Self-Organized Task Allocation to Sequentially Interdependent Tasks in Swarm Robotics

Jimmy Lee, Sawyer Jager, Brad Steiner
Team Amirite?

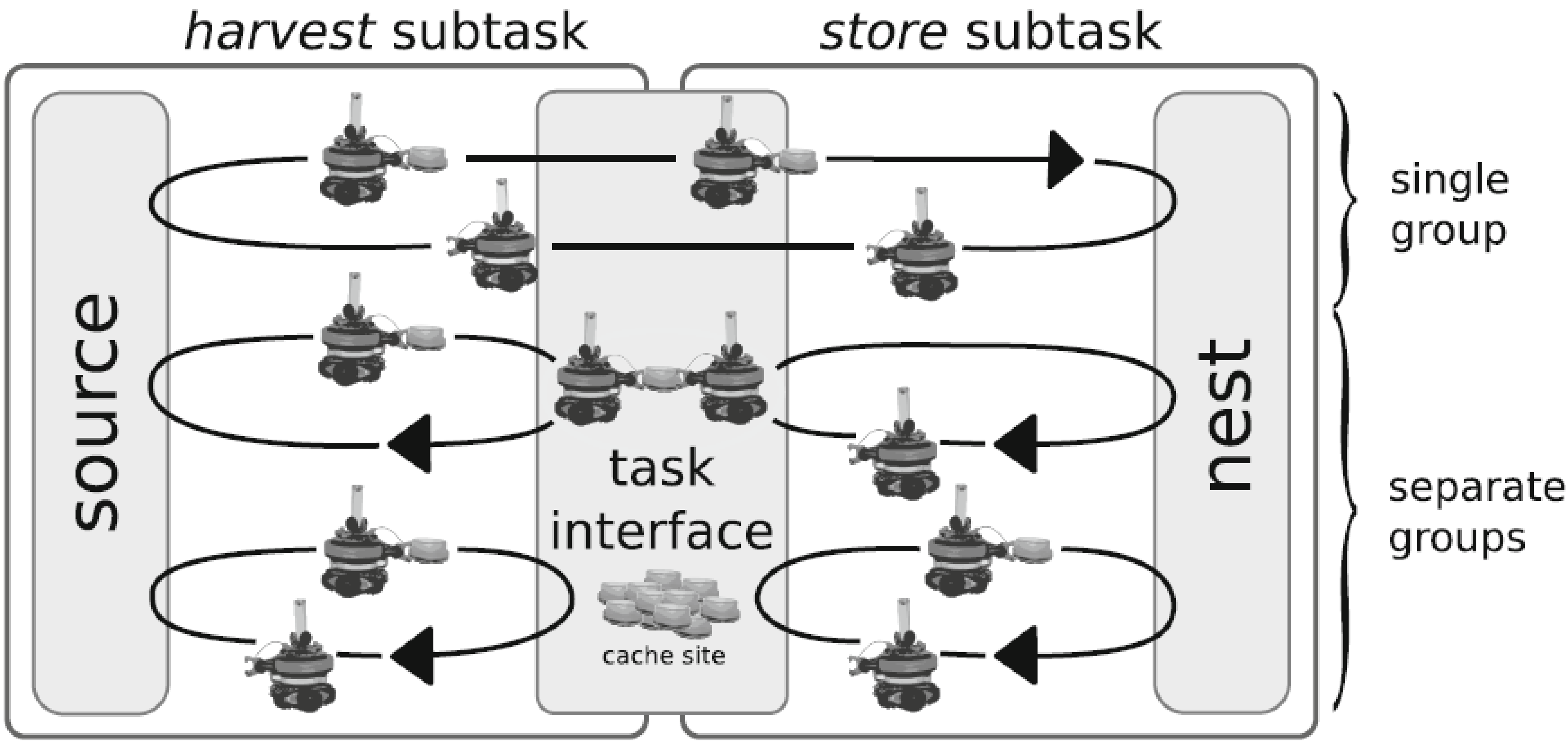
Arne Brutschy, Giovanni Pini, Carlo Pinciroli, Mauro Birattari, and Marco Dorigo (2014). Self-Organized Task Allocation to Sequentially Interdependent Tasks in Swarm Robotics, Autonomous Agents and Multiagent Systems, 28(1):290-336. (Brutschyetal2014.pdf)

How can a swarm of robots allocate themselves to efficiently complete sequentially interdependent tasks?

Sequentially interdependent tasks:

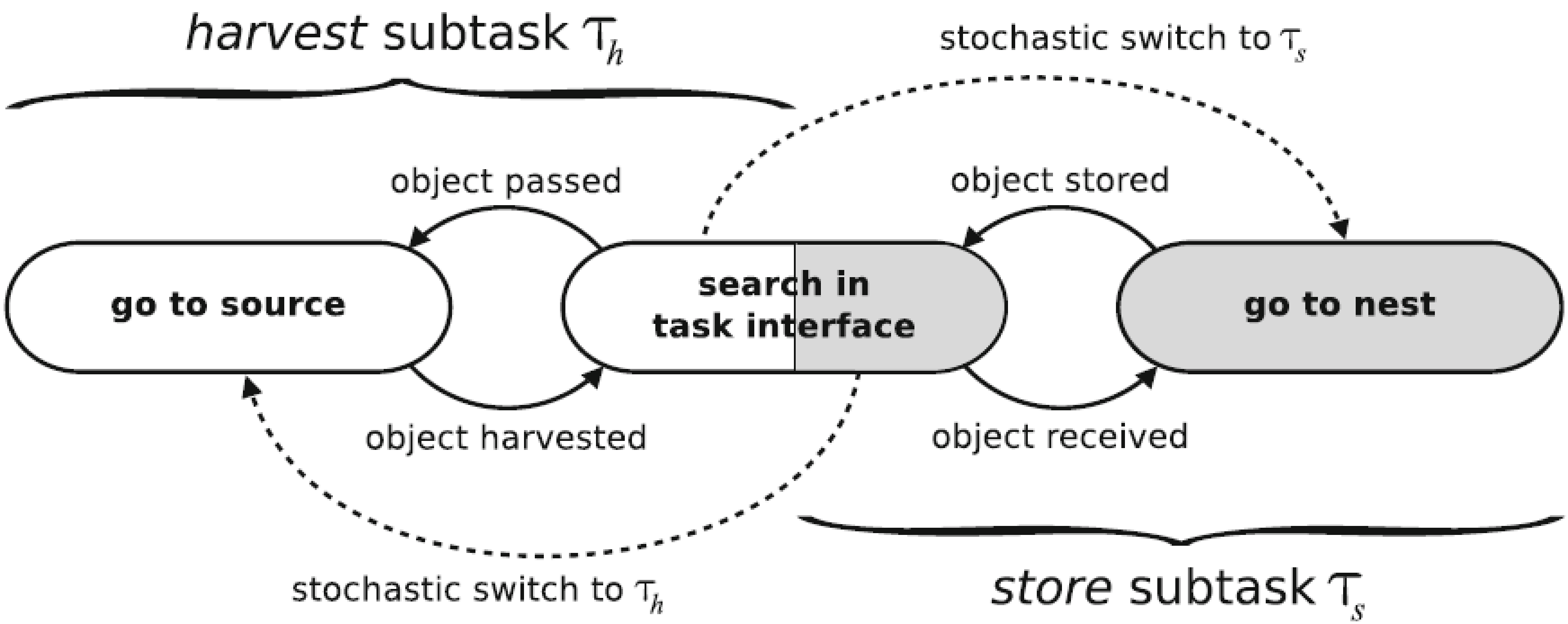
A set of subtasks that must be completed ***one after the other*** to complete the overall task

Setup - Harvesting and Storing



Proposal

- Goal: Maximize number of objects retrieved per time unit
- Robots independently assign themselves to subtasks based on how long they must wait for a robot from the other task to arrive at the task interface.
- Increasing probability of a robot switching from one subtask to another one while waiting at the interface
- Intuitively, a relatively longer wait at the interface indicates that the other subtask is understaffed.
- End result: rates of arrival at the interface are equalized



harvest subtask τ_h
stochastic switch to τ_s
object passed
object stored
go to source
search in task interface
go to nest
object harvested
object received
stochastic switch to τ_h
store subtask τ_s

Proposal

- Does not rely on global knowledge or centralized components

- Does not use any communication

- Self organized

Formal Definitions

Formal Definitions - Tasks/Subtasks

- A task T is composed of subtasks $\tau_1 \ldots \tau_n$
- Sequential interdependence means $\tau_1 \ldots \tau_n$ must be completed in a given order
    - T can also be called a “task sequence”
- $\tau_1 > \tau_2$: τ_1 must be completed before τ_2
    - τ_1 is a predecessor
    - τ_2 is a successor

