

Bigraphical Semantics of Higher-Order Mobile Embedded Resources with Local Names

Mikkel Bundgaard and Thomas Hildebrandt
`{mikkelbu, hilde}@itu.dk`

Department of Theoretical Computer Science
IT University of Copenhagen

Graph Transformation for Verification and Concurrency 2005

Overview of the talk:

Introduction — Bigraphs and Homer

Encoding

Conclusions and Future Work

Bigraphical Programming Languages (BPL)

Context of this work:

- ▶ A research initiative at the **IT University of Copenhagen** consisting of approximately 10 persons at IT University of Copenhagen
- ▶ **Purpose:** in collaboration with Prof. Robin Milner, to develop a programming language and programming environment for context-dependent mobile communication based on the bigraph model.
- ▶ **My part** (general): focusing on how one can give semantics to higher-order mobile active resources with local references when the resources can be copied

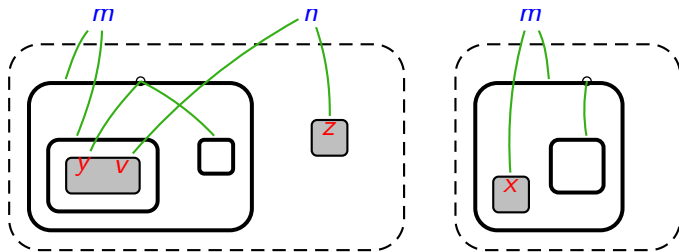
Introduction

- ▶ There already exists **bigraphical presentations** of:
 - ▶ **local names** and scoping (the **π -calculus**)
 - ▶ active processes in **nested locations** (**Mobile Ambients**)
 - ▶ non-linear higher-order **process passing**, by explicit substitutions (the **λ -calculus**)
- ▶ The combination: non-linear, active process mobility in nested locations and local names **needs special care**
- ▶ We give the first bigraphical presentation of this combination, exemplified in the the calculus of **Higher-Order Mobile Embedded Resources (Homer)**

Bigraphical Reactive Systems

- ▶ Bigraphical Reactive Systems (proposed by Robin Milner *et al*)
- ▶ Proposed as a **topographical meta-model** for mobile, distributed agents that can manipulate their own linkages and nested locations.
- ▶ Aiming to **unite calculi** such as λ -calculus, Petri nets, π -calculus, Mobile Ambients etc.
- ▶ A bigraph consists of two structures: the **place graph** and the **link graph**.

A Bigraph



- ▶ inner face outer face
- ▶ A local bigraph $G : (\{y, v\}, \{z\}, \{x\}) \rightarrow (\{m, n\}, \{m\})$
- ▶ Consisting of 2 regions and 3 sites (holes)
- ▶ The ordered ports on nodes can be binding or free
- ▶ We can compose two bigraphs, if their interfaces match

Syntax of asynchronous Homer σ

Assume

- ▶ infinite set of **names** \mathcal{N} ranged over by m and n , and let \tilde{n} range over finite sets of names.
- ▶ let δ range over non-empty sequences of names (called **addresses**)
- ▶ and let $\varphi[r]_{\tilde{n}}$ be a shorthand for locations: $\bar{\delta}\langle r \rangle_{\tilde{n}}$ or $\delta[r]_{\tilde{n}}$

Processes: $p, q, r ::= \mathbf{0} \mid \pi . p \mid p \parallel q \mid (n)p \mid$
 $p[x := q : \tilde{n}] \mid x \mid \bar{\delta}\langle r \rangle_{\tilde{n}} \mid \delta[r]_{\tilde{n}}$

Prefixes: $\pi ::= \delta(x) \mid \bar{\delta}(x)$

Typing asynchronous Homer σ processes

- ▶ We define the **valid typing judgements** of the form $\tilde{x} \vdash p : \tilde{n}$ (we present only some of the rules)

$$\frac{\tilde{x}x \vdash p : \tilde{n} \quad \vdash q : \tilde{m}}{\tilde{x} \vdash p[x := q : \tilde{m}] : \tilde{n} \cup \tilde{m}}$$

$$\frac{\tilde{x} \vdash p : \tilde{n}n}{\tilde{x} \vdash (n)p : \tilde{n}}$$

$$\frac{\tilde{x} \vdash r : \tilde{m}}{\tilde{x} \vdash \varphi[r]_{\tilde{m}} : \tilde{m} \cup \text{fn}(\varphi)}$$

