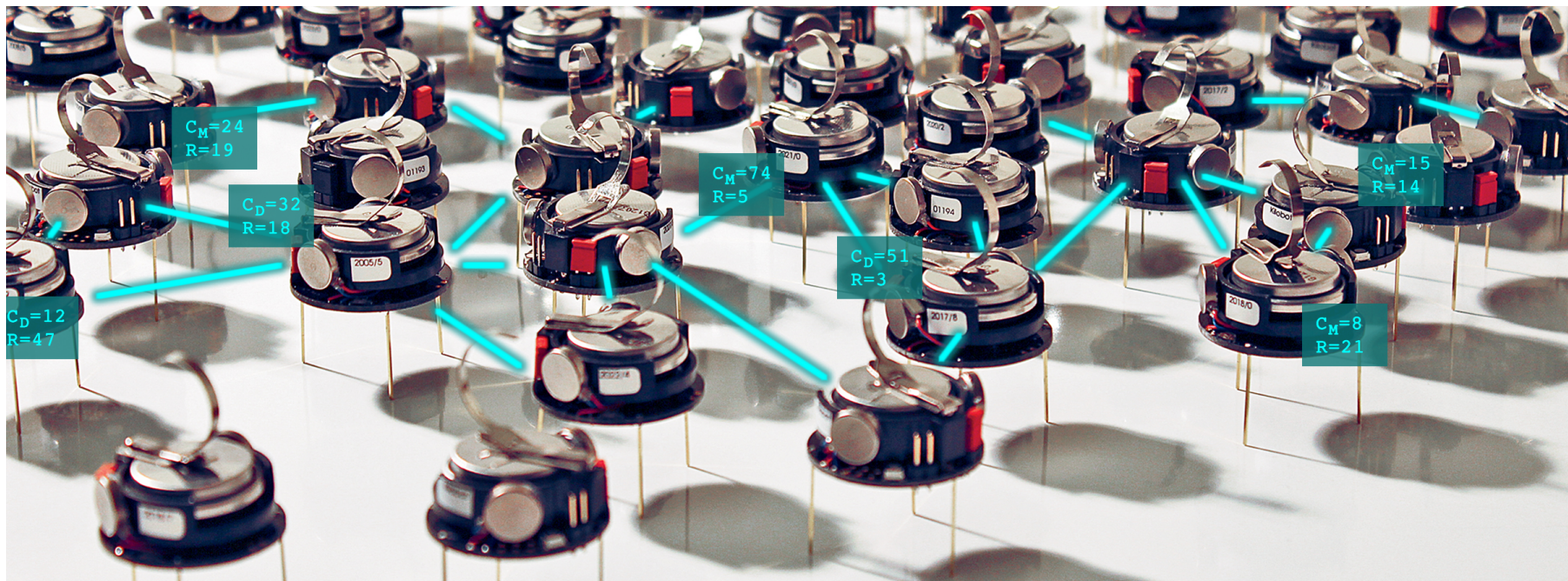


Designing Robot Swarms: A project overview

Lenka Pitonakova

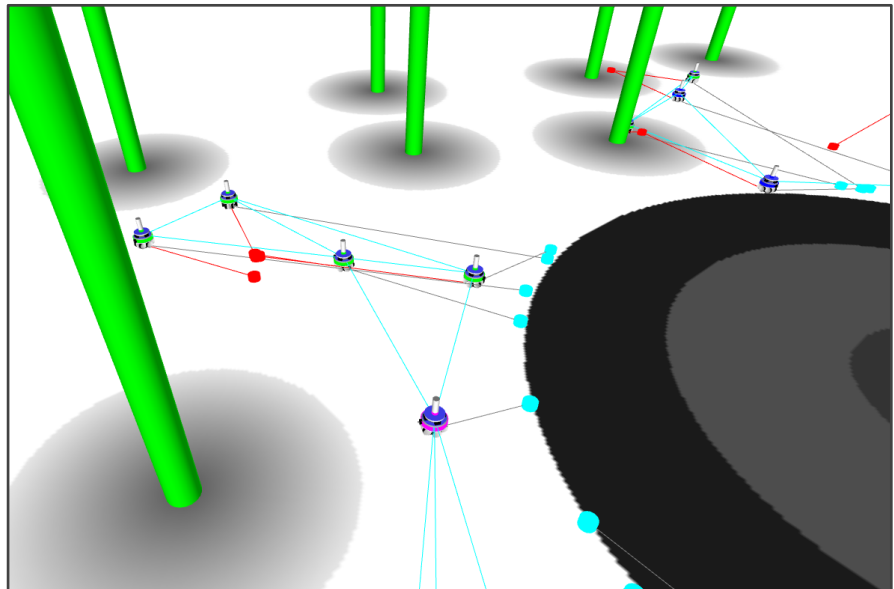


Robot swarms

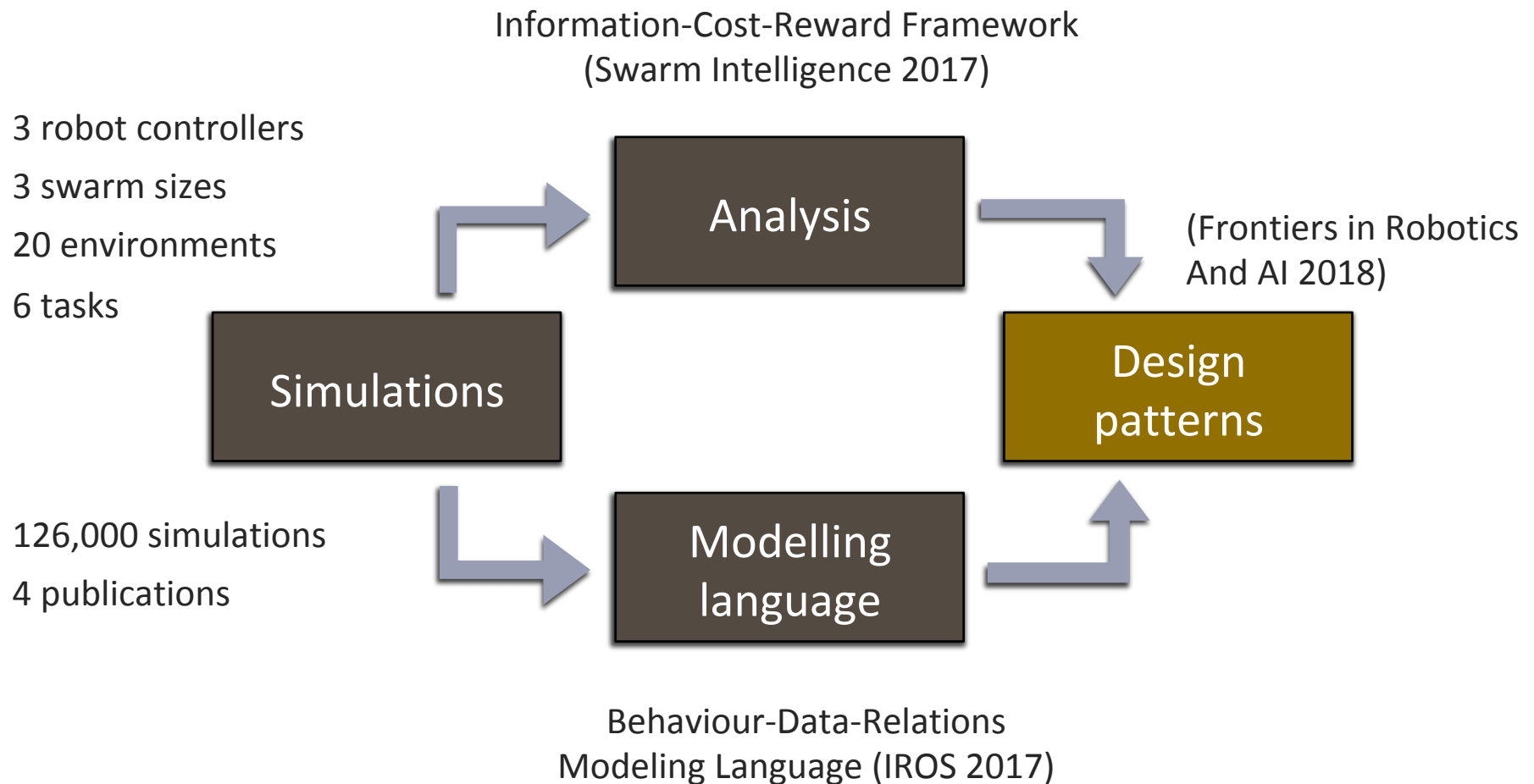
- Collective performance **emerges** as a result of **interactions** among robots and between robots and their environment
- **“Bottom-up” design: How to program the micro behaviour of robots for a desired macro-level outcome?**
 - Imperfect knowledge of the task / environment
 - Localised interactions
 - Parallel code execution
- **Design patterns:** Guidelines for programming modules of robot behaviour (inspired by Object-Oriented Design patterns)

