Self-Stabilizing Task Allocation In Spite of Noise

Self-Stabilizing Task Allocation In Spite of Noise

Anna Dornhaus
University of Arizona, Department of Ecology and Evolutionary Biology
Tucson, AZ, US

Nancy Lynch
MIT, CSAIL
Cambridge, MA, US

Frederik Mallmann-Trenn
King's College London, Department of Informatics
London, UK

Dominik Pajak
Wrocław University of Science and Technology
Wrocław, Poland

Tsvetomira Radeva
MIT, CSAIL
Cambridge, MA, US

ABSTRACT

We study the problem of distributed task allocation by workers in an ant colony in a setting of limited capabilities and noisy environment feedback. We assume that each task has a *demand* that should be satisfied but not exceeded, i.e., there is an optimal number of ants that should be working on this task at a given time. The goal is to assign a near-optimal number of workers to each task in a distributed manner without explicit access to the value of the demand nor to the number of ants working on the task.

We seek to answer the question of how the quality of task allocation depends on the accuracy of assessing by the ants whether too many (`overload`) or not enough (`lack`) ants are currently working on a given task. In our model, each ant receives a binary feedback that depends on the *deficit*, defined as the difference between the demand and the current number of workers in the task. The feedback is modeled as a random variable that takes values `lack` or `overload` with probability given by a sigmoid function of the deficit. The higher the overload or lack of workers for a task, the more likely it is that an ant receives the correct feedback from this task; the closer the deficit is to zero, the less reliable the feedback becomes. Each ant receives the feedback independently about one chosen task. We measure the performance of task allocation algorithms using the notion of *inaccuracy*, defined as the number of steps in which the deficit of some task is beyond certain threshold.

We propose a simple, constant-memory, self-stabilizing, distributed algorithm that converges from any initial assignment to a near-optimal assignment under noisy feedback and keeps the deficit small for all tasks in almost every step. We also prove a lower bound for any constant-memory algorithm, which matches, up to a constant factor, the accuracy achieved by our algorithm.

The authors were supported in part by NSF Award Numbers CCF-1461559 and CCF-0939370. D. Pajak was also supported by the National Science Centre, Poland—Grant Number 2019/33/B/ST6/02988.

SPAA '20, July 15–17, 2020, Virtual Event, USA

ACM ISBN 978-1-4503-6935-0/20/07…$15.00
https://doi.org/10.1145/3350755.3400226

CCS CONCEPTS

• **Theory of computation → Distributed algorithms**; **Design and analysis of algorithms**.

KEYWORDS

ants, biologically inspired algorithms, noise, task-allocation

ACM Reference Format:
Anna Dornhaus, Nancy Lynch, Frederik Mallmann-Trenn, Dominik Pajak, and Tsvetomira Radeva. 2020. Self-Stabilizing Task Allocation In Spite of Noise. In *Proceedings of the 32nd ACM Symposium on Parallelism in Algorithms and Architectures (SPAA '20), July 15–17, 2020, Virtual Event, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3350755.3400226

1 INTRODUCTION

Task allocation in social insect colonies is a process of assigning workers to various tasks such as foraging, scouting, nursing, *etc.* in a way that maximizes the reproductive success of the colony. Each ant is capable of working on each of the tasks but at any time it can only work on at most one of them. Each task has associated demand which is the number of ants that should work on this task that is optimal from the perspective of the colony needs. The ants probably neither know the demand nor can count the current number of ants working on a given task [17]. Moreover, in the colony there is no central control to decide about the actions of each individual [15]. Therefore the allocation has to be performed by each ant locally, based only on feedback from the environment about the tasks and limited local communication with other individuals. The feedback, in biology called 'task stimulus', corresponds, for example, to sensing a too-high temperature in the nest, seeing light through a hole in the nest-wall, or smelling a pheromone produced by hungry brood. Despite using limited communication, local observations, and noisy sensing, many ant species are known to excel at task allocation. How do ants perform task allocation and what can we learn from their behavior?

In [8], the authors proposed a solution to the problem of task allocation in the case where there is no communication between the ants and the feedback received by the ants is binary and always correct. More precisely, in [8] if the *load*, *i.e.,* the number of ants working on the task $j \in [k]$, exceeds the demand $d^{(j)}$, then all ants receive feedback `overload`. Conversely, if the load is below or equals the demand of the task, then all ants receive feedback `lack`. Such a feedback function is rather unrealistic in ant colonies due to its sharp transition between `overload` and `lack`— it requires

each ant to be able to tell the difference between $d^{(j)}$ and $d^{(j)}+1$ number of workers at a task. The authors in [8] therefore pose the open problem of considering a *weaker*, noisy version of the binary-feedback—the focal point of this paper.

We study the performance of task allocation in a realistic, stochastic noise model, in which the feedback from the environment for each ant in each step is a random variable with possible values `lack` and `overload`. The probability that it takes value `lack` equals to the sigmoid function of the deficit (demand minus load). The deficit is negative in case when the load is larger than the demand.

We assume that all the ants regularly receive the feedback. However, there is a delay between the moment when an ant collects the feedback and when it changes its allocation, during which other ants may also make some decisions. To model this delay we assume, similarly to [8], synchronous rounds: at the beginning of each round each ant receives binary feedback of the load of the chosen task. The ants then concurrently make a decision of whether to join some task or to leave their current task. In our model, the ants make choices synchronously as if each of them had a local clock ticking at the same rate. However, we do not assume a global clock which means that the ants do not have access to round number hence cannot for example discriminate odd and even rounds.

How can the ants make independent decisions and achieve a 'good' task allocation in spite of *outdated observations*, in spite of *noise* and in spite of the *lack of global information*—not knowing the demands nor the current loads of the tasks nor the current round number?

In order to define what a 'good' task allocation means we use the notion of *inaccuracy*. The inaccuracy in our setting in some time step for some task means that the absolute value of the deficit (demand minus the load) is larger than the demand times some constant $\varepsilon > 0$. The algorithm will be called ε-accurate if in any (sufficiently long) time interval of length T, in only $o(T)$ steps, the algorithm will be inaccurate for any task. Here we penalize overload and underload equally: An underload corresponds to work that is not being done, and each ant exceeding the demand of a task corresponds to work being wasted (or, even worse, sometimes the excessive number of workers in a task may block each other and decrease the efficiency [9, 12]). Note, that we do not charge any cost for switching tasks.

We first provide a simple, constant-memory, self-stabilizing Algorithm Controlled Oscillations that utilizes the oscillations in the number of workers in each task in order to achieve a stable allocation in the synchronous model. The size of the oscillation at each task in our algorithm is proportional to the demand of the task times the *critical value* of the deficit (see Section 2 for a formal definition). Intuitively, the critical value is a value of the deficit (seen as a fraction of the demand) for which the feedback is correct for each ant with high probability. This corresponds to the smallest value of the deficit at which the sigmoid is very close to 1 and the largest value for which it is very close to 0. We then show that our algorithm is ε-accurate for ε being equal to the critical value times a constant.

As our second result, we prove a lower bound showing that no distributed, constant-memory algorithm can be ε'-accurate for ε' smaller than or equal to the critical value times a constant depending on the number of bits of memory available to the ants. In light of this lower bound, our Algorithm Controlled Oscillations achieves a constant-factor approximation of the optimal accuracy factor. In our lower bound we show that, for any strategy, if the deficit gets too close to 0 for some number of steps then (due to the noisy feedback) it drastically increases. It is therefore impossible to keep the absolute value of the deficit very small for too long. This means that small oscillations in the number of workers at each task are unavoidable in any algorithm that 'tries' to achieve a good and stable allocation. Our proposed solution is to avoid getting too close to 0 with the absolute value of the deficit and use the oscillations (jumping between positive and negative deficit) to achieve stable allocation and asymptotically optimal accuracy factor.

1.1 Related Work

The most related previous work on task allocation is [8], in which the authors also assume synchronous rounds and binary feedback. The authors present a very simple algorithm that converges to an almost-optimum allocation (the allocation that differs from the demand by at most 1 at each task) and analyze its convergence time. Considering a noisy version of the model was left as an open question.

Moreover, the authors of [26] provide a model similar to that of [8] but they also study different versions of the feedback that ants receive from the environment, which varies in the amount of information the ants receive about the deficits of the tasks. The model consists of two feedback components: a *success* component that informs each ant in each round whether it is successful, e.g., needed for the task it is currently working on, and a *choice* component that provides unsuccessful ants with an alternative task to work on. The results in [26] analyze the convergence time of task allocation, and as such are not directly comparable to our work here. In [26], the noise model is very rudimentary: in each round the feedback of the binary success component can be noisy for at most a fixed number of ants. The results do not generalize to our setting.

