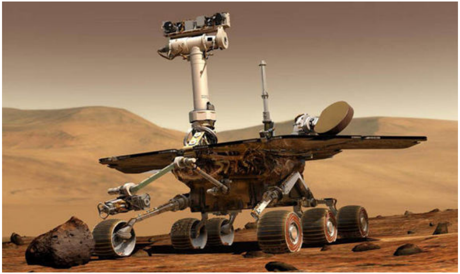
Nondeterministic Bigraphs and Their Use in Modelling Movement

Paulius Dilkas

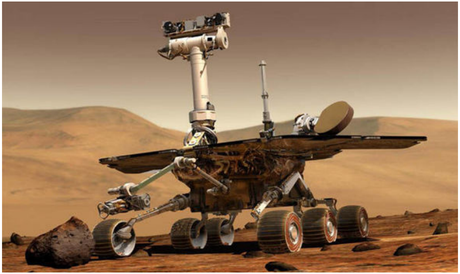
School of Computing Science

2nd September 2018

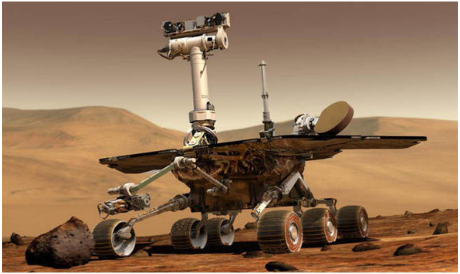
Motivation



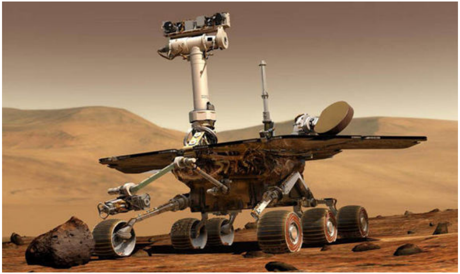
Motivation



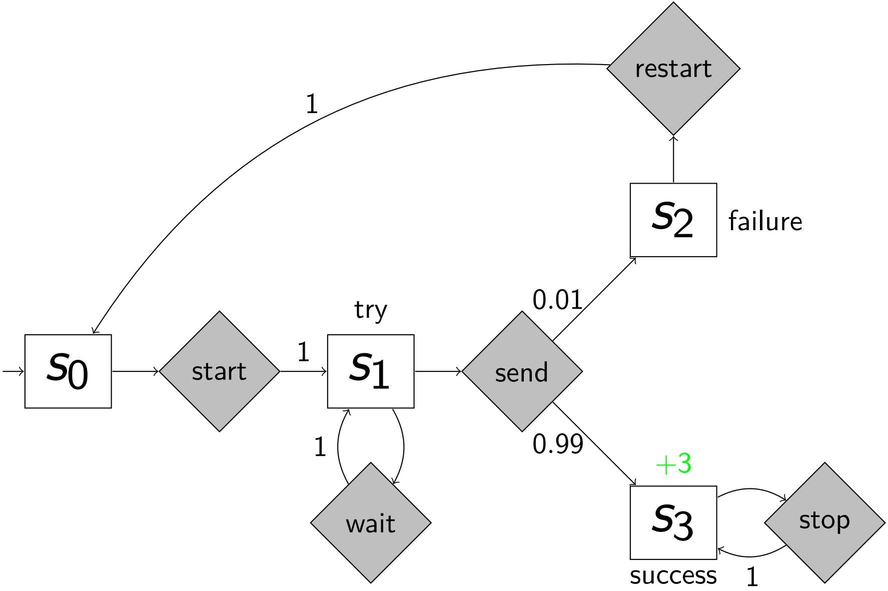
Motivation



Motivation



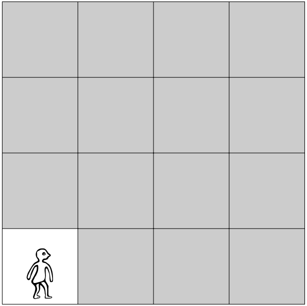
Markov Decision Process



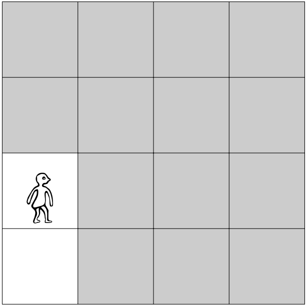
Collecting Objects in a Grid

- Each cell is either visited or unvisited.
- When entering an unvisited cell, with probability p the agent receives an object.
- Once a set number of objects is collected, the agent heads home.