Consequence:

- ▶ A process p is **well-typed** wrt. a finite set of variables \tilde{x} and names \tilde{n} iff the $\text{fn}(p) \subseteq \tilde{n}$ and $\text{fv}(p) \subseteq \tilde{x}$, and
- ▶ for every sub-term $\varphi[r]_{\tilde{m}}$ and $q[x := r : \tilde{m}]$ in p we have that r can be typed with the type \tilde{m} .

Semantics of asynchronous Homer σ (by example)

- ▶ The reaction relation relates processes with the same top-level type $\vdash p \searrow_{\sigma} p' : \tilde{n}'$
- ▶ We can **send** $\vdash \bar{a}\langle r \rangle_{\tilde{n}} \parallel \delta(x) . p \searrow_{\sigma} p[x := r : \tilde{n}] : \tilde{n}'$
- ▶ We can **take** $\vdash a[r]_{\tilde{n}} \parallel \bar{\delta}(x) . p \searrow_{\sigma} p[x := r : \tilde{n}] : \tilde{n}'$
- ▶ But **active** locations permit reactions

$$\vdash r \searrow_{\sigma} r' : \tilde{n} \text{ implies } \vdash a[r]_{\tilde{n}} \searrow a[r']_{\tilde{n}} : \tilde{n}'$$

- ▶ We can **compose addresses**

$$\vdash \bar{a}b\langle r \rangle_{\tilde{n}} \parallel a[b(x) . q \parallel q']_{\tilde{n}'} \searrow_{\sigma} a[q[x := r : \tilde{n}] \parallel q']_{\tilde{n}' \cup \tilde{n}} : \tilde{n}''$$

and dually

$$\vdash a[b[r]_{\tilde{n}} \parallel p]_{\tilde{n}'} \parallel \bar{a}b(x) . q \searrow_{\sigma} a[p]_{\tilde{n}'} \parallel q[x := r : \tilde{n}] : \tilde{n}''$$

Semantics of asynchronous Homer σ cont.

- ▶ We must extend the scope of n **vertically**, through the location boundary, iff the resource r contains the name n free (meaning, if $n \in \tilde{n}$),

$$\vdash a[(n)(b[r]_{\tilde{n}} \parallel p)]_{\tilde{n}'} \parallel \overline{ab}(x) \cdot q \searrow_{\sigma} (n)(a[p]_{\tilde{n}' \cup n} \parallel q[x := r : \tilde{n}]) : \tilde{n}''$$

- ▶ We handle the **explicit substitutions** in standard manner

$$\text{(apply}\sigma\text{)} \quad \vdash \mathcal{C}(x)[x := r : \tilde{n}] \searrow_{\sigma} \tilde{n} \odot \mathcal{C}(r)[x := r : \tilde{n}] : \tilde{n}' ,$$

if \mathcal{C} does not bind x or the names in \tilde{n}

$$\text{(garbage}\sigma\text{)} \quad \vdash p[x := q : \tilde{n}] \searrow_{\sigma} p : \tilde{n}' , \text{ if } x \notin fv(p)$$

Intuition

- ▶ Represent the different term **constructors** using different **controls** in bigraphs
- ▶ Represent the **abstract syntax tree** of a term using nesting of nodes, each having the appropriate control, and **names** and **variables** using links
- ▶ As type annotations are sets, we need a way to associate an arbitrary number of names to a place in an **unordered** way (ports on nodes are **ordered**)
- ▶ Desired properties: **static** and **operational** correspondence