The problem of task allocation in social insect colonies has been well studied in the communities of theoretical and experimental biology. The observations show that social insect colonies are self-organized, with no individuals directing the task choices of others, with interactions between individuals potentially affecting task selection [15]. Workers in a colony may switch tasks as needed [14], although this may come at additional cost [22]. The concept of task switching gives rise to an intriguing question: what is the algorithm used by the ants to decide whether to switch and which tasks to choose? Some notable examples of models of task allocation [3] include (1) the threshold-based model where ants compared the stimulus of a task to their built-in threshold to determine whether to work on a given task, and (2) the 'foraging for work' model [30] where the ants are believed to actively look for work when they are idle or redundant in the current task. In some species the ants are believed to choose the tasks based on physical suitability (*physical polyethism*) [21], whereas in other species, the ants are physically similar and suitable to do any task (*temporal polyethism*) [4]. In this paper, we assume that the ants are identical (no thresholds) and they can work on any task.

Some biological studies have focused on the efficiency of the task allocation process itself, and how it is determined by the specific algorithm used by the ants. For example, [24] and [10] model task allocation determined by social interactions and response thresholds, respectively, and both demonstrate that perfect task allocation of workers to tasks cannot be achieved, potentially due to the speed and accuracy of task allocation trading off against each other. In [29], an algorithm of task allocation is analyzed in a setting where there are no thresholds, the tasks are arranged in a line and there is no noise in sensing of the demand. The goal of [29] is to explain the experimental observations where certain tasks were preferred by older ants.

Additional factors such as individual experience, interactions with other workers, spatial and hierarchical position in the colony, and random encounters with tasks are also known to affect the specific task allocation mechanism employed [5, 11, 15]. Unfortunately, most often it is not precisely known what is the actual algorithm that the ants use to select tasks or how the factors listed above interact to produce variation in preferences across tasks or across individuals [25].

A key property that we observe in our results—oscillations in the task allocation behavior of ants—is also a commonly observed biological phenomenon more generally known as cyclical activity patterns [6]. Although the role of cyclical activity patterns is not completely understood [7], several studies make conjectures that may be related to the conclusions in our paper. First, our assumption that ants perform actions in synchronized rounds and phases as a means of introducing 'delay' between one another's actions is also observed in biological studies. Ants perform actions in bursts of activity and inactivity in order to clear stale information from spreading through the colony [27]. Second, our results suggest that, assuming that ants have constant memory (*i.e.,* they cannot even store the total number of ants n let alone use it), and noisy environmental feedback, the oscillations are inevitable as the deficit becomes small. We conjecture that such cyclic activity patterns (switching between different tasks and being idle) are necessary and a product of the limitations of the ants and the noisy feedback about number of workers at a task.

Another key assumption we make is that the noise follows a sigmoid function (also known as a *logistic sigmoid activation function*). Such functions appear in countless biological contexts (*e.g.* [13, 19, 28]), to model the uncertainty with which the ants sense the need for work at different tasks. We believe that the versatility and applicability to the real-world problems of the sigmoid noise model makes it a good choice to model the noise of the environment in our setting.

Finally, somewhat related load-balancing processes have been studied under the term *user-based migration* in which the tasks move in a network of resources [1, 2, 20] by querying the load of the current resource and moving to a neighbor in case of an overload. However, this line of research assumes that the communication noise-free and that each resource knows an upper bound on how many tasks it can accept.

2 MODEL

We have a collection of n ants and a constant number of k tasks where each task $j \in [k]$ (we use notation $[k] = \{1, 2, \ldots, k\}$) has a fixed demand $d^{(j)}$. We assume, that each ant has some number of bits of memory and that there is no direct communication between the ants. Time is divided into discrete steps. We assume that *at the beginning* of step t, each ant i receives feedback $F_t^{(j)}(i)$ about one, chosen task $j \in [k]$ (the value of the feedback depends on the number of ants working on task j in previous step). Based on this feedback and the current memory state, the ant decides whether or not to work and on which task during this round. The ant also decides about its new memory state and chooses a task about which it will receive feedback at the beginning of the next round. We assume that the ant can choose to receive feedback about any task however in our algorithm it always chooses between the same task as in the previous step or a new task chosen uniformly at random.

Let $W_t^{(j)}, j \in [k]$ denote the number of ants performing task j during step t. In the following we call value $d^{(j)} - W_t^{(j)}$ a *deficit* of task j in step t. Value $W_t^{(j)} - d^{(j)}$ will be referred to as *overload*.

2.1 Noisy Feedback

We seek to model the noise in the sensing of the lack or overload by the ants such that the following conditions are fulfilled. First, in case of a very large overload or lack, almost all ants should notice this (w.h.p. all ants should receive the correct feedback). Second, whenever exactly the correct number of ants are working on a given task, then the 'uncertainty' in this task should be the largest and the ants should receive `lack` and `overload` with equal probability. In the following we define the sigmoid feedback model that fulfills these requirements. At the beginning of round t, ant i receives for chosen task j noisy feedback:

$$F_t^{(j)}(i) = \begin{cases} \texttt{lack} & \text{w. p. } s(d^{(j)} - W_{t-1}^{(j)}), \\ \texttt{overload} & \text{otherwise,} \end{cases}$$

where $s(x) = \frac{1}{1+e^{-\lambda x}}$, for fixed $\lambda \in \mathbb{R}$.

It is not crucial for our results to have a sigmoid function; in fact all our results apply for any monotone antisymmetric function s with exponential decay and $\lim_{x\to-\infty} s(x) = 0$ and $\lim_{x\to\infty} s(x) = 1$ and $s(0) = 1/2$.

Reliability of the feedback depends on how the absolute value of the deficit is far from 0. As the absolute value of the deficit increases, it is more and more likely for ants to receive the correct feedback. We would like to define a critical value of the deficit beyond which the feedback is (almost) completely reliable and an interval which we call a *grey zone*, within which we will not rely on the correctness of the feedback.

Definition 2.1 (critical value and grey zone). *Let* $y(x) = \min_{x' \in \mathbb{R}}\{x' : s(-x' \cdot d^{(j)}) \le x \text{ for all } j \in [k]\}$. *We define the* critical value *to be* $\gamma^* = y(1/n^6)$. *We define for each task* $j \in [k]$ *the* grey zone *to be interval* $[-\gamma^* d^{(j)}, \gamma^* d^{(j)}]$. *We say that a task* j *is in step* t *in grey zone, when its deficit is inside this interval.*

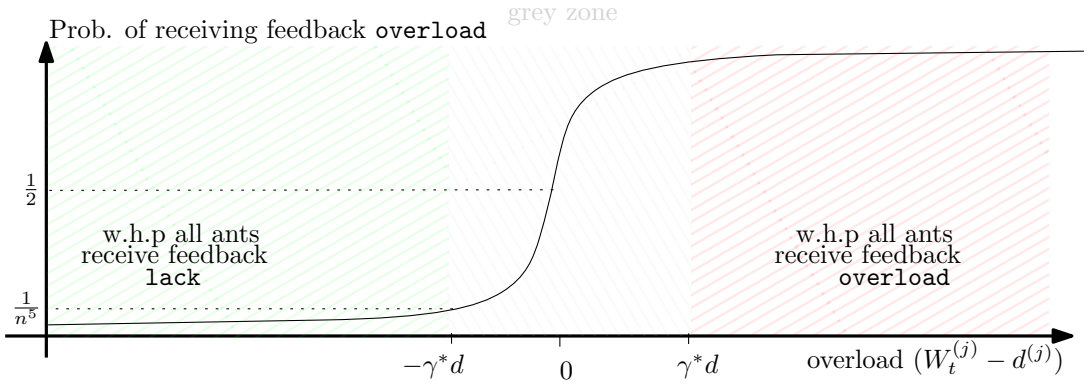

Figure 1: Whenever the overload of a task is in the green (red, respectively) region, all ants will receive w.h.p. the feedback `lack` (`overload`, respectively). Whenever the overload is in the grey region (falling tilling pattern), and the closer the overload is to 0, the more noisy is the feedback received by the ants.

2.2 Accurate algorithms

For any $\varepsilon > 0$ and for any $j \in [k]$ we denote:

$$\mathsf{Inaq}_j^{\varepsilon}(\tau) = \mathbf{1}_{|W_\tau^{(j)} - d^{(j)}| > \varepsilon d^{(j)}}$$

The function $\mathsf{Inaq}_i^{\varepsilon}(\tau)$ indicates whether the algorithm is *inaccurate* in step τ for task j. For any time interval $[1, T]$ we define $\mathsf{Inaq}_j^{\varepsilon}(T) = \sum_{\tau \le T} \mathsf{Inaq}_j^{\varepsilon}(\tau)$, $\mathsf{Inaq}^{\varepsilon}(\tau) = \min\left\{1, \sum_{j\in[k]} \mathsf{Inaq}_j^{\varepsilon}(\tau)\right\}$, and $\mathsf{Inaq}^{\varepsilon}(T) = \sum_{\tau \le T} \mathsf{Inaq}^{\varepsilon}(\tau)$. Function $\mathsf{Inaq}^{\varepsilon}(T)$ indicates in how many steps of the time interval $[1, T]$ the algorithm was inaccurate for at least one task.