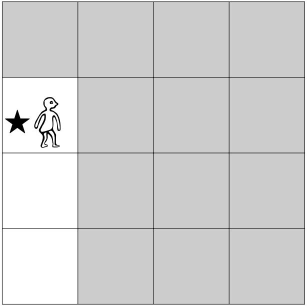
Collecting Objects in a Grid



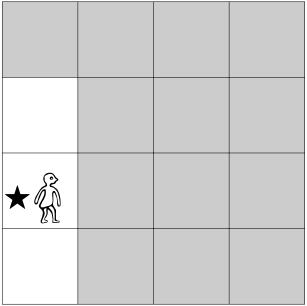
Collecting Objects in a Grid



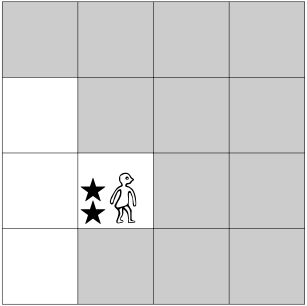
Collecting Objects in a Grid



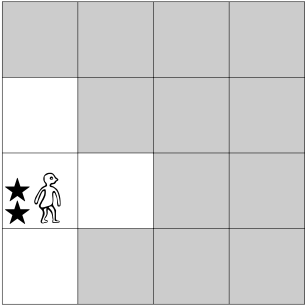
Collecting Objects in a Grid



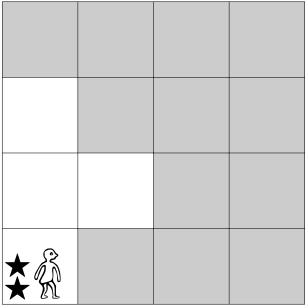
Collecting Objects in a Grid



Collecting Objects in a Grid



Collecting Objects in a Grid



A High Level View

- Controls (types of nodes)

A High Level View

- Controls (types of nodes)
  - `Agent`, `Cell`, `Directions`, `Object`

A High Level View

- Controls (types of nodes)
  - `Agent`, `Cell`, `Directions`, `Object`
  - `North`, `East`, `West`, `South`

A High Level View

- Controls (types of nodes)
  - `Agent`, `Cell`, `Directions`, `Object`
  - `North`, `East`, `West`, `South`
  - `Visited`, `Unvisited`

A High Level View

- Controls (types of nodes)
  - `Agent`, `Cell`, `Directions`, `Object`
  - `North`, `East`, `West`, `South`
  - `Visited`, `Unvisited`
- Predicates (properties to check)
  - `goal`: collected the required number of objects
  - `home`: is in the southwest corner of the grid

A High Level View

- Controls (types of nodes)
  - `Agent`, `Cell`, `Directions`, `Object`
  - `North`, `East`, `West`, `South`
  - `Visited`, `Unvisited`
- Predicates (properties to check)
  - `goal`: collected the required number of objects
  - `home`: is in the southwest corner of the grid
- Reaction rules (how the state changes)

A High Level View

- Controls (types of nodes)
  - `Agent`, `Cell`, `Directions`, `Object`
  - `North`, `East`, `West`, `South`
  - `Visited`, `Unvisited`
- Predicates (properties to check)
  - `goal`: collected the required number of objects
  - `home`: is in the southwest corner of the grid
- Reaction rules (how the state changes)
  - Grouped into actions by direction
  - Different rules for going to visited and unvisited cells

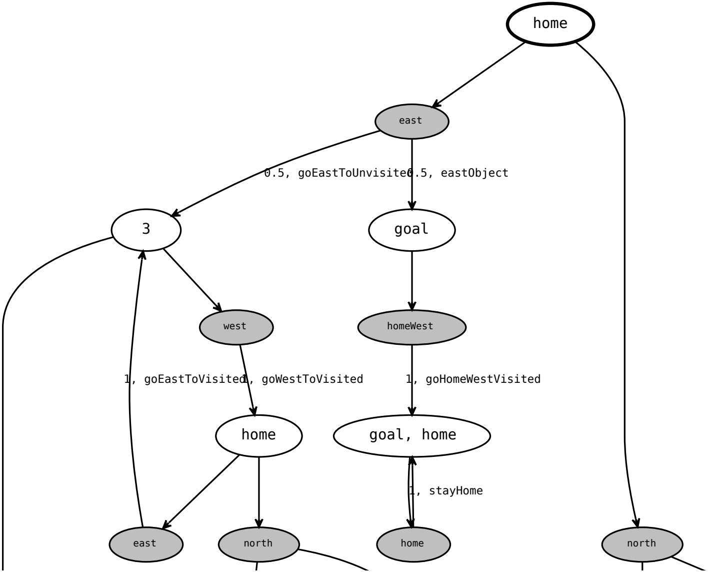
A High Level View

- Controls (types of nodes)
  - `Agent`, `Cell`, `Directions`, `Object`
  - `North`, `East`, `West`, `South`
  - `Visited`, `Unvisited`
- Predicates (properties to check)
  - `goal`: collected the required number of objects
  - `home`: is in the southwest corner of the grid
- Reaction rules (how the state changes)
  - Grouped into actions by direction
  - Different rules for going to visited and unvisited cells
  - Priority 1: going/staying home (5 rules)

