



University
of Glasgow

Bigraphs: From Category Theory to Digital Twins

Michele Sevegnani

5th October 2025, TU Dresden

WORLD
CHANGING
GLASGOW

A WORLD
TOP 100
UNIVERSITY



Outline and format

This talk is organised in **three** parts

1. A primer on the theory of bigraphs
2. BigraphER
3. Two recent applications

It should be accessible to **non-experts** in bigraphs

Please feel free to ask questions or give comments during the talk!

All the work presented today is in **collaboration** with my coauthors



University
of Glasgow

Part 1: Theory





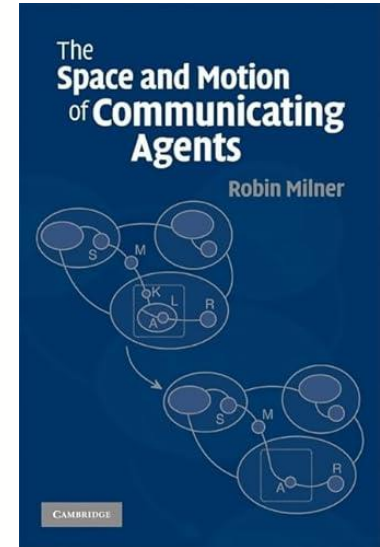
Bigraphs history

Process Algebra

- π -calculus: non-local names mobility
- Mobile Ambients: space

How can we unify process calculi?

- Physical spatially related contexts
- Non-local computational/interaction contexts
- How do we model such systems?

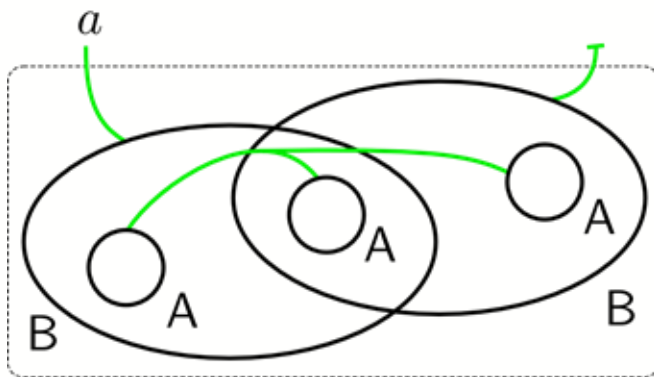


Robin Milner

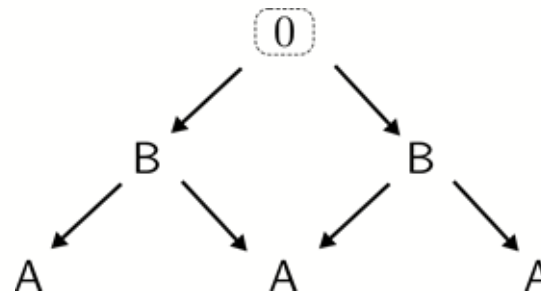


Bigraph structure

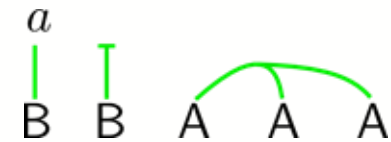
1. Bigraph: superimposition of a **place graph** and a **link graph**
2. Place graph (**with sharing**): **DAG** - topological space - no distances - containment relation
3. Link graph: **Hypergraph** - relationships between sets of entities (e.g. communication capabilities)



Bigraph



Place graph

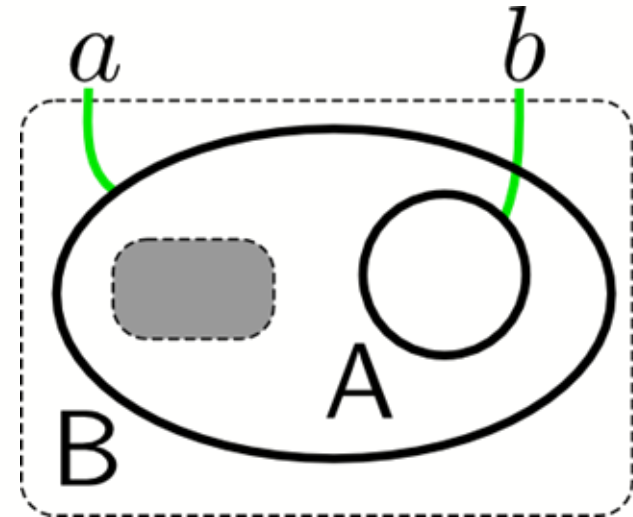


Link graph



Some terminology

- Outer dashed rectangle: **region**
- Grey box: **site**
- A, B: **entities** of type (control) A,B
- Green edges: **links**
- Two entities of the same type are linked to the same number of links
- a,b: outer **names**
- Inner names are displayed below the diagram (not shown here)





Compositionality

1. The **regions** and **outer names** form the **outer interface** of a bigraph
2. The **sites** and **inner names** form the **inner interface** of a bigraph

This allows building larger bigraphs from smaller parts

Powerful: “applied category theory” is mostly “make systems more compositional to make reasoning easier”



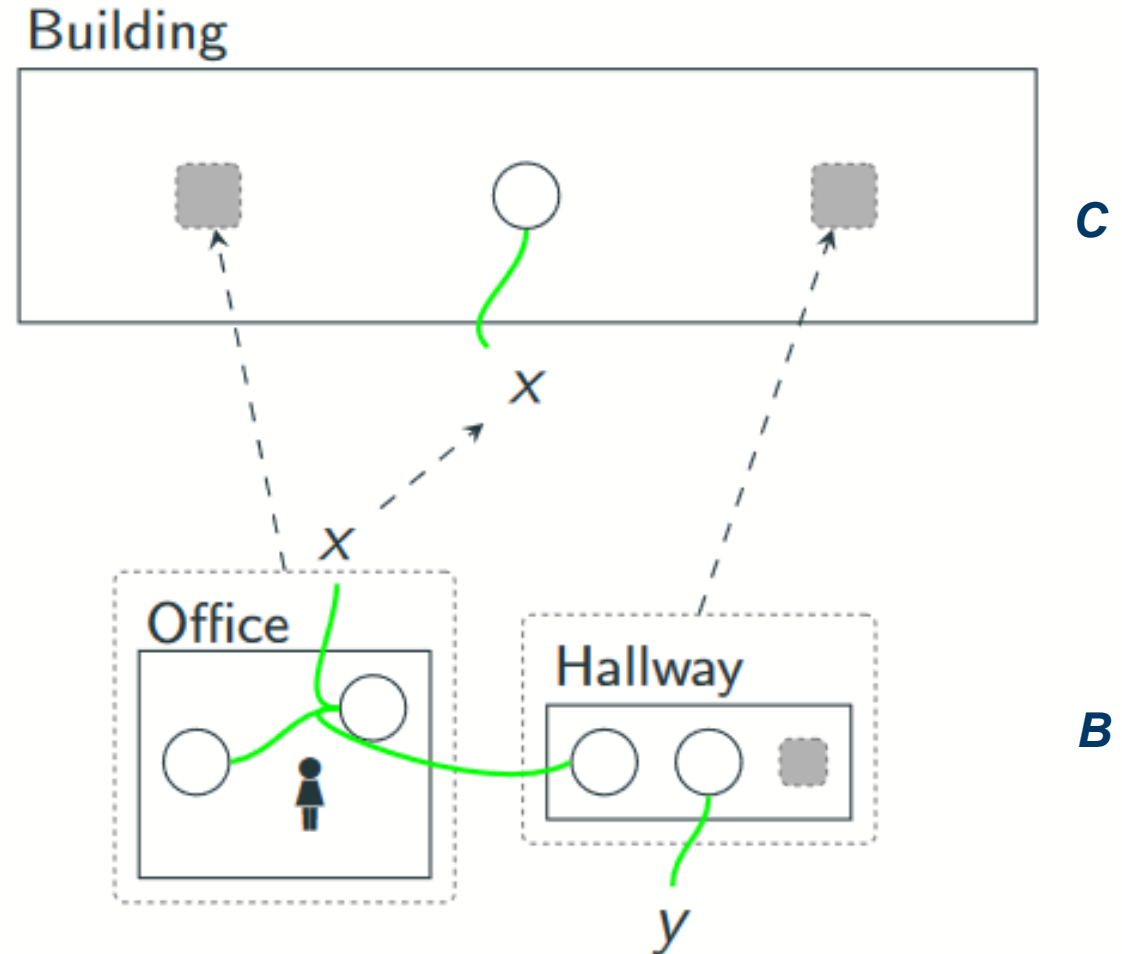
Compositionality – Example

$C \circ B$

1. i -th region of B is inserted in the i -th site of C
2. Links with equal outer names in B and inner names in C are fused