Finally, we will call an algorithm *ε-accurate* if $\mathsf{Inaq}^{\varepsilon}(T) \in o(T)$ for any $T \in \Omega(n)$ and assuming arbitrary initial state. This means that when a colony of ants is using such an algorithm, then almost all the time (*i.e.,* in all steps except a smaller order term) the loads of all the tasks are close to their demands.

We assume that $k = O(1)$ motivated by the fact that the number of tasks is much smaller than the number of ants. Hence the ant can in constant memory store an identifier of a task.

We use the shorthand w.p. for 'with probability'. We say an event happens w.h.p. 'with high probability' to mean that the event happens w.p. at least $1 - O(1/n)$. We say an event happens with *overwhelming probability* if it happens w.p. at least $1 - e^{-\Omega(n)}$.

3 OUR RESULTS

In this section we present the assumptions and the statements of our main results.

3.1 Upper Bound

In our upper bound we use the following assumptions on the demand vector. First, we require the demands to be at least logarithmic in the number of ants. Second we assume that the sum of the demands does not exceed the total number of ants.

Assumption 3.1. *Assume that for all $j \in [k]$ we have $d^{(j)} \ge 600 \ln n/(\gamma^*)^2$ for each $j \in [k]$. Moreover, assume that the sum of the demands satisfies $(1+17\gamma)\sum_{j\in[k]} d^{(j)} \le n$ for some $\gamma \in [\gamma^*, 1/100]$.*

We proceed by giving the precise statements of Theorem 3.2 showing that the assignment of Algorithm Controlled Oscillations is w.h.p. $O(1)$-accurate.

Theorem 3.2. *Assuming Assumption 3.1 with parameter γ holds, then for any initial allocation at time 0. Algorithm Controlled Oscillations with $\gamma \in [\gamma^*, 1/100]$, is w.h.p. 15γ-accurate.*

Note that we allow t to take arbitrary values—in particular, values that are super-polynomial in n.

Remark 3.3. *The guarantees from Theorem 3.2 apply even if the feedback is arbitrarily correlated as long as the marginal probability for each ant to receive incorrect feedback outside the grey zone is $1/n^6$. Moreover, our algorithm also works—due to its self-stabilizing nature—for changing demands.*

3.2 Lower Bound

We show a lower bound for a class of algorithms in which the states of all the ants are always reachable from each other. We do not allow for example algorithms where an ant working on some task can never leave this task. It is well-known (*e.g.* [16]) that ants do not stabilize to a fixed allocation but switch between the tasks whenever it is needed.

Assumption 3.4. *We assume that for any pair of states s_1, s_2 (idle or working on one of the tasks $j \in [k]$), there must exists a finite sequence of feedback values such that an ant being initially in state s_1, after receiving this sequence of feedbacks, transitions with nonzero probability to state s_2.*

Under this assumption we show a lower bound on the accuracy factor of any constant memory algorithm.

Theorem 3.5. *Assume $\gamma^* \le 1$. Let $\varepsilon \in (0, 1/4)$ and let n (number of ants) be a large enough integer. There exists a demand vector $(d^{(1)}, d^{(2)}, \ldots, d^{(k)})$ such that for any collection of n ants executing (possibly distinct) algorithms $A_1, A_2, \ldots, A_n$, each using at most $\lfloor \ln(1/(16\varepsilon)) \rfloor$ bits of memory, the resulting taks-allocation algorithm cannot be $2\varepsilon\gamma^*$-accurate.*

4 UPPER BOUND

In this section, we present the definition of the algorithm, an overview of the execution and the analysis.

4.1 Definition of the Algorithm

Let $a = 2.5$ and $b = 5.6$ and γ be a parameter of the algorithm satisfying $\gamma \in [\gamma^*, 1/100]$. For each task j we have 9 states out of which $w_1^j, \ldots, w_5^j$ are working states and $i_1^j, i_2^j, i_3^j, i_4^j$ are idle states. We also have one idle state i^* that is common for all the tasks (see Figure 2 for an overview of the machine). When an ant is in one of the states $w_1^j, \ldots, w_5^j$ during step t for $j \in [k]$, then it is working on task j during this step. Otherwise the ant is idle.

The state machine for each task $j \in [k]$ is divided into two submachines: the filtering submachine consisting of the states $F^j = \{i_1^j, i_2^j, w_1^j\}$ and the oscillating submachine consisting of the states $O^j = \{i_3^j, i_4^j, w_2^j, w_3^j, w_4^j, w_5^j\}$. Let $S^j = F^j \cup O^j$ denote the set of all states of task j. We say that when an ant is in any of the states from set S^j that it is *committed* to task j in this step. Notice that not all ants that are committed to a task work on the task (ants in states $i_1^j, i_2^j, i_3^j, i_4^j$ are committed to task j but are not working on it) and that the ants may change their commitment.

When an ant transitions to state i^*, it selects a task $j \in [k]$ uniformly at random to collect its feedback. When the observed feedback is `lack` for 5 times in a row, then the ant joins the task (transitions to state w_1^j). Otherwise, when observing `overload` the ant picks another task at random among all k tasks. Whenever the ant is not in i^*, it is in a state in S^j of some task j and then it receives the feedback about task j. Based on the feedback from at most three previous steps and the current state, the ant is making a probabilistic choice of its next state. The exact list of transitions and probabilities is in Table 1 and on Figure 2.

State From	*Feedback*	*State To*	*Prob.*
$w_2^j, w_3^j, w_4^j, w_5^j, i_3^j, i_4^j$	3 times `overload` in a row	w_1^j	1
w_1^j	`overload`	w_1^j	$1 - a\gamma$
w_1^j	`overload`	i_1^j	$a\gamma$
w_1^j	`lack`	w_2^j	$1 - a\gamma$
w_1^j	`lack`	i_3^j	$a\gamma$
i_1^j	`overload`	i_2^j	1
i_1^j	`lack`	i_3^j	1
i_2^j	`overload`	i^*	1
i_2^j	`lack`	i_3^j	1
w_2^j	both	w_4^j	1
i_3^j	both	w_4^j	1
w_3^j	both	w_2^j	1
i_4^j	both	i_3^j	1
w_4^j	both	w_5^j	1
w_5^j	both	w_3^j	$1 - b\gamma$
w_5^j	both	i_4^j	$b\gamma$
i^*	5 times `lack` in a row	w_1^j	1

Table 1: The rules are stated in decreasing order of priority; in case two rules are applicable, the rule with the higher priority is executed.

4.2 Intuition of the Algorithm

The intuition is presented from the perspective of a fixed task $j \in [k]$. For other tasks, the algorithm works analogously. The goal of the filtering machine is to reduce the number of ants carefully until a number close to the demand of the task is reached. These ants then all transition independently to the oscillating machine. The transition is accomplished by the following mechanism (assume for a moment that all the ants committed to task j are in three states w_1^j, i_1^j and i_2^j): when the number of ants in the filtering machine is too large, the ants in state w_1^j receive feedback `overload` and leave with some small probability $a\gamma$. However, before leaving to the common idle state i^*, these leaving ants go through a path of two idle states i_1^j and i_2^j. If one of such decreases changes the number of ants working on the task from more than $(1+\gamma)d^{(j)}$ to less than $(1-\gamma)d^{(j)}$, then we say that we *jump over* the grey zone. In this case, all ants committed to task j receive in the next step feedback `lack` and move from filtering machine to oscillating machine. In this case we say that the ants move to the oscillating machine in one *wave*. In the opposite case, if we end up in the grey zone, some number of ants will receive feedback `overload` and some `lack` and we cannot predict how many will receive which feedback. Our objective in this case is to quickly leave the grey zone because if the system remains in the grey zone for several steps, then we will not be able to say anything meaningful about the number of ants in different states of this task. To avoid this, our state machine is constructed in such a way, that regardless of the feedback, each ant from w_1^j transitions with probability $a\gamma$ to an idle state (i_1^j or i_3^j). The ants that receive `lack` in this step move to the oscillating machine in the first wave. By carefully choosing a, after this step, the number of working ants is already small enough that it is outside of the grey zone (on the other, underload side), and in the next step, all ants receive feedback `lack` from task j. Then the ants that are still in the filtering machine move to the oscillating machine in the second wave.