Robot swarm foraging

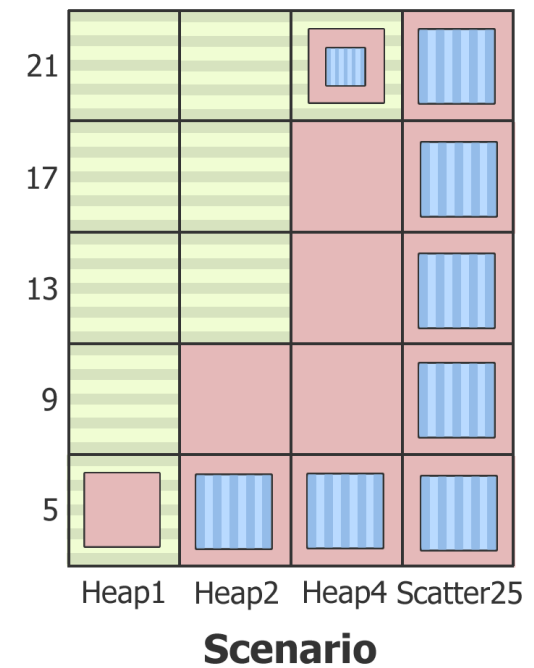
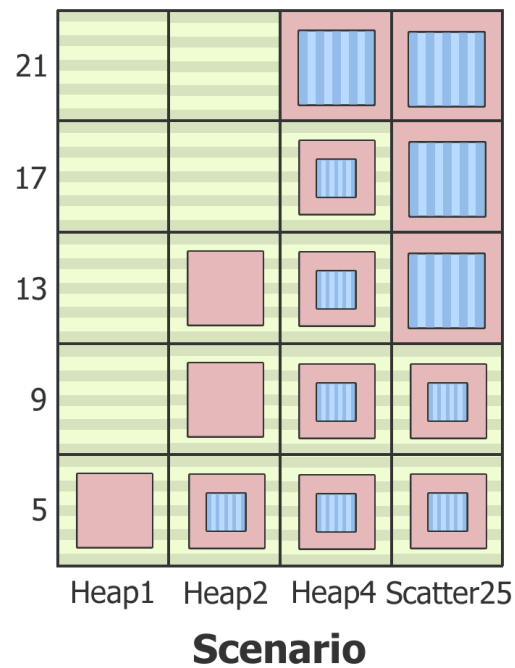
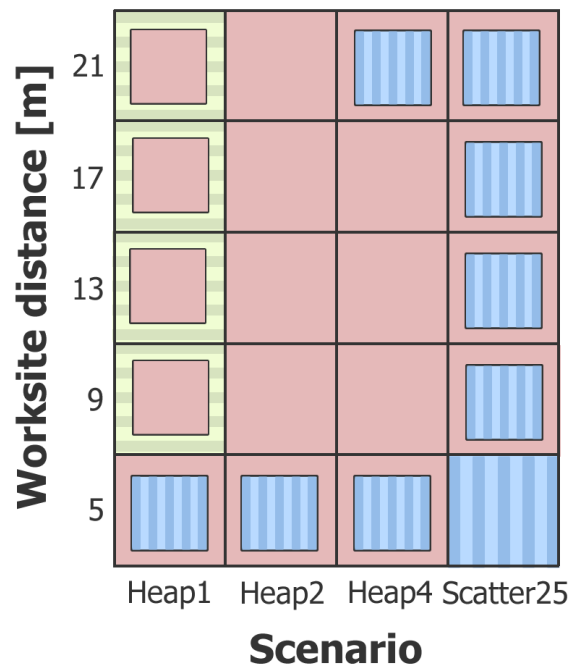
- A group of robots has to search the environment for “worksites” and perform work on them or gather items from them
- Paradigm for
 - Resource collection
 - Warehouse servicing
 - Monitoring
 - Toxic waste clean-up
 - ...






