Optimal Pheromone Utilization

Yehuda Afek
afek@cs.tau.ac.il

Roman Kecher
roman.kecher@cs.tau.ac.il

Moshe Sulamy
moshe.sulamy@cs.tau.ac.il

Blavatnik School of Computer Science, Tel-Aviv University, Israel

Abstract

The minimal amount of pheromones necessary and sufficient to deterministically find a treasure (food) by a colony of ants, each modeled by either a Finite State Machine or a Turing Machine, is considered. The k mobile agents (ants), initially located at the origin (nest) of an infinite grid, communicate only through pheromones to perform a collaborative search for an adversarially hidden treasure at an unknown distance D. We begin by proving a tight lower bound of $\Omega(D)$ on the number of pheromones required by a FSM to even complete the search, and continue to reduce the lower bound to $\Omega(k)$ for the stronger ants modeled as TM. Matching optimal algorithms, in terms of run time as well as pheromone usage, are provided both for synchronous and asynchronous models in each case.

1 Introduction

While performing mostly simple and local computations, ants solve complicated problems with great collaboration. Such collaborative work, with limited communication and computational power, is at the heart of distributed computing. By studying and understanding the behavior of ants from a distributed computing perspective, progress and insights can be made for both the distributed computing and biology fields.

In this paper, we analyze the minimum number of pheromones required by ants to find a treasure—a food item. This is the basis of the *Ants Nearby Treasure Search (ANTS)* problem, first presented by Feinerman et al. [6]. In the original ANTS problem, k identical ants start at the origin (i.e., the *nest*) of an infinite grid, and need to find a treasure located at an unknown distance D. The ants are modeled as randomized mobile Turing machines, traveling along $\mathbb{Z}^2$, and cannot communicate after leaving the nest. Once any ant steps onto the grid point containing the treasure, the treasure is found and the algorithm terminates. A trivial lower bound on the time required to find the treasure is $O(D + D^2/k)$.

Our model is based on the model presented by Lenzen et al. [8], where ants mark *each* visited grid point by a pheromone. Pheromones are a biological resource which might be limited in quantity or costly to be used lightly. In our model, we allow ants to *choose* whether or not to leave a pheromone upon visiting a grid point, thus analyzing the minimal amount of pheromones required to find the treasure for various computational models is a natural research direction.

*This research was supported by the Israel Science Foundation (grant 6693/11).

1.1 Contributions

We present lower bounds on the minimal amount of pheromones required by k ants in order to find a treasure at an unknown distance D, where ants are modeled either by a *Finite State Machine* or a *Turing Machine*. We further prove that our bounds are tight by presenting matching upper bounds for each computational model, in both the synchronous and asynchronous models.

To the best of our knowledge, no such lower bounds have been previously analyzed; the only relevant algorithm, presented in [8], uses $O(D^2)$ pheromones to find the treasure with ants modeled as FSM with extra capabilities.

For the FSM model, we prove that $\Omega(D)$ pheromones are required to find the treasure. We provide optimal algorithms for both the synchronous and asynchronous models that, for k ants, find the treasure in $O(D + D^2/k)$ rounds, while using only $O(D)$ pheromones.

For the TM model, no pheromones are required to find the treasure; however, we prove that $\Omega(k)$ pheromones are required in order to find the treasure in optimal time, in $O(D+D^2/k)$ rounds. Once more, we provide optimal algorithms for both synchronous and asynchronous models that, for k ants, finds the treasure in $O(D + D^2/k)$ rounds, while using $O(k)$ pheromones.

Remark 1. When operating in the synchronous scheduling model, we assume the existence of an *emission scheme* – an algorithm which controls the emission of ants from the nest. The only assumption is that the emission scheme guarantees that no two ants leave the nest simultaneously, and the time it takes for k ants to be emitted is denoted as $t_e(k)$. This is a cost that is added to the maximal number of rounds required by the provided algorithms in the synchronous model.

Outline The rest of this paper is organized as follows. Following next is a summary of related and previous work regarding the ANTS problem. Section 2 covers the models used in this paper: it describes the general model of ANTS with pheromones, as well as the synchronous and asynchronous scheduling models. Sections 3 and 4 provide the computational models of FSM and TM ants, respectively, as well as lower bounds and matching upper bounds for the number of pheromones required by each computational model (under synchronous and asynchronous scheduling models). Finally, section 5 presents a discussion of conclusions and results, as well as future questions following this paper.

1.2 Related work

The *Ants Nearby Treasure Search (ANTS)* problem was introduced by Feinerman, Korman, Lotker, and Sereni [6], showing that matching the lower bound requires knowledge or a constant approximation of k.