In both cases, all ants from the filtering machine move in at most two waves to states i_3^j and w_2^j of the oscillating machine. Notice that the number of ants that move to the oscillating machine is slightly larger than the demand (because also the ants from states i_1^j and i_2^j are triggered by `lack` to move to i_3^j), but not much larger (the reason for this is that the number of ants in i_1^j and i_2^j is much smaller than the number of ants in w_1^j).

Having a correct (but slightly too large) number of ants in the oscillating machine of our task j, now the goal is to maintain it, which means to make sure that the ants do not leave the task and that no new ants join the task. This is accomplished by having some steps in which all ants in the oscillating machine are guaranteed

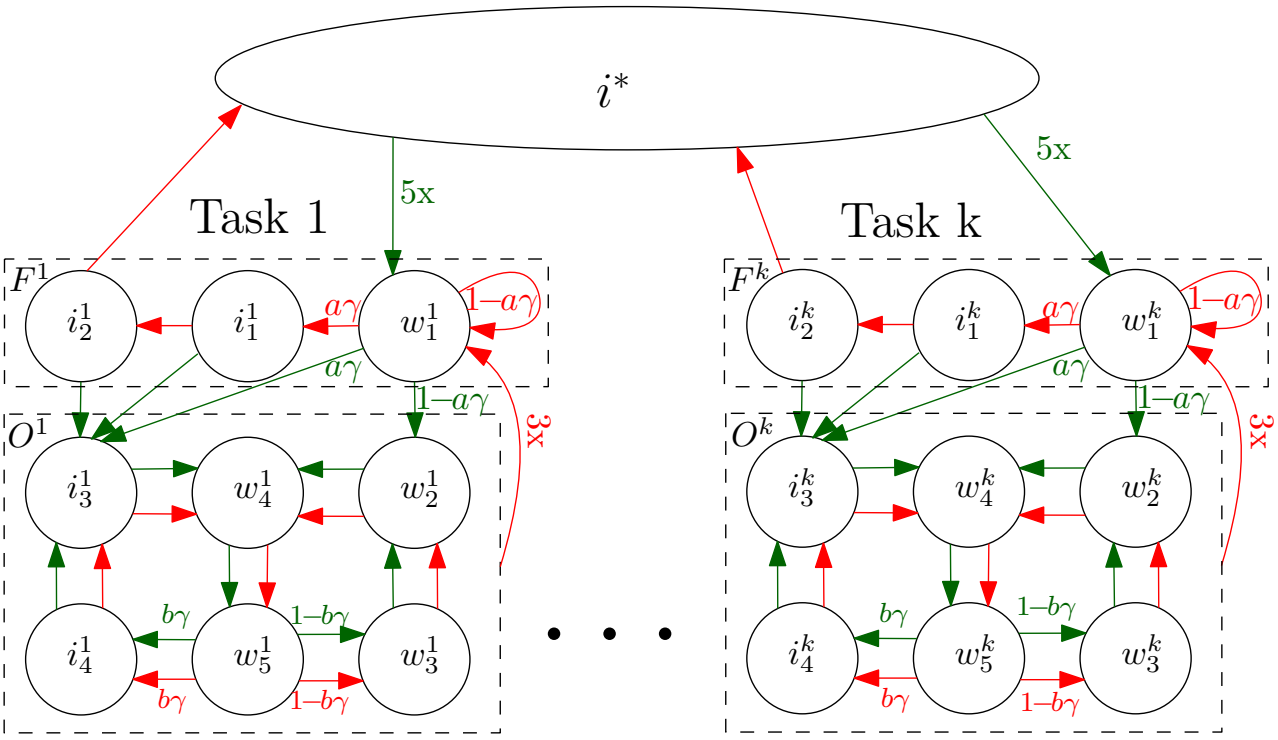

Figure 2: Illustration of the state machine used in Algorithm Controlled Oscillations. Red arrows indicate the transitions taken by the ants that receive feedback `overload` and green arrows by the ones that receive `lack`. Values over the arrows (*e.g.* $a\gamma$ and $1-b\gamma$) denote probabilities of the transitions. Notation 3x (and 5x) indicates that the feedback associated with this transition needs to be received 3 (or 5) times in a row. Transition with 3x is from every state in O^j and has higher priority (*i.e.*, is taken with probability 1 from every state in O^j by each ant that receives 3 times `overload` in a row).

to be working in which case all ants receive `overload` from this task. To achieve this, we use two states w_4^j and w_5^j in which all ants committed to j work on j (we need two states because we might have two waves). This is followed by rounds in which only a fraction of the ants are working; this fraction is small enough so that all ants receive feedback `lack` and large enough to ensure that the underload is small. After leaving w_5^j the ants are split into two groups out of which only one is working (one group moves to the working state w_3^j and the other one to idle state i_4^j). The fraction that moves to i_4^j, transitions in the next step to i_3^j. This is necessary because the ants moved to the oscillating machine in one or two waves. With such a mechanism a roughly $b\gamma$ fraction of the total number of ants in the oscillating machine will be in idle states at least once every 4 time steps. Out of every 4 time steps we have one sure `lack` and one sure `overload` separated by one step of a (possible) grey zone and such a pattern of feedbacks prevents ants from joining and leaving the task hence guarantees a stable allocation.

Our machine has few additional features that ensure its robustness to arbitrary initialization. The oscillating machine gets "cleaned" after the ants see an `overload` 3 times in a row. In such a case all the ants from the oscillating machine transition to w_1^j. The idea is that it is easier to argue about the system when the ants are in every step either in the filtering or in the oscillating machine and since our system has to work from any initial state, we cannot exclude the case where the ants are in both machines. Another reason for this feature is when a small number of ants (*i.e.*, much smaller than the demand) joins the task, we want these ants stay in the oscillating machine until a large number joins w_1^j from i^* (note that these ants cannot wait in the filtering machine because there is a constant "flow" from filtering machine to i^*). Using this feature we can argue that, informally speaking, until the first `overload`, the total number of ants committed to the task is nondecreasing.

The second feature is that the ants from i^* can join w_1^j only after seeing `lack` for 5 times in a row. The idea is that such a situation can occur only when the total number of ants committed to task j is small. If the number of ants in the oscillating machine is close to optimal, we also obtain feedback `lack` (because some ants may in idle states i_4^j and i_5^j) but not 5 times in a row because once every 4 steps there is a step when all the ants from oscillating machine are working on the task.

4.3 Memory

Each ant is equipped with memory in which we want to store its state, previous feedbacks and id of the task from which the ant wants to collect feedback in the next round. In our Algorithm Controlled Oscillations we need $\lceil \log_2 k \rceil$ bits to store task id. If the ant is in i^* it is the id of the task from which the ant is collecting

feedback and otherwise it is the id of the task to which the ant is committed. Note that if the ant is not in i^* then it collects feedback from the environment about the task to which it is committed hence we do not need to store its identifier again. Also, when the ant joins a task from i^* it always joins the task from which it just collected feedback thus remembering only one task id is sufficient.

We need few more bits to encode the remaining information. If the ant is in i^* we need the information about how many (0, 1, 2, 3 or 4) previous feedbacks from currently observed task was `lack`. If the ant is not is i^* we need to store the state (9 values) and how many previous observations of the task feedback returned `overload` (3 values). Thus together we have 5 different values if ant is in i^* and 27 different values otherwise. We can enumerate all these values and store them on 5 bits of memory hence $\lceil \log_2 k \rceil + 5$ bits of memory per ant are sufficient for Algorithm Controlled Oscillations.

4.4 Analysis

In the following analysis we consider an interval of time steps $\mathcal{I} = [1, 2, \ldots, T]$ of length $T \leq n^2$ and we will analyze the evolution of the system and the total number of inequalities in this interval assuming an arbitrary initial configuration in step 1. We will later show how to combine such intervals and obtain the result for arbitrarily large T.

Our proof idea is as follows. We first define several 'bad' events (*e.g.* ant receiving incorrect feedback outside of the grey zone) and prove that w.h.p. none of such events happen during interval $\mathcal{I}$. Conditioning on this, we can analyze the process deterministically by always assuming the worst case within the bounds on which we conditioned. We divide the evolution of the load of each task in time into three *phases* and bound the number of inaccuracies in each of these phases separately, we note that the phases of the tasks are not necessarily aligned across different tasks. Transitions between the phases happen when for the first time the load (or the total number of ants committed to a task) crosses certain fixed threshold value. We will show that the construction of our algorithm guarantees that for each task each such an event can only happen at most once in interval $\mathcal{I}$. In the first phase, the task is underloaded and always returns feedback `lack`. Therefore, the number of ants committed to this task monotonically increases. To bound the inaccuracies caused by the tasks in the first phase we have to first argue about the availability of idle ants (we prove how fast the ants leave the overloaded tasks). Then, even though we cannot argue about how quickly any specific task advances to the second phase, we can show that in each interval of 12 steps there is some progress across all the tasks. Using this we can bound the total number of inaccuracies of all the tasks in the first phase. When the number of ants committed to some task crosses a certain threshold, then the task enters a second phase during which the load of the task is close to the demand. We can guarantee that within this phase the load never falls below a certain value and this phase ends when the number of working ants increases above another threshold. This starts the third phase in which there is a (possibly very) large load at the task. We prove that after such an event all the ants in the task transition to the filtering machine. Then, the ants leave the task (the number of ants in the task decreases geometrically) until an almost correct number remains. This group then moves to the oscillating machine where the number of workers oscillates in a controlled way around the demand ensuring that no ants leave nor join the task.

