Remember the Past and Forget Thresholds

Anna Dornhaus (University of Arizona),
Nicole Leitner (Washington University),
Nancy Lynch (MIT),
Frederik Mallmann-Trenn (MIT $\rightarrow$ King's College London),
Dominik Pajak (Wroclaw University)

Task Allocation

(a) Foraging

(b) Brood care

(c) Farming aphids

(d) Cultivating fungi

How are ants allocated to tasks?

- Age-based: young ants starting doing task in the nest (e.g., brood care and nest maintenance), when they are older they perform tasks outside (e.g., foraging, nest defense, ...)

How are ants allocated to tasks?

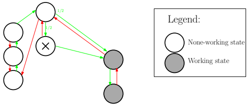
- Age-based: young ants starting doing task in the nest (e.g., brood care and nest maintenance), when they are older they perform tasks outside (e.g., foraging, nest defense, ...)
- Foraging for work: Ants don't have preferences; external space dictates what task an ant works on

How are ants allocated to tasks?

- Age-based: young ants starting doing task in the nest (e.g., brood care and nest maintenance), when they are older they perform tasks outside (e.g., foraging, nest defense, ...)
- Foraging for work: Ants don't have preferences; external space dictates what task an ant works on
- Response thresholds: Different ants have different thresholds for each task, i.e., they perform the task when the stimulus is larger than the threshold.

How are ants allocated to tasks?

- Age-based: young ants starting doing task in the nest (e.g., brood care and nest maintenance), when they are older they perform tasks outside (e.g., foraging, nest defense, ...)
- Foraging for work: Ants don't have preferences; external space dictates what task an ant works on
- Response thresholds: Different ants have different thresholds for each task, i.e., they perform the task when the stimulus is larger than the threshold.
- State machines: Ants have different states and transition from one state to another based on the interaction with the environment.

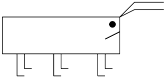


Response Thresholds

- Wilson (76): Observed that ants (*P. dentata*) split into two castes (majors and minors) where each caste performs a different set of tasks.
- Bonabeau et al. (96), study a theoretical version of this. One tasks and different thresholds θ_1, θ_2 for the two castes. Distribution of majors and minors agrees with experimental data of Wilson.
- Page et al. (98) use a slightly different model and assumes that the thresholds are drawn from different distribution.
- Evolution of thresholds has also been studied.

Let’s model this!

- We’ll model ants and tasks!



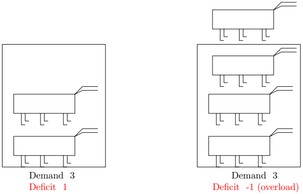
An accurate depiction of the *Squarus Formica*

Model - Tasks

- We have k tasks.
- demand: Each task i $\mathrm{demand}^{(j)}$, i.e., number of ants required to work on task j.
- load: $\mathrm{load}_j^{(t)}$ is the number of ants working on it at time t.
- deficit: $\mathrm{deficit}_j^{(t)}$ is the $\mathrm{demand}^{(j)} - \mathrm{load}_j^{(t)}$ at time t.

Model - Tasks

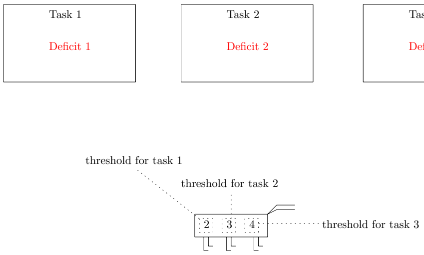
- We have k tasks.
- demand: Each task i demand$^{(j)}$, i.e., number of ants required to work on task j.
- load: $\text{load}_j^{(t)}$ is the number of ants working on it at time t.
- deficit: $\text{deficit}_j^{(t)}$ is the demand$^{(j)}$ $-$ $\text{load}_j^{(t)}$ at time t.



Two tasks, one underloaded and one overloaded.

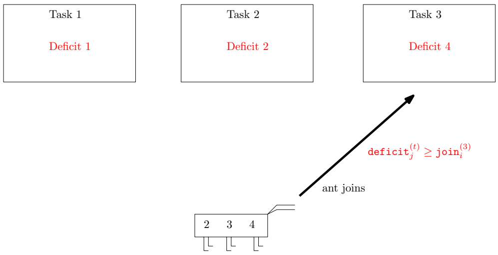
Model - Ants

- Say we have n ants.
- Assume we have synchronous rounds
- For each task j, each ant i has a (joining) threshold $j_i^{(j)}$
- If $\text{deficit}_j^{(t)} \geq j_i^{(j)}$ and ant is idle, then the ant i joins the task j.



