

# Typed Polyadic $\pi$ -calculus and Sortings

Mikkel Bundgaard and Vladimiro Sassone

April 27 2006

Overview of the talk:

Introduction

In More Detail

Conclusion

# The polyadic Pi-calculus and Type Systems

- ▶ Generalisation of the monadic  $\pi$ -calculus as we can send a tuple of names in a single communication
- ▶ Now communication can “go wrong”, hence the need for a type system
- ▶ We consider a variant of the **capability** types of Pierce and Sangiorgi
- ▶ **Types** are assigned to names and they represent how the name can be used, hence types are tuples of types tagged with either an input, output or both **tag**
- ▶ The type system also has a sub type relation with **bounded meets** (a crucial property)

# (Link) Sortings

- ▶ A **sorting** enforces a condition on bigraphs and allows us to only consider bigraphs which satisfy this condition.
- ▶ A **link** sortings enforce requirements on the kind of linkage that can occur in a bigraph
- ▶ Technically this is done by adding sorts to the two interfaces of a bigraph and then stipulate a condition which must be satisfied.

# Goal

- ▶ **Main goal:** To represent type systems using link sortings
- ▶ Prior link sortings have “only” been used to ensure that we only consider well-formed bigraphs (Leifer and Milner and O’Conchuir)
- ▶ Note the similarity between sortings and type system in both settings we add some information and then use this information to rule out ill-typed processes/bigraphs

# Extending the Theory

(or sorted binding bigraphs and sortings in 3 slides)

# Binding Bigraphs with Controls for Edges and Sorts

- ▶ We will not introduce binding bigraphs in this presentation
- ▶ However, note that we have augmented the definition with a set of **controls** for edges (as we want to represent binders with sort annotation)
- ▶ The theory remains **unchanged** despite this augmentation

## Sorted binding bigraphs

- ▶ We add **sorts** to the interfaces and to edge controls
- ▶ We do not assign sorts to ports, but only the node (control) as a whole

# Weakly reflecting pushout, bounded meets, etc.

- ▶ We generalise reflects pushout to **weakly** reflects pushout (for IPOs) to allow for a larger class of sortings
- ▶ In a **sub sorting**  $\Sigma = (\Theta, \mathcal{K}, \mathcal{E}, \Phi)$  the sorts are a set  $\mathcal{S}$  with a preorder  $\leq$  which must have **bounded meets**, and we have a bijective function  $pack : \mathcal{S}^* \times Q \rightarrow \mathcal{S}$ , which takes a tuple of sorts and an element from a set  $Q$  and returns a sort.
- ▶ **Condition**
  - ▶ For every link  $l : S$  and for every point  $p : S'$  of  $l$  we must have  $S \leq S'$ . (think **subsumption**)
  - ▶ For every control  $K \in \mathcal{K}$  with  $ar(K) = n > 0$  we associate it with an element  $q \in Q$  and require that  $s_0 \leq pack(s_1, \dots, s_{n-1}, q)$ , where  $s_0$  is the sort of the first port on  $K$ ,  $s_1$  the sort of the second, and so forth.

# Sub sorting creates RPOs, weakly reflects pushout

- ▶ Sub sorting **creates RPOs** and **weakly reflects pushout** (proven by hand and not using safeness)
  - ▶ For this we need that  $\leq$  is a **preorder** and that it has **bounded meets**
- ▶ Most sub sorted bigraphs are not **opcartesian** (simple counterexample using subtyping)
- ▶ We can use most of the existing results (bisimilarity in ST is a **congruence** and FPE is **adequate** for ST)

# Polyadic $\pi$ -calculus with Capability Types

## Syntax

$$P ::= \mathbf{0} \mid P \mid P' \mid (\nu n : S)P \\ \mid \bar{n}\langle m_1, \dots, m_n \rangle . P \mid n(m_1 : S_1, \dots, m_n : S_n) . P$$

## Reaction Relation

---

$$n(m_1 : S_1, \dots, m_i : S_i) . P \mid \bar{n}\langle m'_1, \dots, m'_i \rangle . Q \rightarrow_{\pi} P\{m'_1, \dots, m'_i / m_1, \dots, m_i\} \mid Q$$

- ▶ Only allow communication when the parties **agree** on the length of the tuple being communicated
- ▶ We need a **type system** to ensure this (and that this is preserved under reaction)

# Sorts

We define the set of **sorts** of our type system using the following rules (letting  $l ::= b \mid r \mid w$ )

$$\frac{}{()^l :: Type} \quad \frac{T_1 \dots T_n :: Type}{(T_1, \dots, T_n)^r :: Type} \quad \frac{T_1 \dots T_n :: Type}{(T_1, \dots, T_n)^w :: Type}$$
$$\frac{T_1 \dots T_n :: Type \quad S_1 \dots S_n :: Type \quad S_i \leq T_i}{(T_1, \dots, T_n; S_1, \dots, S_n)^b :: Type}$$

