Behaviour-Data Relations Modelling Language For Multi-Robot Control Algorithms

Lenka Pitonakova
Richard Crowder

Seth Bullock

UNIVERSITY OF Southampton

University of BRISTOL

Multi-robot systems

- Collective performance **emerges** as a result of **interactions** among robots and between robots and their environment

- **“Bottom-up” design:** How to program the micro behaviour of robots for desired macro-level outcome?
  - Imperfect knowledge of the task / environment
  - Localised interactions
  - Parallel code execution

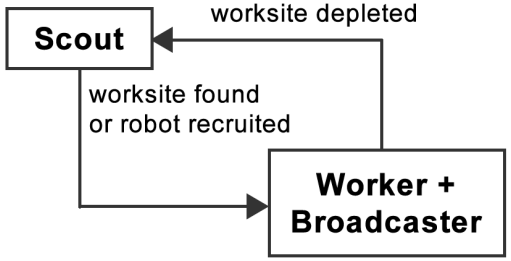
Modelling robot code

- A good modeling language is essential for:
  - **Designing** algorithms (faster on paper than in code)
  - **Representing** algorithms (facilitate quick understanding)
  - **Reproducing** algorithms (based on good representations)

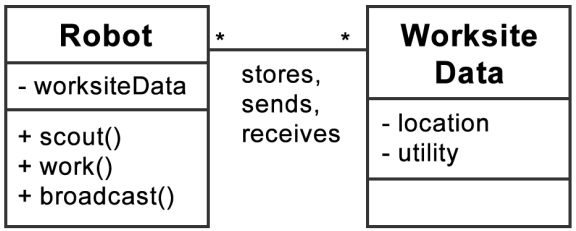
- Visual representation more accessible than text
  - Ease of understanding vs accuracy concerns

Existing methods

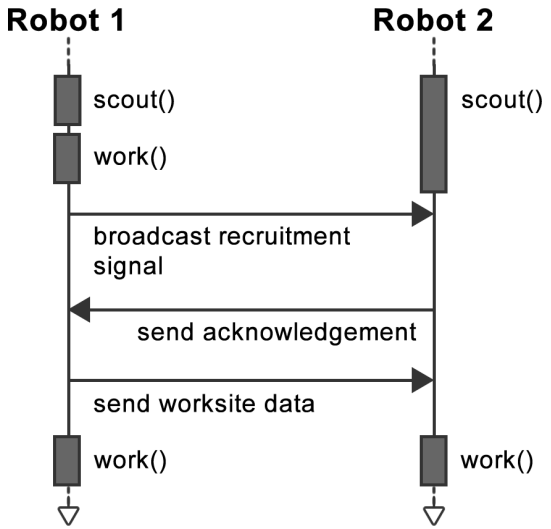
Algorithm: Search for worksites in the environment. Perform work them. Recruit nearby robots while working (e.g. customer servicing).



Statechart



Class diagram



Sequence chart

Existing methods

- These methods were invented when programs were simpler and more linear

- **Problems for multi-robot systems:**
  - Assumptions of finite-state machines with well-defined, predictable interactions
  - No explicit representation of data or of influences external to the system

BDRML

- A BDRML diagram consists of **visual** and **textual** description (1-to-1 correspondence, can be used together or separately)
  - Primitives (behaviours, data structures)
  - Conditional relations between them

- Describes robot **behaviours**, not states.
  - “Work” behaviour versus “Worker” state
  - Model finite-state machines, neural network controllers, behaviour-based controllers, etc.

