_j : S_j\}_{j \in J}$, $J \subseteq I$ and $\forall j \in J. (T_j, S_j) \in \mathcal{R}$;*
4. *if $T = \mu \mathbf{t}.T'$ then $(T'\{T/\mathbf{t}\}, S) \in \mathcal{R}$.*

T is a synchronous subtype of S, written $T \leq S$, if there is a synchronous subtyping relation $\mathcal{R}$ such that $(T, S) \in \mathcal{R}$.

Input Context [I&C15]

Session typing and asynchronous subtyping for the higher-order π-calculus

Dimitris Mostrous [a,*], Nobuko Yoshida [b]

Definition (Input Context). *An input context $\mathcal{A}$ is a session type with multiple holes defined by the syntax:*

$$\mathcal{A} \ ::= \quad []^n \quad | \quad \&\{l_i : \mathcal{A}_i\}_{i \in I}$$

Input Context [I&C15]

Session typing and asynchronous subtyping for the higher-order π-calculus

Dimitris Mostrous [a,*], Nobuko Yoshida [b]

Definition (Input Context). *An input context $\mathcal{A}$ is a session type with multiple holes defined by the syntax:*

$$\mathcal{A} \ ::= \quad []^n \quad | \quad \&\{l_i : \mathcal{A}_i\}_{i \in I}$$

- How to use **input contexts**:

$\mathcal{A}[T_k]^{k \in \{1,\ldots,m\}}$ *denotes the type obtained by filling each hole k in $\mathcal{A}$ with T_k*

$$\&\{l_1 : \oplus\{l : T_1\}, l_2 : \oplus\{l : T_2\}\} \quad = \quad \&\{l_1 : [\,]^1, l_2 : [\,]^2\}$$

$$\oplus\{l : T_1\} \qquad \oplus\{l : T_2\}$$

Asynchronous Session Subtyping

Definition (Asynchronous Subtyping, $\leq$). *$\mathcal{R}$ is an asynchronous subtyping relation whenever $(T, S) \in \mathcal{R}$ implies that:*

1. *if $T = \mathbf{end}$ then $\exists n \geq 0$ such that $\mathsf{unfold}^n(S) = \mathbf{end}$;*
2. *if $T = \oplus\{l_i : T_i\}_{i \in I}$ then $\exists n \geq 0, \mathcal{A}$ such that*
   - $\mathsf{unfold}^n(S) = \mathcal{A}[\oplus\{l_j : S_{kj}\}_{j \in J_k}]^{k \in \{1,\ldots,m\}}$,
   - $\forall k \in \{1, \ldots, m\}. I \subseteq J_k$ *and*
   - $\forall i \in I. (T_i, \mathcal{A}[S_{ki}]^{k \in \{1,\ldots,m\}}) \in \mathcal{R}$;
3. *if $T = \&\{l_i : T_i\}_{i \in I}$ then $\exists n \geq 0$ such that $\mathsf{unfold}^n(S) = \&\{l_j : S_j\}_{j \in J}$, $J \subseteq I$ and $\forall j \in J. (T_j, S_j) \in \mathcal{R}$;*
4. *if $T = \mu \mathbf{t}.T'$ then $(T'\{T/\mathbf{t}\}, S) \in \mathcal{R}$.*

T is an asynchronous subtype of S, written $T \leq S$, if there is an asynchronous subtyping relation $\mathcal{R}$ such that $(T, S) \in \mathcal{R}$.

Structure of the presentation

- Binary Session Subtyping
  - Gay-Hole
- Asynchronous Session Subtyping
  - Mostrous-Yoshida
- **Fair asynchronous session subtyping**
  - Bravetti-Lange-Zavattaro
- A couple of open-problems

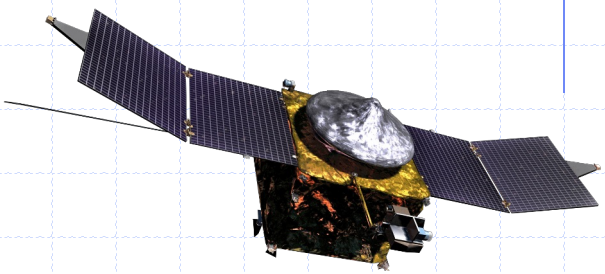
Space telecom
motivating example in [FOSSACS21]

