An EMOF-Compliant Abstract Syntax for Bigraphs

Timo Kehrer, Christos Tsigkanos and Carlo Ghezzi
Politecnico di Milano, Italy

Graphs as Models (GaM) @ ETAPS 2016
The idea of bigraphs:

- Fundamental concepts: locality (placing) and connectivity (linking)

Constituents of a bigraph:

- **Place graph**: Forest defined over a set of nodes representing entities in terms of a containment structure.
- **Link graph**: Hypergraph composed over the same set of nodes representing arbitrary linking among entities.
Bigraphs and Bigraphical Reactive Systems

The idea of bigraphs:

- Fundamental concepts: locality (placing) and connectivity (linking)
- Constituents of a bigraph:
  - Place graph: Forest defined over a set of nodes representing entities in terms of a containment structure.
  - Link graph: Hypergraph composed over the same set of nodes representing arbitrary linking among entities.

Bigraphical reactive systems:

- Possible local reconfigurations expressed as a set of reaction rules.
- Rewriting rules which consist of two bigraphs; redex and reactum.
- Redex specifies a bigraphical pattern whose occurrence found in a “host bigraph” enables replacement by the reactum.
Bigraphs as Domain-specific Language for Simulation and Development

What’s readily available...

- Fundamental modeling formalism
- Rich theory, shown to be general enough as a meta-calculus (can embed existing process calculi)
- Precise visual syntax and algebraic notation (textual syntax)

What’s missing...

- Bigraphs yet lack a definition of an abstract syntax
- Available research prototypes use proprietary formats and are thus poorly integrated
- Hampers the exchange of models across tool boundaries and the development of sophisticated tool chains
Bigraphs as Domain-specific Language for Simulation and Development

What’s readily available...

- Fundamental modeling formalism
- Rich theory, shown to be general enough as a meta-calculus (can embed existing process calculi)
- Precise visual syntax and algebraic notation (textual syntax)

What’s missing...

- Bigraphs yet lack a definition of an abstract syntax
- Available research prototypes use proprietary formats and are thus poorly integrated
- Hampers the exchange of models across tool boundaries and the development of sophisticated tool chains
Outline

Preliminaries:

- The Idea of Bigraphs
- Graph-based Representation of EMOF Models

Contributions:

- A Canonical Mapping of Bigraphs to Typed Graphs
- Handling of Application-specific Variability
Bigraph Model of a Context-aware Printing System

- An office environment comprised of two rooms
- One room contains a computer and a printer
- The other room contains a user who holds a print job
- A user can submit the job for printing through the computer connected to the printer

Bigraph (visual syntax):
The Anatomy of Bigraphs

Place = Root or Node or Site
Link = Edge or Outer Name
Point = Port or Inner Name

Signature:
{Job:0, User:1, Room:1, Spool:1, Printer:2, Computer:1}

Place Graph:

Link Graph:
Graph-based Representation of EMOF Models

Natural formalization of object-oriented principles:

- Models as directed, unlabelled, typed graph (ASG)
- Types of nodes and edges are defined by a type graph (Meta-model);
  Special graph which includes the definition of
  - an inheritance hierarchy including abstract node types,
  - a containment structure,
  - opposite edges representing bidirectional edge types,
  - multiplicities attached to edge types.
Basic Type Graph Modeling the Anatomy of Bigraphs

Place graph structure

- **BPlace**
  - bChild: 0..*
  - bPoint: 0..1

- **BNode**
  - bNode: 1..1

- **BRoot**
  - index: int

Link graph structure

- **BPoint**
  - bPoints: 1..*

- **BPort**
  - index: int
  - bPorts: 0..*

- **BInnerName**

- **BEdge**

- **BOuterName**
Control-compatible Extension of the Basic Type Graph (w.r.t. a given bigraph signature)

**General principle:**

For each control defined by the signature, we introduce a corresponding subtype of the generic node type $BNode$. 
Control-compatible Extension of the Basic Type Graph (w.r.t. a given bigraph signature)

**General principle:**

For each control defined by the signature, we introduce a corresponding subtype of the generic node type $BNode$.

**Example:**
Handling of Arities defined by a Signature

Generic well-formedness rule:

For each subtype of BNode, restrict the multiplicity [0..∗] defined by the generic edge type bPorts to the fixed arity value.
Handling of Arities defined by a Signature

Generic well-formedness rule:
For each subtype of \textit{BNode}, restrict the multiplicity \([0..\ast]\) defined by the generic edge type \textit{bPorts} to the fixed arity value.

Example:
Mapping Bigraphs to Typed Graphs

**Basic element mapping:**

Bijectively map the elements of a bigraph $B$ to the nodes of a typed graph $G$, with nodes in $G$ properly typed.