Composition is defined only if C 's inner interface and B 's outer interface are compatible

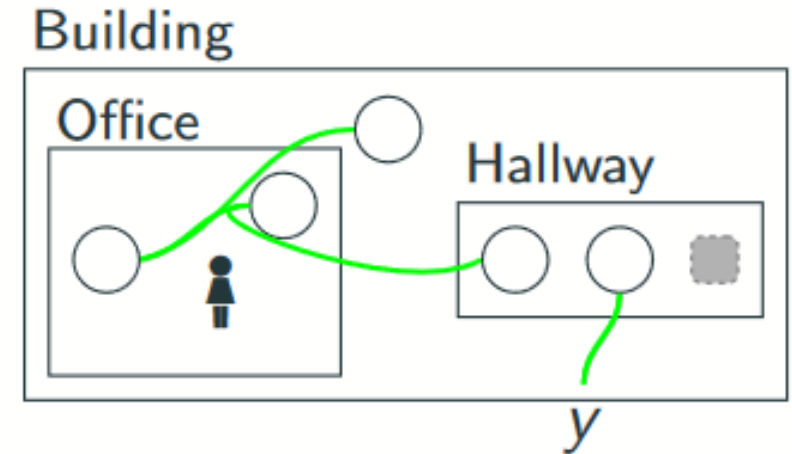
$C: \langle 2, x \rangle \rightarrow \langle 1, \{ \} \rangle$ $B: \langle 1, y \rangle \rightarrow \langle 2, x \rangle$





Compositionality – Example (cont.)

$C \circ B: \langle 1, y \rangle \rightarrow \langle 1, \{ \} \rangle$





Bigraph categories

1. Interfaces are the **objects** of the precategory of **concrete** bigraphs
 - More details in the next slide
 - Recall composition is **partial**
2. The **arrows** in this category are the concrete bigraphs

Departure from GTS

Note place graphs are **self-dual** thanks to sharing



Concrete & abstract bigraphs

1. Entities and closed links can have unique identifiers (**support**)
 - We call these bigraphs **concrete**
 - This is how we represent/manipulate bigraphs on a computer
2. We usually work over **congruence classes** of bigraphs
 - An **abstract** bigraph is a lean-support equivalence class of concrete bigraphs
 - We can turn a concrete bigraph into an abstract one by **discarding** all its **identifiers** and **idle edges**



Bigraph categories (cont.)

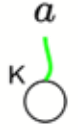
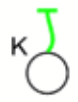
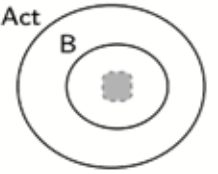
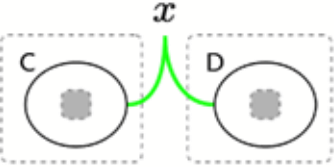
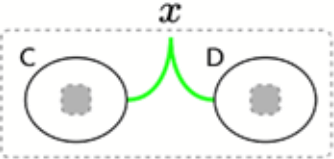
We can also define a horizontal composition called **tensor product**

- It places two bigraphs side-by-side
- Partial since names in the interfaces need to be disjoint

This defines a **partial monoidal precategory**

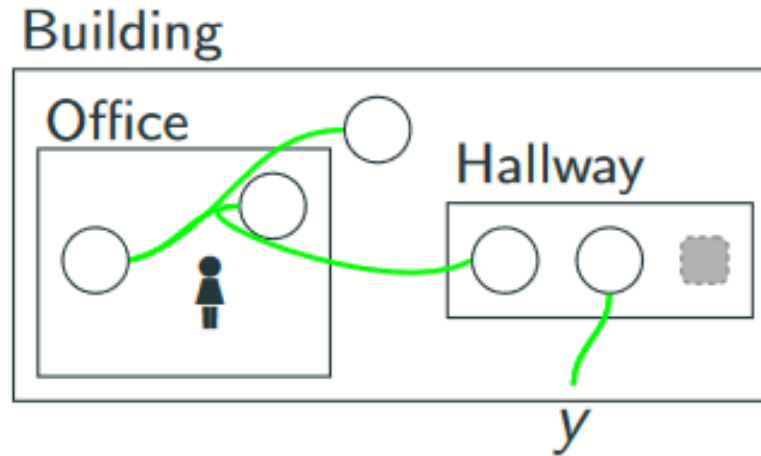
By applying a **lean-support quotient functor** we define the category of abstract bigraphs

Algebraic notation

Component/Operation	Algebraic Form	Diagrammatic Form
Entity of arity 1	K_a	
Name closure	$/a K_a$	
Site	id	
Region	1	
Nesting	$Act.B.id$	
Parallel product	$C_x.id \parallel D_x.id$	
Merge product	$C_x.id \mid D_x.id$	



Algebraic notation – Example



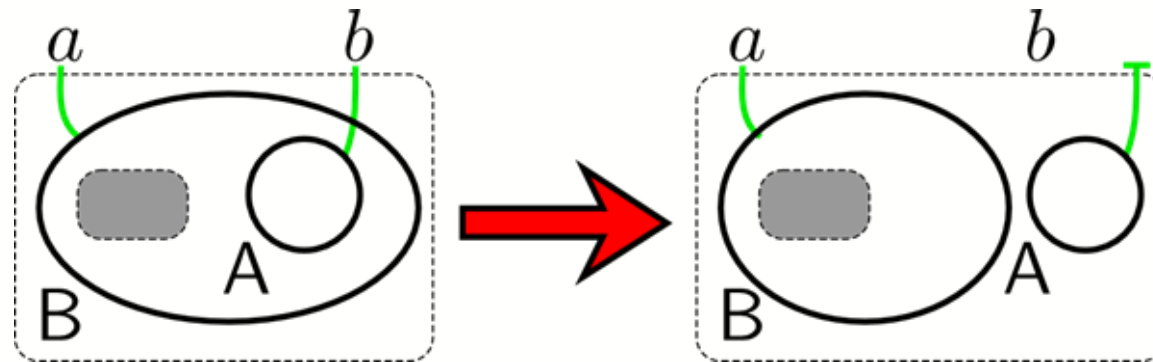
$$/x/y \text{ (Building. (Office. (C}_x \mid \text{User} \mid C_x)) \mid C_x \mid \text{Hallway. (C}_x \mid C_y \mid \text{id})) \mid \text{id}_y \text{)}$$

A complete axiomatisation can be given by the equational theory of **qualitative commutative bialgebras**



Reaction rules

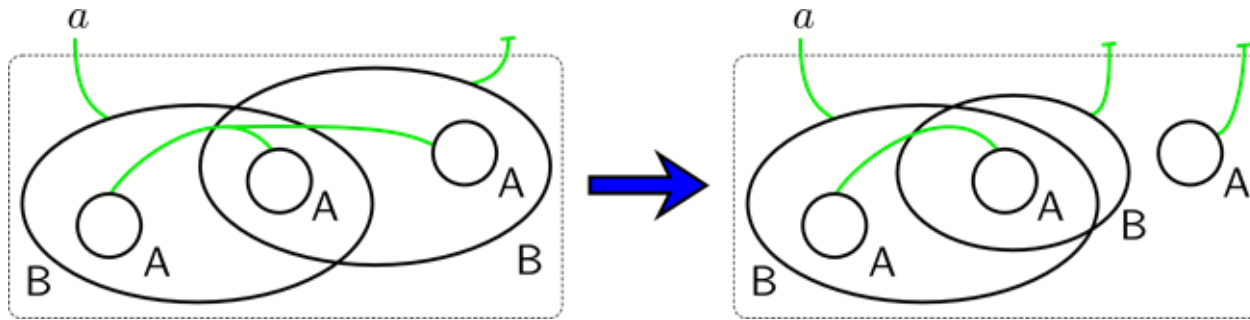
- A set of reaction rules specify the **dynamics** of the system
- How to apply a rule to a bigraph (rewriting):
 1. Identify occurrences of the lhs in the bigraph
 2. Substitute each of them with the rhs