A High Level View

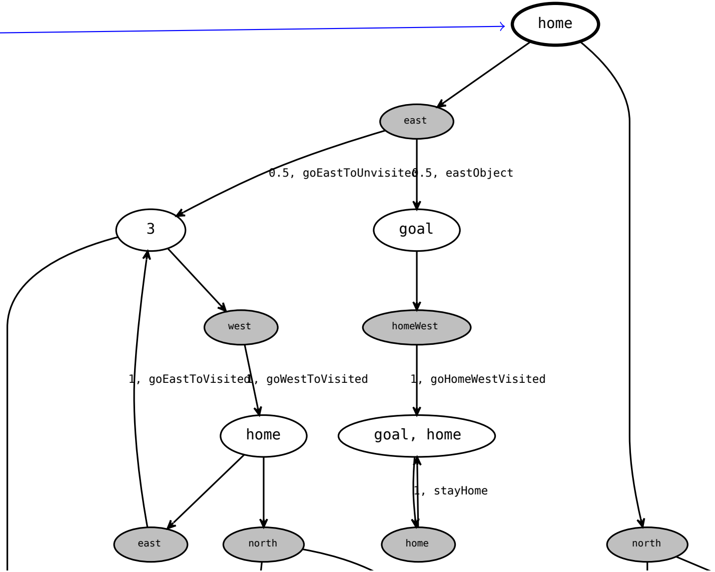
- Controls (types of nodes)
  - `Agent`, `Cell`, `Directions`, `Object`
  - `North`, `East`, `West`, `South`
  - `Visited`, `Unvisited`
- Predicates (properties to check)
  - `goal`: collected the required number of objects
  - `home`: is in the southwest corner of the grid
- Reaction rules (how the state changes)
  - Grouped into actions by direction
  - Different rules for going to visited and unvisited cells
  - Priority 1: going/staying home (5 rules)
  - Priority 2: 3 rules for each direction
    - visited
    - unvisited
    - unvisited + object