Project overview



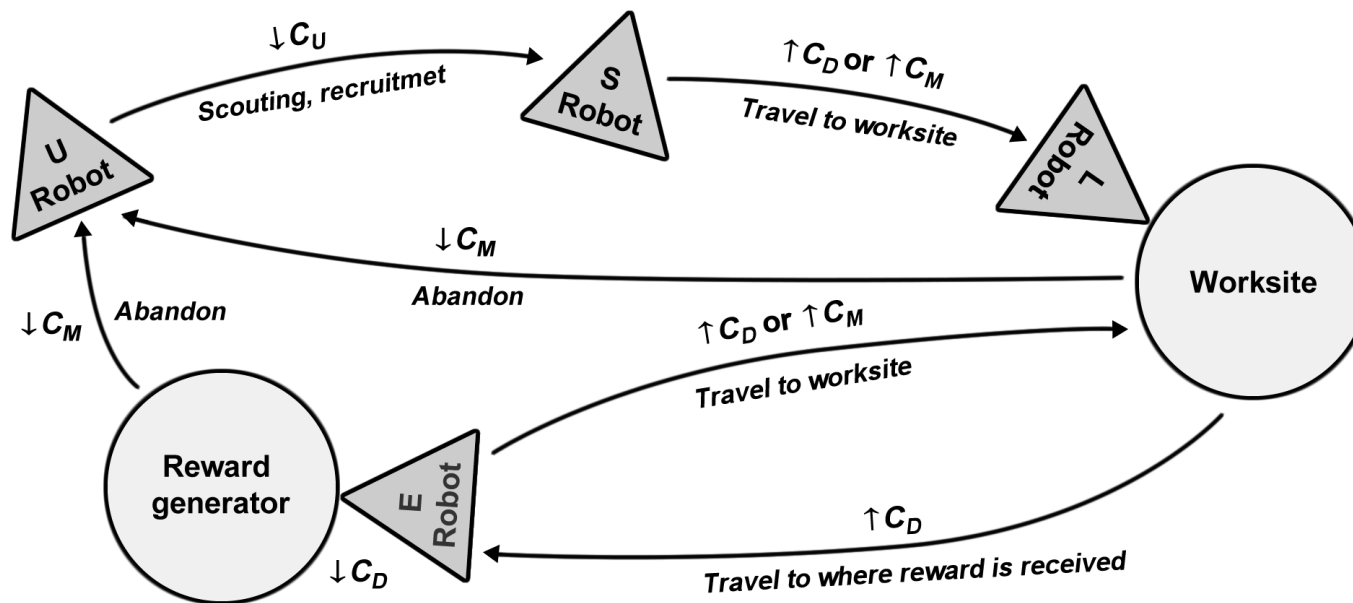
Foraging simulations



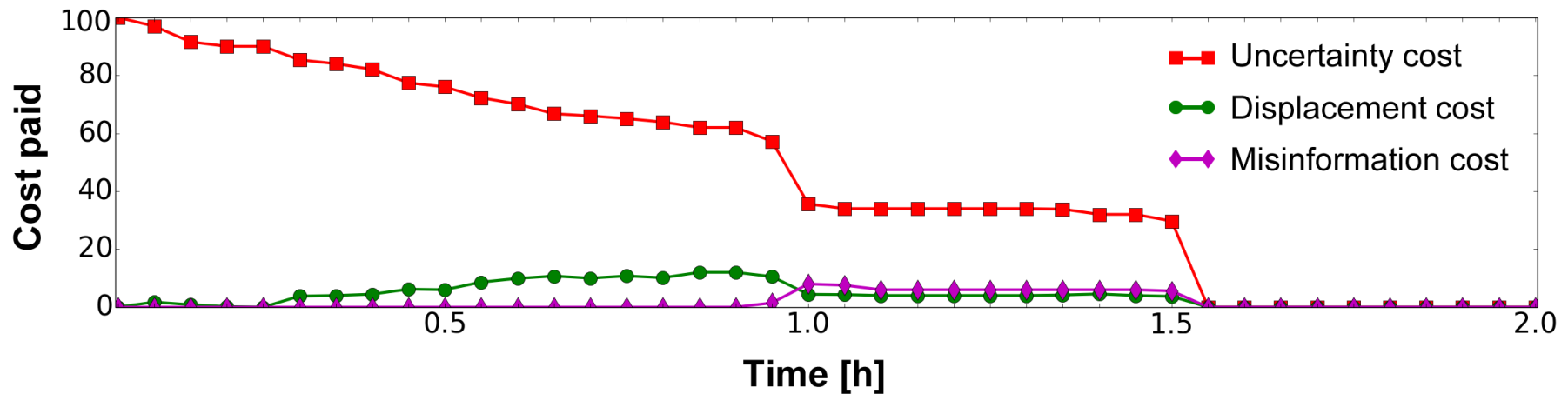
 Solitary swarm  Local broadcasters  Bee swarm