Spacecraft

```
s := new Connection()
tm[] := prepareTelemetries()
for (i := 0; i < tm.len; i++)
{
    s.send("TM", tm[i])
}
s.send("OVER", nil)
for {
    select x := s.receive() {
    case x = "TC":
        processTC(x)

    case x = "DONE":
        break
    }
}
```

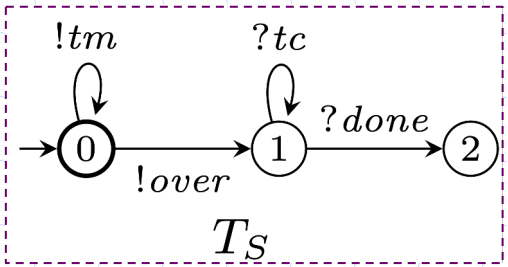
Ground station

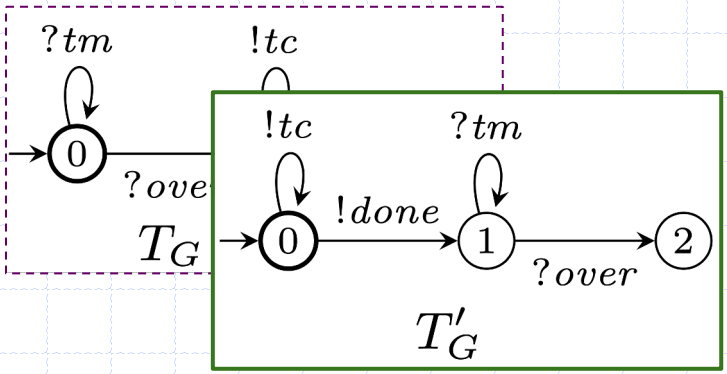
```
s := new Connection()
for {
    select x := s.receive() {
    case x = "TM":
        processTM(x)

    case x = "OVER":
        break
    }
}
tc[] := prepareTelecommands()
for (i := 0; i < tc.len; i++) {
    s.send("TC", tc[i])
}
s.send("DONE", nil)
```



Telecommand

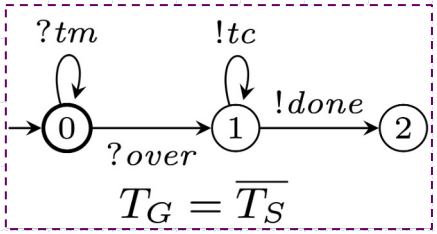
Telemetry



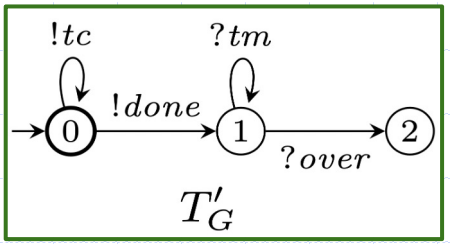


Is T'_G a subtype of T_G ?

- No!
  - The input context used in the output rule is **finite**:



VI



2. *if* $T = \oplus\{l_i : T_i\}_{i\in I}$ *then* $\exists n \geq 0, \mathcal{A}$ *such that*
   - $\text{unfold}^n(S) = \mathcal{A}[\oplus\{l_j : S_{kj}\}_{j\in J_k}]^{k\in\{1,\ldots,m\}}$,
   - $\forall k \in \{1,\ldots,m\}. I \subseteq J_k$ *and*
   - $\forall i \in I. (T_i, \mathcal{A}[S_{ki}]^{k\in\{1,\ldots,m\}}) \in \mathcal{R}$;

$$\mathcal{A} \;::=\; []^n \;\mid\; \&\{l_i : \mathcal{A}_i\}_{i\in I}$$

Infinite input contexts

- We consider input contexts with **recursion**..

$$\mathcal{A} ::= \quad []^k \quad | \quad \&\{l_i : \mathcal{A}_i\}_{i \in I} \quad | \quad \mu \mathbf{t}.\mathcal{A} \quad | \quad \mathbf{t}$$

Infinite input contexts

- We consider input contexts with **recursion**..

$$\mathcal{A} ::= \quad []^k \quad | \quad \&\{l_i : \mathcal{A}_i\}_{i \in I} \quad | \quad \mu \mathbf{t}.\mathcal{A} \quad | \quad \mathbf{t}$$

- ..which require a new **unfolding**
  - to avoid **output** constructs including free recursive variables