Transition System



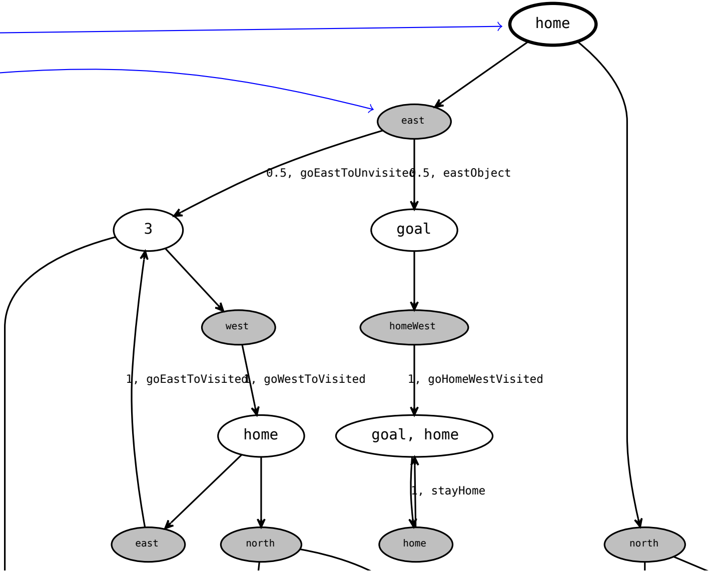
Transition System

- States



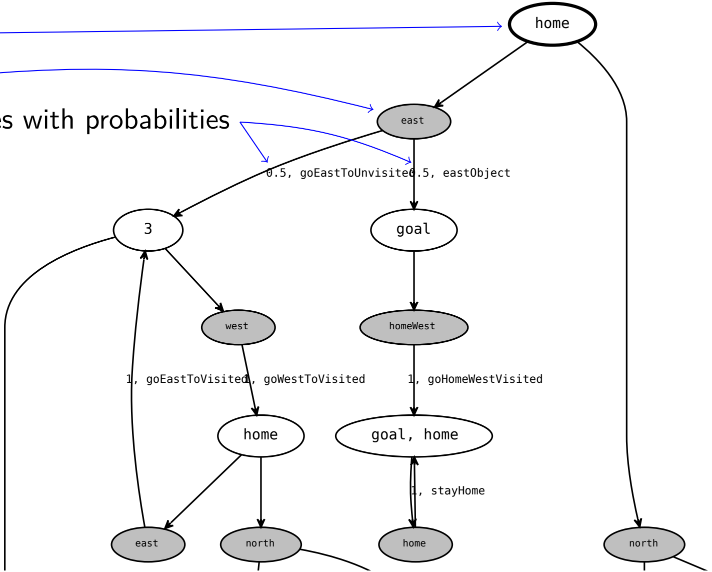
Transition System

- States
- Actions



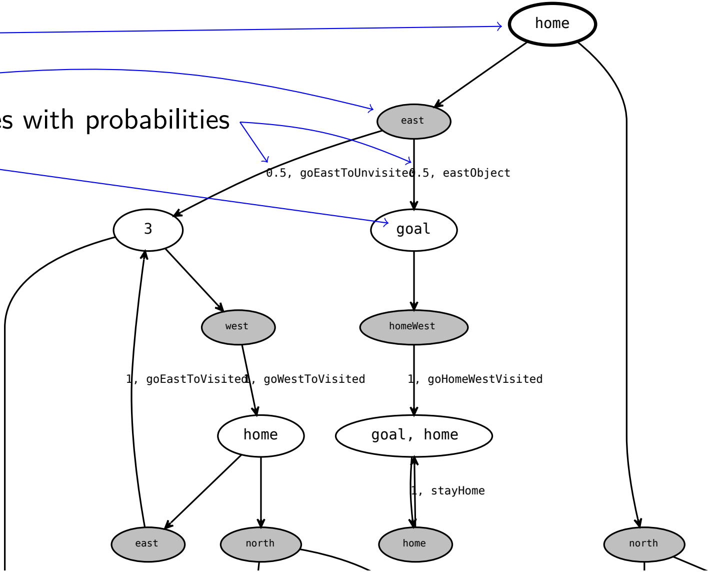
Transition System

- States
- Actions
- Reaction rules with probabilities



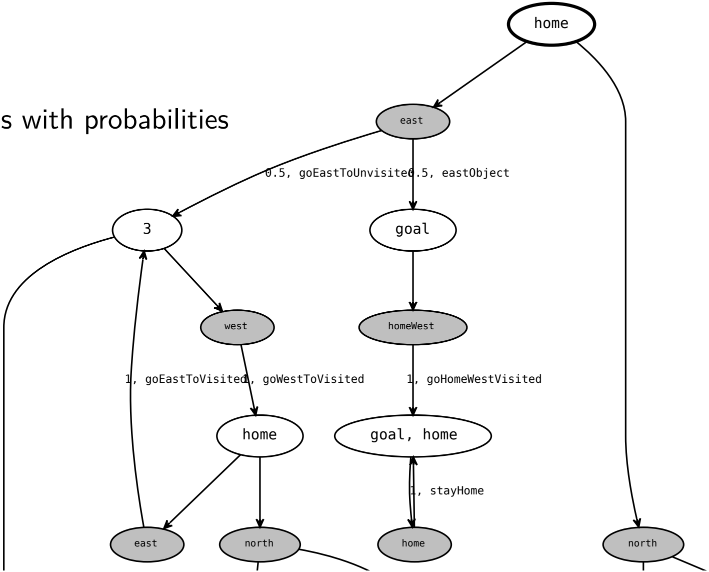
Transition System

- States
- Actions
- Reaction rules with probabilities
- Predicates

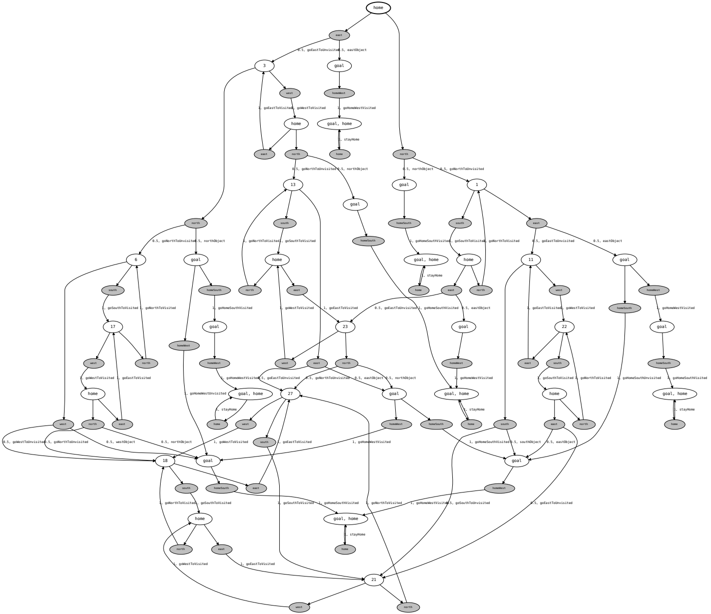


Transition System

- States
- Actions
- Reaction rules with probabilities
- Predicates



Transition System



Mechanics of Workflow

- Start with an initial state

Mechanics of Workflow

- Start with an initial state
- Find all applicable reaction rules (from the highest non-empty priority class)
    - Priorities and actions are orthogonal concepts