Formal Definitions - Allocation

- g_i : A group of robots working on τ_i
- N_i : Number of robots in g_i
- $N_1 + N_2 + \ldots + N_n = N$: total number of robots
  - There are no idle robots

Formal Definitions - Process

A robot allocated to τ_i waiting at the interface for a robot from τ_j experiences ***interface delay*** denoted d_{ij} (seconds)

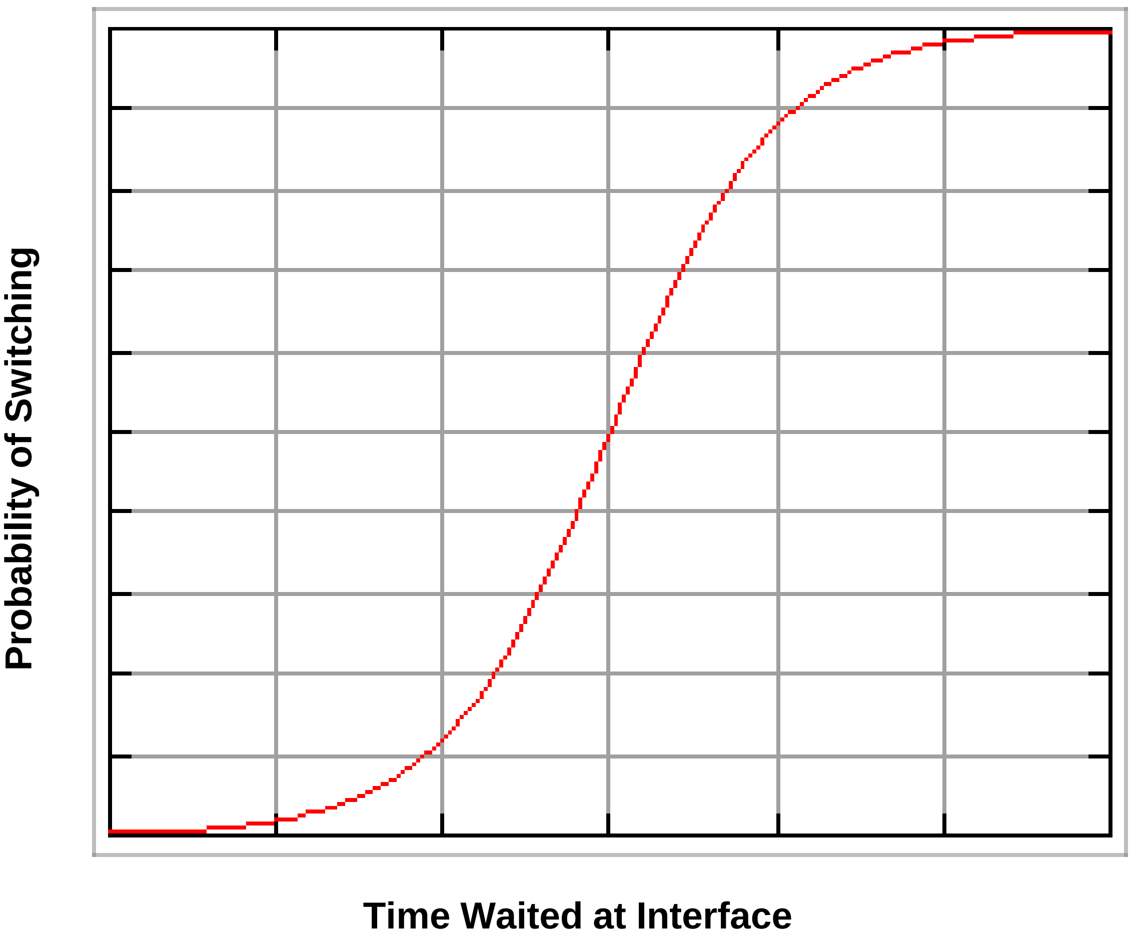
Robots keep track of how long they've waited at either end of the interface. Average wait time of d_{ij} denoted $\hat{d}_{ij}$

Probability of a robot switching from τ_i to τ_j before d_{ij} seconds have elapsed: $P_{ij}(d_{ij})$

- $1 - P_{ij}(d_{ij})$: Probability a robot will wait d_{ij} seconds
- Formally, $P_{ij}(d_{ij} ; \hat{d}_{ij} , \hat{d}_{ji})$