Information-Cost-Reward Framework

Costs: How much reward is not being obtained?



Robot workcycle stages: U = unemployed, S = subscribed, L = laden, E = earning reward
Costs: C_U = uncertainty, C_D = displacement, C_M = misinformation



Local broadcasters, Consumption from two worksites

$$\text{Costs} = R' - R$$

- Account for all possible states robots can be in during foraging
- Identify behaviours that improve / damage robot performance

Behaviour-Data-Relations ML

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



Scout

Work

**Worksite
location**

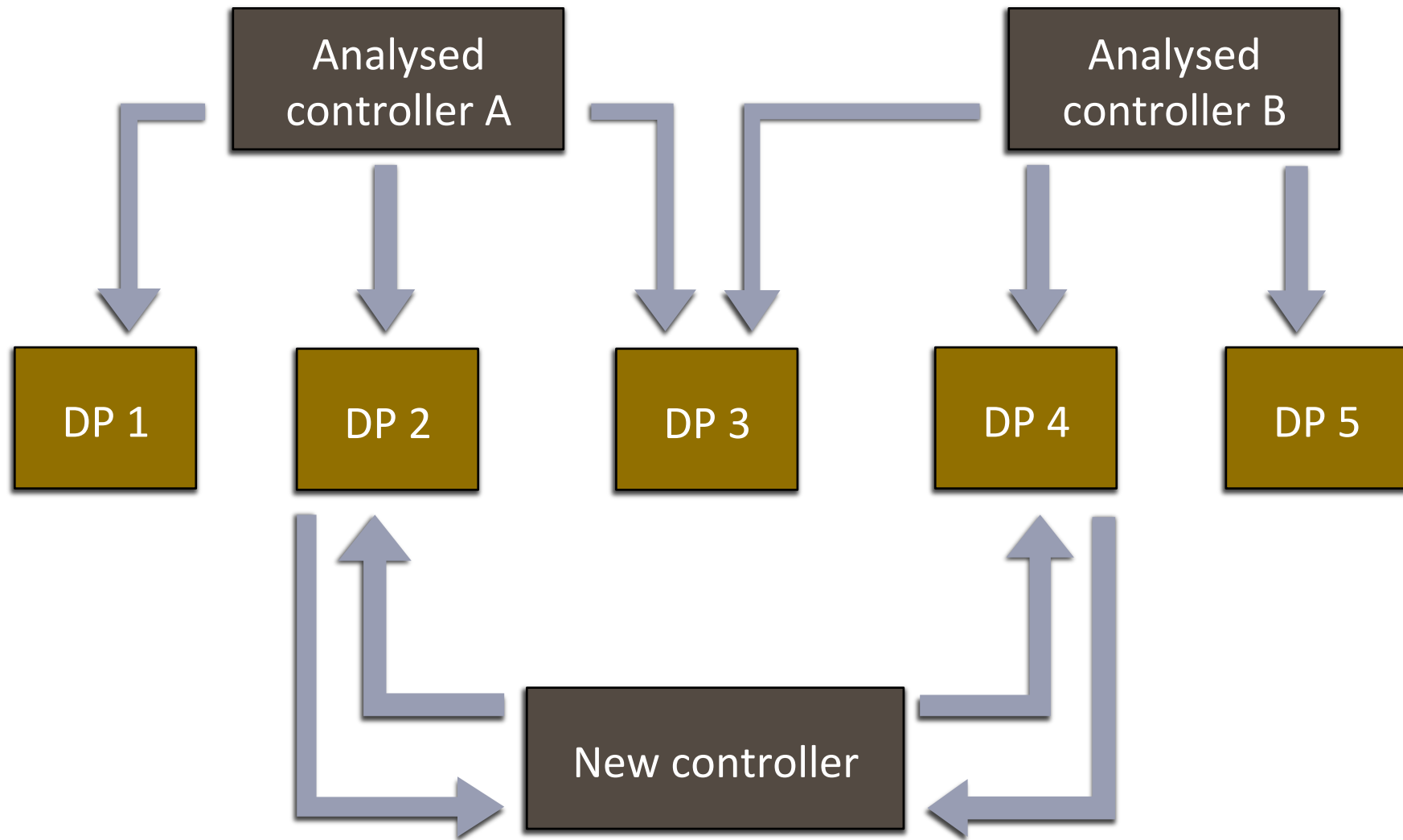
$B = \{\text{Scout}, \text{Work}\}$


$D_i = \{\text{Worksite location} : \text{object}\}$

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

Design patterns for robot swarms

- Describe a particular **stand-alone module of a robot control algorithm**: robot *behaviours*, relevant *internal and external data structures* and *relationships* between them.
- Provide a description of **suitable environments and swarm tasks**, in which the pattern is understood to be an appropriate design choice.
- Be possible to **combine** with other design patterns.
- Be **implementation-generic**: only describe high-level behaviour, rather than an implementation



- 
- **Information transmitter patterns:** what entity transmits or stores information, what information is used by behaviours and under what conditions