Not a valid session type

$$T = \mu \mathbf{t}.\&\{l_1 : \mathbf{t},\ l_2 : \oplus\{l_3 : \mathbf{t}\}\}$$

Infinite input contexts

- We consider input contexts with **recursion**..

$$\mathcal{A} ::= \quad []^k \quad | \quad \&\{l_i : \mathcal{A}_i\}_{i \in I} \quad | \quad \mu \mathbf{t}.\mathcal{A} \quad | \quad \mathbf{t}$$

- ..which require a new **unfolding**
  - to avoid **output** constructs including free recursive variables

A valid session type

$$T = \mu \mathbf{t}.\&\{l_1 : \mathbf{t},\ l_2 : \oplus\{l_3 : \mathbf{t}\}\}$$

$$\text{selUnfold}(T) = \mu \mathbf{t}.\&\{l_1 : \mathbf{t},\ l_2 : \oplus\{l_3 : \mu \mathbf{t}.\ \&\{l_1 : \mathbf{t},\ l_2 : \oplus\{l_3 : \mathbf{t}\}\}\}\}$$

Fair asynchronous subtyping

Definition 12 (Fair Asynchronous Subtyping, $\leq$).

A relation $\mathcal{R}$ on session types is a controllable subtyping relation whenever $(T, S) \in \mathcal{R}$ implies:

1. *if* $T = \mathbf{end}$ *then* $\mathsf{unfold}(S) = \mathbf{end}$;
2. *if* $T = \mu \mathbf{t}.T'$ *then* $(T'\{^T/_{\mathbf{t}}\}, S) \in \mathcal{R}$;
3. *if* $T = \&\{l_i : T_i\}_{i \in I}$ *then*
   - $\mathsf{unfold}(S) = \&\{l_j : S_j\}_{j \in J}$,
   - $K = \{k \in J \mid S_k \text{ is controllable}\}$,
   - $I \supseteq K$, *and* $\forall k \in K.\, (T_k, S_k) \in \mathcal{R}$;
4. *if* $T = \oplus\{l_i : T_i\}_{i \in I}$ *then*
   - $\mathsf{selUnfold}(S) = \mathcal{A}[\oplus\{l_i : S_{ki}\}_{i \in I}]^{k \in K}$ *and*
   - $\forall i \in I.\, (T_i, \mathcal{A}[S_{ki}]^{k \in K}) \in \mathcal{R}$.

T is a controllable subtype of S if there is a controllable subtyping relation $\mathcal{R}$ s.t. $(T, S) \in \mathcal{R}$.

T is a fair asynchronous subtype *of S, written $T \leq S$, whenever: S controllable implies that T is a controllable subtype of S.*

Which are the properties preserved by this subtyping ?

- Our subtyping preserves the following **fair termination** property:

Definition 5 (Fair Compliance). *Two types T and S are* fair compliant *if whenever* $[T, \epsilon]||[S, \epsilon] \rightarrow^* s'$, *there exists a continuation* $s' \rightarrow^* [\mathbf{end}, \epsilon]||[\mathbf{end}, \epsilon]$.

  - for every **reachable** configuration, the communication can continue until **termination**

Attention to output covariance

- Notice: subtype and supertype have the **same** output labels:

$$4.\ \ \textit{if}\ T = \oplus\{l_i : T_i\}_{i \in I}\ \textit{then}$$
$$- \ \mathsf{selUnfold}(S) = \mathcal{A}[\oplus\{l_i : S_{ki}\}_{i \in I}]^{k \in K}\ \ \textit{and}$$
$$- \ \forall i \in I.\, (T_i, \mathcal{A}[S_{ki}]^{k \in K}) \in \mathcal{R}.$$

  - Choices necessary to reach **termination** must be present also in the subtype

Structure of the presentation

- Binary Session Subtyping
  - Gay-Hole
- Asynchronous Session Subtyping
  - Mostrous-Yoshida
- Fair asynchronous session subtyping
  - Bravetti-Lange-Zavattaro
- **A couple of open-problems**

A couple of open-problems

- Characterize the **sound** and **complete** subtyping w.r.t. fair termination
  - use techniques already used for the **synchronous** case (see Padovani's talk)
- (Fair) asynchronous session subtypings are **undecidable**
  - we have already a **sound** algorithmic approximation; other techniques (eg. Dagnino et al.) could detect other interesting patterns