Mechanics of Workflow

- Start with an initial state
- Find all applicable reaction rules (from the highest non-empty priority class)
    - Priorities and actions are orthogonal concepts
- Normalise probabilities per action

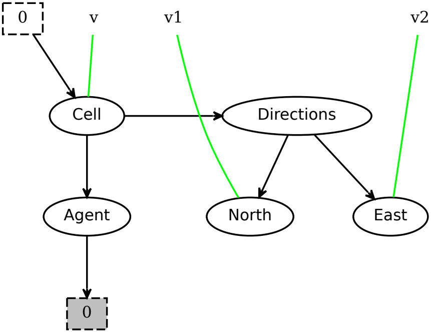
Mechanics of Workflow

- Start with an initial state
- Find all applicable reaction rules (from the highest non-empty priority class)
    - Priorities and actions are orthogonal concepts
- Normalise probabilities per action
    - Caveat: one rule can sometimes be applied in multiple ways
    - In that case, each outcome is equally likely

Mechanics of Workflow

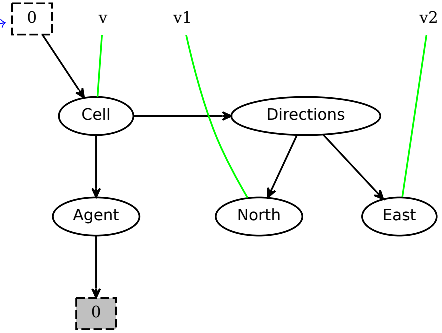
- Start with an initial state
- Find all applicable reaction rules (from the highest non-empty priority class)
  - Priorities and actions are orthogonal concepts
- Normalise probabilities per action
  - Caveat: one rule can sometimes be applied in multiple ways
  - In that case, each outcome is equally likely
- Either:
  - Breadth first search to generate the full transition system
  - Or select the next state randomly for a simulation