 - Individualist
 - Broadcaster
 - Information Storage
 - **Information aggregation patterns:** where information exchange takes place and what behaviours are responsible for the exchange
 - Information Exchange Anywhere
 - Information Exchange at Worksites
 - Information Exchange Centre

Broadcaster

Category: Information Transmitter pattern

Problem:

- Robots need to find and exploit worksites as quickly as possible, but the task characteristics (e.g., robot or worksite density) make it difficult for robots to discover worksites.

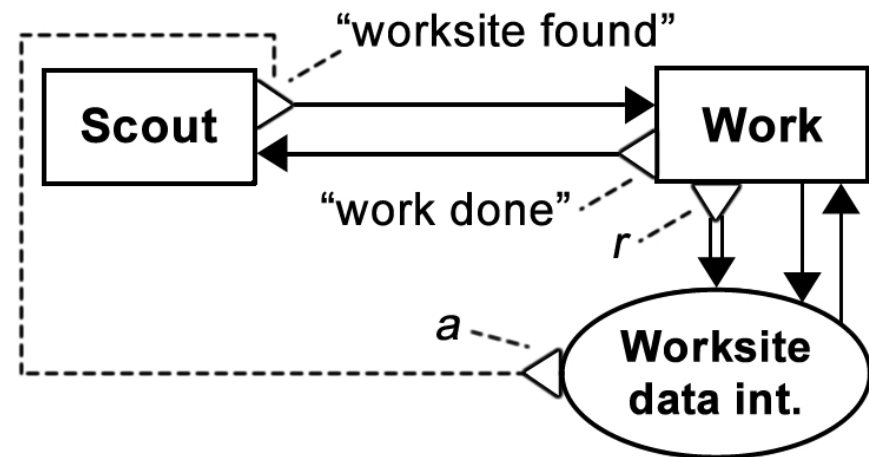
Applicability:

- Robots are capable of directly communicating with each other

Solution:

A robot scouts for worksites in the environment. A successful scout stores information about its worksite, such as its location, in an internal data structure (“Worksite data int.”), and begins work. The data structure may be updated and used periodically while the robot works. A robot that is engaged in the “Work” behaviour may send information about its worksite to another robot, provided that a boolean *recruitment* function, r , returns *true*. When a robot receives worksite data in this way, it stores it in its own internal data structure and transitions from the “Scout” to the “Work” behaviour, provided that a boolean *adoption* function, a , returns *true*.