The ANTS problem is further discussed in [2, 3, 4, 7, 8] in various models. In [5], lower bounds on the memory required by the ants to find the treasure are given. In [3, 4], ants modeled by a weaker computational model, that of a *Finite State Machine*, are presented, in both synchronous [3] and asynchronous [4] models. Emek et al. take a computability point of view [2], by analyzing the minimal number of ants required to find the treasure within finite time. In [7], the *Selection Complexity* is studied, a measure of how likely a given algorithmic strategy is to arise in nature.

Pheromones were introduced to ANTS by Lenzen et al. [8], who assumed that ants mark *each* visited grid cell. Our model differs in that the ants are allowed to *choose* whether or not to emit pheromones.

2 Model

Throughout the paper, regardless of the computational model in use, we assume the ants are deterministic. We assume that k, the total number of participating ants, is unknown to the ants. We only discuss *uniform algorithms*, in the sense that a single algorithm (whatever the employed computational model is) works for all $k \in \mathbb{N}$. The ants have no means of communication other than pheromones; upon entering a grid cell, an ant is capable of (i) sensing the existence (or lack thereof) of a pheromone on that grid location, and (ii) releasing a pheromone (or not), all as a single operation. This means that an ant senses the existence of a pheromone, places a pheromone (if it so chooses) at its current location, and moves one space to any chosen direction, all as a single atomic operation.

This model deviates from the model presented in [8] in two key points: the ants do not have to emit pheromones on every single grid cell they visit, and we do not assume that the ants know for free the direction or path back to the nest. We only assume that the ants share a common orientation (being able to tell north, east, south and west directions) and derive everything from that, the pheromones, and the computational model.

During the search for the treasure, the ants are able to emit pheromones in order to coordinate the collaborative search in the grid. Essentially, this is a distributed search which is carried out by many different mobile agents, each performing its own set of moves. The scheduling of said moves is analyzed under two different models: a synchronous model, in which all ants make their next move simultaneously, and an asynchronous model, in which an adversary repeats the process of scheduling only a single ant to make its next move. Naturally, each model imposes different limitations and requires a different run time analysis. This section covers these topics.

2.1 Synchronous model

In the synchronous model, we divide the run time into multiple synchronous rounds. In each round, all the ants take a single step, simultaneously.

Consider what happens if two ants start the search simultaneously: both of them will operate in exactly the same way from that point forward, since the ants are deterministic and their view of the world is exactly the same. These ants have no different initial knowledge, nor are there any available means for breaking the symmetry since we assume that in the synchronous model everything happens at the same time – including the check for the existence of a pheromone on the current location and the emission of a new pheromone. This is a single operation which the two ants are going to perform concurrently, unaware of their partner doing the same. Consequently, it is impossible to separate any number of ants that leave the nest at the same round and they will effectively behave as a single ant. Therefore, under this model we make a fundamental basic assumption: no two ants leave the nest at the same round.

Emission scheme An emission scheme is a mechanism which decides the order and rate of ants leaving the nest. The idea of emission schemes was first introduced in [3,4], where ants are assumed to be randomized FSM, and various properties of those mechanisms (such as the delays between consecutive groups of ants leaving the nest) are discussed. We adapt the idea and assume the existence of such a mechanism, whose only premise in our case is that no two ants leave the nest at the same round. We also denote the time it takes for k ants to leave the nest as $t_e(k) \geq k$.

Following the definition of the synchronous model it is apparent that unless an emission scheme is given, there is no way of breaking the inherent symmetry of the model without resorting to

randomization. We leave the open question of devising such an efficient emission scheme under our settings for future work.

2.2 Asynchronous model

In the asynchronous model, ants' moves are governed by an adversarial scheduler. The scheduler selects a single ant to make a single move, and once finished, the scheduler proceeds to select another ant (possibly the same one) to make its next move. This process repeats Ad infinitum, in an iterative manner. Effectively, this implies that no two ants take a step at the exact same time. The placement of a pheromone on a grid cell is like a *test-and-set* operation, an ant atomically in one operation checks that the cell is not marked and marks it with a pheromone. Notice that, compared to the previous model of scheduling, the current asynchronous setting is closer to the reality.

Our definition of a round in the given asynchronous model is as follows: round $i \in \mathbb{N}$ ends after every ant has made at least a single step since round $i-1$ has ended. Round 0 is defined as the initial state, in which no ant took any step yet.

Notice that if we instead use an alternative definition of a round in which round r ends as soon as all ants took at least r steps, such as defined in [8], then our own algorithms (as well as some provided in predating work) would actually run in $O(D^2)$ time under this very definition. That is due to the fact that an adversary scheduler could let just one ant perform almost the entire search, covering all $D^2 - 1$ cells other than the one with the treasure, and then have all other ants run for at least that number of moves.

Remark 2. The actual number of pheromones used during an asynchronous search is potentially unlimited, e.g., a scheduler might stop scheduling any ant right before it reaches the treasure, whilst keep scheduling the rest. Therefore, we consider the maximal number of pheromones which is *required* to ensure that the treasure is found, given that every ant is scheduled infinitely often.