Bigraphs



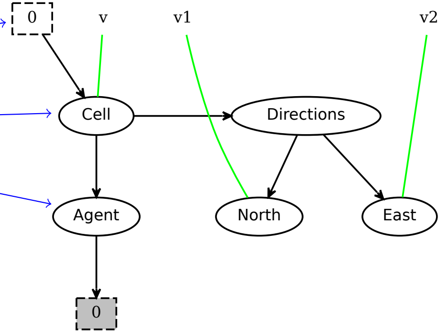
Bigraphs

- Region



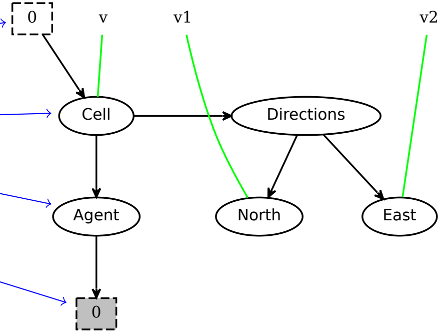
Bigraphs

- Region
- Nodes



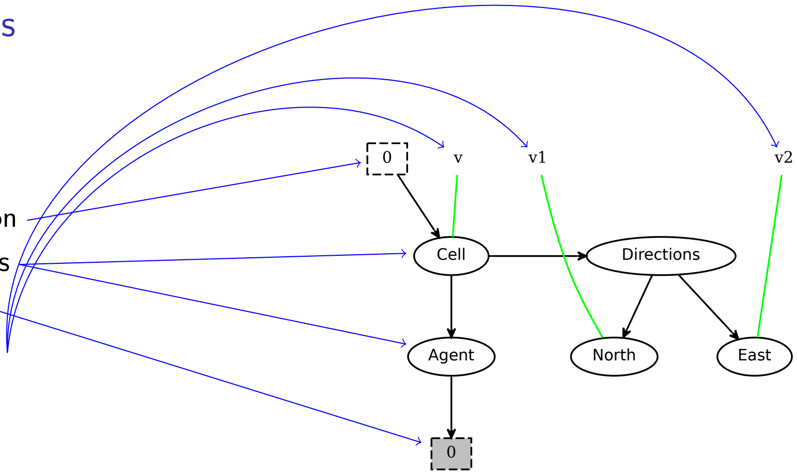
Bigraphs

- Region
- Nodes
- Site

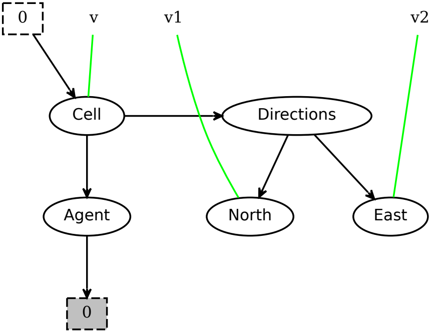


Bigraphs

- Region
- Nodes
- Site
- Links



Bigraphs

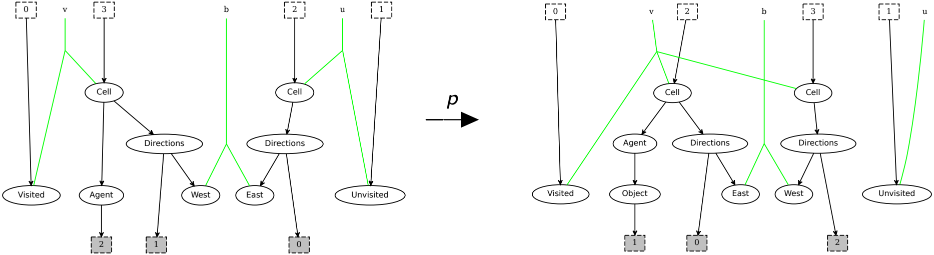


```
big home = Cell{v}.(Directions.(North{v1}
                              | East{v2})
                    | Agent);
```

Initial State

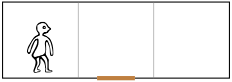
```
big initial = Visited{v}
           || Unvisited{u}
           # bottom left
           || Cell{v}.(Directions.(North{a}
                                 | East{b})
                     | Agent.1)
           # top left
           || Cell{u}.Directions.(East{c}
                                | South{a})
           # bottom right
           || Cell{u}.Directions.(North{d}
                                | West{b})
           # top right
           || Cell{u}.Directions.(West{c}
                                | South{d});
```

Reaction Rule: Go West and Collect an Object

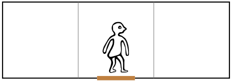


A Tale of Schrödinger's Wall...

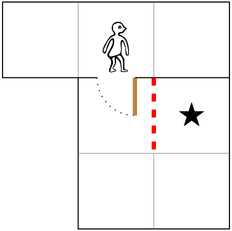
A Tale of Schrödinger's Wall...



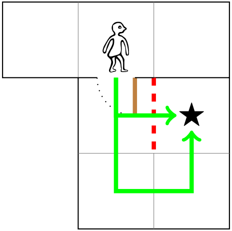
A Tale of Schrödinger's Wall...



A Tale of Schrödinger's Wall...



A Tale of Schrödinger's Wall...



A High Level View

- Controls
  - `Agent`, `Cell`, `Directions`, `Goal`, `Node`

A High Level View

- Controls
  - `Agent`, `Cell`, `Directions`, `Goal`, `Node`
  - `North`, `East`, `West`, `South`
  - no `Object`, no `Visited`/`Unvisited`

A High Level View

- Controls
  - Agent, Cell, Directions, Goal, Node
  - North, East, West, South
  - no Object, no Visited/Unvisited
- Reaction rules
  - Priority 1: generating the room (2 rules in 1 action)

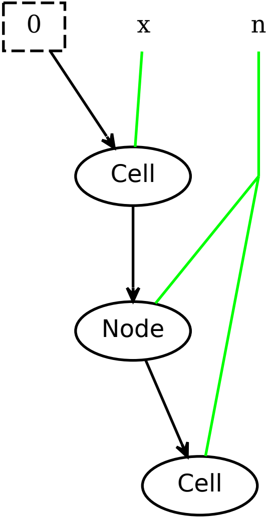
A High Level View

- Controls
  - `Agent`, `Cell`, `Directions`, `Goal`, `Node`
  - `North`, `East`, `West`, `South`
  - no `Object`, no `Visited`/`Unvisited`
- Reaction rules
  - Priority 1: generating the room (2 rules in 1 action)
  - Priority 2: movement in 6 directions (including going in/out)
    - ⋆ each rule in a separate action

A High Level View

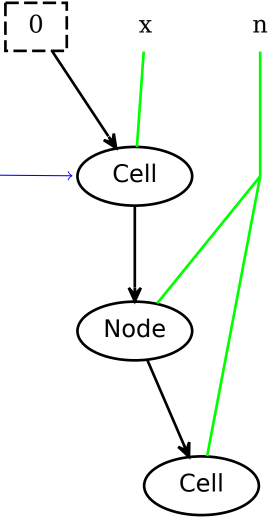
- Controls
  - `Agent`, `Cell`, `Directions`, `Goal`, `Node`
  - `North`, `East`, `West`, `South`
  - no `Object`, no `Visited`/`Unvisited`
- Reaction rules
  - Priority 1: generating the room (2 rules in 1 action)
  - Priority 2: movement in 6 directions (including going in/out)
    - ★ each rule in a separate action
- Predicate
  - is `Agent` and `Goal` in the same cell?