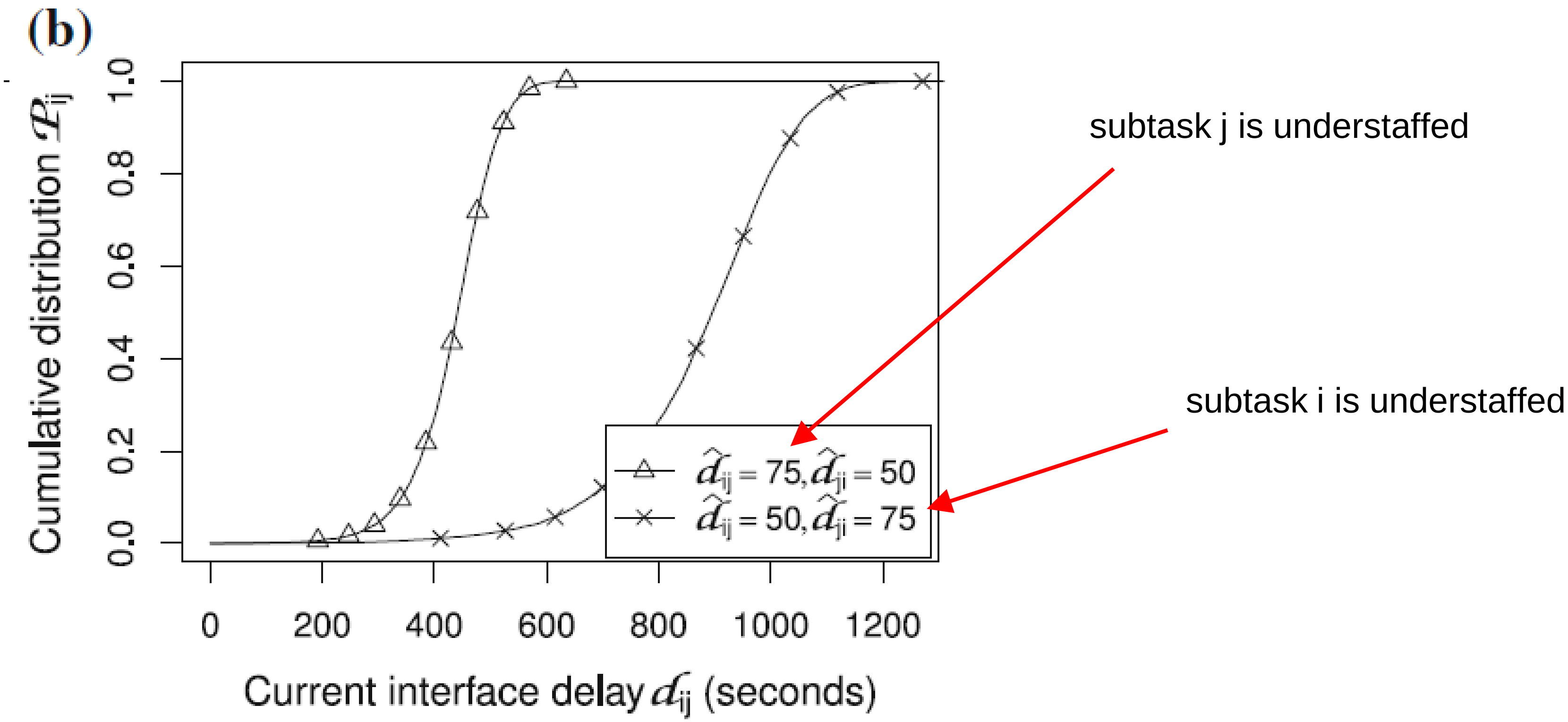
Probability Function

Sigmoid Curve



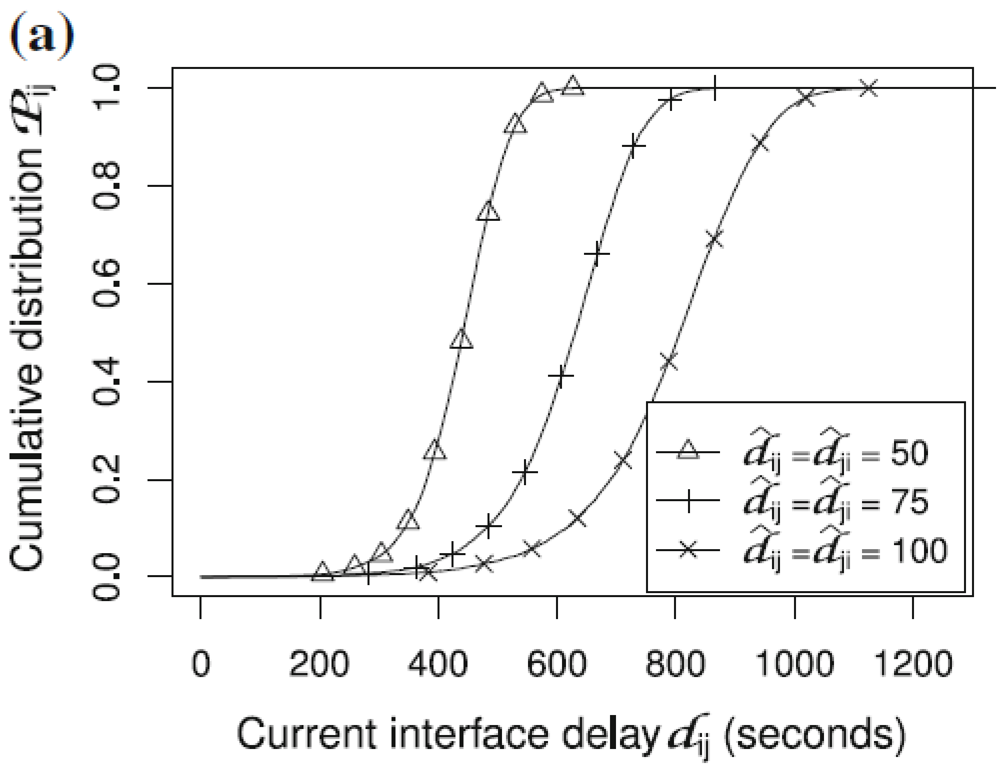
Probability Function

Designed so that robots tend to switch to the subtask with a longer wait time quicker



Probability Function

Function is independent of absolute values of subtask duration and delays



Probability curve automatically calibrates itself

If wait times are generally longer, the probability of switching doesn't increase until more time has passed

Probability Function Formal Def.

Robots evaluate probability function at discrete points in time (dictated by control cycles).

At each control cycle, the robot switches from one task to another with a probability of p_{ij} (switching probability, not the same as P_{ij})

Probability Function Formal Def.

Probability that the robot does not switch subtasks within the first d_{ij} control cycles.

$$P_{ij}(d_{ij} ; \hat{d}_{ij} , \hat{d}_{ji}) = 1 - \prod_{q=1}^{d_{ij}} (1 - p_{ij}(q ; \hat{d}_{ij} , \hat{d}_{ji}))$$

$$p_{ij}(d_{ij}; \hat{d}_{ij}, \hat{d}_{ji}) = \frac{1}{1 + e^{-\theta(d_{ij}; \hat{d}_{ij}, \hat{d}_{ji})}} \cdot \gamma$$

$$\theta(d_{ij}; \hat{d}_{ij}, \hat{d}_{ji}) = \frac{1}{k} \left(\frac{d_{ij}}{r(\hat{d}_{ij}, \hat{d}_{ji})} - m \right)$$

$$r(\hat{d}_{ij}, \hat{d}_{ji}) = \frac{\hat{d}_{ij} \cdot \max(\hat{d}_{ij}, \hat{d}_{ji})}{\hat{d}_{ji}} = \begin{cases} \hat{d}_{ij}^2 / \hat{d}_{ji} & \text{if } \hat{d}_{ij} > \hat{d}_{ji} \\ \hat{d}_{ij} & \text{if } \hat{d}_{ij} \le \hat{d}_{ji} \end{cases}$$

Probability Function Formal Def.

$$\theta(d_{ij}; \hat{d}_{ij}, \hat{d}_{ji}) = \frac{1}{k}\left(\frac{d_{ij}}{r(\hat{d}_{ij}, \hat{d}_{ji})} - m\right)$$