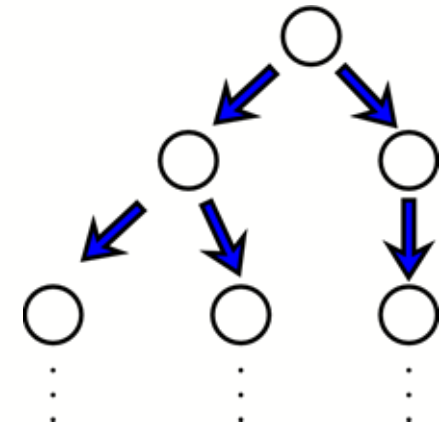


Bigraphical Reactive Systems

- An **initial** bigraph and a **set** of reaction rules
- By performing all the rewriting steps we find all the reachable configurations of the system
- The resulting transition system can be labelled and model checked



Transition



Transition system



Bigraphical Reactive Systems – Extensions

Instantiation maps, i.e. copying/deleting sites

Conditional rewriting

Probabilistic semantics

- DTMC
- CTMC
- MDP



Some relevant papers

- M. Sevegnani and M. Calder. **Bigraphs with sharing**, in *Theoretical Computer Science*, volume 577, pp 43--73, Elsevier, 2015
- B. Archibald et al. **Conditional Bigraphs**, in proceedings of *International Conference on Graph Transformation (ICGT)*, LNCS, volume 12150, pp 3--19, 2020
- B. Archibald et al. **Probabilistic Bigraphs**, in *ACM Formal Aspects of Computing Journal*, volume 34, issue 2, 2022
- C. Marcon et al. **BiCoq : Bigraphs Formalisation with Coq**, in *ACM/SIGAPP Symposium On Applied Computing (Software Verification and Testing Track)*, 2025



University
of Glasgow

Part 2: Tooling





BigraphER

An implementation of Bigraphical Reactive System in **OCaml**

Key features include:

- CLI tool & OCaml APIs
- Correct handling of abstract bigraphs
- Several matching engines (SAT, PB, and sub-iso solver)
- JSON interop
- Probabilistic/stochastic reaction rules, rule priorities and conditional rules
- Instantiation maps



BigraphER (cont.)

- Graphical output (TikZ/graphviz)
- Simulation
- Interactive rewriting
- Exhaustive state space exploration
- Compatible with the PRISM and Storm probabilistic model checker
- Predicate checking and rewards

Several new features being actively developed/tested on the v2 branch

<https://bitbucket.org/uog-bigraph/bigraph-tools/branch/v2>



BigraphER language

Based on the standard **algebraic form** with some **syntactic sugar** for ease of modelling

- Parameterised controls to handle data in models and for pseudo-identifiers
- Priority classes (sometimes more natural control flow than conditional rules)
- Instantaneous rules to hide intermediate states not useful for model analysis

Some basic type checking already supported



BigraphER language – Example

Control definitions (**signature**)

```
##### Controls
```

```
ctrl Locale = 1;  
ctrl Localeattack = 1;  
ctrl Aura = 1;  
fun ctrl Area(i) = 1;
```

```
atomic ctrl Lion = 1;  
atomic ctrl Lionattack = 2;  
atomic ctrl Liongroup = 2;
```

```
atomic ctrl Impala = 1;  
atomic ctrl Impalaseen = 2;  
atomic ctrl Impalaheld = 2;
```

```
atomic ctrl Field = 0;
```

```
atomic fun ctrl GPS(i) = 1;
```

```
atomic ctrl Child = 1;  
atomic ctrl Childattack = 1;  
atomic ctrl Childgroup = 1;
```

BigraphER language – Example (cont.)

Some reaction rules

```
##### Rules of the game

big attack_lhs =
  Child{a} || Locale{m}.(Lion{a} | Impala{b} | id(1));
big attack_rhs =
  /l (Childattack{a} || Localeattack{m}.(Lionattack{a,l} | Impalaseen{b,l} | id(1)));

react attack =
  attack_lhs -> attack_rhs;

react join2 =
  /l (Child{a} || (Lion{a} | Liongroup{a',l} | Liongroup{a'',l} | Impalaheld{b,l}))
  ->
  /l (Childgroup{a}
    || (Liongroup{a,l} | Liongroup{a',l} | Liongroup{a'',l} | Impalaheld{b,l}));

fun react kill(i) =
  /b/l (Childgroup{a} || Childgroup{a'} || Childgroup{a''}
    || Localeattack{m}.(Liongroup{a,l} | Liongroup{a',l} | Liongroup{a'',l}
      | Impalaheld{b,l} | id(1))
    || GPS(i){b})
  ->
  Child{a} || Child{a'} || Child{a''}
  || Locale{m}.(Lion{a} | Lion{a'} | Lion{a''} | id(1))
  || 1;
```



BigraphER language – Example (cont.)

BRS block

```
##### Bigraphical reactive system

brs
  init field;
  # GPS events only for the example trace
  rules = [
    { attack, timeout1, join1, timeout2, join2, kill(0),
      mov2(1, 0, 2, 5), mov2(1, 0, 3, 6),  mov2(0, 1, 5, 7),
      mov2(2, 0, 4, 8),  mov2(0, 2, 8, 9) }
  ];
endbrs
```

More examples at <https://uog-bigraph.bitbucket.io/examples.html>



Some relevant papers

- M. Sevegnani and M. Calder. **BigraphER: rewriting and analysis engine for bigraphs**, in proceedings of *Computer Aided Verification (CAV)*, LNCS, volume 9780 Part II, pp 494--501, Springer, 2016
- B. Archibald et al. **Practical Bigraphs via Subgraph Isomorphism**, in proceedings of *27th International Conference on Principles and Practice of Constraint Programming (CP)*, 2021
- B. Archibald et al. **Practical Modelling with Bigraphs**, in *ACM Formal Aspects of Computing Journal*, 2025



University
of Glasgow

Part 3: Applications

**WORLD
CHANGERS
WELCOME**



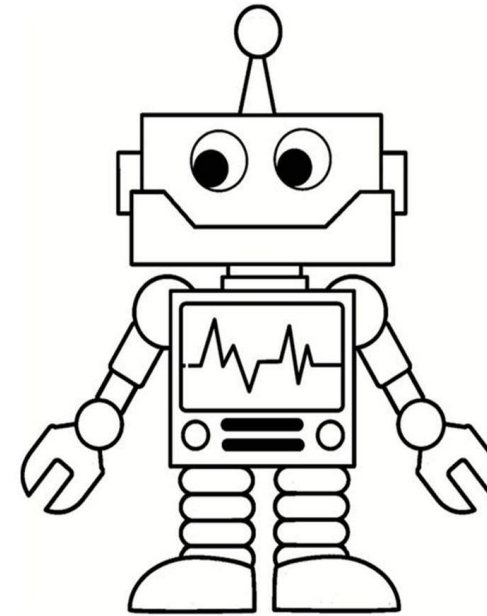


Autonomous agents

An entity

- which **perceives** its situated environment
- which **deliberates** accordingly
- which **takes actions** autonomously