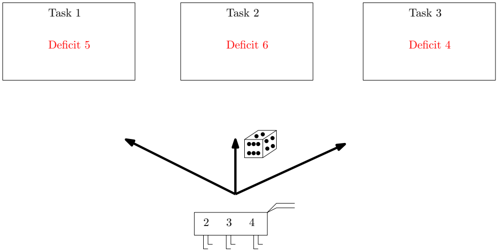
Model - Ants

- Say we have n ants.
- Assume we have synchronous rounds
- For each task j, each ant i has a (joining) threshold $j_i^{(j)}$
- If $\text{deficit}_j^{(t)} \geq j_i^{(j)}$ and ant is idle, then the ant i joins the task j.



Model - Ants

- Say we have n ants.
- Assume we have synchronous rounds
- For each task j, each ant i has a (joining) threshold $j_i^{(j)}$
- If $\text{deficit}_j^{(t)} \geq j_i^{(j)}$ and ant is idle, then the ant i joins the task j.
- In case there are several tasks exceeding the joining threshold, the ant picks one *uniformly at random*.



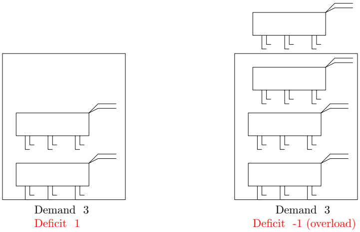
All deficits are above the threshold. The ants joins a task chosen uniformly at random.

When do ants leave?

- We assume all ants have the same leaving threshold
- If the overload exceeds this threshold, all ants will leave.

What is possible?

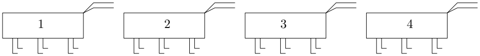
- Quality measure Q for task allocation: Sum of absolute deficits.
  For example: $Q = |1| + |-1| = 2$



- 1 Task: Perfect allocation possible ($Q = 0$)
- 2 Tasks: Perfect allocation possible ($Q = 0$)
- 3 Tasks: ??? (our main result)
- I'll talk about all three

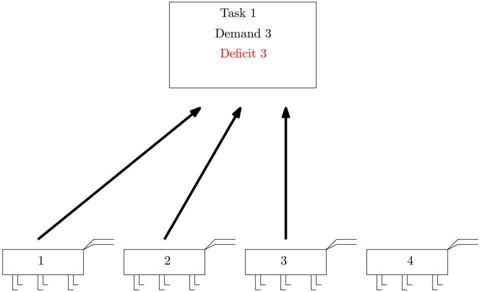
1 Task

- Give numbers from 1 to n to the ant.
- Ant i has a joining threshold of i.
- The leaving threshold (for all ants) is arbitrary



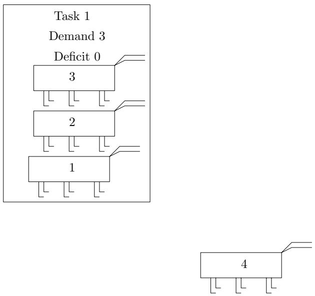
1 Task

- Give numbers from 1 to n to the ant.
- Ant i has a joining threshold of i.
- The leaving threshold (for all ants) is arbitrary



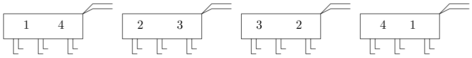
1 Task

- Give numbers from 1 to n to the ant.
- Ant i has a joining threshold of i.
- The leaving threshold (for all ants) is arbitrary



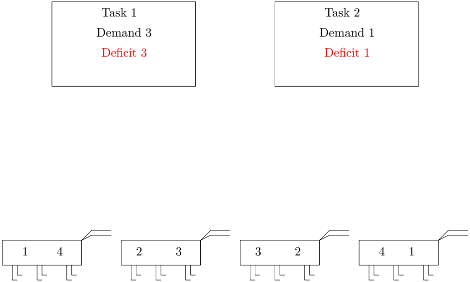
2 Tasks

- Give numbers from 1 to n to the ant.
- Ant i has a joining thresholds i and $n - i$ (task 1; task 2)
- The leaving threshold (for all ants) is some arbitrary positive value



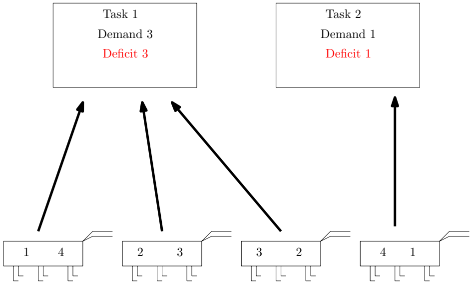
2 Tasks

- Give numbers from 1 to n to the ant.
- Ant i has a joining thresholds i and $n - i$ (task 1; task 2)
- The leaving threshold (for all ants) is some arbitrary positive value