- ▶ Given a  $(T_1, \dots, T_n; S_1, \dots, S_n)^b$ ,  $T_1, \dots, T_n$  **input capability** and  $S_1, \dots, S_n$  **output capability**.
- ▶ Uses the sub sorting relation defined on the next slide

## Sub sort relation

$$\begin{array}{c}
 \text{for each } i, T_i \leq T'_i \\
 \hline
 (T_1, \dots, T_n)^r \leq (T'_1, \dots, T'_n)^r
 \end{array}
 \quad
 \begin{array}{c}
 \text{for each } i, T_i \leq T'_i \\
 \hline
 (T'_1, \dots, T'_n)^w \leq (T_1, \dots, T_n)^w
 \end{array}$$

$$\begin{array}{c}
 \text{for each } i, T_i \leq T'_i \text{ and } S_i \leq S'_i \\
 \hline
 (T_1, \dots, T_n; S'_1, \dots, S'_n)^b \leq (T'_1, \dots, T'_n; S_1, \dots, S_n)^b
 \end{array}$$

$$\begin{array}{c}
 \text{for each } i, T_i \leq T'_i \\
 \hline
 (T_1, \dots, T_n; S_1, \dots, S_n)^b \leq (T'_1, \dots, T'_n)^r
 \end{array}
 \quad
 \begin{array}{c}
 \text{for each } i, T_i \leq T'_i \\
 \hline
 (S_1, \dots, S_n; T'_1, \dots, T'_n)^b \leq (T_1, \dots, T_n)^w
 \end{array}$$

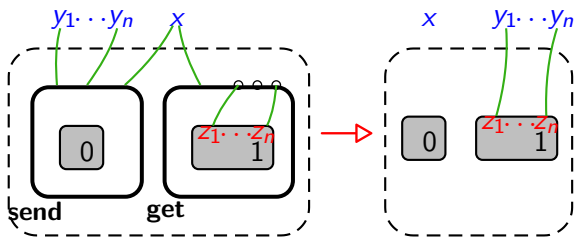
- ▶ r-tag (w-tag) is a **covariant** (**contravariant**) constructor
- ▶ For **b-tags** the operator is covariant in the first part and contravariant in the second part
- ▶ This variant has **bounded meets** (contrary to the original version)

# The typing relation

$$\begin{array}{c}
 \frac{\Gamma \vdash P : \circ \quad \Gamma \vdash Q : \circ}{\Gamma \vdash P \mid Q : \circ} \quad \frac{\Gamma, n : S \vdash P : \circ}{\Gamma \vdash (\nu n : S)P : \circ} \quad \frac{}{\Gamma \vdash \mathbf{0} : \circ} \\
 \\
 \frac{\Gamma(n) \leq (\Gamma(m_1), \dots, \Gamma(m_n))^w \quad \Gamma \vdash P : \circ}{\Gamma \vdash \bar{n}\langle m_1, \dots, m_n \rangle . P : \circ} \\
 \\
 \frac{\Gamma(n) \leq (S_1, \dots, S_n)^r \quad \Gamma, m_1 : S_1, \dots, m_n : S_n \vdash P : \circ}{\Gamma \vdash n(m_1 : S_1, \dots, m_n : S_n) . P : \circ}
 \end{array}$$

- ▶ Only input and output are interesting
- ▶ We can prove the **standard properties** (weakening, narrowing, subject relation)

# Bigraphical Presentation

The sorted BRS  $S_{\text{BBG}}^{\pi_{\leq}}$ 

- ▶ The sorted BRS is as **expected**, where we associate **send** with a contravariant operator and **get** with covariant operator
- ▶  $y_1, \dots, y_n$  have sorts  $T_1, \dots, T_n$ ,  $z_1, \dots, z_n$  and the edges they are connected to have sort  $U_1, \dots, U_n$ , and the name  $x$  has sort  $(U_1, \dots, U_n; T_1, \dots, T_n)^b$
- ▶ the family of reaction rules is **indexed** by  $n$  and the sorts
- ▶ however, this is **not** an optimal solution (for deriving labels)

# Translation and standard properties

- ▶ we translate a well-typed process  $\Gamma \vdash P : \circ$ , inductively in the typing derivation of  $P$  into the homset  $(\epsilon, \langle \Gamma \rangle)$   
 $\langle \Gamma \rangle$  shorthand for  $\langle 1, (\emptyset), \text{dom}(\Gamma), \Gamma \rangle$

## static correspondence

$\Gamma \vdash P : \circ \equiv_{\pi} \Gamma \vdash P' : \circ$  if and only if  $\llbracket \Gamma \vdash P : \circ \rrbracket = \llbracket \Gamma \vdash P' : \circ \rrbracket$

- ▶ **weakening**, we can express weakening as tensoring with the idle name in bigraphs
- ▶ **narrowing**, we can express narrowing as composition

## dynamic correspondence

For every well-typed process  $\Gamma \vdash P : \circ$  and agent  $a : \langle \Gamma \rangle$  we have

$$\llbracket \Gamma \vdash P : \circ \rrbracket \rightarrow a \text{ if and only if } P \rightarrow_{\pi} a_{\pi}$$