3 Ants as Deterministic Finite State Machines

In this section we assume that all mobiles agents, referred to as ants, are implemented by a Deterministic Finite State Machines (FSM). Formally, such an FSM is defined by the following 3-tuple:

$$\Pi = \langle Q, s_0, \delta \rangle ,$$

where Q is the set of states, $q_0 \in Q$ is the initial state, and

$$\delta : Q \times \{P, \overline{P}\} \to Q \times \{P, \overline{P}\} \times \{North, East, South, West, Stay\}$$

is the transition function, which given a current state and the existence of a pheromone on the current grid cell, determines the next state, whether or not to emit a pheromone at the current cell, and the direction in which the ant should move (if any). Note that the orientation is assumed to be shared between all the ants, so that the directions $\{North, East, South, West\}$ are assumed to have the same meaning and represent the same axis for any involved ant.

Throughout the paper FSM algorithms are not described through explicit use of this formal definition, but rather through common algorithmic pseudo-code. In every such description it is straightforward and immediately apparent that an explicit description of the aforementioned formal form exists, as the main limitations of this model are lack of any memory and the sheer necessity of making only "local transitions" as there is no global information other than what is encoded in the constant number of states and the existence (or lack) of a pheromone at the current grid cell occupied by the ant.

3.1 Lower bound

Theorem 3.1. *In the Finite State Machine computational model, an ant has to emit $\Omega(D)$ pheromones to search every grid cell at distance $d \leq D$. Assuming the FSM is uniform, i.e., a single FSM works for all $D \in \mathbb{N}$.*

Proof. Let us assume by contradiction that there exists FSM (ant) A that is capable of finding an adversarially hidden treasure at any unknown distance $D \in \mathbb{N}$ using $o(D)$ pheromones. Let S be the number of states in A.

Define *layer* n as the set of grid cells (x, y) such that $|x| + |y| = n$. There exists a distance D such that if we look at its search space, there must exist $S + 1$ consecutive layers with no pheromones, otherwise there are $\Omega(D)$ pheromones overall, which is a contradiction. Looking at these consecutive layers, we denote the layer furthest away from the origin as layer l.

Let us consider the first time the ant arrives at layer l. Denote its state at that point as s'. Looking at its path to this point up to S steps back, the ant must have passed state s' once more. Therefore we have a path that starts at some layer $l-i$ $(1 \leq i \leq S)$ with state s' and ends at layer l in the same state, during which the ant has not placed any pheromones nor encountered any. This path therefore must be repeated infinitely from layer l onwards. But since layer l contains more grid cells than all previous layers (and this is the first time for the ant to get as far as l from the origin), the ant does not cover all grid cells with distance $d \leq D$. Therefore, there exists a grid cell which is not covered, and the ant would not find the treasure if it is placed there. □

Corollary 3.2. *A lower bound on the number of pheromones emitted during an exhaustive search for a treasure which is located at an unknown distance D from the origin in the Finite State Machine computational model is $\Omega(D)$, regardless of the number of ants taking part in the search.*

3.2 Synchronous Finite State Machines

The proposed algorithm is as follows. We use north, east, south and west rays of pheromones, forming a plus sign with the nest in the center (using pheromones as guides, an idea inspired by [3, 4]). Those rays serve as markers, so that the ants know when to turn. The intuition is that using FSM as a computational model implies that the ants only have a finite number of move patterns to use and can only rely on pheromones in order to change their behavior, as they have no available (non constant) memory of their own.

Therefore, the ants start their exploration from the northern ray, and perform pairs of east-south "zigzag" moves (a move to the east followed by a move to the south) until they encounter the eastern ray of pheromones, which indicates that they should now emit two pheromones to extend the east ray and then switch to west-south "zigzag" moves in order to reach and extend the southern ray. Continuing this process, the ants reach the western ray and then get back to the northern ray, at which point an ant moves north to find a pheromone-free grid cell which indicates its next layer to explore. Note that considering only odd-indexed cells is enough. Important details are described next, while the main idea is depicted in Figure 1 and a more formal description is provided in Algorithm 1.

The main issue with the synchronous model is the need to prevent any two ants from colliding and performing the same work, e.g., one ant might arrive at a pheromone-free grid cell at the north ray concurrently with another ant and both proceed together. We tackle this issue by partitioning the ants to two logical groups, each following a slightly different algorithm. The first group is that of newbie ants, which are ants that have yet to explore their first layer, and non-newbie ants.