- Both **behaviours** and **data** are primitives, so they can relate to each other
  - Explicit representation of what information is communicated and where it is stored
  - Combines capabilities of statecharts and class diagrams (describe control algorithm) and of sequence charts (describe communication)

- Allows to specify relations between behaviours and data **external** to a robot's memory
  - Represent communication between robots and interactions with their environment

BDRML primitives

Behaviour:

b = Behaviour name

Internal data structure:

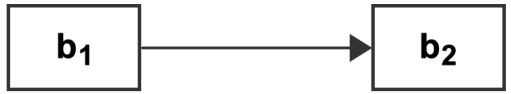
d_i = Data name : data type

External data structure:

d_e = Data name : data type

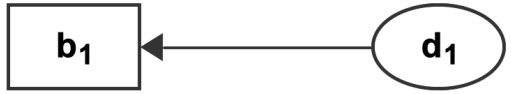
BDRML relations

Transition:



trans(b_1,b_2)

Read:



read(d_1,b_1)

Write:

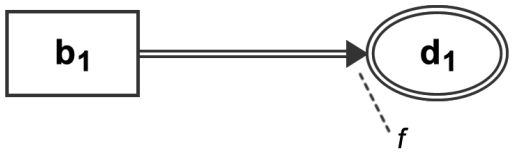
write(f : d_1,b_1)

write(+1 : d_1,b_1)

Receive:

receive(d_1,b_1)

Send:



send(f : d_1,b_1)

Copy:

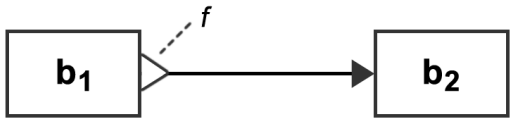
copy(d_1,d_2)

Update:

update(f : d_1)

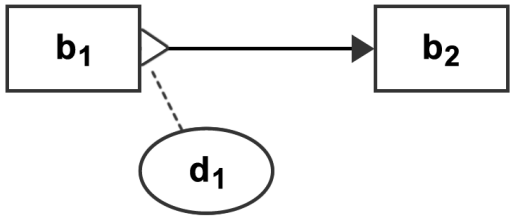
BDRML conditions

A function as a condition:

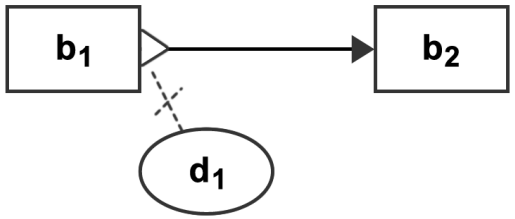


trans(b_1,b_2) : {f}

Existence and non-existence of data as a condition:



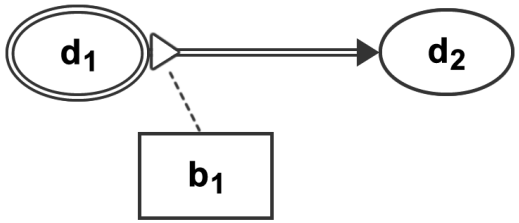
trans(b_1,b_2) : { $\exists d_1$}



trans(b_1,b_2) : { $\nexists d_1$}

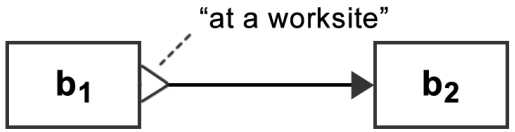
BDRML conditions

Current behaviour as a condition:



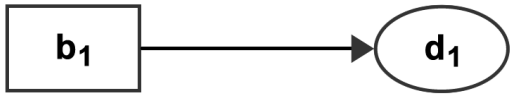
$copy(d_1,d_2) : \{b=b_1\}$

A textual description as a condition:



$trans(b_1,b_2) : \{$"at a worksite"$\}$

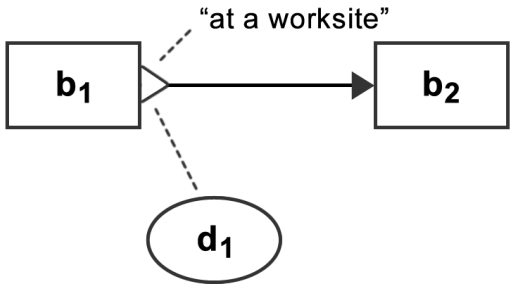
"Always" condition:



$write(d_1,b_1) : \{*\}$

BDRML conditions

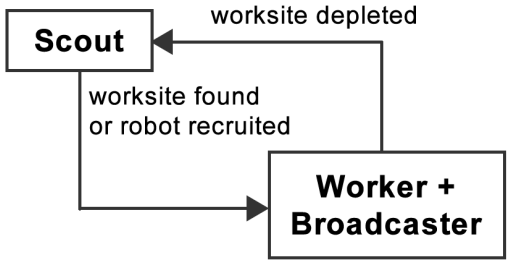
A combination of conditions:



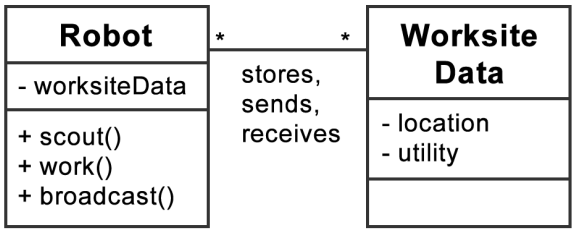
trans(b_1,b_2) : {"at a worksite",
$\exists d_1$}

BDRML vs existing methods

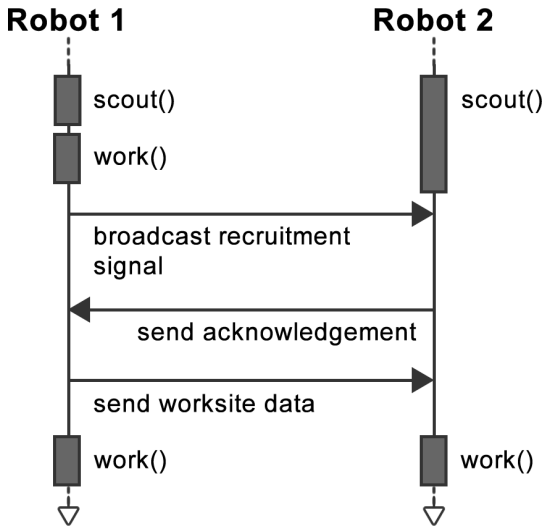
Algorithm: Search for worksites in the environment. Perform work them. Recruit nearby robots while working (e.g. customer servicing).