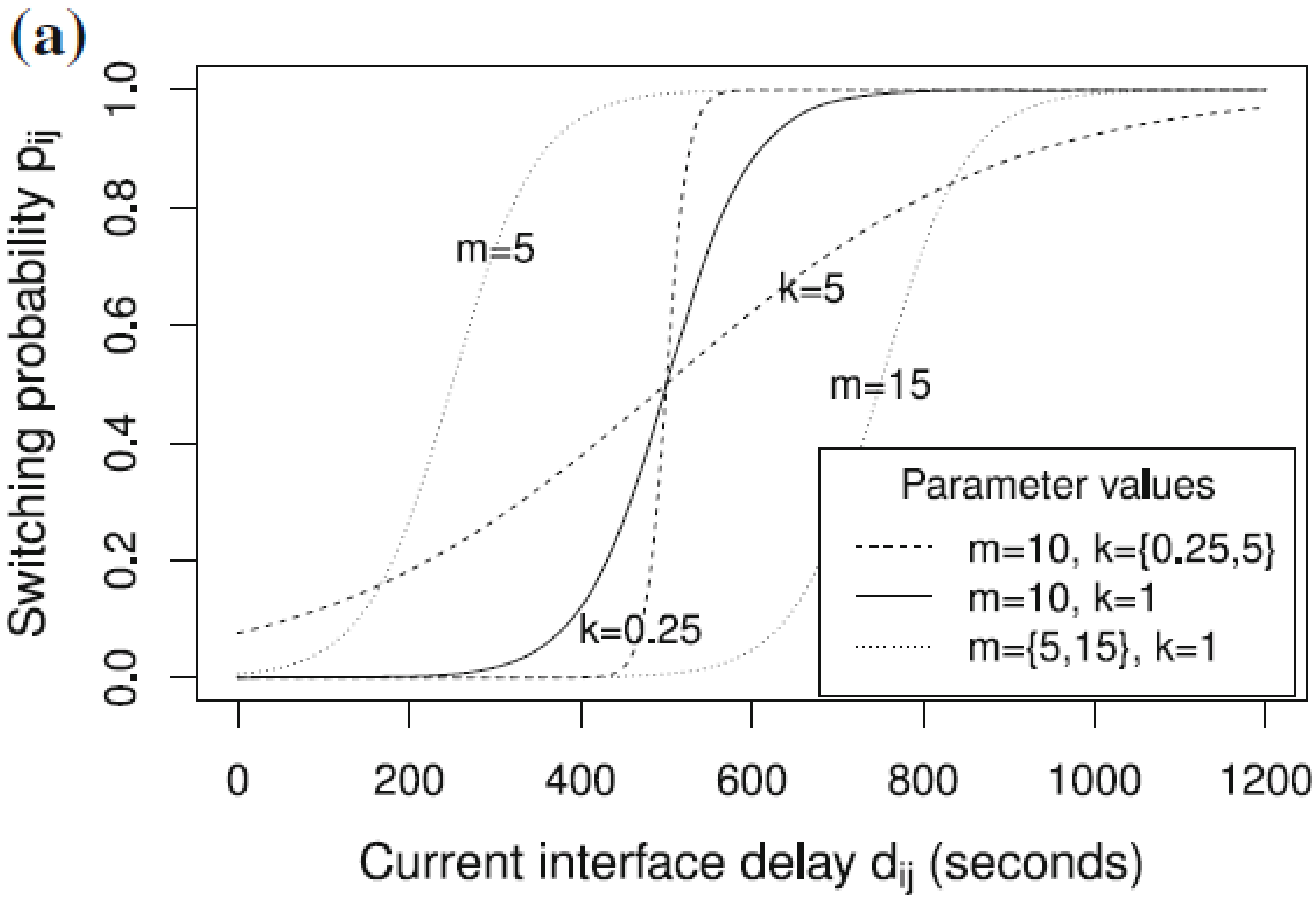
k = steepness parameter

lower k => steeper curve

m = shift parameter

higher m => delay longer

Steepness and Shift



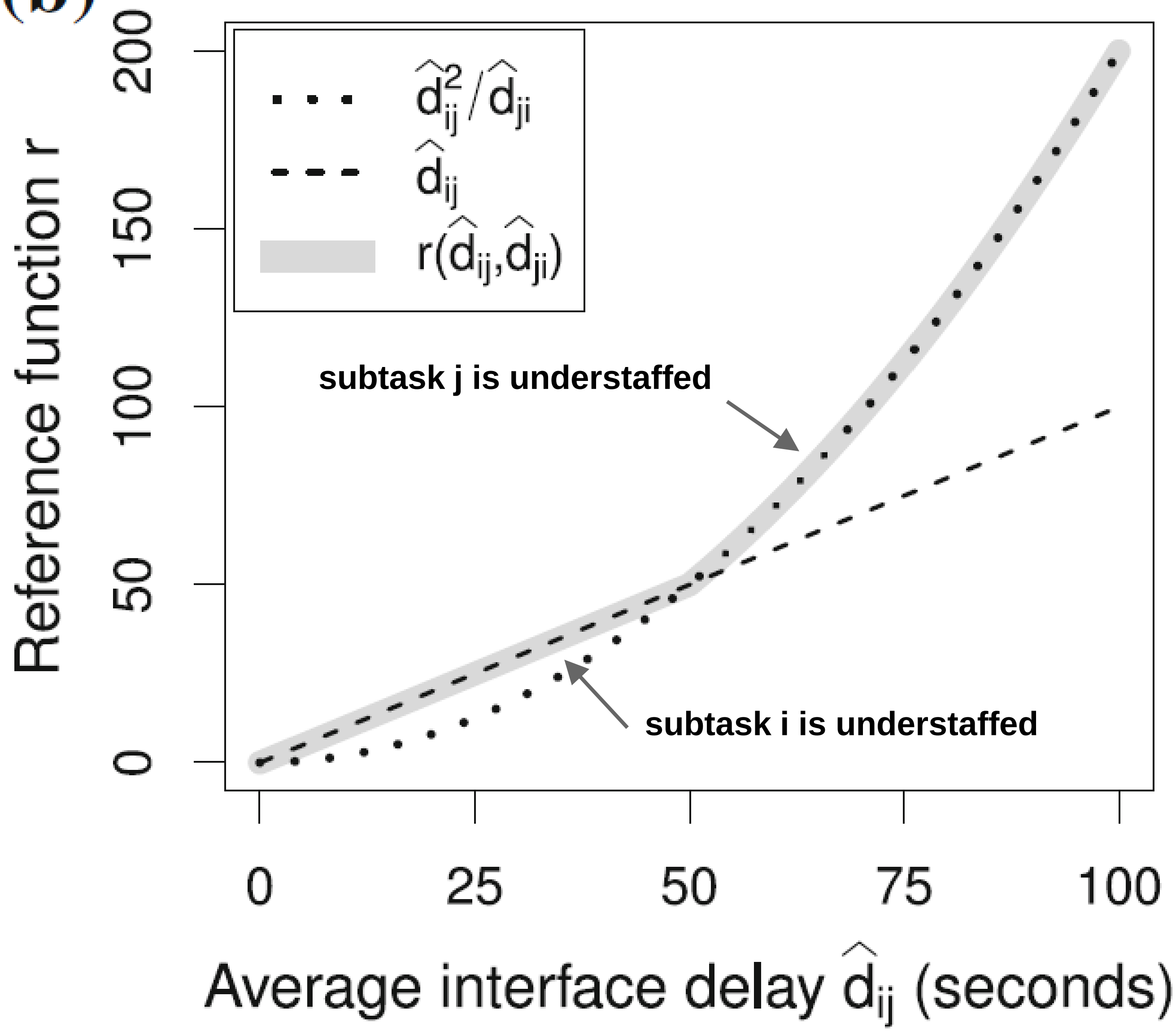
Probability Function Formal Def.

$$r(\hat{d}_{ij}, \hat{d}_{ji}) = \frac{\hat{d}_{ij} \cdot \max(\hat{d}_{ij}, \hat{d}_{ji})}{\hat{d}_{ji}}$$
$$= \begin{cases} \hat{d}_{ij}^2 / \hat{d}_{ji} & \text{if } \hat{d}_{ij} > \hat{d}_{ji} \\ \hat{d}_{ij} & \text{if } \hat{d}_{ij} \leq \hat{d}_{ji} \end{cases}$$

Relates the average interface times

Ensures robots tend to switch to the understaffed group and not the other way around.

(b)

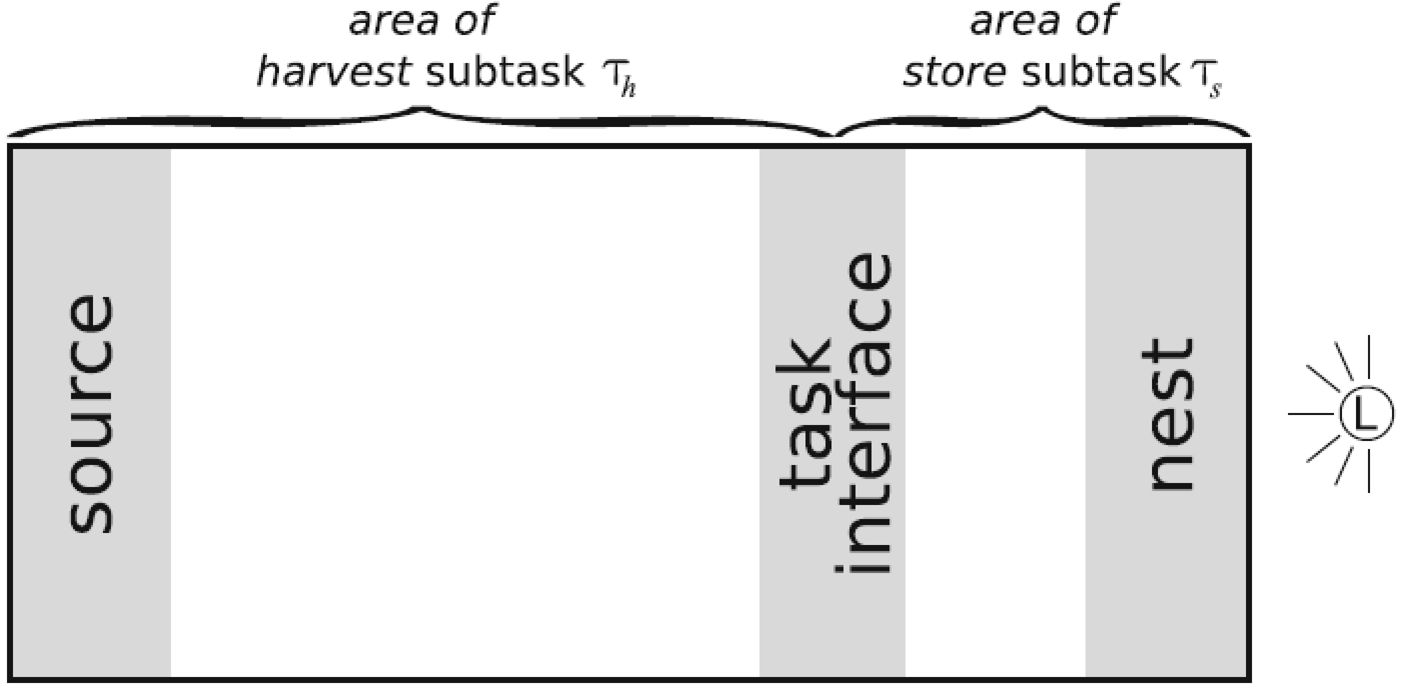


200
150
100
50
0
Reference function r
$\hat{d}_{ij}^2/\hat{d}_{ji}$
$\hat{d}_{ij}$
$r(\hat{d}_{ij},\hat{d}_{ji})$
subtask j is understaffed
subtask i is understaffed
0
25
50
75
100
Average interface delay $\hat{d}_{ij}$ (seconds)

Experiment

Experimental Framework

Shown:
Asymmetric environment (arena ratio = .33)



Experimental Framework

Two subtasks: harvesting and storing

Two environments: symmetric and asymmetric (.67)

Resources must be harvested before they can be stored

Resources cannot be cached at interface area (direct handoffs only)

Robots that switch subtasks experience a switching cost, c_s

Task switching only occurs at the task interface

Metrics

P : Swarm Performance = # of objects collected by the swarm

R : allocation ratio = fraction of robots allocated to storage subtask

S_{tot} = total number of switches performed by robots in a run of the simulation.