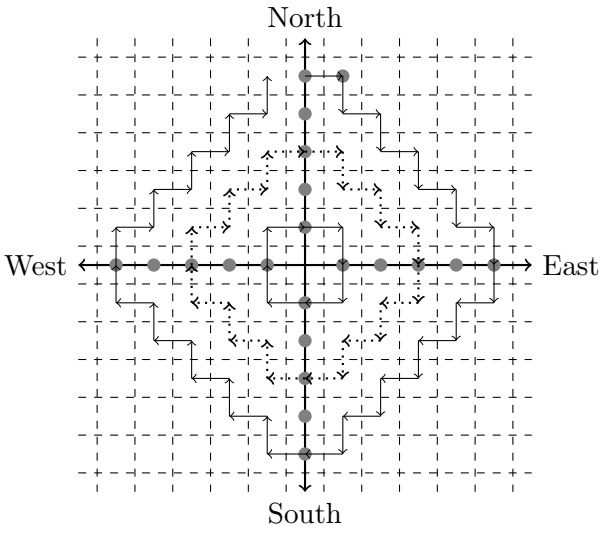

Figure 1: Two synchronous FSM ants searching for food. Circles indicate pheromones.

Naturally, an ant is considered a newbie when it is emitted, and becomes non-newbie when it explores its first layer.

Non-newbie ants travel north until they encounter the first odd-indexed pheromone-free grid cell, which marks their next layer to explore. Upon reaching such a cell, the non-newbie ant marks it with a pheromone and also emits another pheromone on the first grid cell it visits during its exploration phase (which is directly to the east). This pheromone is later checked by newbie ants; a newbie ant looks for the first odd-indexed pheromone-free grid cell on the north ray, exactly as do non-newbie ants, but after marking such a cell it then makes one step to the east and waits a single round to make sure that it has not collided with a non-newbie ant. If it detects a collision (by discovering a pheromone), priority is given to the non-newbie ant and the newbie ant just proceeds to search for a different free layer. Note that in order to keep the gaps introduced by the emission scheme, all newbie ants perform the same operation of checking the eastern grid cell for every odd-indexed cell on the north ray, even if they are marked with pheromones.

Lemma 3.3. *Newbie ants cannot collide with other newbie ants.*

Proof. The only requirement from the given emission scheme is that no two ants leave the nest at the same time. The gap that is introduced by the emission scheme is kept by making all newbie ants perform the same number of steps for every grid cell they pass before becoming non-newbies. □

Lemma 3.4. *Newbie ants cannot explore same layer (collide) as non-newbie ants.*

Proof. Newbie ants give precedence to non-newbie ants by checking the grid cell east to the current one. Such eastern cells can only be marked by non-newbie ants, and when encountered cause a skip of the current layer for the newbie ants. □

Lemma 3.5. *Non-newbie ants cannot collide with other non-newbie ants.*

Proof. Let us assume by contradiction that two non-newbie ants collide at some grid cell. Denote as $d_1 < d_2$ the last layers completed by these ants before they collided (those exist because the ants were newbies once). Then we have $t_{start}(d_1) < t_{start}(d_2)$, otherwise the second ant would not have passed layer d_1 on its way to d_2. And $t_{end}(d_2) - t_{end}(d_1) = t_{start}(d_2) - t_{start}(d_1) + 8(d_2 - d_1) > 8$, which

Algorithm 1 Synchronous FSM distributed treasure search

```
1: EmitPheromone(East, South, West)                                  ▷ Initial setup
2: newbie ← true
3: while newbie = true do                          ▷ Identify first free layer, give precedence
4:     Move(North)
5:     free = CheckMarkCurrentCheckEast()                  ▷ Takes a fixed number of rounds
6:     if free = true then
7:         ExploreLayer()          ▷ Turn (and extend ray) when encountering a pheromone
8:         newbie ← false
9:     end if
10: end while
11: while true do                                                    ▷ Explore free layers
12:     Move(North)
13:     free = CheckMarkCurrent()
14:     if free = true then
15:         EmitPheromone(East)
16:         ExploreLayer()          ▷ Turn (and extend ray) when encountering a pheromone
17:     end if
18: end while
```

means that there is a gap between the ants after their last separate layer explorations. Therefore, the gap can not be closed until their following exploration, in contradiction to the assumption. □

Lemma 3.6. *Every $O(1)$ steps, every ant either explores a new grid cell or increases its distance from the nest.*

Proof. Lemmas 3.3, 3.4 and 3.5 prove that no two ants collide, therefore only a single ant explores every layer, while the rest skip that layer and increase their distance from the nest (at least once every some constant number of steps). □

Theorem 3.7. *Algorithm 1 finds the treasure in $O(t_e(k) + D + D^2/k)$ time and utilizes $O(D)$ pheromones, hence it is optimal given a matching emission scheme.*

Proof. It takes $t_e(k)$ time for all k ants to be emitted. Following Lemma 3.6, after $O(1) \cdot (D + D^2/k)$ steps there is at least one ant at distance D from the nest. If there is an ant at distance D then all layers $i < D$ which are not yet fully explored contain an ant (an ant only continues after fully exploring its layer), and from that point it takes at most $8D$ rounds for all layers $i \leq D$ to be fully explored, for a total of $O(D + D^2/k)$. □

3.3 Asynchronous Finite State Machines

The proposed algorithm for the asynchronous model is a variation of the aforementioned synchronous algorithm. The key observation is that no ant can make any assumption on the progress of other ants, and hence cannot rely on pheromones emitted by other ants for its navigation clues. We tackle this problem by visiting the rays in a certain order, which guarantees that every ant only has to rely on the pheromones that had to be emitted prior to when it has started the exploration phase (either by itself or other ants).