The Main Idea



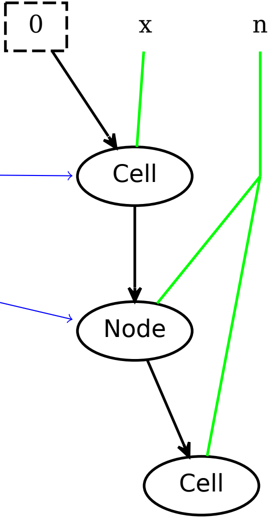
The Main Idea

- Outside the door



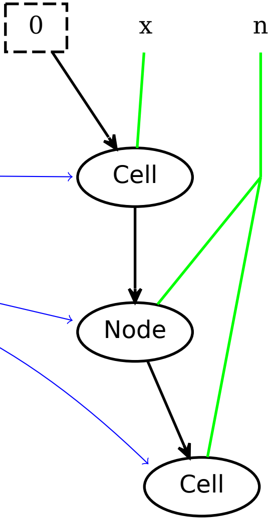
The Main Idea

- Outside the door
- The room



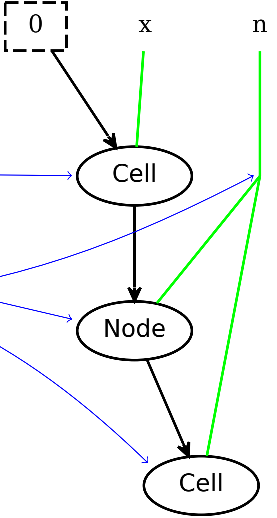
The Main Idea

- Outside the door
- The room
- Inside the door

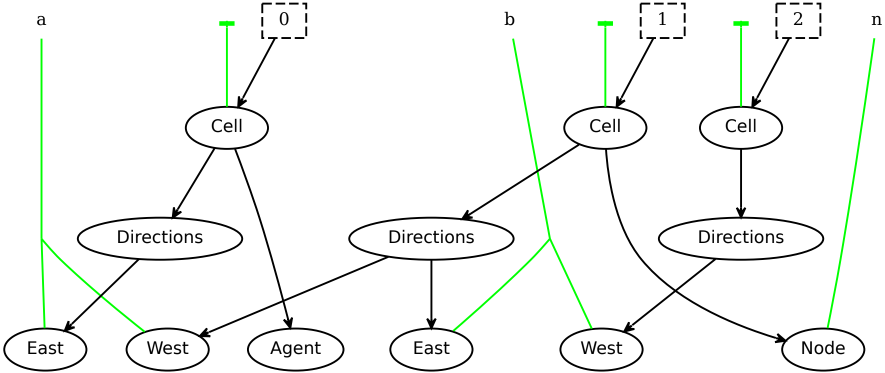


The Main Idea

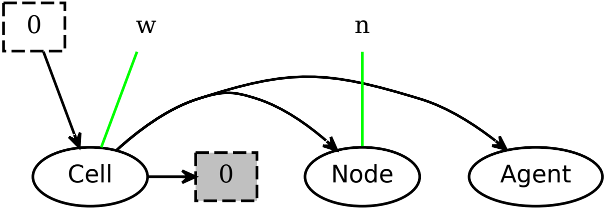
- Outside the door
- The room
- Inside the door
- Which cell is closest to the door?