Experiments

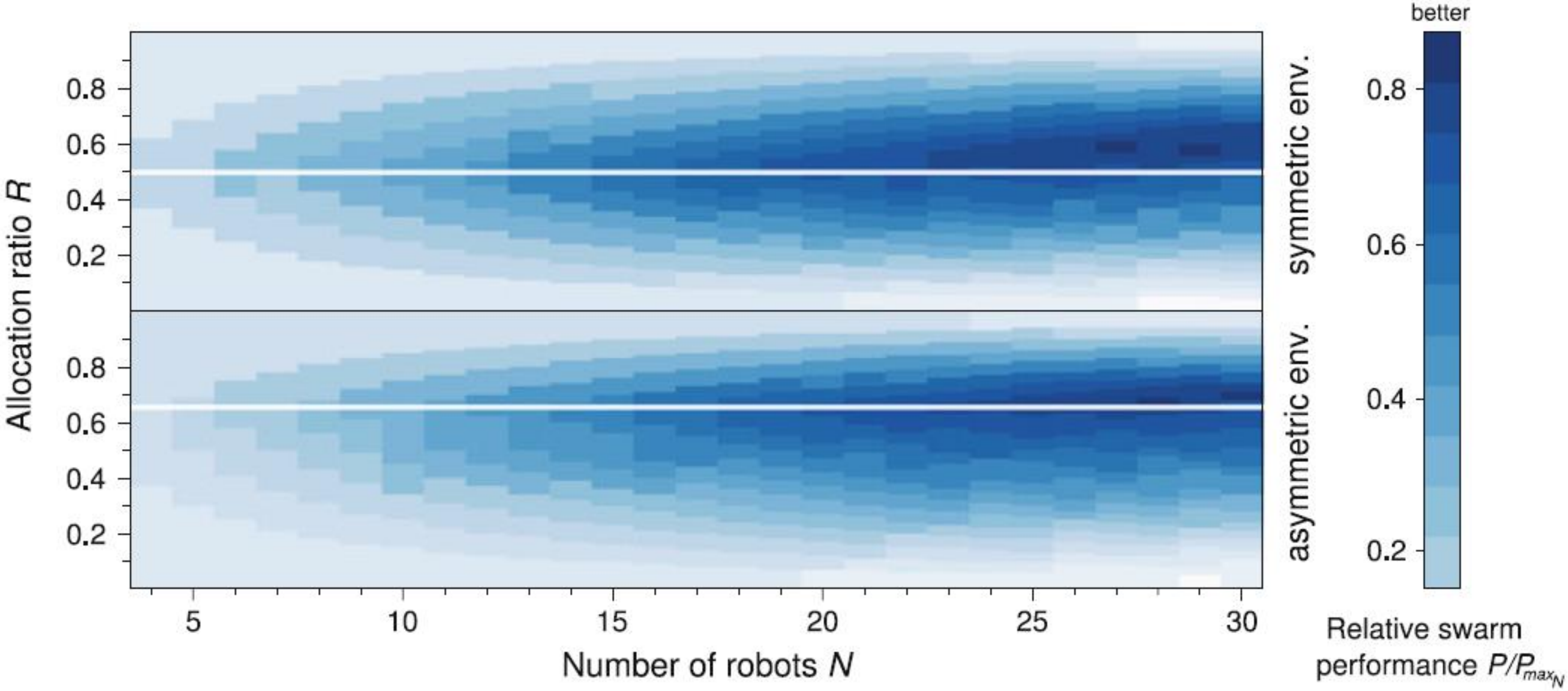
1. Optimal Allocation
2. Task Switching Cost
3. Parameter Study
4. Scalability
5. Adaptivity

1. Optimal Allocation

Brute force search for the optimal ratio of allocation (R_{opt}) for various numbers of robots

Robots are not allowed to switch subtasks

Symmetric Environment



Asymmetric Environment

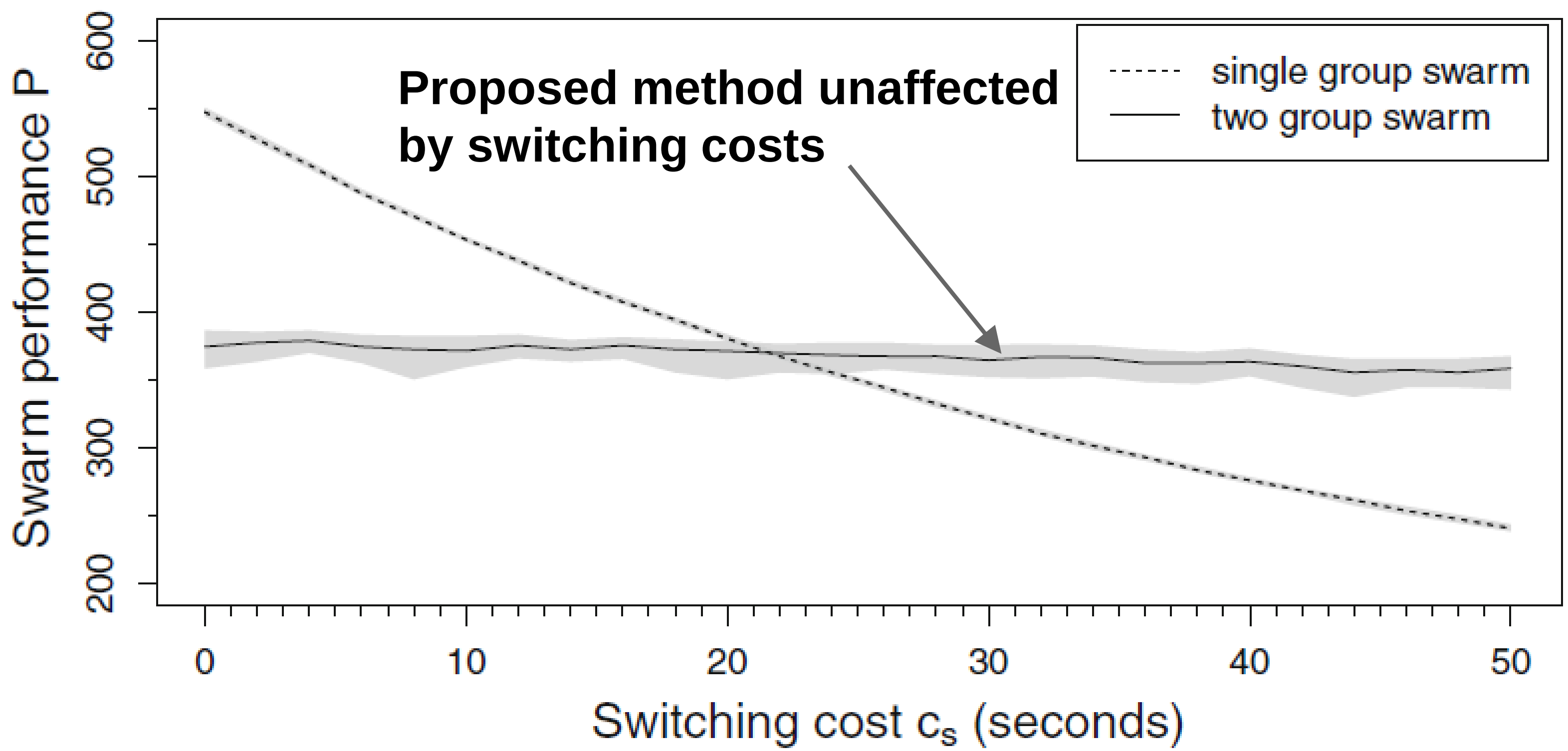
2. Task switching cost

Examine influence of task switching cost

Experiment with two groups:

- Group 1: Each robot required to complete both subtasks
- Group 2: Robots switch using the proposed method