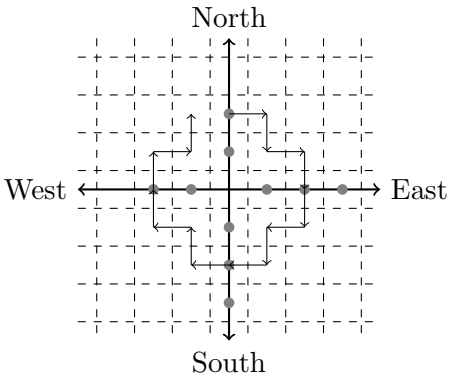

Figure 2: One asynchronous FSM ant searching for food. Circles indicate pheromones.

Notice that in the synchronous model we had to make sure that no two ants explore the same location at the same time, while in the asynchronous case we rely on the test-and-set operation of placing a pheromone to ensure this. This operation not only prevents two ants from following the same route, but also renders the problem of breaking the symmetry a non-issue.

The full solution is described in Algorithm 2 and illustrated in Figure 2. The main idea is almost the same as before; the colony constructs four rays of pheromones which form a plus shape with the nest at its center. The ants can then start by extending the east, south and west rays, and then begin their exploration phase from the north ray. Using this chain of operations guarantees that every ant starts the search in a layer which already has pheromones in all the rays – thus ensuring that the ant is able to finish the exploration successfully. Note that the nest remains free of pheromones throughout the search in order to let ants easily identify it, as every ant has to return to the nest in between exploration phases.

Lemma 3.8. *An ant is always able to successfully finish exploring any layer it started.*

Proof. The north ray is never longer than the east, south or west rays. Therefore, any ant is always able to find pheromones on the other three rays, which allow it to navigate successfully and finish its exploration procedure regardless of how the other ants are scheduled. □

Theorem 3.9. *Algorithm 2 terminates successfully in $O(D + D^2/k)$ rounds and the total number of emitted pheromones is $O(D)$, which make it optimal.*

Proof. We make two assumptions in order to simplify the analysis process. First, we assume that all ants move exactly once in every round, otherwise the total run time (number of rounds) would only improve. Second, we analyze the algorithm k layers at a time, again ignoring some extra progress as it does not affect our asymptotic analysis of the expected runtime.

It takes k rounds to place k pheromones on the east ray, and after a total of at most $7k$ rounds, all the ants are ready at the north ray and the first k layers are ready to be explored. At most $8k$ rounds later all the first k layers are fully explored, and an extra k rounds are required for the ants to get back at the nest. To summarize, the first k layers are fully explored after at most $16k$ rounds.

Marking the rays for the next k layers takes at most $14k$ rounds, after which the ants are ready to explore those layers. Exploring layers $k < i \leq 2k$ takes up to $16k$ rounds to complete, and another $2k$ rounds are required for the ants to get back to the nest. In total, it takes an extra $32k = 2 \cdot 16k$ rounds, at most, to explore the second set of k layers.

Inductively, we have that the total time for k ants to explore D layers is bound by $\Sigma_{d=1}^{D/k} d \cdot 16k = 8(D + D^2/k)$ asynchronous rounds. □

Algorithm 2 Asynchronous FSM distributed treasure search

```
1: while true do
2:     MoveUntilFreeAndMark(East), MoveUntilFree(West)
3:     MoveUntilFreeAndMark(South), MoveUntilFree(North)
4:     MoveUntilFreeAndMark(West), MoveUntilFree(East)
5:     MoveUntilFreeAndMark(North)          ▷ Explore layer starting from north ray
6:     ExploreLayer()                       ▷ Turn when encountering a pheromone
7:     MoveUntilFree(South)
8: end while
```

3.4 Summary

We have shown a lower bound of $\Omega(D)$ on the number of pheromones that must be emitted for a FSM to perform an exhaustive search. That lower bound remains valid regardless of the way the ants are scheduled, and is a pure computational limit of the FSM model. Following that, we have presented algorithms that match the lower bound in the number of emitted pheromones, but still allow k ants to cooperate and find the treasure in optimal time, thus the lower bound is tight. The presented algorithms propose feasible solutions for both synchronous and asynchronous models, and have only minor differences that result from the slightly different scheduling models.