Translation

We define the **translation** of a Homer σ -term p inductively in the inference of $\tilde{x} \vdash p : \tilde{n}$ (we present only some of the cases)

$$\begin{aligned} \llbracket \tilde{x} \vdash \mathbf{0} : \tilde{n} \rrbracket &= \tilde{n} \overline{\oplus} \tilde{x} \\ \llbracket \tilde{x} \vdash (n)p : \tilde{n} \rrbracket &= /n \circ (\llbracket \tilde{x} \vdash p : \tilde{n}n \rrbracket) \\ \llbracket \tilde{x} \vdash \delta[r]_{\tilde{n}'} : \tilde{n}' \cup fn(\delta) \rrbracket &= (\mathbf{loca}_{\delta} \overline{\oplus} \mathbf{id}_{\tilde{n}', \tilde{x}})(\llbracket \tilde{x} \vdash r : \tilde{n}' \rrbracket \mid (\mathbf{ann} \overline{\oplus} \mathbf{id}_{\tilde{n}'})\llbracket \tilde{n}' \rrbracket) \\ \llbracket \tilde{x} \vdash \overline{\delta}(x) . p : \tilde{n} \cup fn(\delta) \rrbracket &= (\mathbf{take}_{\delta(x)} \overline{\oplus} \mathbf{id}_{\tilde{n}, \tilde{x}})\llbracket \tilde{x}x \vdash p : \tilde{n} \rrbracket \end{aligned}$$

and **type annotations** as follows: $\llbracket \tilde{n} \rrbracket = \prod_{n \in \tilde{n}} \mathbf{tname}_n$.

Important points:

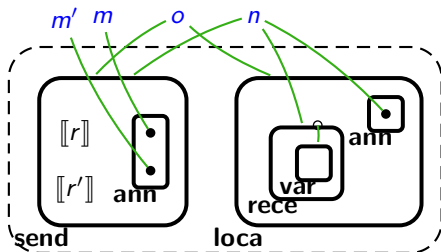
- ▶ No control for neither $\mathbf{0}$ nor (n) (ensure static correspondence)
- ▶ Instead we encode **restriction** using the **closure operator**

Translation of $\overline{on}\langle r \parallel r' \rangle_{\{m, m'\}} \parallel o[n(x) \cdot x]_{\{n\}}$

Example (The Translation)

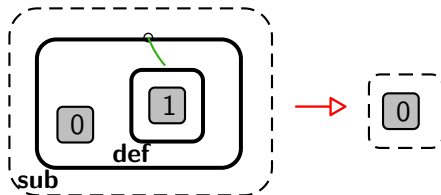
Example on **translation** of the term

$\overline{on}\langle r \parallel r' \rangle_{\{m, m'\}} \parallel o[n(x) \cdot x]_{\{n\}}$ into a bigraph



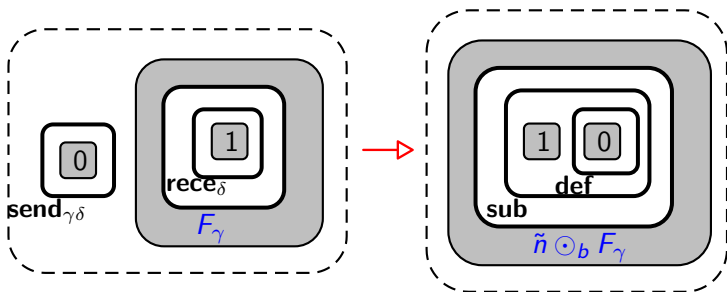
Translating a Reaction Rule

- ▶ Recall the **garbage rule**
 $\vdash p[x := q : \tilde{n}] \searrow_{\sigma} p : \tilde{n}'$, if $x \notin fv(p)$
- ▶ In the bigraph **term language**
 $R = (\mathbf{sub}_{(x)} \oplus \mathbf{id}_{\tilde{n}'}) (\mathbf{id}_{\tilde{n}'} \mid (\mathbf{def}_x \oplus \mathbf{id}_{\tilde{n}}))$,
 $R' = \mathbf{id}_{\tilde{n}'}$,
 $\eta = \{0 \mapsto 0\}$
- ▶ **Visually** we have (eliding the outer names and inner names)



A More Complicated Rule (if time allow)

- ▶ A visual sketch of the **send rule** (the send and take rule are parametrised over **contexts**)



- ▶ Done in syntax by the use of **rule schemes**
- ▶ Cannot be an **evaluation context**, since we may need to update it, and since it has to have a certain form