Not an implementation





Feedback loops:

- Sharing of worksite information represents a positive feedback loop that can be regulated via the pattern's parameters.

Parameters: (Detailed description of each)

Forces: (What affects the pattern's effectiveness)

- A sufficient communication range must be available in order for recruitment to take place, depending on the worksite and robot density

Consequences: (On swarm-level characteristics - ICR framework)

- Causes the robots to incur displacement cost, associated with traveling to worksites after being recruited

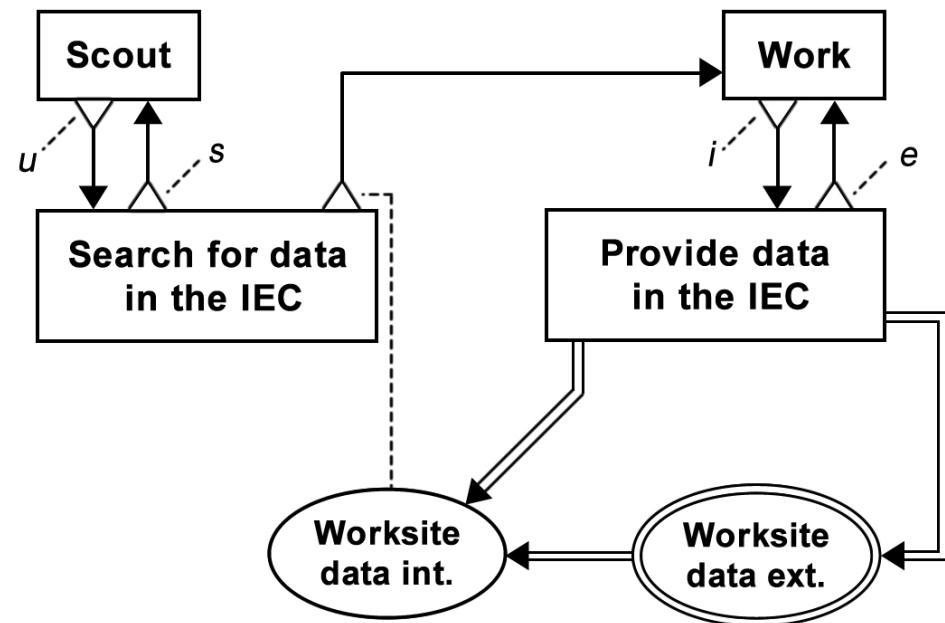
Known uses, Related patterns: Examples from the literature

Information exchange centre

Category: Information Aggregation pattern

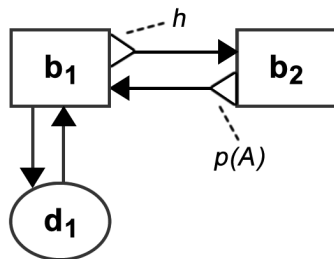
Problem:

- Robots have a low probability of meeting each other during foraging due to low density of worksites and/or robots



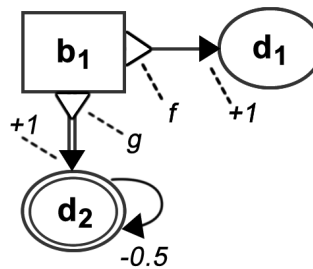
Creating robot control strategies

- A set of BDRML rules for combining multiple patterns into a robot control strategy



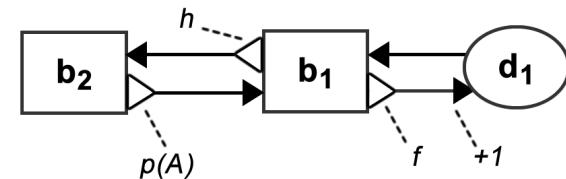
$B = \{b_1, b_2\}$ $D_i = \{d_1: \text{bool}, \text{int}\}$
 $\text{trans}(b_1, b_2) : \{h\}$
 $\text{trans}(b_2, b_1) : \{p(A)\}$
 $\text{write}(d_1, b_1) : \{*\}$
 $\text{read}(d_1, b_1) : \{*\}$

+



$B = \{b_1\}$
 $D_i = \{d_1: \text{int}\}$ $D_e = \{d_2: \text{int}\}$
 $\text{write}(+1: d_1, b_1) : \{f\}$
 $\text{send}(+1: d_2, b_1) : \{g\}$
 $\text{update}(-0.5: d_2) : \{*\}$