Table 1 summarizes our results. As aforementioned, the only open issue is the emission scheme on which we rely in the synchronous case. The only requirement that is posed on such a mechanism is to prevent two ants from leaving the nest simultaneously, which is considered a reasonable assumption in the real world of biological creatures, where no two actions take place at the same exact moment.

The next immediate direction to head in is to further reduce the number of pheromones. It is apparent that in order to do so, a stronger computational model must be employed. Therefore, the following section provides an analysis of ANTS implemented by Turing Machines.

4 Ants as Turing Machines

A Turing Machine (TM) is a well known model of computation. Essentially, a TM can be thought of as a FSM sitting on an infinitely long tape (memory) containing symbols from some alphabet Γ, which the machine is able to alter at will through a read/write head that is able to move anywhere along the tape.

More formally, under our settings, a Turing Machine M is a 4-tuple

$$M = \langle Q, s_0, \Gamma, \delta \rangle ,$$

where Q is the set of states, $s_0 \in Q$ is the initial state, $\Gamma = \{0, 1\}$ is the tape alphabet, and

$$\delta : Q \times \Gamma \times \{P, \overline{P}\} \to Q \times \Gamma \times \{\triangleleft, \triangleright\} \times \{P, \overline{P}\} \times \{North, East, South, West, Stay\}$$

is the transition function, which given a current state, the contents of the tape, and the existence of a pheromone on the current grid cell, determines the next state, the operation of the tape's

head, whether or not to emit a pheromone at the current cell, and the direction in which the ant should move (if any). Note that, as before, the orientation is assumed to be shared between all the different ants, so that the directions $\{North, East, South, West\}$ are assumed to have the same meaning and represent the same axes for any involved ant. Also, once more, algorithms will be described as pseudo-code rather than in their explicit forms under the given model.

Also worth noting is that we assume that any TM is able to make quite a few internal transitions until it decides on the next move. For instance, when it has to compute the next value of a counter which has just been incremented. We treat the sequence of such internal transitions as a single move of the machine. Formally, it is possible to add another set of outputs which denotes the end of the current computation.

It is reasonable to assume that real world ants do not posses the full set of TM capabilities, e.g., they certainly do not have unlimited memory. Therefore, the only added value of employing the more capable computational model of Turing Machines lies in the new ability of maintaining a counter; despite the theoretically unlimited size of the tape, the suggested algorithms require only $O(log(D))$ bits of information for various counters. These are the only contents that are actually stored on the tape.

4.1 Lower bound

Obviously, even a single ant implemented as a TM could find the treasure in $O(D^2)$ rounds by performing a spiral shaped search, and it would need absolutely no pheromones in order to do so. Therefore, the only meaningful lower bound to consider in this section would be the number of pheromones that have to be emitted in order to perform the search cooperatively; i.e., achieve an optimal run time of $O(D + D^2/k)$ rounds for k ants.

Theorem 4.1. *Performing a distributed search for a treasure at an unknown distance $D \in \mathbb{N}$ by k TM ants in $O(t_e(k) + D + D^2/k)$ rounds under the synchronous model, requires $\Omega(k)$ pheromones.*

Proof. Assume by contradiction that T is a TM (ant) which is able to find a treasure at any distance D in $O(t_e(k) + D + D^2/k)$ synchronous rounds (for any emission scheme) with any k ants, while only using $o(k)$ pheromones. We will devise a specific emission scheme, dependent on T, which does not achieve the assumed run time – thus reaching a contradiction.

Let us analyze the behavior of a group of k ants implemented by TM T. We start by emitting only one ant at round $t_0 + 1 = 1$. That ant must stop emitting pheromones at some round (perhaps right away), denoted as t_1. T only emits $o(k)$ pheromones, therefore t_1 must exist. We emit the second ant at round $t_1 + 1$, and again wait until round t_2 when both ants finish emitting all of their pheromones. We emit the third ant at time $t_2 + 1$ and repeat the process until we get to emit the first ant which does not emit any pheromones: ant p, emitted at time $t_{p-1} + 1$. There is at least one such ant, since the total number of pheromones is $o(k)$. Notice that two invariants hold once ant p is emitted: (i) no extra pheromones will be emitted by all ants $i \leq p$ according to our assumption (their view of the world does not change), and (ii) all ants $j > p$ operate exactly as ant p, since they are deterministic and share a common view of the world. Also worth noting is that t_i, for all $i \leq p$, are constants which only depend on the specification of TM T.