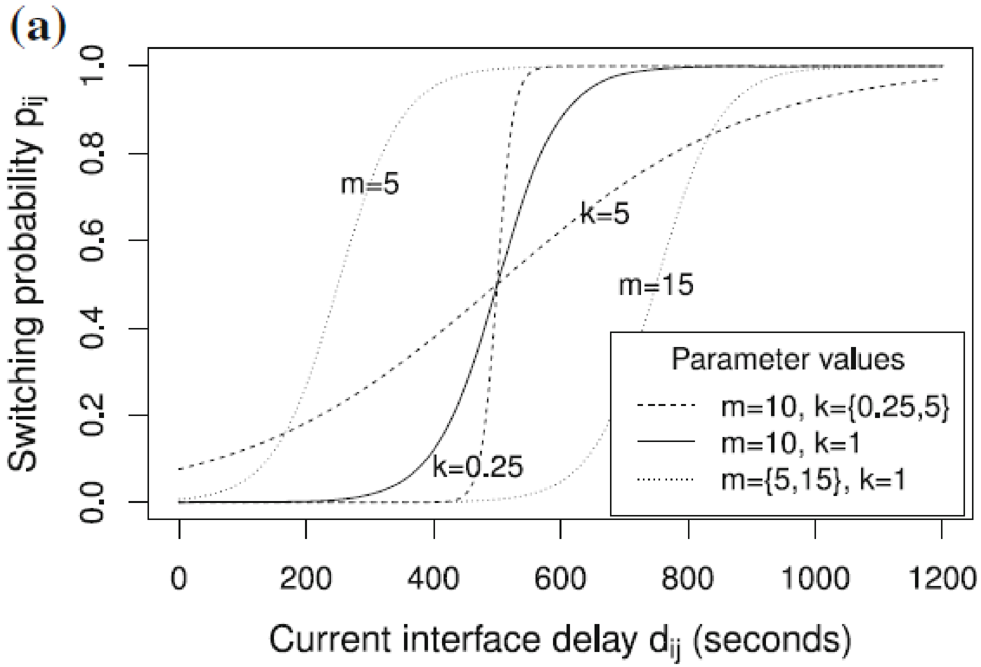
N = 18

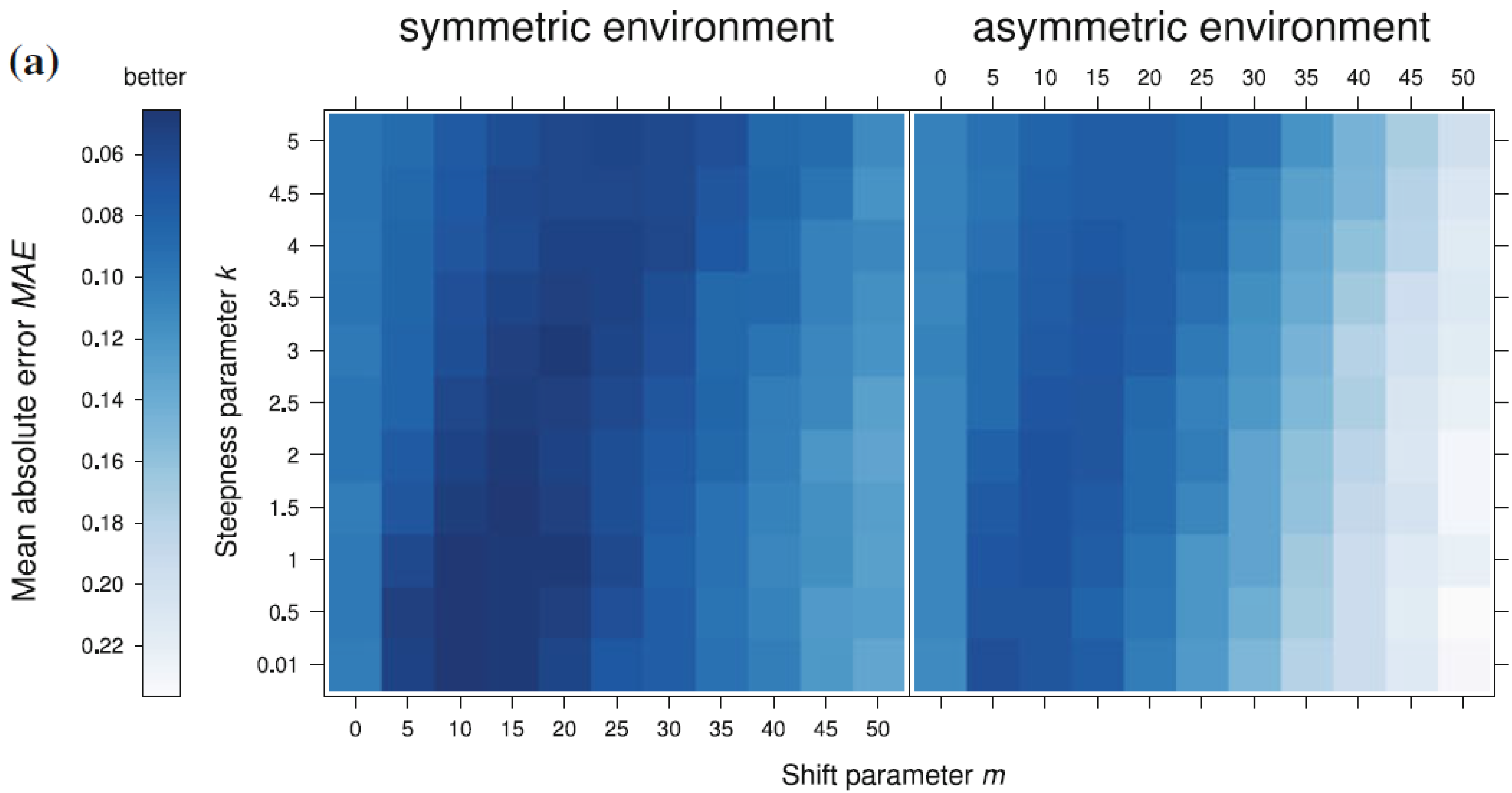


Proposed method unaffected by switching costs
single group swarm
two group swarm
Swarm performance P
200
300
400
500
600
0
10
20
30
40
50
Switching cost c_s (seconds)

3. Parameter Study

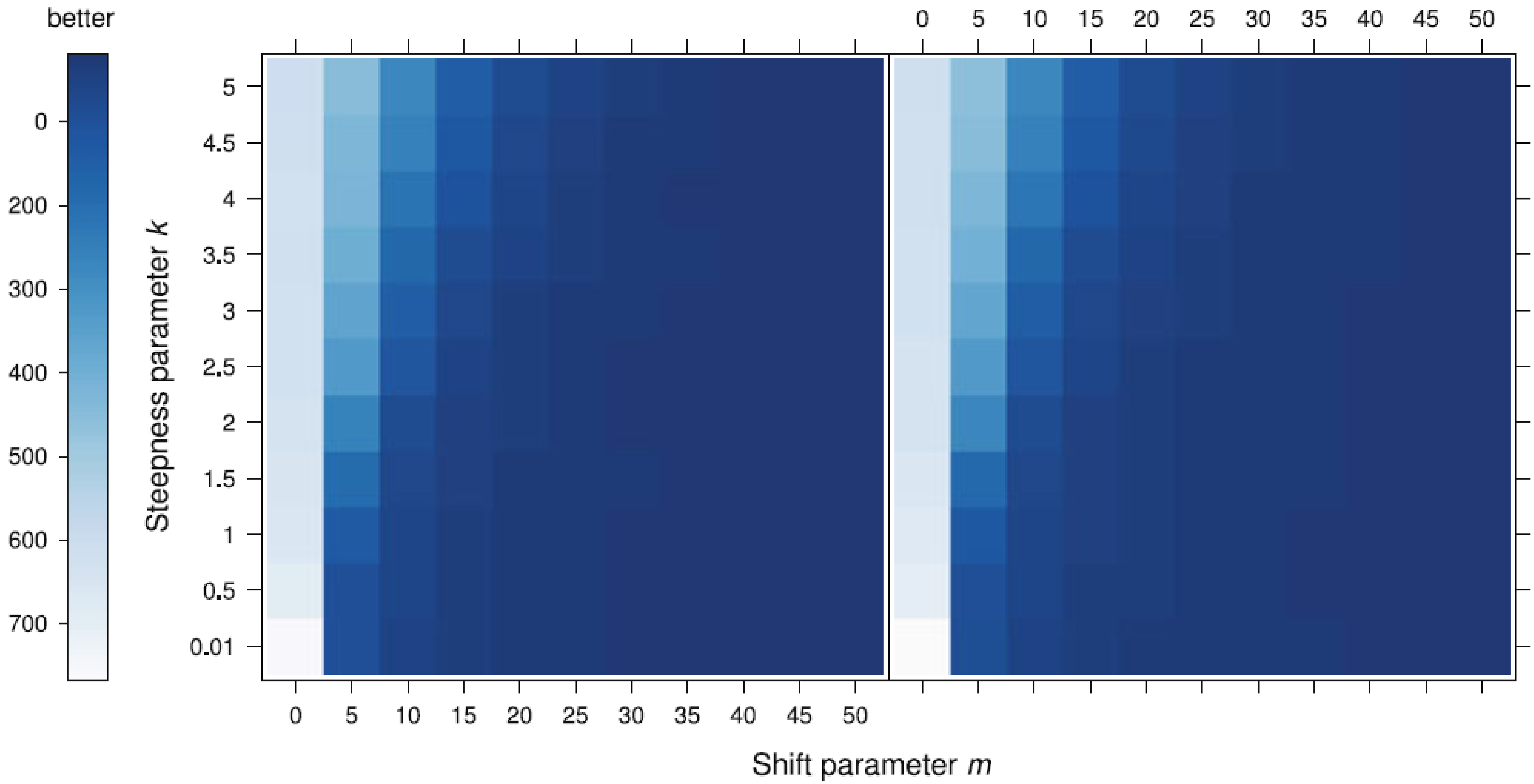
Explore the effects of m (shift parameter) and k (steepness parameter) on performance