For each ant i we define $a_t^{(i)}$ to be the state in which ant i is during round t. We denote by $f_t^{(s)}$ the number of ants that are in state s during round t, *i.e.*, $f_t^{(s)} = \sum_i \mathbf{1}_{a_t^{(i)}=s}$. We abuse the notation slightly and write $f_t^{(j)}$ to denote the total number of ants committed to task j, *i.e.*, $f_t^{(j)} = \sum_{s \in S^j} f_t^{(s)}$. By, $f_t^{(i^*)}$ we denote the number of ants that are not committed to any task at time t. The total number of ants working on task j in step t equals $W_t^{(j)} = \sum_{s \in \{w_1^j, w_2^j, w_3^j, w_4^j, w_5^j\}} f_t^{(s)}$ and we can similarly define the number of idle ants committed to task j in step t as: $I_t^{(j)} = \sum_{s \in \{i_1^j, i_2^j, i_3^j, i_4^j\}} f_t^{(s)}$. Let Δ be the set of transitions of our (probabilistic) state machine. Let random variable $\Lambda_t^{(\delta)}$ denote the number of ants applying transition $\delta \in \Delta$ in step t.

We start by defining several 'good' events. We will later prove that all of these events occur w.h.p. in all steps $t \in \mathcal{I}$ for every ant $i \in [n]$ and for each task $j \in [k]$.

(1) $\mathcal{E}_{feedback}$: If $|d^{(j)} - W_{t-1}^{(j)}| \geq \gamma^* d^{(j)}$ then at the beginning of step t, the feedback from j received by all the ants is *correct*, meaning that $F_t^{(j)}(i) = \texttt{lack}$ if $d^{(j)} - W_{t-1}^{(j)} \geq \gamma^* d^{(j)}$ and $F_t^{(j)}(i) = \texttt{overload}$ if $d^{(j)} - W_{t-1}^{(j)} \leq -\gamma^* d^{(j)}$ for each ant i that receives feedback from task j.

(2) $\mathcal{E}_{transition}$: For any transition $\delta \in \Delta$, the number of ants applying this transition is close to its expected value:
$$\left| \Lambda_t^{(\delta)} - \mathbb{E}\left[\Lambda_t^{(\delta)} \right] \right| \leq \max\left\{ \mathbb{E}\left[\Lambda_t^{(\delta)} \right] / 20, 360 \ln n \right\}.$$

(3) $\mathcal{E}_{selection}$: If $f_t^{(i^*)} > 48k \ln n$ then feedback about j is collected by at least $f_t^{(i^*)}/(2k)$ ants in state i^*.

(4) $\mathcal{E}_{idle}$: The number of ants in idle states of any task is bounded by a fraction of ants in the working states of this task: $I_t^{(j)} \leq \max\{12\gamma W_t^{(j)}, 1296 \ln n\}$ and if $f_t^{(j)} > 110 \ln n/\gamma$, then the total number of ants committed to any task does not decrease too quickly: $f_{t+1}^{(j)} \geq (1 - 12\gamma) f_t^{(j)}$.

(5) $\mathcal{E}_{leave}$: If $W_t^{(j)} \geq 360 \ln n/\gamma$ then the number of ants working on any task does not decrease too quickly: $W_{t+1}^{(j)} \geq (1 - 1.1b\gamma) W_t^{(j)}$.

Using Chernoff bound we can prove that event $\mathcal{E} = \mathcal{E}_{feedback} \cap \mathcal{E}_{transition} \cap \mathcal{E}_{selection} \cap \mathcal{E}_{idle} \cap \mathcal{E}_{leave}$ happens w.h.p.

Lemma 4.1. $\mathbb{P}[\mathcal{E}] \geq 1 - 1/n^2$.

Assuming that event $\mathcal{E}$ takes place, the remaining analysis of Algorithm Controlled Oscillations in interval $\mathcal{I}$ is deterministic. Whenever the load of some fixed task is inside the grey zone, we make no assumption about the feedback received by the ants about this task. However if it is outside of the grey zone, then conditioning on $\mathcal{E}_{feedback}$, the ants receive the same, correct feedback. In this case, the ants in the same state apply the transitions from this state with the same probability and using event $\mathcal{E}_{transition}$ we can argue about the number of ants applying each transition.

In the following we define time steps that mark the transitions between phases of each task. Let $\tau_1^{(j)}$ be the smallest step in interval

$\mathcal{I}$ such that the total number of ants comitted to j is at least $f^{(j)}_{\tau^{(j)}_1} \geq (1-\gamma)d^{(j)}$ and let $\tau^{(j)}_2$ be the smallest step in interval $\mathcal{I}$ such that the number of ants working on j is at least $(1+15\gamma)d^{(j)}$ ($W^{(j)}_{\tau^{(j)}_2} \geq (1+15\gamma)d^{(j)}$). If $f^{(j)}_t < (1-\gamma)d^{(j)}$ for all steps $t \in \mathcal{I}$ we set $\tau^{(j)}_1 = T+1$, and similarly if $W^{(j)}_t < (1+15\gamma)d^{(j)}$ for all steps $t \in \mathcal{I}$ we set $\tau^{(j)}_2 = T+1$. We say that until $\tau^{(j)}_1$, task j is in the first phase and in $\tau^{(j)}_1$ it is advanced to the second phase that lasts until $\tau^{(j)}_2$. In step $\tau^{(j)}_2$ task j is advanced to the third phase which lasts until the end of interval $\mathcal{I}$.

We will bound the total inaccuracy of the algorithm separately for each phase. We first bound the total inaccuracy caused by the tasks in the first phases. To show this we need to argue about how quickly the tasks get advanced to their second phases. We know that a task in its first phase always returns feedback lack (by $\mathcal{E}_{feedback}$) hence if ants in i^* are available, some fraction of them will join each such task (we can use $\mathcal{E}_{selection}$ to lowerbound the number of ants that choose any underloaded task). What is missing is at what rate the ants from overloaded tasks return to i^* to become available to join the underloaded ones. We bound this rate in the next lemma.

Lemma 4.2. *Assume event $\mathcal{E}$ and that $\gamma \leq 0.01$. For any $t \in \mathcal{I}$ and $j \in [k]$ such that $f^{(j)}_t \geq (1+15\gamma)d^{(j)}$ we have that in the interval $[t, t+5]$ at least $\gamma f^{(j)}_t/2$ ants transition from i^j_2 to state i^*.*

Now we are ready to bound the inaccuracy of our algorithm. We fix $\gamma \in [\gamma^*, 1/100]$ and split the $\mathsf{Inaq}^{15\cdot\gamma}(T)$ into three components $\mathsf{Inaq}^{15\cdot\gamma}(T) = \mathsf{Inaq}^{15\cdot\gamma}(T,1)+\mathsf{Inaq}^{15\cdot\gamma}(T,2)+\mathsf{Inaq}^{15\cdot\gamma}(T,3)$, where $\mathsf{Inaq}^{15\cdot\gamma}(T,i)$ denotes the inaccuracy of the algorithm in interval $[1,T]$, due to the tasks in i-th phase for $i = 1,2,3$.

Lemma 4.3. *[Inaccuracy First Phase] Assume event $\mathcal{E}$ and that $\gamma \leq 0.01$. We have $\mathsf{Inaq}^{15\gamma}(T,1) = O(k/\gamma^2)$.*

Proof. We define the following pair of potentials. The number of tasks still in the first phase in step t: $\Phi(t) = \sum_{j\in[k]} \mathbf{1}_{t<\tau^{(j)}_1}$, the number of ants below a certain threshold in each task in step t: $\Xi(t) = \sum_{j\in[k]} \max\{0, (1-\gamma)d^{(j)} - f^{(j)}_t\}$. By $\mathcal{E}_{feedback}$, both these potentials are non-increasing. Moreover by the definition of $\tau^{(j)}_1$, we have $\Phi(t) = 0$ if and only if $\Xi(t) = 0$. Hence we need only to show how many steps are needed until both these potentials will reach 0.