We shall now increase the number of ants and keep emitting them, a single ant every round, until a total of $k' \gg k \geq p$ ants are emitted. Ants do not know k nor k', therefore the aforementioned behavior does not change. However, since all $k' - p$ (at least) ants operate the same, there exists some $D \gg k'$ such that it takes the ants $\Theta(t_e(k') + D + D^2/p) > O(t_e(k') + D + D^2/k')$ rounds to find the treasure, in contradiction to our assumptions. □

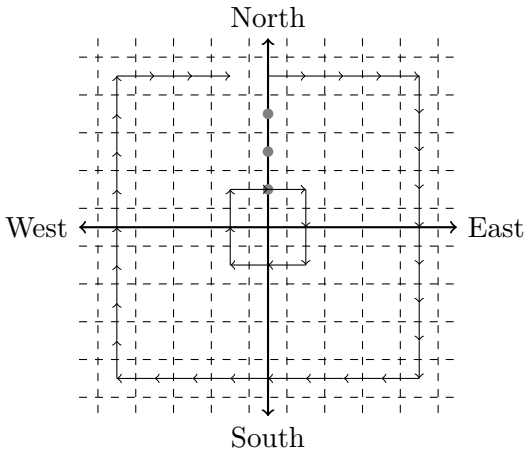

Figure 3: (A)Synchronous TM ant searching for food. First ant of a total of three.

Theorem 4.2. *Performing a distributed search for a treasure at an unknown distance $D \in \mathbb{N}$ by k TM ants in $O(D + D^2/k)$ rounds under the asynchronous model, requires $\Omega(k)$ pheromones.*

Proof. Let us assume by contradiction that T is a TM (ant) which is able to find a treasure at any distance D in $O(D+D^2/k)$ asynchronous rounds for any k ants, while only using $o(k)$ pheromones.

Let there be some k, D and scheduling S, such that it takes the k ants $\Theta(D+D^2/k)$ rounds and less than k pheromones to find the treasure. This is a tight lower bound on the runtime, therefore there must exist such a scenario. there is at least one ant which has not emitted any pheromones throughout the run; let us denote one such ant as p.

Consider a different group of $k' \gg k$ ants, where some k ants are scheduled exactly as in schedule S and the other $k' - k$ ants are scheduled to take a single step each, one after the other, right after each step of ant p. Ant p emits no pheromones, so it shares its view of the world with the $k' - k$ ants that follow it, and all those ants operate exactly the same (due to T being deterministic). The number of rounds remains $\Theta(D+D^2/k) > O(D+D^2/k')$, in contradiction to our assumptions. □

4.2 Synchronous Turing Machines

The main idea here is a static partition of the search space. Pheromones are used for two purposes: (i) give every ant a unique id, and (ii) provide a way of estimating the total number of participating ants. Each ant then keeps a current estimate of the total number of ants and updates that estimation after each successful exploration of a new layer. Note that a layer is defined slightly differently in this section; *layer* l contains the set of grid cells (x, y) where $\max(|x|, |y|) = l$. Naturally, an ant with id $1 \leq i \leq k$ and estimated number of ants k_i covers layers l_i such that $l_i \equiv i \pmod{k_i}$. The algorithm is illustrated in Image 3 and a more formal description is given in Algorithm 3.

Note that if an ant learns that it has to update its current estimate of the total number of ants, its search space has to be restarted. Otherwise, there would exist an emission scheme which would cause the exploring ants to miss a few layers.

Theorem 4.3. *Algorithm 3 completes in $O(t_e(k)+D+D^2/k)$ rounds, after emitting $O(k)$ pheromones. Hence, it is optimal given a matching emission scheme.*

Proof. Every ant obtains a unique id since the emission scheme provides the guarantee that no two ants leave the nest simultaneously. Following the acquisition of an id, each ant starts exploring the

layers which correspond to its id. Upon completing a layer, the estimate for the total number of ants is rechecked, and updated if needed; every time an update occures, the ants restart their search from the first layer after their id. Thus, each and every layer will be covered by the searching ants. As for the run time of the algorithm: it takes $t_e(k) + k$ rounds until the last pheromone is emitted, after that it takes at most $D + 8D$ rounds for all ants to notice and update their estimates, and then it takes another $\Theta(D^2/k)$ rounds for the ants to finish exploring all required layers. Therefore, the algorithm executes in $O(t_e(k) + D + D^2/k)$ rounds. □

Algorithm 3 TM distributed treasure search

```
1:  id ← 1
2:  Move(North)
3:  while SensePheromone(Current) do          ▷ Search for the first free spot
4:      id ← id + 1
5:      Move(North)
6:  end while
7:  EmitPheromone(Current)                    ▷ Mark taken id
8:  total ← id
9:  layer ← id
10: while true do
11:     ExploreLayer(layer)
12:     if UpdateTotal(total) then            ▷ Start over if a new ant joined
13:         layer ← id
14:     end if
15:     layer ← layer + total
16: end while
```

4.3 Asynchronous Turing Machines

Algorithm 3, depicted in Figure 3, works exactly the same in the asynchronous model as it does in the synchronous case. Therefore, Theorem 4.3 still holds – without the cost of $t_e(k)$.