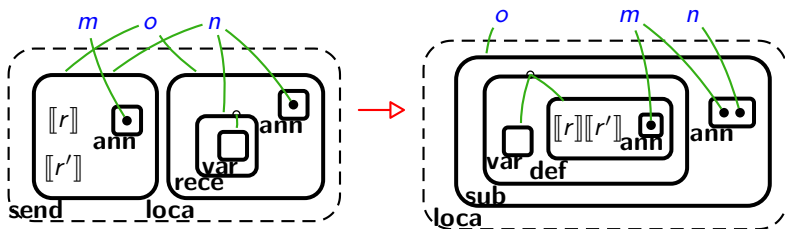
An Example

Example (Mimicking Reactions)

We consider the following reaction (top-level types omitted).

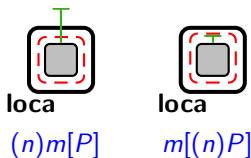
$$\overline{on}\langle r \parallel r' \rangle_{\{m\}} \parallel o[n(x) . x]_{\{n\}} \searrow_{\sigma} o[x[x := (r \parallel r') : \{m\}]]_{\{n,m\}}$$

which we can **mimic** in bigraphs as



Problems with the Location of a Link

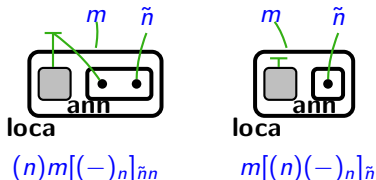
- ▶ $(n)m[P]$ and $m[(n)P]$ ($n \neq m$) are **not** structural congruent, since this would lead to problems, when mobile processes may be **copied**
- ▶ **without** the type annotations, however, the two processes will give rise to **isomorphic** bigraphs



- ▶ since the closed link, representing the **restriction** constructor, has **no location**

Problems with the Location of a Link — Solution

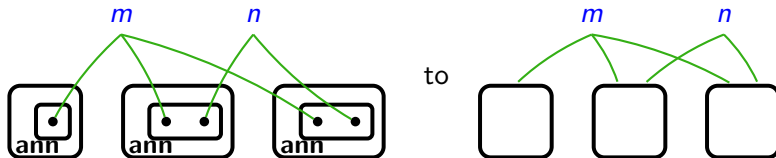
- ▶ **Static solution:** localisation using the type annotations



- ▶ **Dynamic solution:** keep **explicit** track of free names in the parametric reaction rules. We require that the interface of the parameter must **equal** the interface of the annotation

Proposed Extension — Localised Links

- ▶ Intuitively, we allow for that an arbitrary number of names can be associated to a place in an **unordered** way
- ▶ Useful for representing the **type annotations** succinctly



- ▶ A name can both be used as a **regular link** and as a **localised link**, as in e.g. $m[p]_{\{m\}}$

Main Results and Conclusions

Main results: the following **correspondences**:

- ▶ **Structural** correspondence:

$$\tilde{x} \vdash p \equiv_{\sigma} q : \tilde{n} \text{ if and only if } \llbracket \tilde{x} \vdash p : \tilde{n} \rrbracket = \llbracket \tilde{x} \vdash q : \tilde{n} \rrbracket$$

- ▶ **Operational** correspondence:

$$\vdash p \searrow_{\sigma} p' : \tilde{n} \text{ if and only if } \llbracket \vdash p : \tilde{n} \rrbracket \rightarrow \llbracket \vdash p' : \tilde{n} \rrbracket .$$

Conclusions:

- ▶ **Presentation** of a higher-order calculus with non-linear active process mobility and local names in bigraphs
- ▶ In calculi with non-linear active process mobility and local names, it is important to keep **explicit track of free names** in the reaction rules, and
- ▶ to **localise** closed free links (representing names) explicitly

Future Work

Future Work:

- ▶ Examine the **LTS bisimulation congruence** derivable using the general theory of bigraphs
- ▶ Examine the extension of **localised links**, if it retains relative pushouts and examine its expressive power
- ▶ Proof techniques of bigraphs: **up-to proof technique** known from process calculi
- ▶ Examine the connection between **sortings** and **logics** for bigraphs, and see if we can enforce a more strict **control** with the movement and locations of closed free links