Fix any time step $t > 11$. We want to show that if $\Xi(t) > 0$ then in interval $[t-11, t]$ one of two events can happen:

(1) $\Phi(t-11) - \Phi(t) \geq 1$
(2) $\Xi(t-11) - \Xi(t) \geq \gamma^2 n/(24k)$.

This means that either of the potential decreases in each time interval of length at least 12.

We define a shorthand notation for the total demand as $\bar{n} = \sum_{j\in[k]} d^{(j)}$. We also denote the number of ants beyond a certain threshold: $\Psi(t) = \sum_{j\in[k]} \max\{0, f^{(j)}_t - (1+15\gamma)d^{(j)}\}$.

In any time τ, each ant is either idle or committed to some task:

$$
\begin{aligned}
n &= \sum_{j\in[k]} f^{(j)}_\tau + f^{(i^*)}_\tau = f^{(i^*)}_\tau + \sum_{f^{(j)}_\tau \leq (1-\gamma)d^{(j)}} f^{(j)}_\tau \\
&+ \sum_{(1-\gamma)d^{(j)} < f^{(j)}_\tau < d^{(j)}} f^{(j)}_\tau + \sum_{d^{(j)} \leq f^{(j)}_\tau < (1+15\gamma)d^{(j)}} f^{(j)}_\tau \\
&+ \sum_{f^{(j)}_\tau \geq (1+15\gamma)d^{(j)}} f^{(j)}_\tau - \sum_{j\in[k]} d^{(j)} + \sum_{j\in[k]} d^{(j)} \\
&= f^{(i^*)}_\tau - \sum_{f^{(j)}_\tau \leq (1-\gamma)d^{(j)}} \left((1-\gamma)d^{(j)} - f^{(j)}_\tau\right) \\
&+ \sum_{(1-\gamma)d^{(j)} < f^{(j)}_\tau < d^{(j)}} \left(f^{(j)}_\tau - (1-\gamma)d^{(j)}\right) \\
&- \sum_{d^{(j)} \leq f^{(j)}_\tau < (1+15\gamma)d^{(j)}} \left((1+15\gamma)d^{(j)}) - f^{(j)}_\tau\right) \\
&+ \sum_{f^{(j)}_\tau \geq (1+15\gamma)d^{(j)}} \left(f^{(j)}_\tau - (1+15\gamma)d^{(j)})\right) \\
&+ \bar{n} - \sum_{d^{(j)} > f^{(j)}_\tau} \gamma d^{(j)} + 15 \sum_{d^{(j)} \leq f^{(j)}_\tau} \gamma d^{(j)} \\
&\leq f^{(i^*)}_\tau - \Xi(\tau) + \Psi(\tau) + (1+15\gamma)\bar{n},
\end{aligned}
$$

where the last inequality is true because, out of six sums after the second equality, the first and fourth sums equal exactly to $\Xi(\tau)$ and $\Psi(\tau)$ respectively, the second sum gets canceled with the fifth one (giving the inequality) and the third sum is always negative. This inequality, knowing that $n \geq (1+17\gamma)\sum_{j\in[k]} d^{(j)}$ gives us $\Psi(\tau)+f^{(i^*)}_\tau \geq 2\gamma\bar{n}$. Thus at least one of the following is true: $f^{(i^*)}_{t-11} \geq \gamma\bar{n}$ or $\Psi(t-11) \geq \gamma\bar{n}$. If the latter is true then by Lemma 4.2, within the next 6 time steps at least $\gamma^2\bar{n}/2$ ants transition to i^*. Hence in both cases there exists a step $\tau \in \{t-5, t-4, \ldots, t\}$ in which we have $f^{(i^*)}_\tau \geq \gamma^2\bar{n}/12$. Since $\gamma^2\bar{n}/12 \geq 48k\ln n$ (by Assumption 3.1) we have, conditioned on $\mathcal{E}_{selection}$ that in step τ each task is selected by at least $\gamma^2\bar{n}/(24k)$ ants. Take any task j that in step τ contributes to sum in $\Xi(t)$ (*i.e.,* $t < \tau^{(j)}_1$) – such a task has to exist since $\Xi(t) > 0$. This task is selected by at least $\gamma^2\bar{n}/(24k)$ idle ants. In the following 5 steps, these ants will collect feedback from task j. If in all of these steps we will have $f^{(j)}_\tau < (1-\gamma)d^{(j)}$, then by $\mathcal{E}_{feedback}$ all ants will receive lack and join the task, which will result in a decrease of Ξ by at least $\gamma^2\bar{n}/(24k)$ (event (2)). Or in any of these steps we have $f^{(j)}_t \geq (1-\gamma)d^{(j)}$, which means that the task advanced to the second phase – which means that Φ decreases by 1 (event (1)).

Since each task can advance to the second phase at most once, event (1) can happen at most k times. Since $\Xi(0) \leq \bar{n}$, event (2) can happen at most $24k/\gamma^2$ times. After these at most $k + 24k/\gamma^2$ intervals hence at most $12(k+24k/\gamma^2)$ steps we must have $\Xi(t) = 0$ and $\Phi(t) = 0$. □

To show that the inaccuracy in the second phase is 0, we will prove that during the whole second phase the load of each task is always close to the demand. The inaccuracy in second and third phases will be bounded for each task separately. We denote by

$\mathsf{Inaq}_j^{\mathcal{E}}(T,i)$, the inaccuracy of j-th task in i-th phase during interval $[1,T]$.

Lemma 4.4. *[Inaccuracy Second Phase] Assume $\mathcal{E}$ and that $\gamma \le 0.01$, then we have for any task $j \in [k]$: $\mathsf{Inaq}_j^{15\gamma}(T,2) = 0$.*

Proof. We need to first show that $f_t^{(j)} \ge (1-\gamma)d^{(j)}$ for each time step t in interval $[\tau_1^{(j)}, T]$. Assume that $f_t^{(j)} \ge (1-\gamma)d^{(j)}$ and we will show it for $t+1$. If any ants are leaving task j (transitioning to i^*) in step $t+1$, this means that these ants received feedback `overload` from step t, which means by $\mathcal{E}_{feedback}$ that $W_t^{(j)} > (1-\gamma)d^{(j)}$. Since connection to i^* is only from an idle state i_2^j, we have $f_{t+1}^{(j)} \ge W_t^{(j)} > (1-\gamma)d^{(j)}$. We showed that in the whole interval $[\tau_1^{(j)}, T]$ the total number of ants committed to task j is at least $(1-\gamma)d^{(j)}$.

If $f_t^{(j)} \ge (1-\gamma)d^{(j)}$, we have conditioned on $\mathcal{E}_{idle}$ that $I_t^{(j)} \le 12\gamma W_t^{(j)}$, thus:

$$(1-\gamma)d^{(j)} \le W_t^{(j)} + 12\gamma W_t^{(j)},$$

and since $(1-\gamma)/(1+12\gamma) \ge 1-15\gamma$, we get the that in every step $\tau \in [\tau_1^{(j)}, \tau_2^{(j)} - 1]$, $|W_j^{(\tau)} - d^{(j)}| \le 15\gamma d^{(j)}$, thus $\mathsf{Inaq}_j^{15\gamma}(\tau) = 0$. Since the last equation is true in each step of the second phase (of task j), the claim holds. □

Finally we bound the inaccuracy in the third phase. The following lemma shows that shortly after the advancement of the task to the third phase, all the ants committed to this task will only be in the filtering machine. Secondly, this shows that the number of ants in state w_1^j will be at least $(1+\gamma)d^{(j)}$, hence the task will not be in the grey zone. We will later show that from such a configuration with at least $(1+\gamma)d^{(j)}$ ants in w_1^j and no ants in O^j, the system quickly reaches an almost-optimal allocation of ants to this task and stabilizes.

Lemma 4.5. *Assume $\mathcal{E}$ holds and that $\gamma \le 0.01$. Then, for each task $j \in [k]$ if $\tau_2^{(j)} + 2 \le T$ we have: $\forall_{s \in O^j} f^{(s)}_{\tau_2^{(j)}+2} = 0$, and $f^{(w_1^j)}_{\tau_2^{(j)}+2} \ge (1+\gamma)d^{(j)}$.*

It might happen that a very large number of ants is committed to task j in step $\tau_2^{(j)}$. Since having a much too large number of ants in a task also counts as inaccuracy, we need to show that the ants will leave task j. In the next lemma we have to prove that the ants slowly leave the filtering machine and transition to i^* until a correct (*i.e.*, close to the demand) number of ants remains. The total inaccuracy in this interval can be bounded because the decrease in the number of ants working on the task is geometric. After that, the ants move in at most two waves to the oscillating machine and remain there until the end of interval $\mathcal{I}$ maintaining an (almost) optimal number of working ants.