in order to achieve its design objectives



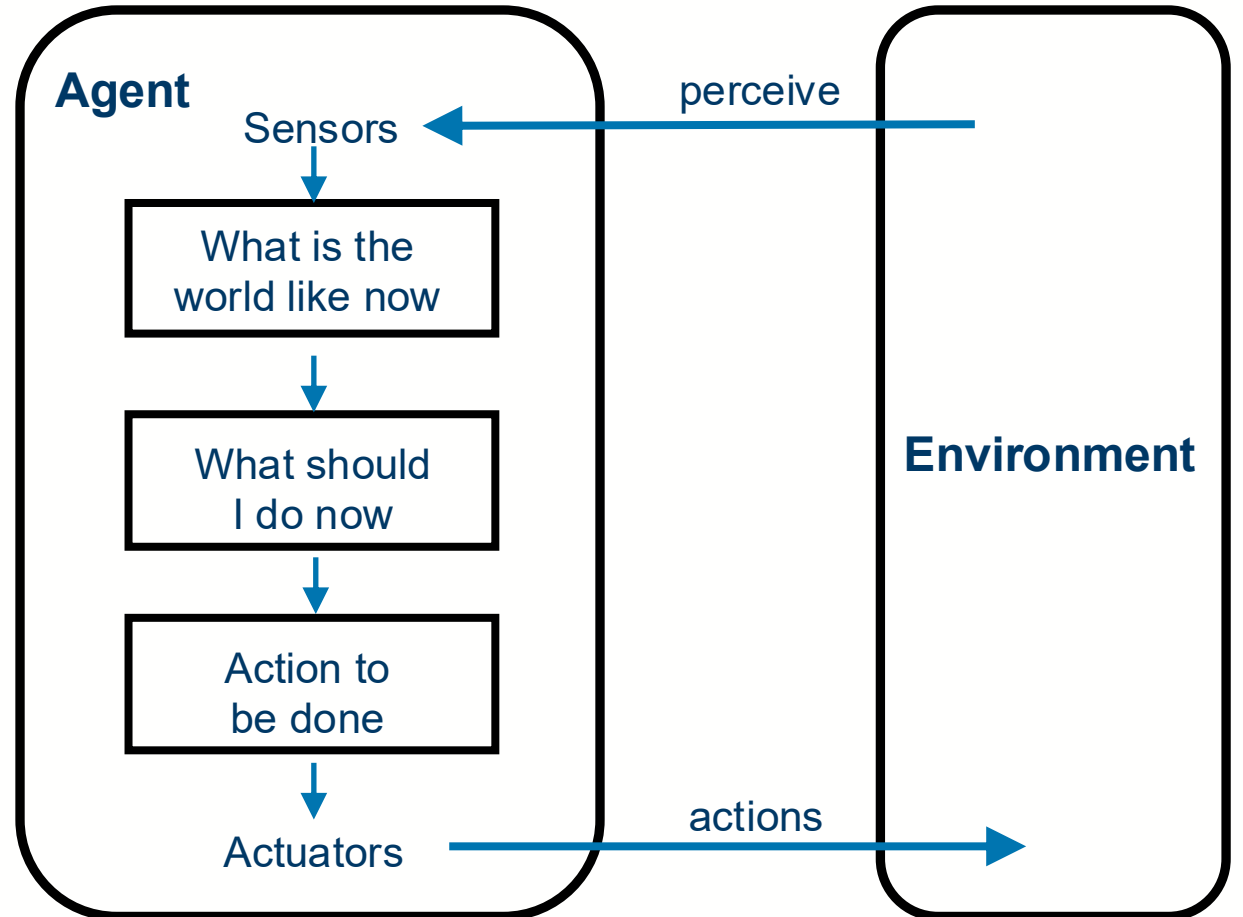


Autonomous agents

An entity

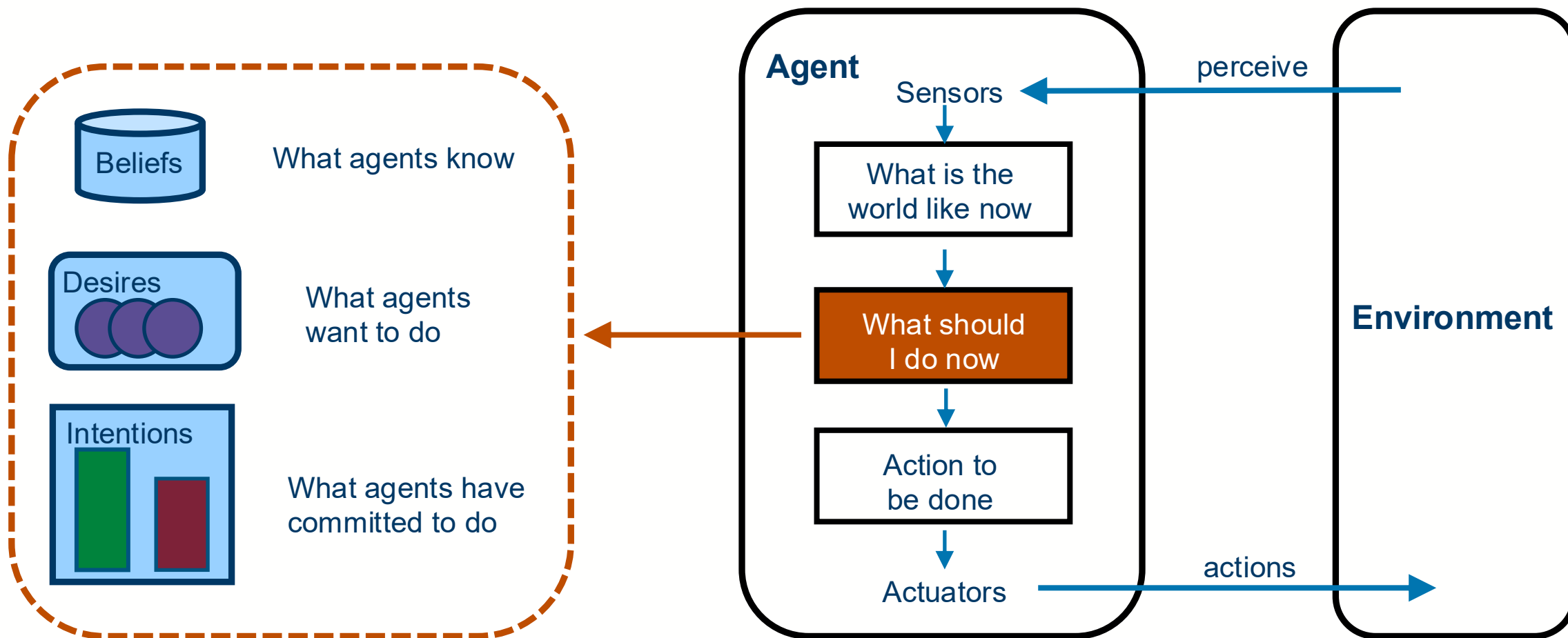
- which **perceives** its situated environment
- which **deliberates** accordingly
- which **takes actions** autonomously