# Deriving Labelled Transitions

The labels are **characterised** using the same method as Jensen and Milner. Given  $a \xrightarrow{L} a'$ . We can characterise  $a$ ,  $L$ , and  $a'$  in the following forms

$$\begin{aligned} a &= (/Z : \tilde{S})(r_a \mid b) : \langle \text{sort} \rangle \\ L &= \langle \sigma \rangle \mid r_L : \langle \text{sort} \rangle \rightarrow \langle \text{sort}' \rangle \\ a' &= \sigma(/Z : \tilde{S})(^{y_1, \dots, y_n} /_{(z_1, \dots, z_n)} c_2 \mid c_1 \mid b) : \langle \text{sort}' \rangle \end{aligned}$$

where one of the following cases holds

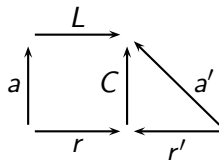
$r_a$	$r_L$	$\sigma$
<b>send</b> <sub><math>xy_1 \dots y_n</math></sub> $c_1$	<b>get</b> <sub><math>x(z_1 \dots z_n)</math></sub> $c_2$	sub $_X$
<b>get</b> <sub><math>x(z_1 \dots z_n)</math></sub> $c_2$	<b>send</b> <sub><math>xy_1 \dots y_n</math></sub> $c_1$	sub $_X$
<b>send</b> <sub><math>x_0 y_1 \dots y_n</math></sub> $c_1$   <b>get</b> <sub><math>x_1(z_1 \dots z_n)</math></sub> $c_2$	1	sub $_{X \setminus x_i} \mid x_i / x_i$
<b>send</b> <sub><math>xy_1 \dots y_n</math></sub> $c_1$   <b>get</b> <sub><math>x(z_1 \dots z_n)</math></sub> $c_2$	1	sub $_X$

# Redundant Labels

- ▶ The structure of the label is as **expected** (for synchronous  $\pi$ -calculus)
- ▶ But what about the **sorts** in the outer face of the agent ?
- ▶ Ideally, in a transition  $a \xrightarrow{L} a'$ , we would like that  $L$  only changes the sort of a name if this is absolute necessary (the only name that might need sub sorting is the name communicated over)
- ▶ However, the way parametric reaction rules are defined creates problems:
  - ▶ it is not possible to give **one minimal** sorted parametric rule as this conflicts with the sorting condition
  - ▶ the **grounding** of parametric rules (the agent we ground with should be minimal sorted in some sense)

# Redundant Labels Cont.

- ▶ The IPO property of a transition,  $a \xrightarrow{L} a'$ ,



enforces that the label  $L$  is **minimal** compared to the given agent  $a$  and the chosen ground reaction rule  $r$ .

- ▶ but we have an **infinite** number of reaction rules
- ▶ we want to choose the **minimal** reaction rule (among reaction rules which only differ on the sorts in their outer face), i.e. the reaction rule that introduce the least sub sorting when deriving the label.

# Solution

- ▶ We look at the **sub-TS**  $\mathcal{M}$  where we only consider labels without redundant sub sorting

We use that the LTSs have the following **properties** (letting  $\varphi$  be a resorting)

- ▶ In the full TS  $\mathcal{L}$  (the TS with redundant labels)  $\varphi a \xrightarrow{L} a'$  iff  $a \xrightarrow{L\varphi} a'$
- ▶ If  $a \xrightarrow{L} a'$  in  $\mathcal{L}$  the  $\exists \varphi. a \xrightarrow{M} a''$  and  $L = \varphi M$  and  $a' = \varphi a''$  in  $\mathcal{M}$

These properties are enough to ensure **adequacy** of the sub-TS

# Comparison with work on straight transitions

For **straight** transitions

- ▶ **problem:** one reaction rule can be matched in several ways
- ▶ transitions are only defined up-to isomorphism
- ▶ we need not to consider the agent for defining straightness
- ▶ every link graph is isomorphic to a straight link graph
- ▶ we know that iso-classes are adequate

For labels **without redundant sub sorting**

- ▶ **problem:** several (similar) redundant rules for a given agent
- ▶ a precise label is defined in terms of the agent involved
- ▶ we derive redundant labels and then prove that we can disregard those instead of proving that we can disregard them from the beginning

# Conclusions and Future Work

## Conclusions:

- ▶ It is possible to **represent** type systems for the  $\pi$ -calculus as sortings in bigraphs and to derive **reasonable** LTS
- ▶ To this end we have **conservatively extended** and lifted some of the theory of BRSs (edge controls, weakly reflect pushouts, sortings for binding bigraphs, etc.)

## Future Work

- ▶ More **advanced** type systems (linearity, behavioural and recursive types)
- ▶ **Compare** the derived congruence with typed equivalences of Hennessy *et al* and Deng and Sangiorgi
- ▶ More **general connection** between type system features and requirements on sorts (subtyping vs. bounded meets)

Finished