Lemma 4.6. *[Inaccuracy Third Phase] Assume $\mathcal{E}$ and that $\gamma \le 0.01$, then we have for any $j \in [k]$: $\mathsf{Inaq}_j^{15\gamma}(T,3) = O(\log n/\gamma)$.*

Proof. If $\tau_2^{(j)} + 100 > T$ then the claim holds because $\gamma < 1/100$.

In the opposite case by Lemma 4.5 we know that in step $\tau_2^{(j)} + 2$ all ants committed to task j can be only in the filtering machine and $f^{(w_1^j)}_{\tau_2^{(j)}+2} \ge (1+\gamma)d^{(j)}$. We want to show that starting from step $\tau_2^{(j)} + 3$, the overload of task j will decrease geometrically and it will stabilize to a value at most $5.5\gamma d^{(j)}$ and will not increase above this value within interval $\mathcal{I}$.

Observe that since $f^{(w_1^j)}_{\tau_2^{(j)}+2} \ge (1+\gamma)d^{(j)}$ then by $\mathcal{E}_{feedback}$ all ants receive `overload` from task j in step $\tau_2^{(j)} + 2$. This means that after $\tau_2^{(j)} + 2$, as long as the number of ants in state w_1^j is at least $(1+\gamma)d^{(j)}$ then no ant committed to task j will be in any of the states from O^j.

Note that ants in state w_1^j that receive feedback `overload` transition to idle state i_1^j with probability $a\gamma$. Thus the number of working ants decreases geometrically until it drops below $(1+\gamma)d^{(j)}$.

Consider time steps $t = \tau_2^{(j)} + 2, \tau_2^{(j)} + 3, \ldots$. We want to show that if in step t, value $f_t^{(w_1^j)}$ satisfies $f_t^{(w_1^j)} \cdot (1 - 1.05a\gamma) \ge (1+\gamma)d^{(j)}$ then in step $t+1$ we do not enter the grey zone. In this case we have $W_t^{(j)} \ge (1+\gamma)d^{(j)}$ and by $\mathcal{E}_{feedback}$ all ants receive feedback `overload` from task j thus ants leave state w_1^j with probability $a\gamma$ and no ant joins task j. Thus all ants working on task j in step $t+1$ are in state w_1^j and by $\mathcal{E}_{transition}$ we have:

$$W_{t+1}^{(j)} = f_{t+1}^{(w_1^j)} \ge f_t^{(w_1^j)} - f_t^{(w_1^j)} \cdot 1.05a\gamma \ge (1+\gamma)d^{(j)}.$$

Consider the smallest time step $t^* \ge \tau_2^{(j)} + 2$ such that $f_{t^*}^{(w_1^j)} \cdot (1 - 1.05a\gamma) \le (1+\gamma)d^{(j)}$. We know that until this step we had in task j an overload by a factor of at least $1+\gamma$ hence all ants received feedback `overload` by $\mathcal{E}_{feedback}$. Using this fact we can bound the rate of decrease of the load for task j in steps $\tau_2^{(j)} + 2, \tau_2^{(j)} + 3, \ldots, t^*$. By $\mathcal{E}_{transition}$, $W_\tau^{(j)} \le (1 - 0.95a\gamma)W_{\tau-1}^{(j)}$. Thus:

$$\begin{aligned} t^* - \tau_2^{(t)} &\le \log_{1/(1-0.95a\gamma)} n = \frac{\log n}{\log\left(1 - \frac{0.95a\gamma}{1-0.95a\gamma}\right)} \\ &\le 2.5 \log n/\gamma. \end{aligned}$$

In the remaining part of the proof we will show that in steps $[t^*, T]$, the load of the task will be accurate. We prove that in the next few steps after t^*, the ants committed to this task move from the filtering to the oscillating machine. Denote two sets of ants:

$$A_1 = \{i \in [n] : a_{t^*}^{(i)} = w_1^j\},$$
$$A_2 = \{i \in [n] : a_{t^*}^{(i)} = i_1^j\}.$$

We know that $(1+\gamma)d^{(j)}/(1 - 1.05a\gamma) \ge |A_1| \ge (1+\gamma)d^{(j)}$. Moreover by $\mathcal{E}_{feedback}$ we have $|A_1| \ge (1 - 1.05a\gamma)f_{t^*-1}^{(w_1^j)}$. The only way to be in state i_1^j in step t^* is to transition from w_1^j and this transition happens with probability $a\gamma$ thus $\mathbb{E}[\,|A_2|\,] = a\gamma f_{t^*-1}^{(w_1^j)}$

and by $\mathcal{E}_{feedback}$:

$$|A_2| \le 1.05a\gamma f_{t^*-1}^{(w_1^j)} \le \frac{1.05a\gamma|A_1|}{1-1.05a\gamma} \le \frac{1.05a\gamma(1+\gamma)d^{(j)}}{(1-1.05a\gamma)^2} \le 2.8\gamma d^{(j)},$$

where in the last equation we used the fact that $\gamma < 0.01$ and $a = 2.5$.

Now we want to show that out of these two groups A_1 and A_2, all of A_1 and some subset of A_2 will transition to the oscillating machine in steps $t^* + 2$ and $t^* + 3$ and then remain there for a polynomial number of steps.

Consider the number of ants working on task j in step $t^* + 2$. Observe that by the definition of the algorithm, since in step t^* all ants received feedback `overload` from task j and all states in O^j were empty then the number of working ants in steps $t^* + 1$ and $t^* + 2$ decreases in expectancy by a factor of $1 - a\gamma$. Moreover no new ants can join the task in these two steps from i^* because all ants received `overload` in step t^*. Hence we have by $\mathcal{E}_{transition}$:

$$W_{t^*+1}^{(j)} \le W_{t^*}^{(j)}(1-0.95a\gamma)$$

Now, in step $t^* + 1$ all ants working on task j are still in state w_1^j but in this step the load might be in the grey zone. Thus in this step, from w_1^j there are two possible transitions to idle states (i_1^j and i_3^j). And conditioned on $\mathcal{E}_{feedback}$ we get:

$$W_{t^*+2}^{(j)} \le W_{t^*}^{(j)}(1-0.95a\gamma)^2 + 720\ln n \le \frac{(1+\gamma)(1-0.95a\gamma)^2 d^{(j)}}{1-1.05a\gamma} \le (1-\gamma)d^{(j)}.$$

Now, conditioned on $\mathcal{E}_{feedback}$ all ants receive feedback `lack` from task j in step $t^* + 2$. This means that all remaining ants that are in step $t^* + 2$ in the filtering machine transition in step $t^* + 3$ to states i_3^j and w_2^j. Observe that all the ants from A_1 join states i_3^j or w_2^j in steps $t^* + 2$ and $t^* + 3$ because in step $t^* + 1$ these ants are either in w_1^j or i_1^j hence no ant from this set can transition to i^* in step $t^* + 1$. In step $t^* + 2$ some of these ants transition to w_2^j or i_3^j. In $t^* + 3$ all remaining of ants from A_1 that are in states from F^j transition to w_2^j or i_3^j, because they receive feedback `lack`. Observe secondly that some of the ants from A_2 may also join states w_2^j or i_3^j in steps $t^* + 2$ and $t^* + 3$ but no other ant can.

In step $t^* + 3$ we have the following:

$$(1+\gamma)d^{(j)} \le |A_1| \le f_{t^*+3}^{(w_2^j)} + f_{t^*+3}^{(w_4^j)} + f_{t^*+3}^{(i_3^j)} \le |A_1| + |A_2| \le (1+3.8\gamma)d^{(j)},$$

and no ant committed to task j is in any other state. Thus we have two "waves" of ants, where the second wave is one step behind the first one (it may happen that one of the waves is empty). In step $t^* + 4$ all ants from both waves are working (in states w_4^j and w_5^j), which results in `overload` conditioned on $\mathcal{E}_{feedback}$. Since ants from i^* need at least a sequence of 5 times `lack` in a row, no new ants will join this task. In the next two steps, both waves are split into working and idle states, where each ant transitions into an idle state with probability $b\gamma$. We have then by $\mathcal{E}_{feedback}$:

$$W_{t^*+6}^{(j)} \le (1+3.8\gamma)(1-0.9b\gamma)d^{(j)} + 720\ln n \le (1-\gamma)d^{(j)},$$

$$W_{t^*+6}^{(j)} \ge (1+\gamma)(1-1.1b\gamma)d^{(j)} - 720\ln n \ge (1-5.5\gamma)d^{(j)}$$

which means that by $\mathcal{E}_{feedback}$ all ants receive feedback `lack` in step $t^* + 6$. By repeating the same argument inductively we have that for all $i = 0, 1, \ldots$ in each step $t^* + 4 + 4i$ ants receive feedback `overload` and in each step t^*+6+4i each ant receives feedback `lack`. This means that no new ant joins task j and no ant leaves the states O^j at least until the end of interval $\mathcal{I}$ and the difference between the load of the task j and the demand $d^{(j)}$ is at most $5.5\gamma d^{(j)}$ in every step between t^* and T. This means that $\mathsf{Inaq}_j^{15\gamma}(T, 3) \le t^* - \tau_2^{(j)} = O(\log n/\gamma)$.