in order to achieve its design objectives



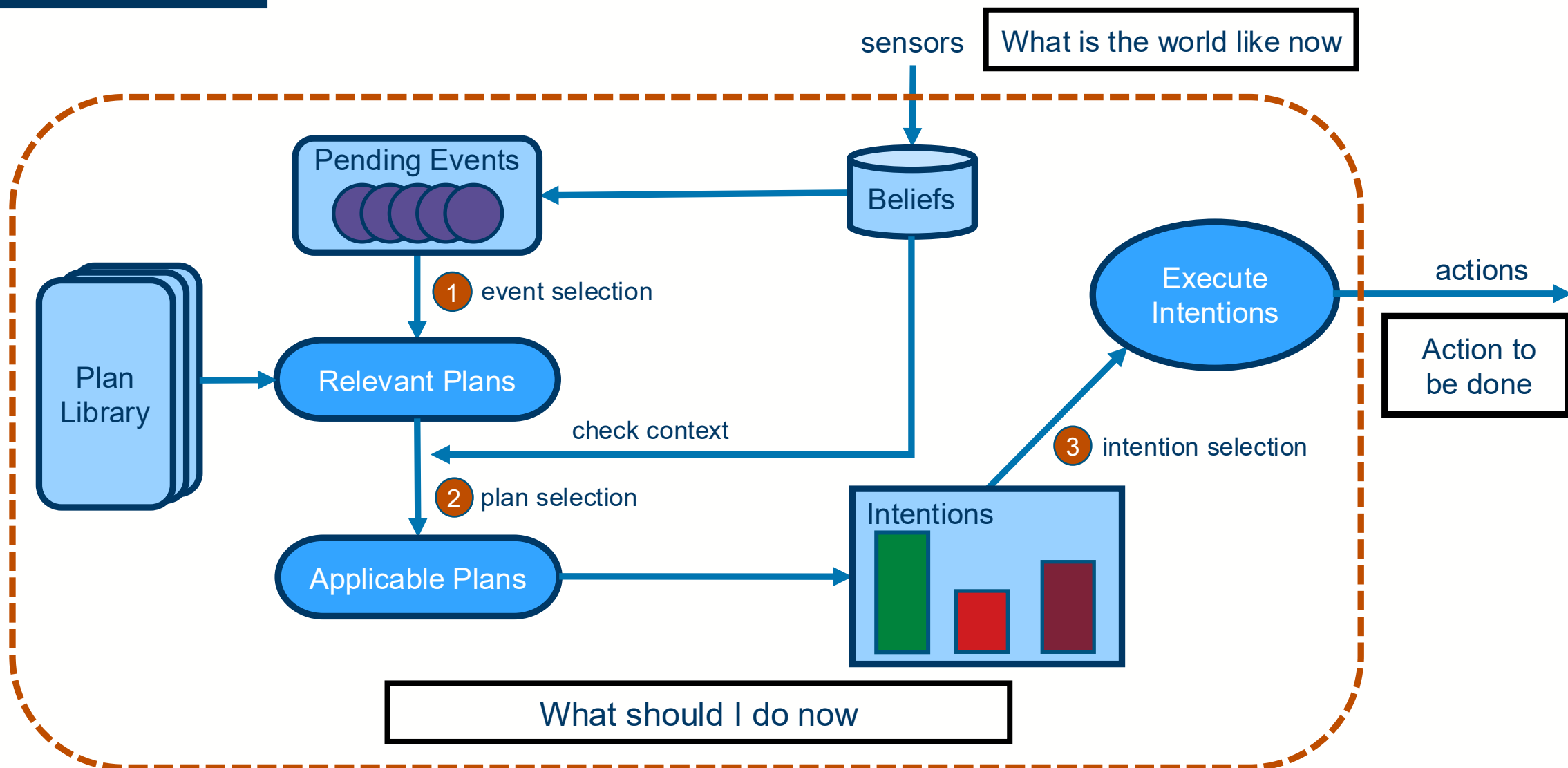


Belief-Desire-Intention (BDI) agents





Belief-Desire-Intention (BDI) agents (cont.)





Motivation: building trustworthy agents

Writing correct BDI programs is not always easy

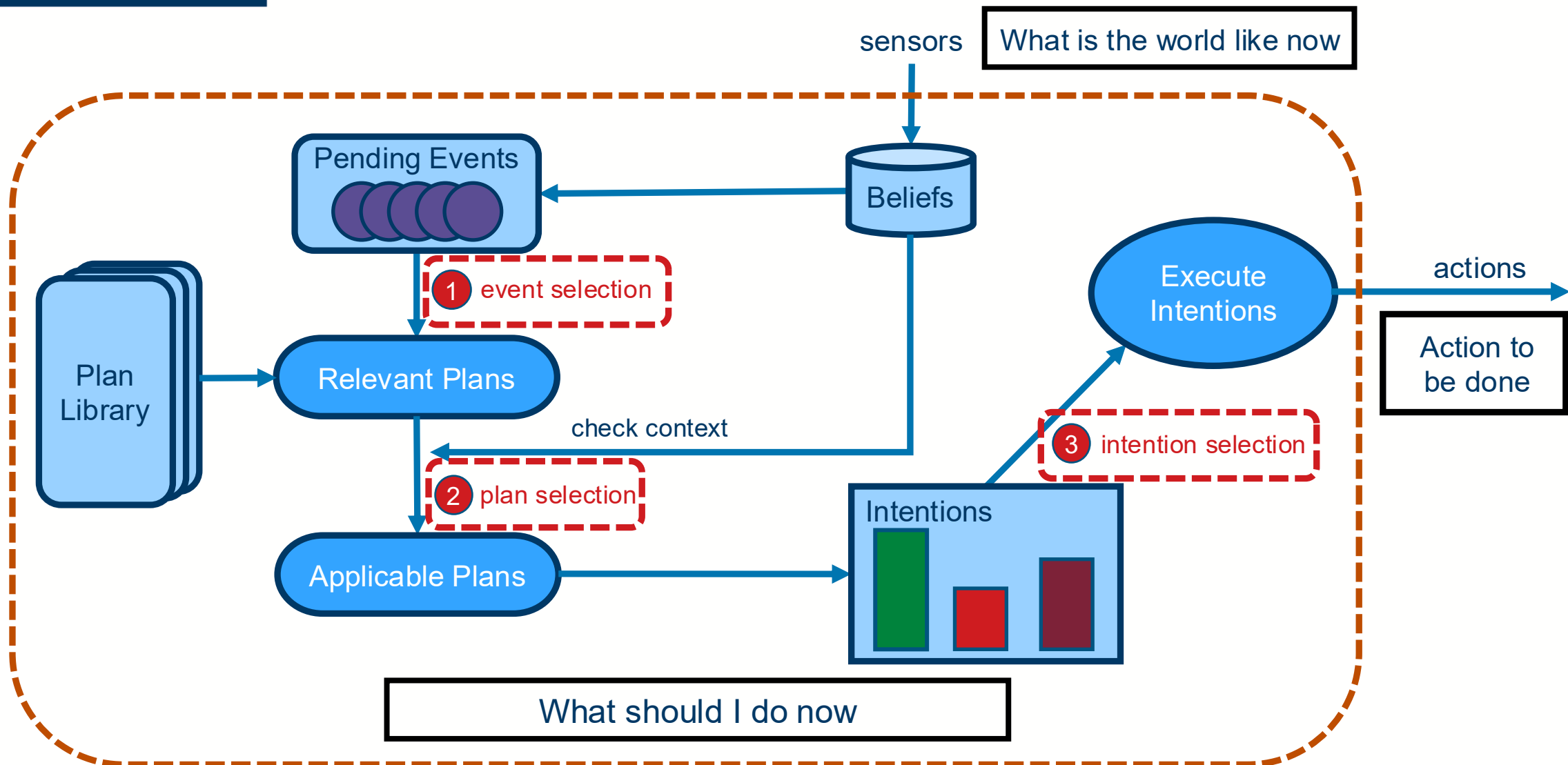
- Plans can include complex constructs like **declarative goals**, **failure recovery**, and **interleaved concurrency**

We need a model that allows us to

- **Verify** the probability an agent successfully completes a mission under environmental **uncertainty**
- **Synthesise** optimal strategies for internal agent decision making