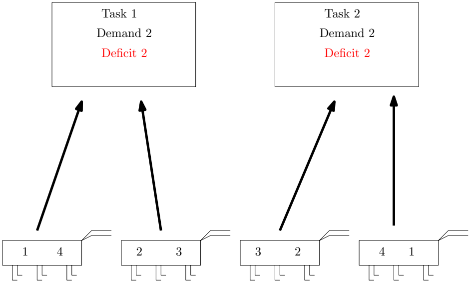
2 Tasks

- Give numbers from 1 to n to the ant.
- Ant i has a joining thresholds i and $n - i$ (task 1; task 2)
- The leaving threshold (for all ants) is some arbitrary positive value



2 Tasks

- Give numbers from 1 to n to the ant.
- Ant i has a joining thresholds i and $n - i$ (task 1; task 2)
- The leaving threshold (for all ants) is some arbitrary positive value



Demands changed from $(3, 1)$ to $(2, 2)$

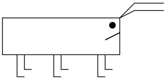
2 Tasks

- Give numbers from 1 to n to the ant.
- Ant i has a joining thresholds i and $n - i$ (task 1; task 2)
- The leaving threshold (for all ants) is some arbitrary positive value

Note: We always assume that the sum of demands are at most n; otherwise it's impossible anyway ...

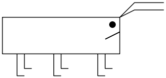
3 Tasks - Our Theorem

- In general, it is impossible to reach a perfect allocation ($Q = 0$)!
- In particular, for any setting of thresholds, there is a demand vector where the sum of absolute values is linear in n.



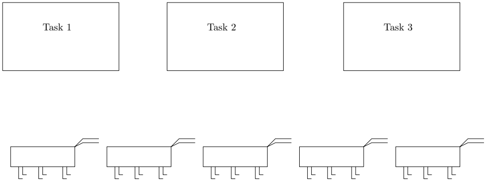
3 Tasks - Our Theorem

- In general, it is impossible to reach a perfect allocation ($Q = 0$)!
- In particular, for any setting of thresholds, there is a demand vector where the sum of absolute values is linear in n.
- Disclaimer: This does not mean that ants don't do it, but if they do, it's highly inefficient.
- Note, state-machines are capable of reaching perfect allocation ($Q = 0$) in even harsher settings



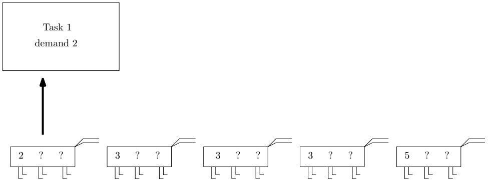
3 Tasks

- Say we have n with arbitrary joining thresholds.
- Assume they have the same leaving threshold.
- Assume we have 3 tasks.



3 Tasks

- Say we have n with arbitrary joining thresholds.
- Assume they have the same leaving threshold.
- Assume we have 3 tasks.
- Consider the number of ants n_1 that would join task n_1 if it had a demand of 2.
- Case $n_1 < 2$: then too few ants will join

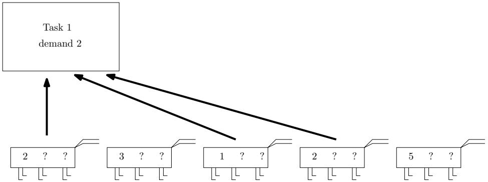


Deficit will always remain 1 :(

3 Tasks

- Say we have n with arbitrary joining thresholds.
- Assume they have the same leaving threshold.
- Assume we have 3 tasks.
- Consider the number of ants n_1 that would join task n_1 if it had a demand of 2.

- If $n_1 > 2$ then too many ants join



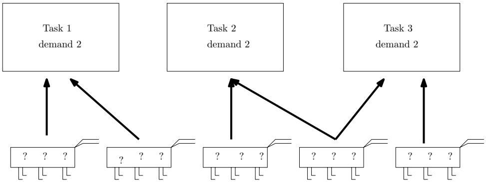
Two cases, both bad: Deficit will always remain -1 or endless oscillations

3 Tasks

- Thus we want $n_1 = 2$
- By the same logic we also need $n_2 = n_3 = 2$

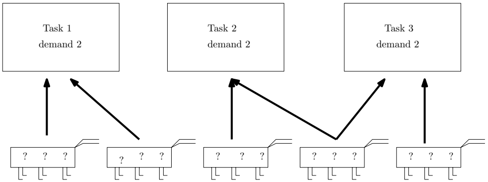
3 Tasks

- Thus we want $n_1 = 2$
- By the same logic we also need $n_2 = n_3 = 2$
- The problem is that when we open two carefully chosen tasks with demand 2, then one task won't have enough ants.