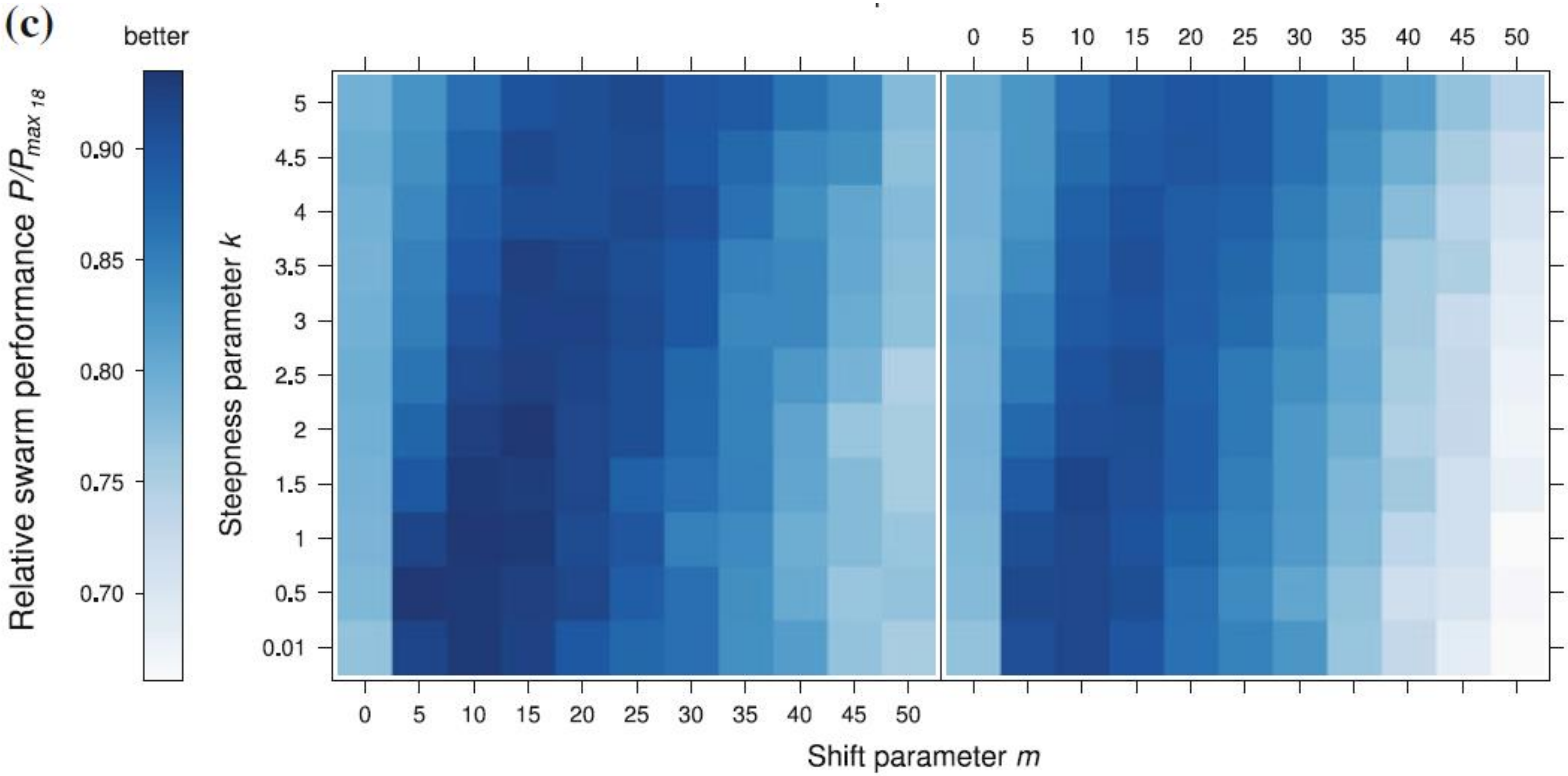
(a)
symmetric environment
asymmetric environment
better
Mean absolute error MAE
0.06
0.08
0.10
0.12
0.14
0.16
0.18
0.20
0.22
Steepness parameter k
5
4.5
4
3.5
3
2.5
2
1.5
1
0.5
0.01
0 5 10 15 20 25 30 35 40 45 50
Shift parameter m

(b)

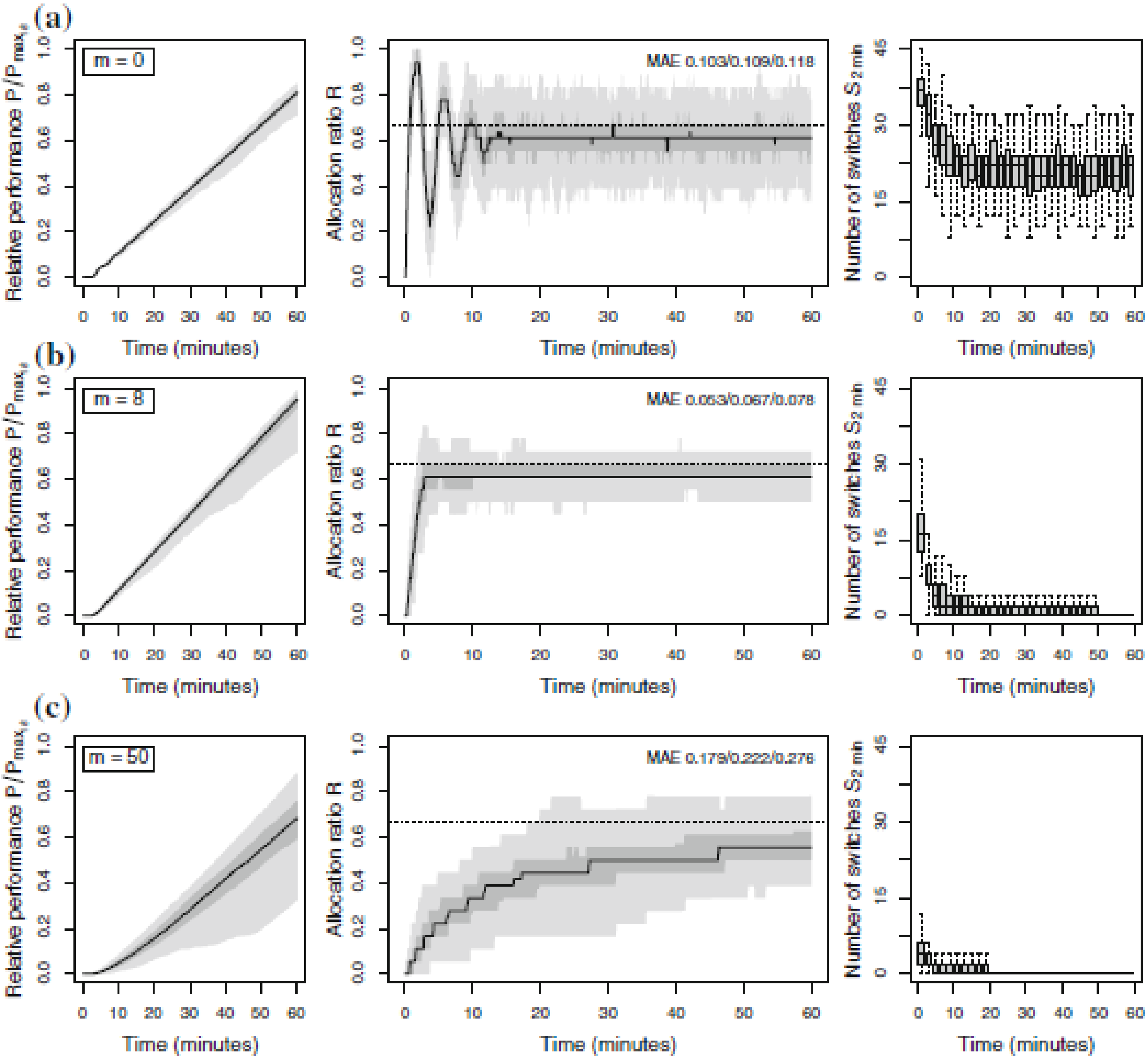


better
Number of switches S_{tot}
0
200
300
400
500
600
700
Steepness parameter k
5
4.5
4
3.5
3
2.5
2
1.5
1
0.5
0.01
0 5 10 15 20 25 30 35 40 45 50
0 5 10 15 20 25 30 35 40 45 50
Shift parameter m