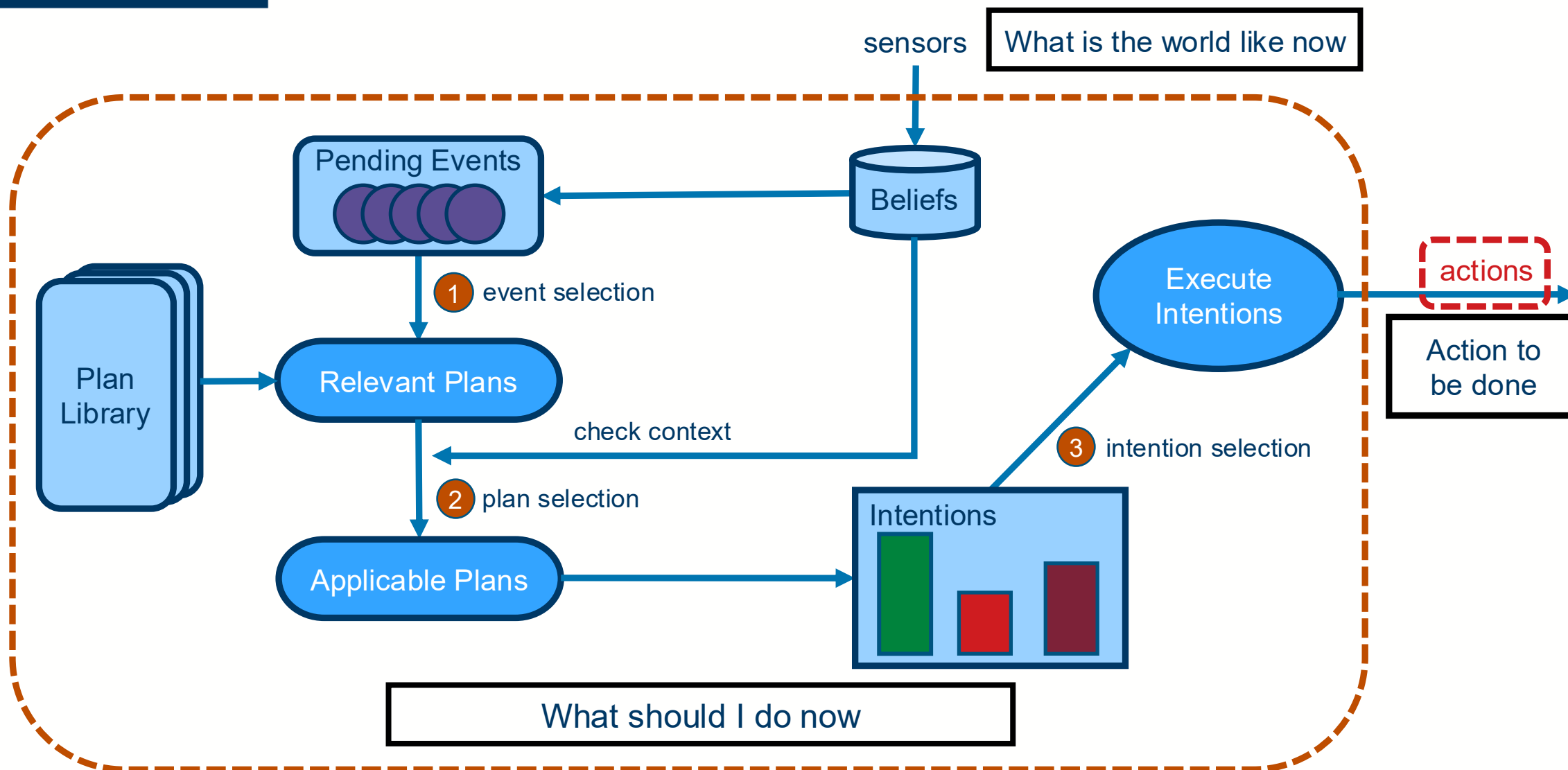


Our approach: identifying non-determinism





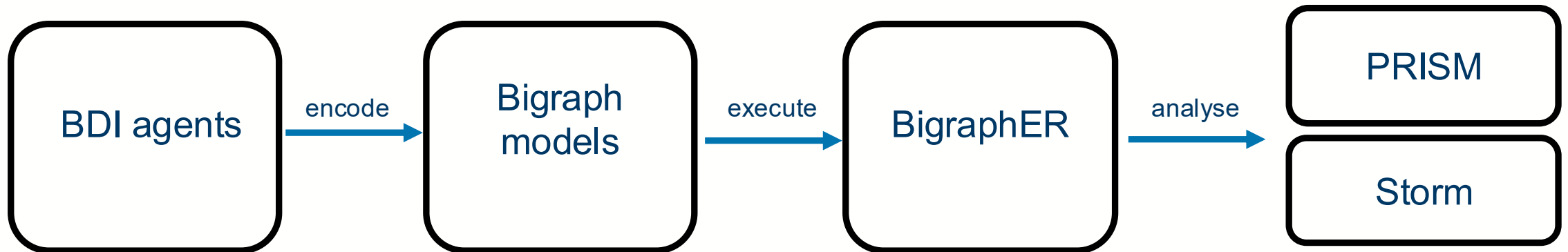
Our approach: introducing probabilistic outcomes





Our approach: encoding and verification pipeline

1. An MDP semantics for BDI to support non-deterministic selections and probabilistic action outcomes
2. An encoding of BDI agents into **bigraphs**
 - Excellent at providing **executable verifiable semantics** for DSLs and other computational models





Some technical details on our encoding

- We extend the semantics of the **CAN** language for BDI agents
- Example for probabilistic action outcomes

original BDI semantics



probabilistic BDI semantics

$$\frac{act: \varphi \leftarrow \langle \phi^-, \phi^+ \rangle \quad \mathcal{B} \models \varphi}{\langle \mathcal{B}, act \rangle \rightarrow \langle (\mathcal{B} \setminus \phi^- \cup \phi^+), nil \rangle} \quad act$$

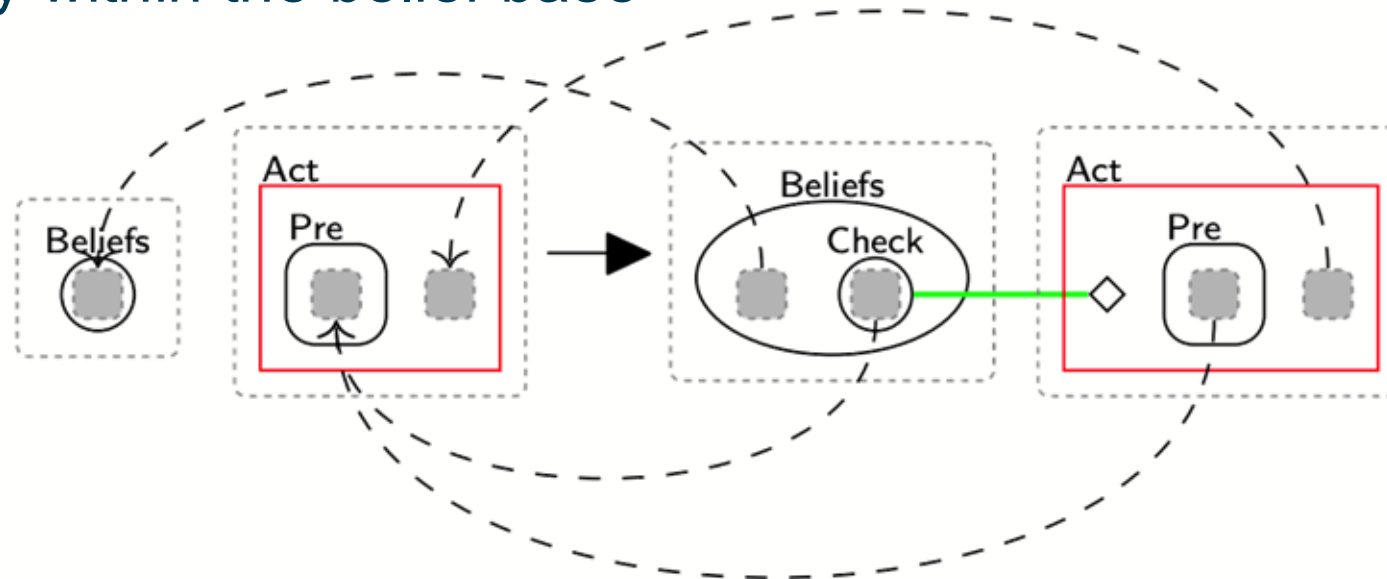
$$\frac{act: \varphi \leftarrow \mu \quad \mu(\phi_i^-, \phi_i^+) = p_i \quad \mathcal{B} \models \varphi}{\langle \mathcal{B}, act \rangle \rightarrow_{p_i} \langle (\mathcal{B} \setminus \phi_i^- \cup \phi_i^+), nil \rangle} \quad act^p$$

$$\mu = [(\phi_1^-, \phi_1^+) \mapsto p_1, \dots, (\phi_n^-, \phi_n^+) \mapsto p_n] \quad \sum_{i=1}^n p_i = 1$$



Some technical details on our encoding (cont.)

- Then we encode it as a **set** of reaction rules
- We make **extensive** use of priorities, actions (for MDP semantics), and instantaneous rules
- Example: requests the action pre-condition to be checked by nesting a Check entity within the belief base





Example

- A robotic production line for packaging items with two types of wrapping
 - Cheap wrapping might break
 - Expensive wrapping never breaks
- Items decay when temperature rises over time if not wrapped

```
1 // Initial belief bases
2 deadline1 = 10, deadline2 = 14
3 // External events
4 product1, product2
5 // Plan library
6 product1 : true <- goal(success1, process_product1, failure1).
7 process_product1 : deadline1 ≥ 3 <- wrap_standard1; move_product_standard1.
8 process_product1 : deadline1 ≥ 0 <- wrap_premium1; move_product_premium1.
9 product2 : true <- goal(success2, process_product2, failure2).
10 process_product2 : deadline2 ≥ 3 <- wrap_standard2; move_product_standard2.
11 process_product2 : deadline1 ≥ 3 <- wrap_premium2; move_product_premium2.
```



Example: analysis and synthesis

- The max/min probability of both products being processed successfully over all possible adversaries
- Optimal strategy synthesis
 1. Wrap more urgent products first until they are packed
 2. Then switch to wrap the other products.
 3. Only after both are wrapped the robot moves them to storage
- Multi-objective analysis: obtaining high success rate while keeping the overall bag cost



Some relevant papers

- B. Archibald et al. **Probabilistic BDI Agents: Actions, Plans, and Intentions**, in proceedings of *19th International Conference on Software Engineering and Formal Methods (SEFM)*, 2021
- B. Archibald et al. **Modelling and Verifying BDI Agents with Bigraphs**, in *Science of Computer Programming*, volume 215, pp 288--310, Elsevier, 2022
- B. Archibald et al. **Quantitative Verification and Strategy Synthesis for BDI Agents**, in *NASA Formal Methods Symposium (NFM)*, 2023
- B. Archibald et al. **Quantitative Modelling and Analysis of BDI Agents**, in *International Journal on Software and Systems Modeling (SoSyM)*, volume 23, pp 343--367, 2023



University
of Glasgow

Part 3: Applications (cont.)