□

Directly from Lemma 4.3, Lemma 4.4 and Lemma 4.6 we get that $\mathsf{Inaq}^{15\gamma}(T) = O(\log n/\gamma)$ for any T, such that $T \le n^2$ with probability at least $1 - 1/n^2$. For longer intervals, we simply split them into subintervals of length n^2. We then bound the number of intervals in which event $\mathcal{E}$ fails. Our main Theorem 3.2 follows.

5 LOWER BOUND

The idea of our lower bound is as follows. From the bound on the memory size we can easily derive an upper bound s on the number of states of any ant. By Assumption 3.4 we know that from any state there exists a sequence of at most s feedbacks that an ant transitions from the original state to a working state of any task j with constant probability. We set the demand vector to be $d^{(j)} = \sqrt{n}$ for all $j \in [k]$. We show that if in s consecutive steps an absolute value of the deficit of some task j is smaller than $2s\varepsilon\gamma^* d^{(j)}$, then with high probability $\Omega(n^{2/3})$ ants will join task j during this interval. This shows that the algorithm cannot be $2s\varepsilon\gamma^*$-accurate.

6 CONCLUSIONS

We presented a simple proof-of-concept algorithm which shows that the problem of distributed task allocation by constant-memory agents (that cannot even store the number of all agents) can be solved in the model with binary feedback even if the feedback is noisy. The algorithm is very resilient to noise, and our preliminary simulations show that it can also adapt to changes in demands, changes of the number of ants and even changes of the number of tasks. It would be interesting to get a concrete result in a dynamic environment. Our results also suggest that there might exist a memory-accuracy tradeoff.

The algorithm embraces the seeming obstacle of synchronization (which we introduced to model the delay of information) to perform controlled oscillations. It would be interesting to see if variations of this algorithm also work in settings of less synchronization (what if some ants collect feedback more frequently than others *i.e.,* what if local clocks of the ants do not tick at the same rate?).

Moreover, it remains an open problem to understand if and by how much simple communication among the ants can help. This leads to the question of which other noise models would make sense to study and how to design experiments with real ants to gather more knowledge about the way noise affects the sensing.

REFERENCES

[1] Heiner Ackermann, Simon Fischer, Martin Hoefer, and Marcel Schöngens. 2011. Distributed algorithms for QoS load balancing. *Distributed Computing* 23, 5-6 (2011), 321–330.

[2] Petra Berenbrink, Tom Friedetzky, Frederik Mallmann-Trenn, Sepehr Meshkinfamfard, and Chris Wastell. 2018. Threshold load balancing with weighted tasks. *J. Parallel Distrib. Comput.* 113 (2018), 218–226. https://doi.org/10.1016/j.jpdc.2017.10.012

[3] Samuel N. Beshers and Jennifer H. Fewell. 2001. Models of division of labor in social insects. *Annual review of entomology* 46, 1 (2001), 413–440.

[4] Prassede Calabi. 1988. Behavioral flexibility in Hymenoptera: a re-examination of the concept of caste. *Advances in myrmecology* (1988), 237–258.

[5] Daniel Charbonneau and Anna Dornhaus. 2015. When doing nothing is something. How task allocation mechanisms compromise between flexibility, efficiency, and inactive agents. *Journal of Bioeconomics* 17 (2015), 217–242.

[6] Blaine J Cole. 1991. Short-term activity cycles in ants: generation of periodicity by worker interaction. *The American Naturalist* 137, 2 (1991), 244–259.

[7] Blaine J Cole and Franc I Trampus. 1999. Activity cycles in ant colonies: worker interactions and decentralized control. In *Information processing in social insects*. Springer, 289–307.

[8] Alejandro Cornejo, Anna R. Dornhaus, Nancy A. Lynch, and Radhika Nagpal. 2014. Task Allocation in Ant Colonies. In *Distributed Computing - 28th International Symposium, DISC 2014, Austin, TX, USA, October 12-15, 2014. Proceedings*. 46–60. https://doi.org/10.1007/978-3-662-45174-8_4

[9] Tomer J Czaczkes, Christoph Grüter, and Francis LW Ratnieks. 2013. Negative feedback in ants: crowding results in less trail pheromone deposition. *Journal of the Royal Society Interface* 10, 81 (2013), 20121009.

[10] Ana Duarte, Ido Pen, Laurent Keller, and Franz J. Weissing. 2012. Evolution of self-organized division of labor in a response threshold model. *Behavioral Ecology and Sociobiology* 66, 6 (2012), 947–957.

[11] Ana Duarte, Franz J. Weissing, Ido Pen, and Laurent Keller. 2011. An evolutionary perspective on self-organized division of labor in social insects. *Annual Review of Ecology, Evolution, and Systematics* 42 (2011), 91–110.

[12] Audrey Dussutour, Stamatios C Nicolis, Jean-Louis Deneubourg, and Vincent Fourcassié. 2006. Collective decisions in ants when foraging under crowded conditions. *Behavioral Ecology and Sociobiology* 61, 1 (2006), 17–30.

[13] Clive A Edwards and Patrick J Bohlen. 1996. *Biology and ecology of earthworms*. Vol. 3. Springer Science & Business Media.

[14] Deborah M. Gordon. 1989. Dynamics of task switching in harvester ants. *Animal Behaviour* 38, 2 (1989), 194–204.

[15] D. M. Gordon. 1996. The organization of work in social insect colonies. *Nature* 380, 14 March (1996), 121–124.

[16] Deborah M Gordon. 1999. *Ants at work: how an insect society is organized*. Simon and Schuster.

[17] Deborah M. Gordon and Natasha J. Mehdiabadi. 1999. Encounter rate and task allocation in harvester ants. *Behavioral Ecology and Sociobiology* 45, 5 (01 Apr 1999), 370–377. https://doi.org/10.1007/s002650050573

[18] Torben Hagerup and Christine Rüb. 1990. A Guided Tour of Chernoff Bounds. *Inform. Process. Lett.* 33, 6 (1990), 305–308.

[19] MP Hassell and HN Comins. 1978. Sigmoid functional responses and population stability. *Theoretical Population Biology* 14, 1 (1978), 62–67.

[20] Martin Hoefer and Thomas Sauerwald. 2013. Brief Announcement: Threshold Load Balancing in Networks. In *32nd Symposium on Principles of Distributed Computing PODC*. ACM, Montreal, Canada, 54–56. The full version is available at https://arxiv.org/abs/1306.1402.

[21] Bert Hölldobler and Edward O Wilson. 1990. *The ants*. Harvard University Press.

[22] Gavin M. Leighton, Daniel Charbonneau, and Anna Dornhaus. 2016. Task switching is associated with temporal delays in Temnothorax rugatulus ants. *Behavioral Ecology* 28, 1 (2016), 319–327.

[23] Michael Mitzenmacher and Eli Upfal. 2005. *Probability and Computing: Randomized Algorithms and Probabilistic Analysis*. Cambridge University Press.

[24] Henrique M. Pereira and Deborah M. Gordon. 2001. A trade-off in task allocation between sensitivity to the environment and response time. *Journal of Theoretical Biology* 208, 2 (2001), 165–184.

[25] N. Pinter-Wollman, J. Hubler, J. A. Holley, N. R. Franks, and A. Dornhaus. 2012. How is activity distributed among and within tasks in Temnothorax ants? *Behavioral Ecology and Sociobiology* 66, 10 (2012), 1407–1420.

[26] Tsvetomira Radeva, Anna Dornhaus, Nancy Lynch, Radhika Nagpal, and Hsin-Hao Su. 2017. Costs of task allocation with local feedback: Effects of colony size and extra workers in social insects and other multi-agent systems. *PLoS computational biology* 13, 12 (2017), e1005904.

[27] Thomas O Richardson, Jonas I Liechti, Nathalie Stroeymeyt, Sebastian Bonhoeffer, and Laurent Keller. 2017. Short-term activity cycles impede information transmission in ant colonies. *PLoS computational biology* 13, 5 (2017), e1005527.

[28] Horst R Thieme. 2003. *Mathematics in population biology*. Princeton University Press.

[29] Chris Tofts. 1993. Algorithms for task allocation in ants.(A study of temporal polyethism: theory). *Bulletin of mathematical biology* 55, 5 (1993), 891–918.

[30] Chris Tofts and Nigel R Franks. 1992. Doing the right thing: ants, honeybees and naked mole-rats. *Trends in ecology & evolution* 7, 10 (1992), 346–349.