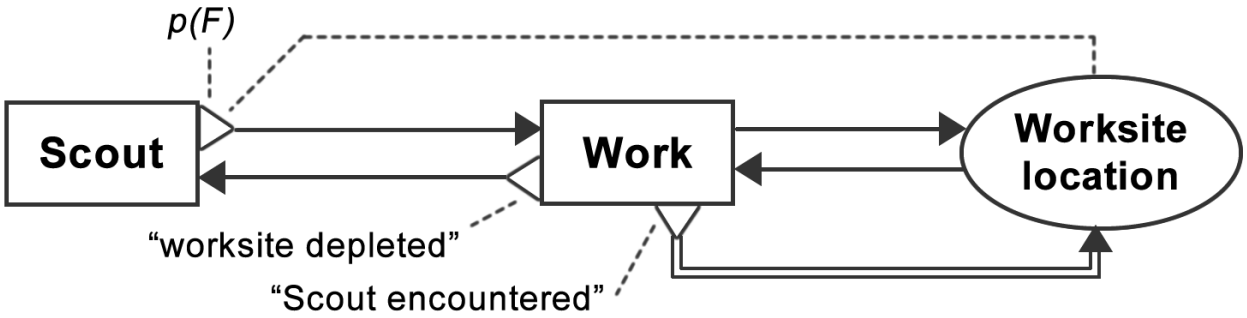
Statechart



Class diagram



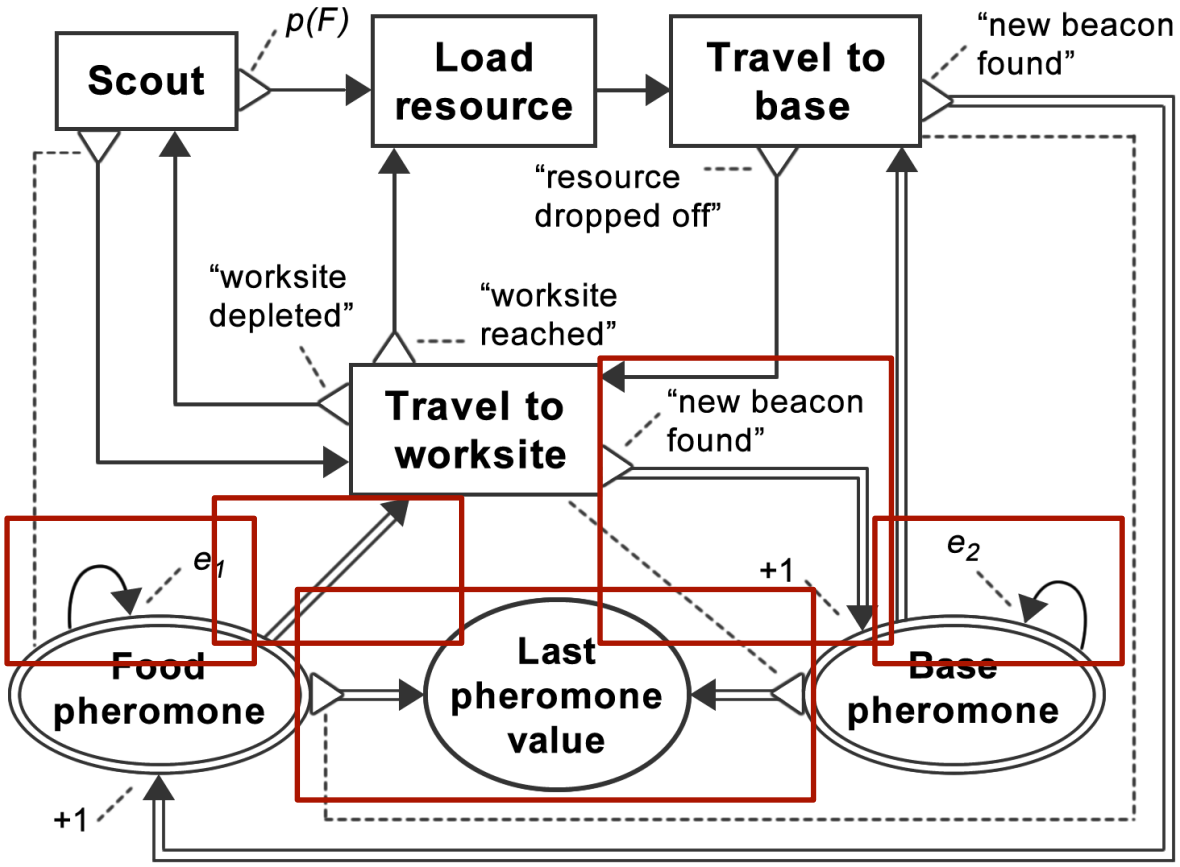
Sequence chart



B = {Scout, Work}
D_i = {Worksite location : object}
trans(Scout, Work) : {*p(F)*, ∃ Worksite location}
trans(Work, Scout) : {"worksite depleted"}
write(Worksite location, Work) : {*}
read(Worksite location, Work) : {*}
send(Worksite location, Work) : {"Scout encountered"}

- **Explicit representation of recruitment**:
  - *Send* relation between *Work* and *Worksite location*
  - Conditional transition between *Scout* and *Work*

Algorithm: Search for worksites in the environment. Use beacons to store pheromone for navigation and recruitment.



- Pheromone evaporation
- Interactions of robot behaviours and pheromone
- Storing pheromone values in robot's memory

Thank you

- Find our paper: L. Pitonakova, R. Crowder, S. Bullock: ***Behaviour-Data Relations Modelling Language For Multi-Robot Control Algorithms***
- Reach me on **contact@lenkaspace.net**