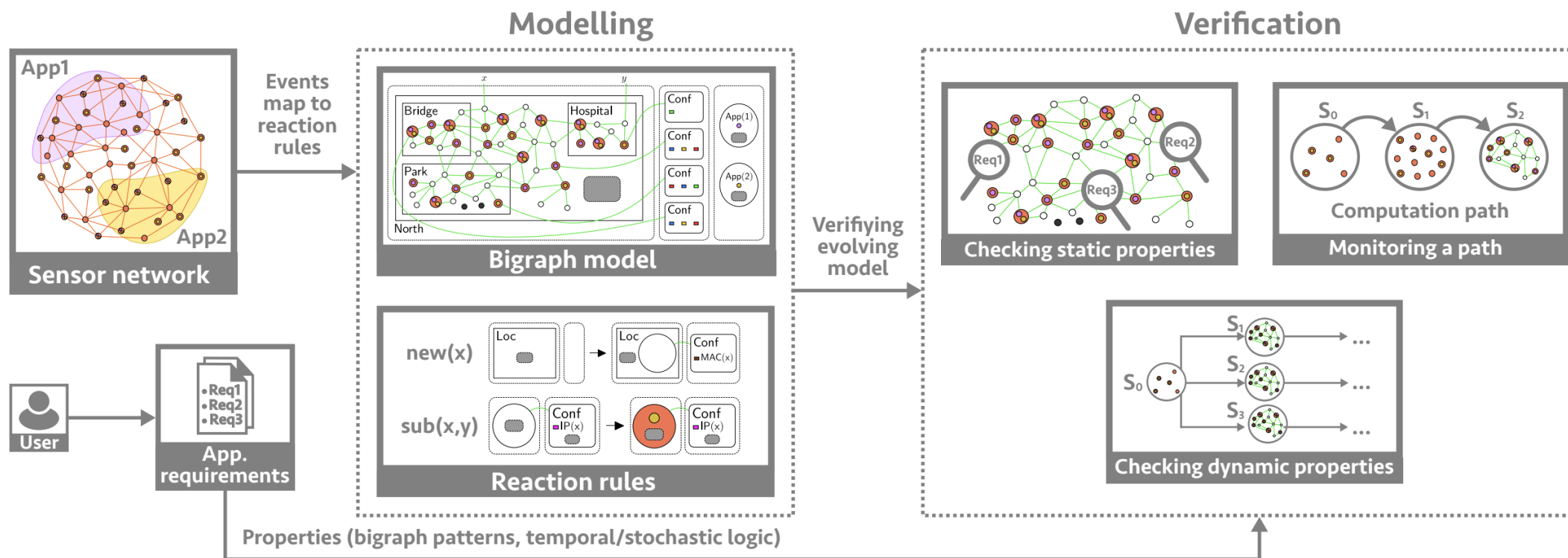


Digital twinning with bigraphs

1. Define a **bigraph model** of the twinned system
2. As new **events** are observed, map them to **reaction rules** and apply them to the model to keep it **in sync** with the system
3. Leverage the bigraph model for
 - What-if scenarios (rapid prototyping with parameters and new rules)
 - Forecasting & monitoring via runtime model checking (e.g. transient properties to look ahead n hours)
 - Optimisation/planning using MDP models
 - Explainability via the graphical notation (especially important in human-in-the-loop scenarios)

Digital twins (DTs) with bigraphs (cont.)

Example: multitenant sensor network infrastructure





Digital Twins for rail networks

Electrification scenario rollout and adoption of **battery-powered trains**

- Identify the track **segments** to electrify first so that all trains are guaranteed to reach their destination with a safe amount of charge left
- Includes the impact of (probabilistic) power blackouts

We use **model checking** on the CTMCs and MDPs generated by BigraphER

Models can be built automatically (OSM data & rail published schedules)

They are accessible to **both** railway engineers and policy makers



Bigraph model

Simple model based on segments

- Each segment corresponds to a section of the tracks of a given length
- Interconnections are modelled with links

Trains are battery-powered

- Battery discharges on non-electrified segments
- Recharged otherwise



Bigraph model (cont.)

Trains move over the segment topology either

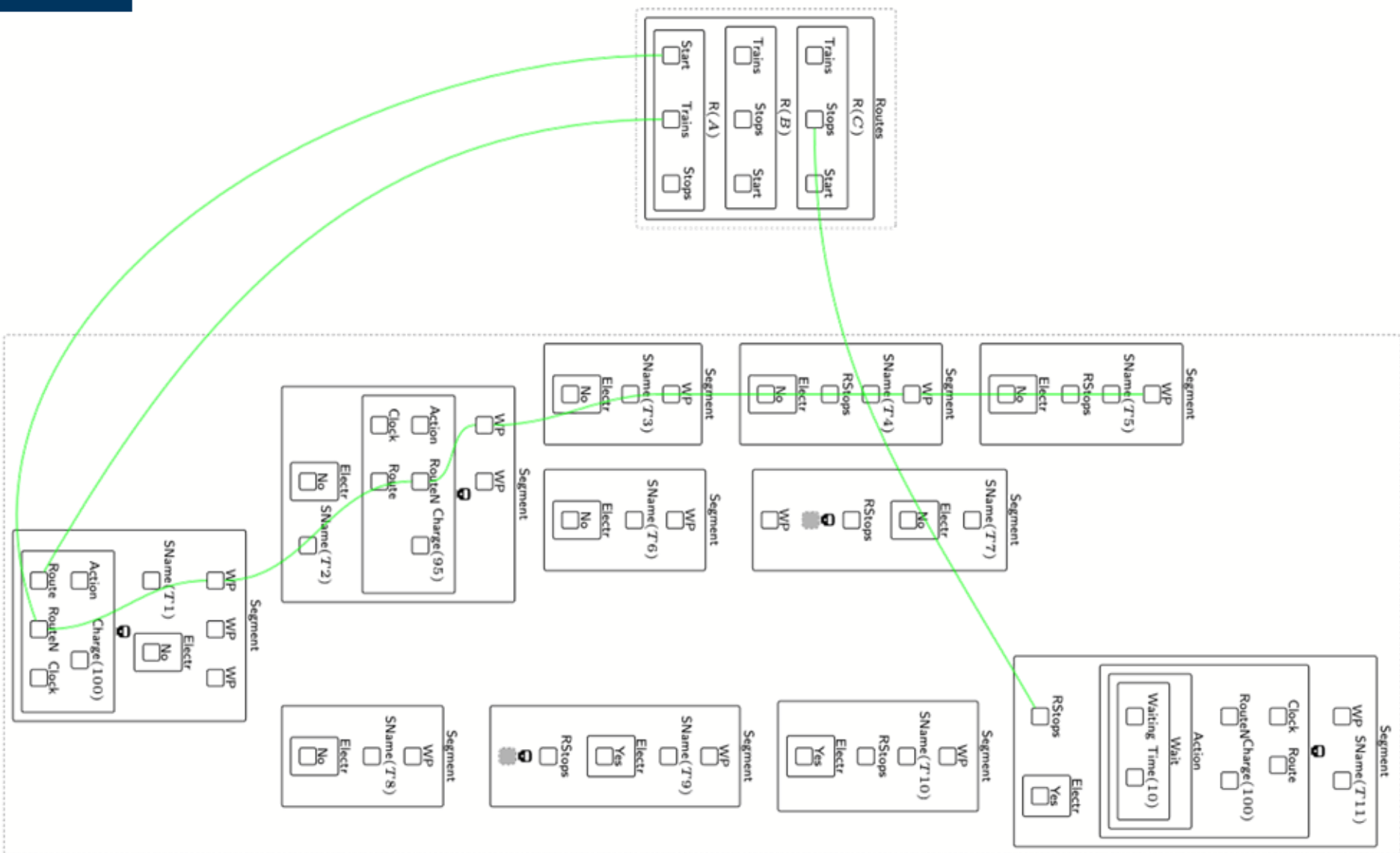
- When movement events get recorded
- Via stochastic rules specifying speed and battery charge/discharge