**Additional soundness criteria:**

Additional soundness criteria defined over the basic element mapping such that mapping induces a unique transformation from bigraphs to typed graphs (and vice versa).
Basic Element Mapping

Mapping of bigraph roots
Basic Element Mapping

Mapping of bigraph nodes
Basic Element Mapping

Mapping of sites

```
: BSite
  index = 0

: Room
  bChld

: BPort
  index = 0
  bPoints
  bNode

: BPort
  index = 0
  bPoints
  bNode

: BPort
  index = 0
  bPoints
  bNode

: BRoot
  index = 0
  bPoints
  bNode

jeff : BOuterName

v2 : Room
  bPrnt
  bChld

v4 : Room
  bPrnt
  bChld

v5 : User
  bPrnt
  bPoints
  bNode

v6 : Job
  bPrnt
  bPoints
```

13
Basic Element Mapping

Mapping of ports
Basic Element Mapping

Mapping of names
Basic Element Mapping

Mapping of bigraph edges
Additional Soundness Criteria

Nesting of places in $B$ must coincide with the containment structure in $G$
Additional Soundness Criteria

Linking structure in $B$ must coincide with the linking structure in $G$.
Additional Soundness Criteria

Indexing of roots in $B$ must be consistent to indices in $G$
Indexing of sites in $B$ must be consistent to indices in $G$
Additional Soundness Criteria

Indexing of ports in $B$ must be consistent to ownership and indices in $G$
Definition of an appropriate abstract syntax depends on several design decisions, many of them being application-specific.

**Variability model:**

```
Bigraphical Abstract Syntax (AS)

Typing (T)                   Root Indices (RI)     Site Indices (SI)
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Strongly Typed (ST)</td>
<td>Explicit Roots (ER)</td>
<td>Explicit Sites (ES)</td>
</tr>
<tr>
<td>Weakly Typed (WT)</td>
<td>Root Indices (RI)</td>
<td>Site Indices (SI)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Concrete F                  Abstract F

Mandatory ❌                 Optional ○

Alternative ▲
```
Implementing Type-level Variability

150% type graph:
Implementing Instance-level Variability

Delta-oriented approach:

- Core variant: canonical graph representation derived from a given bigraph
- Alternative variants: Obtained from the core variant by applying a set of deltas

Example:
Delta \( \Delta(ST, WT) \) to switch from the strongly typed variant to the weakly typed one:

- Application condition: \( WT \)
- Specification: Each node representing a bigraphical node has to be retyped to the generic node type \( BNode \) and the value of the attribute \( control \) has to be set accordingly.
Implementing Instance-level Variability

Delta-oriented approach:

- Core variant: canonical graph representation derived from a given bigraph
- Alternative variants: Obtained from the core variant by applying a set of deltas

Example:

Delta $\Delta(ST, WT)$ to switch from the strongly typed variant to the the weakly typed one:

- Application condition: $WT$
- Specification: Each node representing a bigraphical node has to be retyped to the generic node type $BNode$ and the value of the attribute $control$ has to be set accordingly.
Tool Integration and Example Application

Bigraph modeling environment

Big Red

Eclipse Modeling Framework (EMF)

Example application:

Additional constraint checking facility based on the OCL

Example invariant:

A Spool may only contain Jobs and Sites as nested places.

Specification in OCL:

context Spool
inv:

\[ \text{self.bChld} \rightarrow \text{forall}(c \mid c.oclIsTypeOf(BSite) \text{ or } c.oclIsTypeOf(Job)) \]
Future Work

Conceptually:

- Formalize the relation between bigraphical reactive systems and graph transformation systems
- Extension to other forms of bigraphs, e.g. bigraphs with sharing
Future Work

Conceptually:

- Formalize the relation between bigraphical reactive systems and graph transformation systems
- Extension to other forms of bigraphs, e.g. bigraphs with sharing

Technically:

- Provide a reference implementation based on Eclipse and EMF
Summary

Goals:

- Abstract syntax for bigraphs which is compliant with the EMOF standard defined by the OMG
- Facilitates:
  - Interoperability of bigraphical modeling and analysis tools
  - Integration with mainstream MDE technologies
Summary

Goals:

- Abstract syntax for bigraphs which is compliant with the EMOF standard defined by the OMG
- Facilitates:
  - Interoperability of bigraphical modeling and analysis tools
  - Integration with mainstream MDE technologies

Approach:

- Typed graphs as a formal underpinning of EMOF-based models
- Canonical mapping which maps bigraphs to typed graphs in a natural way
- Handling of application-specific variability using standard techniques from SPL engineering