Proof. As established before, the run time of the worst case scenario is achieved when all ants operate at the same speed. Let us look at the last ant to place a pheromone; this ant has placed the pheromone after exactly k asynchronous rounds. After at most $2D + 8D$ rounds every other ant will have already noticed the new pheromone, and from that point onwards the asynchronous analysis follows the synchronous one to the letter. □

4.4 Summary

Choosing a stronger computational model of Turing Machines over Finite State Machines, we were able to utilize less pheromones to solve the problem of ANTS. The previous (computational) lower bound of $\Omega(D)$ pheromones was lifted and replaced by a new (cooperative) lower bound of $\Omega(k)$ for the new computational model of TM. We have also provided algorithms which allow k ants to find the treasure, still in optimal time, by using only $O(k)$ pheromones (under both synchronous and asynchronous scheduling models), proving that the the new lower bound is tight.

Table 1 illustrates the current state of affairs. At this point it is worth noting that the performance figures (both the number of pheromones and asymptotic run time) are optimal; the only

possible way of improving on those is through providing the ants with extra information, such as a unique id for every ant and/or the total number of ants participating in the search.

5 Conclusions and future work

We have presented different approaches to solving the problem of ANTS with pheromones. We have analyzed the problem under different scheduling models and various computational limits. The main focus has been on utilizing as little pheromones as possible, since this is a biological resource which might only be scarcely available in a real scenario. The upper bounds results are summarized in Table 1.

We have presented a lower bound on the number of pheromones that must be emitted in order to solve ANTS with Finite State Machines, and provided algorithms which match this lower bound while still achieving optimal run time under both the synchronous and asynchronous scheduling models.

In order to further reduce the lower bound on the number of pheromones we have moved to a stronger computational model: that of Turing Machines. We have shown a refined lower bound on the number of pheromones that must be utilized in this new computational model as well, this time deriving the limitation not from the basic requirement of solving the problem, but rather from the need of achieving optimal run time and having all k ants cooperatively perform the search. Once again, we have provided algorithms which match the aforementioned lower bounds on the number of pheromones and the total run time, both for the synchronous and asynchronous scheduling models.

Table 1: Summary of various solutions for the ANTS problem

	Synchronous Model		Asynchronous Model	
FSM	Runtime:	$O(t_e(k) + D + D^2/k)$	Runtime:	$O(D + D^2/k)$
	Pheromones:	$O(D)$	Pheromones:	$O(D)$
TM	Runtime:	$O(t_e(k) + D + D^2/k)$	Runtime:	$O(D + D^2/k)$
	Pheromones:	$O(k)$	Pheromones:	$O(k)$

Open Questions Throughout our discussions of the synchronous model, we have made an assumption on the existence of an emission scheme with a single premise: no two ants leave the nest at the same time. While we feel this is a natural assumption to make in the real world, an open question that remains is devising an optimal randomized algorithm for such an emission scheme. We do denote the impossibility of achieving this goal with any deterministic synchronous computational model, such as those considered in this paper.

A major part of real ants' foraging process is what happens once the food (treasure) is actually found; commonly, the discovering ant has to find its way back to the nest and inform other ants of the discovery in order to get assistance in carrying the food back, for instance. We consider this closely related, follow-up problem a good candidate for future research.

Other topics that were not covered in the current paper but might be important to consider in the future are, for instance: coping with obstacles, ants failing (i.e., due to getting eaten), same cooperative search but from different initial nest locations, or what happens if the ants are capable of utilizing different flavors of pheromones (perhaps even with limited lifespan) or can employ pheromones together with randomization in their decision making process.

References

[1] Y. Afek, N. Alon, O. Barad, E. Hornstein, N. Barkai, and Z. Bar-joseph. A biological solution to a fundamental distributed computing problem. In *Science*, volume 331, pages 183–185, 2011.

[2] Y. Emek, T. Langner, D. Stolz, J. Uitto, and R. Wattenhofer. How many ants does it take to find the food? In *SIROCCO*, pages 263–278, 2014.

[3] Y. Emek, T. Langner, J. Uitto, and R. Wattenhofer. Ants: Mobile finite state machines. *CoRR*, abs/1311.3062, 2013.

[4] Y. Emek, T. Langner, J. Uitto, and R. Wattenhofer. Solving the ants problem with asynchronous finite state machines. In *ICALP (2)*, pages 471–482, 2014.

[5] O. Feinerman and A. Korman. Memory lower bounds for randomized collaborative search and implications for biology. In *DISC*, pages 61–75, 2012.

[6] O. Feinerman, A. Korman, Z. Lotker, and J.-S. Sereni. Collaborative search on the plane without communication. In *PODC*, pages 77–86, 2012.

[7] C. Lenzen, N. A. Lynch, C. C. Newport, and T. Radeva. Trade-offs between selection complexity and performance when searching the plane without communication. In *PODC*, pages 252–261, 2014.

[8] C. Lenzen and T. Radeva. The power of pheromones in ant foraging. *BDA*, 2013.