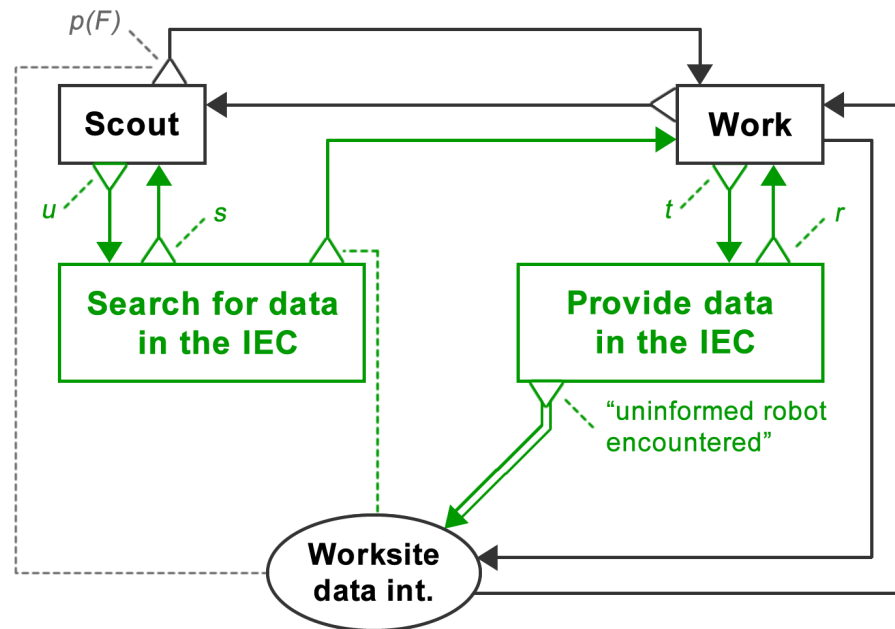
=



$B = \{b_1, b_2\}$ $D_i = \{d_1: \text{bool}, \text{int}\}$ ~~$D_e = \{d_2: \text{int}\}$~~
 $\text{trans}(b_1, b_2) : \{h\}$
 $\text{trans}(b_2, b_1) : \{p(A)\}$
 $\text{write}(+1: d_1, b_1) : \{f\}$
 $\text{read}(d_1, b_1) : \{*\}$
 ~~$\text{send}(+1: d_2, b_1) : \{g\}$~~
 ~~$\text{update}(-0.5: d_2) : \{*\}$~~

A bee-inspired robot controller

Broadcaster + Information Exchange Centre



```

V = {Scout, Search for data in the IEC, Work, Provide data in the IEC, Worksite data int.,
Worksite data ext.}
trans(Scout, Work) : { $p(F)$ ,  $\exists$  Worksite data int.}
trans(Work, Scout) : {"work done"}
write(Worksite data int., Work) : {"*"}
read(Worksite data int., Work) : {"*"}
send(Worksite data int., Work) : {"uninformed robot encountered"}
trans(Scout, Search for data in the IEC) : { $u$ }
trans(Search for data in the IEC, Scout) : { $s$ }
trans(Search for data in the IEC, Work) : { $\exists$  Worksite data int.}
trans(Work, Provide data in the IEC) : { $t$ }
trans(Provide data in the IEC, Work) : { $r$ }
send(Provide data in the IEC, Worksite data int.) : {"uninformed robot encountered"}
send(Worksite data ext., Provide data in the IEC) : {"*"}
copy(Worksite data ext., Worksite data int.) : { $\#$  Worksite data int.}
    
```

Design pattern applications

- **Reusing** known solutions to a class of similar problems
- **Checking** whether a particular behaviour is suitable, given hardware constraints
Can I use a recruitment strategy that relies on large communication range?
- Guide **adaptation**
Should robots change their behaviour to a more suitable one, given their current environment?
- **Guide artificial evolution** and make its search space smaller

Thank you

- **Project website:**
<http://designing-robot-swarms.lenkaspaces.net>
- Design patterns paper pre-print:
<http://lenkaspaces.net/publications>
- Reach me on **contact@lenkaspaces.net**