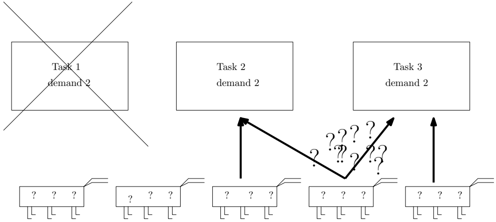
3 Tasks

- Thus we want $n_1 = 2$
- By the same logic we also need $n_2 = n_3 = 2$
- The problem is that when we open two carefully chosen tasks with demand 2, then one task won't have enough ants.
- Incoming edges of the task should be 6
- Thus (at least) one ant would join two different tasks.



3 Tasks

- Thus we want $n_1 = 2$
- By the same logic we also need $n_2 = n_3 = 2$
- The problem is that when we open two carefully chosen tasks with demand 2, then one task won't have enough ants.
- Incoming edges of the task should be 6



3 Tasks

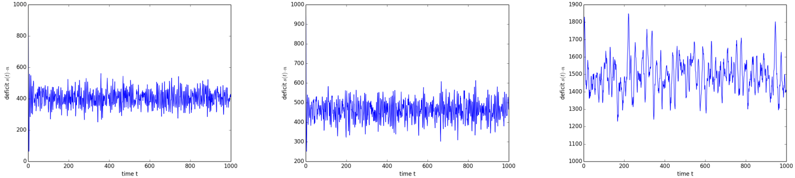
- More generally speaking: For any k and n large enough, the sum of absolute deficits will be linear in the number of ants

3 Tasks

- More generally speaking: For any k and n large enough, the sum of absolute deficits will be linear in the number of ants
- Why hasn't this been noticed before?
- First, many papers only study two tasks ...
- Second, most of the papers don't analyze any quality measure. If they do, it's often the proportion of minors and majors in the nest.

Boneabeau Model - Huge oscillations

The following figure shows our simulation of this protocol with different fractions of cast 1 (n_1/n). In all different settings of n_1/n we see large oscillations of the deficit. The oscillations appear to be linear in the number of ants.

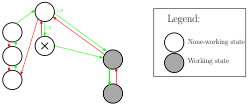


$n_1/n = 0.2$ $\quad$ $n_1/n = 0.3$ $\quad$ $n_1/n = 0.7$

Figure: Parameters are $N = 1000$, $\theta_1 = 8$, $\theta_2 = 1$, $\delta = 1$,$\alpha = 3$ as suggested by Bonabeau et al. (96) and Bonabeau et al. (98).

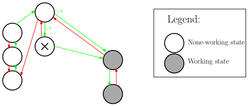
State Machines

- The performance of state-machines can be much better $Q \leq 1$
- In every time step an ant is in a state (marked with an **x**) and receives some feedback.

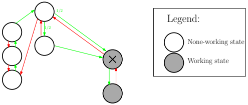


State Machines

- The performance of state-machines can be much better $Q \leq 1$
- In every time step an ant is in a state (marked with an **x**) and receives some feedback.



- Based on that feedback (e.g. $\rightarrow$) the ant transitions to the next state

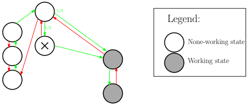


State Machines

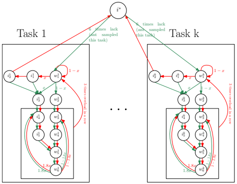
- Meyer et al. showed that
  - threshold model (for bumblebees) does not match experimental results.
  - They suggest a time-resolved model (which can be seen as a simple state machine).

State Machines

- Cornejo et al. proves that for binary feedback (overload/underload of task) achieves a 'perfect allocation' ($Q \leq 1$)



- Dornhaus et al. (BDA last year) show that this still works ($Q << n$) even when the feedback (overload/underload of task) is noisy



Summary: “Remember the Past and Forget Threshold”

- Response threshold theory assumes workers to differ from one another.
- They were initially introduced to avoid massive oscillations (all ants behave exactly the same)
- Response thresholds are nonetheless far from optimal

Summary: "Remember the Past and Forget Threshold"

- Response threshold theory assumes workers to differ from one another.
- They were initially introduced to avoid massive oscillations (all ants behave exactly the same)
- Response thresholds are nonetheless far from optimal
- State machines perform much better (w.r.t. Q)
- Indicating that there is no need for <u>permanently</u> different
- Using a tiny bit of memory (state machines) allows ants to 'differentiate' <u>temporarily</u>

Summary: “Remember the Past and Forget Threshold”

- Response threshold theory assumes workers to differ from one another.
- They were initially introduced to avoid massive oscillations (all ants behave exactly the same)
- Response thresholds are nonetheless far from optimal
- State machines perform much better (w.r.t. Q)
- Indicating that there is no need for permanently different
- Using a tiny bit of memory (state machines) allows ants to ‘differentiate’ temporarily
- Of course you could combine both models, but is there an advantage?

Thank You

Questions?