(c)



better
Relative swarm performance $P/P_{max\ 18}$
0.90
0.85
0.80
0.75
0.70
Steepness parameter k
5
4.5
4
3.5
3
2.5
2
1.5
1
0.5
0.01
0 5 10 15 20 25 30 35 40 45 50
Shift parameter m

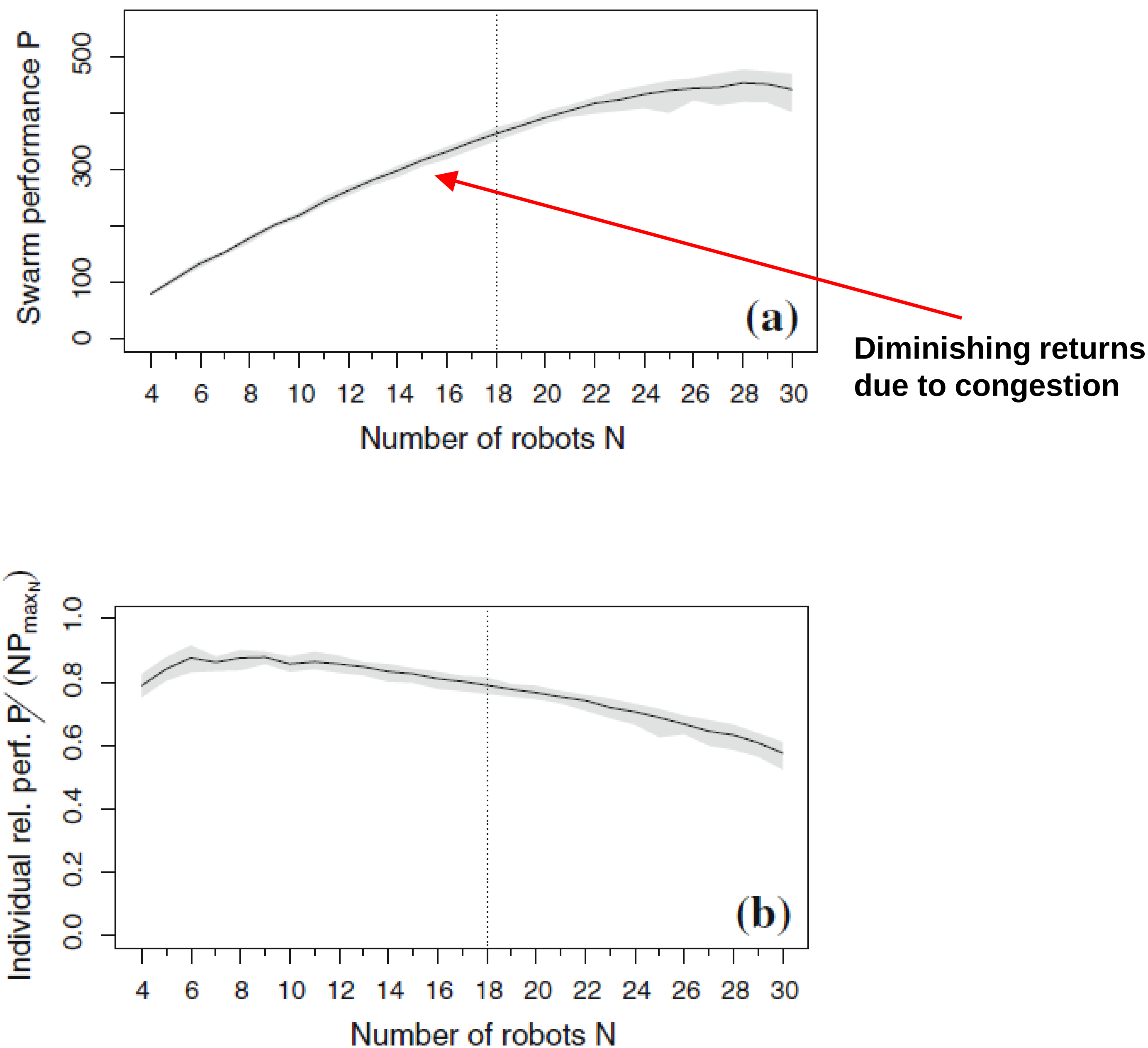


(a)
m = 0
MAE 0.103/0.109/0.118
(b)
m = 8
MAE 0.063/0.067/0.078
(c)
m = 50
MAE 0.179/0.222/0.276
Relative performance P / P max
Allocation ratio R
Number of switches S2 min
Time (minutes)

4. Scalability

Examine how increasing the number of robots affects performance

Physical space is considered though this is a virtual simulation



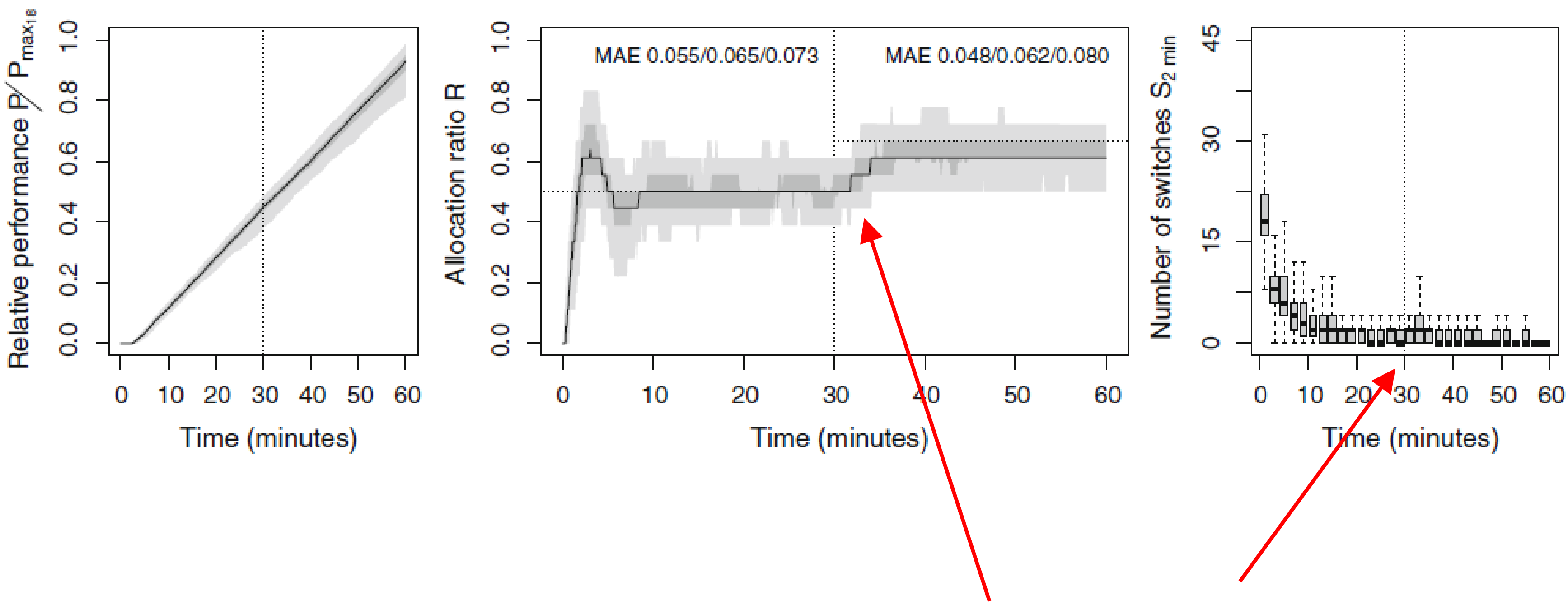
Swarm performance P
0
100
300
500
4
6
8
10
12
14
16
18
20
22
24
26
28
30
Number of robots N
(a)
Diminishing returns due to congestion
Individual rel. perf. P/ (NP_{maxN})
0.0
0.2
0.4
0.6
0.8
1.0
4
6
8
10
12
14
16
18
20
22
24
26
28
30
Number of robots N
(b)

5. Adaptivity

Test flexibility of the proposed method

Change the environment halfway through the simulation

@ t = 30, symmetric => asymmetric



Quick adaptation

Real Simulation

http://iridia.ulb.ac.be/supp/IridiaSupp2011-002/sbot_experiment_run1_30x.ogv

Conclusions

- Proposed method allows for self-organization
- Does not require communication
- Achieves near-optimal allocation
- Environment specific factors have very little influence
- Adaptive

Questions?

No questions?

Then we have questions for you!
Kidding.