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.
There already exists bigraphical presentations of:

- local names and scoping (the $\pi$-calculus)
- active processes in nested locations (Mobile Ambients)
- non-linear higher-order process passing, by explicit substitutions (the $\lambda$-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 $\lambda$-calculus, Petri nets, $\pi$-calculus, Mobile Ambients etc.
- A bigraph consists of two structures: the **place graph** and the **link graph**.
A Bigraph

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$\sigma$

Assume

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

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

Prefixes: $\pi ::= \delta(x) \mid \overline{\delta}(x)$
Typing asynchronous Homer$_\sigma$ processes

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

\[
\begin{align*}
\tilde{x} & \vdash p : \tilde{n} \quad \vdash q : \tilde{m} \quad \tilde{x} \vdash p : \tilde{n} \cup \tilde{m} \\
\tilde{x} & \vdash p[x := q : \tilde{m}] : \tilde{n} \cup \tilde{m} \quad \tilde{x} \vdash (n)p : \tilde{n} \\
\tilde{x} & \vdash r : \tilde{m} \quad \tilde{x} \vdash \varphi[r]_{\tilde{m}} : \tilde{m} \cup fn(\varphi)
\end{align*}
\]

Consequence:

- A process $p$ is well-typed wrt. a finite set of variables $\tilde{x}$ and names $\tilde{n}$ iff the $fn(p) \subseteq \tilde{n}$ and $fv(p) \subseteq \tilde{x}$, 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$_\sigma$ (by example)

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

- We can compose addresses
  
  $\vdash \overline{ab}\langle r\rangle_{\tilde{n}} \parallel a[b(x) \cdot q \parallel q']_{\tilde{n}'} \triangleleft_{\alpha\sigma} a[q[x := r : \tilde{n}] \parallel q']_{\tilde{n}' \cup \tilde{n}} : \tilde{n}''$

and dually

$\vdash a[b[r]_{\tilde{n}} \parallel p]_{\tilde{n}'} \parallel \overline{ab}(x) \cdot q \triangleleft_{\alpha\sigma} a[p]_{\tilde{n}'} \parallel q[x := r : \tilde{n}] : \tilde{n}''$
We must extended the scope of $n$ vertically, through the location boundary, iff the resource $r$ contains the name $n$ free (meaning, if $n \in \tilde{n}$),

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

We handle the explicit substitutions in standard manner

$$(\text{apply}\sigma) \quad \vdash C(x)[x := r : \tilde{n}] \triangleleft_{\sigma} \tilde{n} \circ C(r)[x := r : \tilde{n}] : \tilde{n}'$$, if $C$ does not bind $x$ or the names in $\tilde{n}$

$$(\text{garbage}\sigma) \quad \vdash p[x := q : \tilde{n}] \triangleleft_{\sigma} p : \tilde{n}'$$, if $x \notin \text{fv}(p)$
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$\sigma$-term $p$ inductively in the inference of $\tilde{x} \vdash p : \tilde{n}$ (we present only some of the cases)

\[
\begin{align*}
[\tilde{x} \vdash 0 : \tilde{n}] &= \tilde{n} \ominus \tilde{x} \\
[\tilde{x} \vdash (n)p : \tilde{n}] &= /n \circ ([\tilde{x} \vdash p : \tilde{nn}]) \\
[\tilde{x} \vdash \delta[r] \tilde{n}' : \tilde{n}' \cup fn(\delta)] &= (\text{lca}_{\delta} \ominus \text{id}_{\tilde{n}',\tilde{x}})([\tilde{x} \vdash r : \tilde{n}'] | (\text{ann} \ominus \text{id}_{\tilde{n}'})[\tilde{n}']) \\
[\tilde{x} \vdash \delta(x) \cdot p : \tilde{n} \cup fn(\delta)] &= (\text{take}_{\delta(x)} \ominus \text{id}_{\tilde{n},\tilde{x}})[\tilde{x} \vdash p : \tilde{n}]
\end{align*}
\]

and type annotations as follows: $[\tilde{n}] = \{\text{tname}_{n} \mid n \in \tilde{n}\}$.

Important points:

- No control for neither $0$ nor $(n)$ (ensure static correspondence)
- Instead we encode restriction using the closure operator
Translation of \( \overline{on} \langle r \parallel r' \rangle \{m,m'\} \parallel o[n(x).x]\{n\} \)

Example (The Translation)

Example on translation of the term \( \overline{on} \langle r \parallel r' \rangle \{m,m'\} \parallel o[n(x).x]\{n\} \) into a bigraph
Translating a Reaction Rule

- Recall the garbage rule
  \[ \vdash p[x := q : \tilde{n}] \xrightarrow{\sigma} p : \tilde{n}' \text{, if } x \not\in \text{fv}(p) \]

- In the bigraph term language
  \[ R = (\text{sub}_x \oplus \text{id}_{\tilde{n}'}) (\text{id}_{\tilde{n}'} | (\text{def}_x \oplus \text{id}_{\tilde{n}})), \]
  \[ R' = \text{id}_{\tilde{n}'}, \]
  \[ \eta = \{0 \mapsto 0\} \]

- Visually we have (eliding the outer names and inner names)

\[ \text{sub} \quad \text{def} \]

\[ \text{0} \quad \text{1} \]

\[ \rightarrow \]

\[ \text{0} \]
A More Complicated Rule (if time allow)

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

\[
\begin{array}{c}
\text{send}_{\gamma\delta} \\
\text{rece}_\delta \\
F_\gamma \\
\end{array}
\rightarrow
\begin{array}{c}
\text{sub} \\
\text{def} \\
\tilde{n} \otimes b F_\gamma \\
\end{array}
\]

- 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
Example (Mimicking Reactions)

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

\[
\overline{on}\langle r \parallel r'\rangle_m \parallel o[n(x) \cdot x]_n \searrow_{\sigma} o[x[x := (r \parallel r') : \{m\}]]_{n,m}
\]

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**: localization using the type annotations

  \[ \text{Static solution: } \text{localisation using the type annotations} \]

  ![Diagram showing static solution](image)

  \[ (n)m[(-)_n]\tilde{n} \]

- **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

  ![Diagram showing dynamic solution](image)

  \[ m[(n)(-)_n]\tilde{n} \]
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:**
  \[
  \vdash p \downarrow_\sigma p' : \tilde{n} \text{ if and only if } \llbracket \vdash p : \tilde{n} \rrbracket \rightarrow \llbracket \vdash p' : \tilde{n} \rrbracket.
  \]

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:

- 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.