Other live events include

- Rail schedule update
- Delays/ failures
- Weather alerts

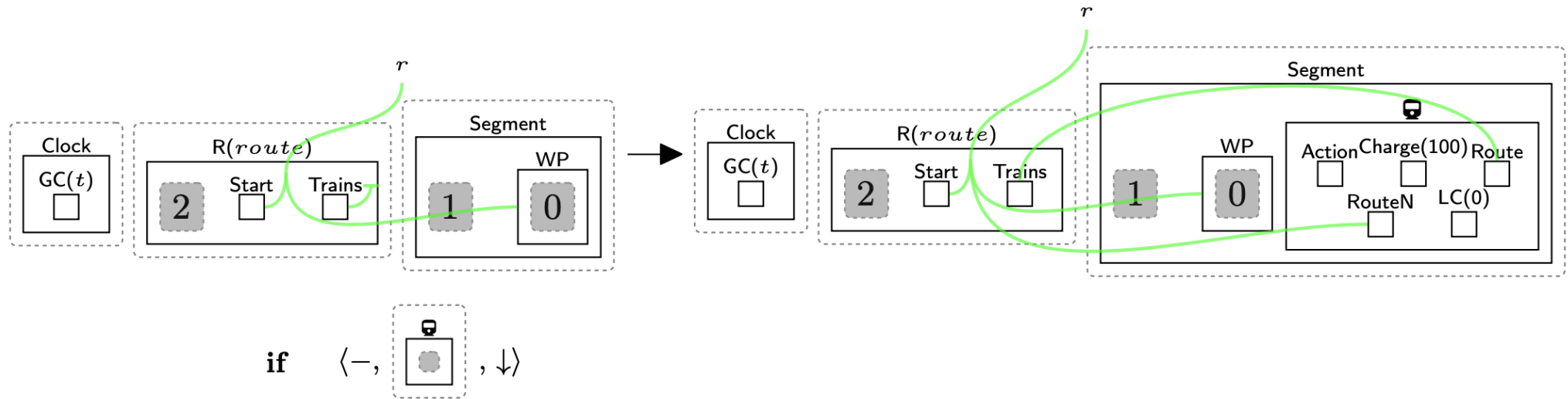


Bigraph model: initial state



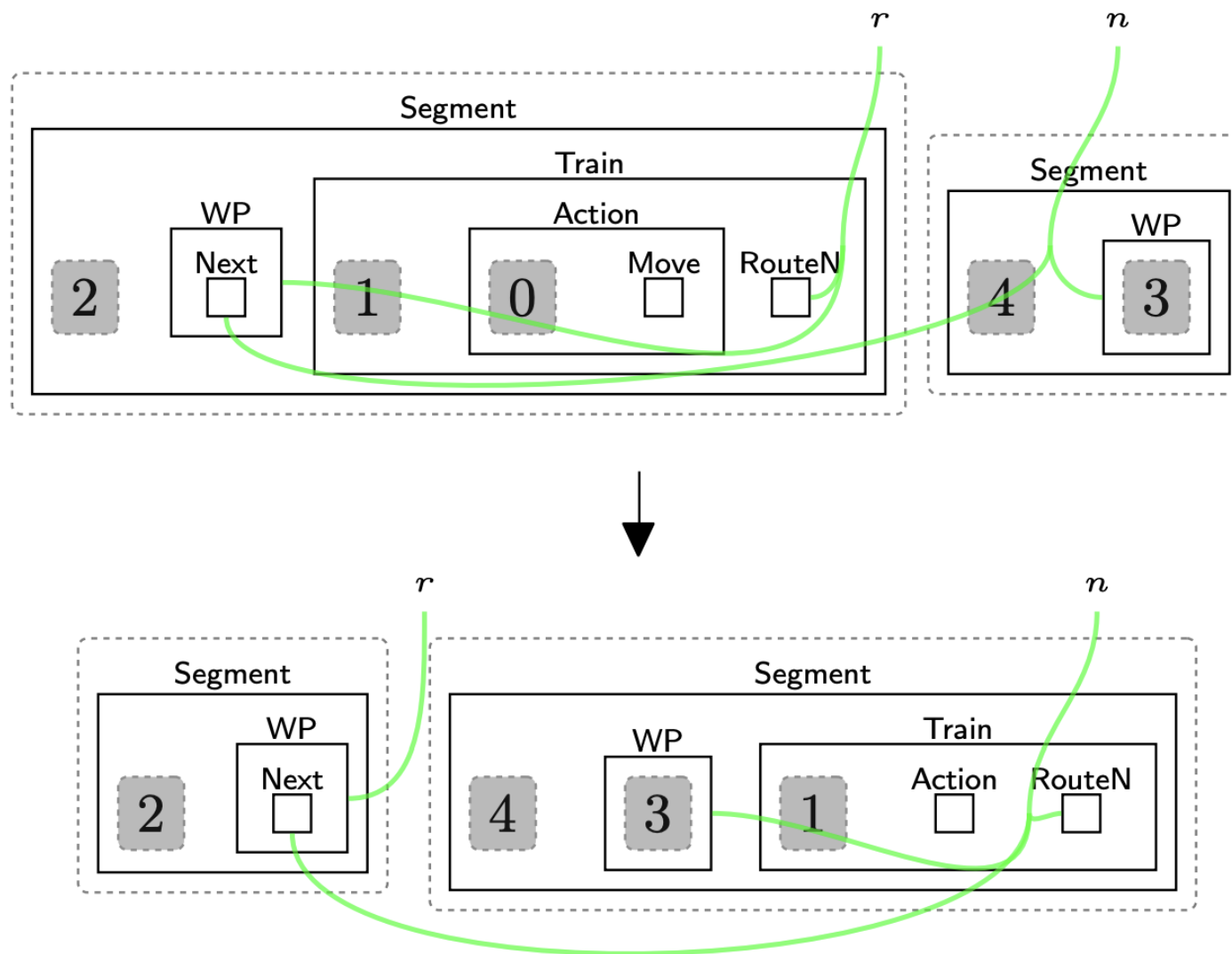


Bigraph model: reaction rules





Bigraph model: reaction rules (cont.)





Some relevant papers

- M. Sevegnani et al. **Modelling and Verification of Large-Scale Sensor Network Infrastructures**, in proceedings of *International Conference on Engineering of Complex Computer Systems (ICECCS)*, pp 71--81, IEEE, 2018
- S. Das et al. **Formal Analysis of Resilience in Transport Systems with Bigraphs**, in *International Workshop on Safety/Reliability/Trustworthiness of Intelligent Transportation Systems (SRTolTS)*, 2025
- R. Almeida et al. **Modelling Railway Networks with Bigraphs: Electrification, Failures, and Optimisation**, 2025 (to appear)



University
of Glasgow

Final thoughts





Current and future research directions

Sorts support and Rocq formalisation

- Code extraction for a certified implementation of BigraphER
- More expressive spatial logics (SLCS style, transitive closures)

Algorithms and implementations for RPO construction

- Model learning, neurosymbolic applications
- Bisimulation techniques

Support for other stochastic processes (POMDP, PTA) and games

Support for efficient statistical model checking

Improve support for runtime monitoring and scalability for realistic twins



University
of Glasgow

Thank you!

Questions?

Acknowledgments

- EPSRC grant EP/Z533221/1, TransiT: Digital Twinning Research Hub for Decarbonising Transport
- Amazon Research Award: Automated Reasoning

 **@michele_seve**