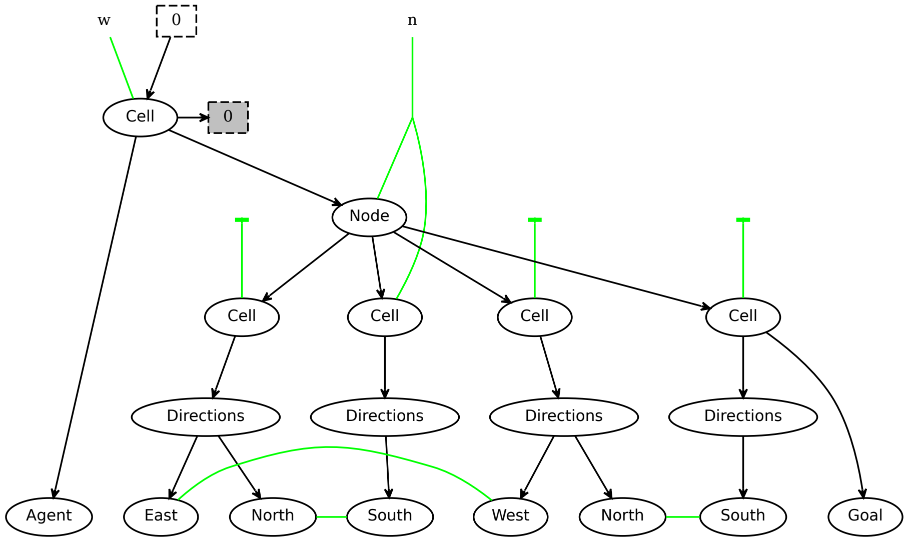
Initial State



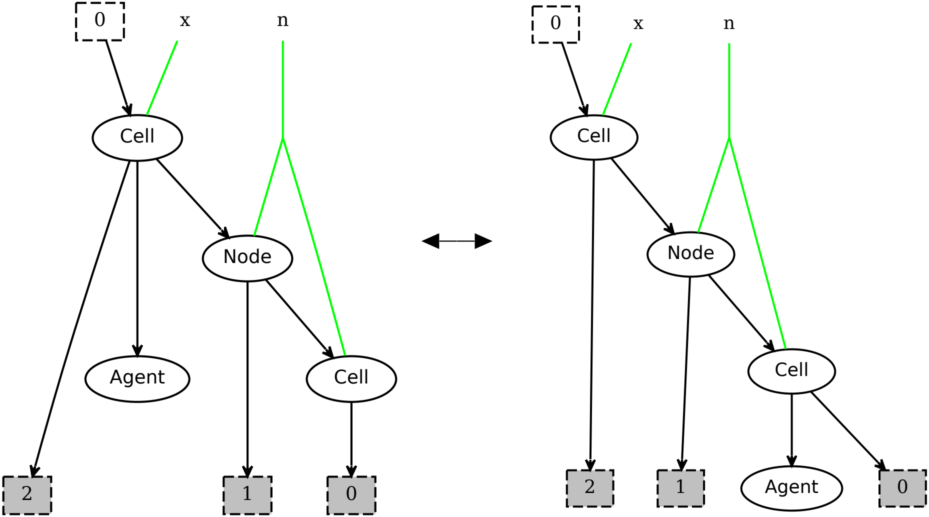
Opening the Door



Opening the Door



Entering/Leaving a Room



Entering/Leaving a Room

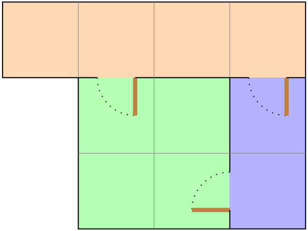
```
action goIn
  react goIn = Cell{x}.(Agent | Node{n}.(Cell{n}
                                       | id)
                      | id)
               -[1.0]->
               Cell{x}.(Node{n}.(Cell{n}.(Agent
                                        | id)
                               | id)
                      | id);
end
```

Extensions

- Multiple rooms (make each `Node` uniquely identifiable)

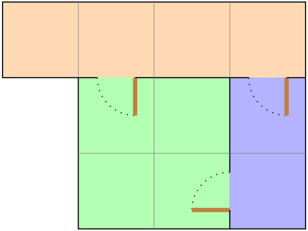
Extensions

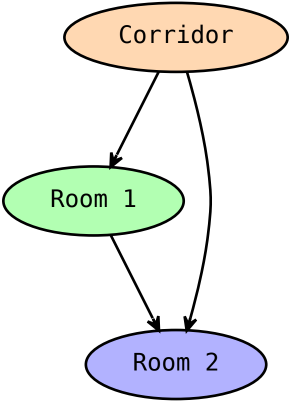
- Multiple rooms (make each `Node` uniquely identifiable)
- Arbitrarily complex configurations (via bigraphs with sharing)



Extensions

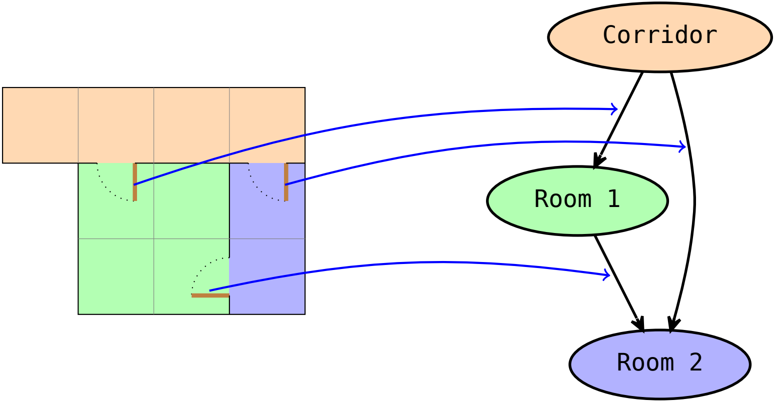
- Multiple rooms (make each Node uniquely identifiable)
- Arbitrarily complex configurations (via bigraphs with sharing)





Extensions

- Multiple rooms (make each `Node` uniquely identifiable)
- Arbitrarily complex configurations (via bigraphs with sharing)



Conclusions

- + A direct visual representation of the modelled situation
- + Easy to represent complicated spatial structures and uncertainty about them
- + Succinct and easy to modify
- – Some simple ideas are impossible or hard to implement
- – Not every aspect of a model can be exported for quantitative analysis
- More work to be done on probabilistic space