A Pheromone-Based Utility Model for Collaborative Foraging

Liviu Panait
Department of Computer Science
George Mason University
lpanait@cs.gmu.edu

Sean Luke
Department of Computer Science
George Mason University
sean@cs.gmu.edu

Abstract

Multi-agent research often borrows from biology, where remarkable examples of collective intelligence may be found. One interesting example is ant colonies' use of pheromones as a joint communication mechanism. In this paper we propose two pheromone-based algorithms for artificial agent foraging, trail-creation, and other tasks. Whereas practically all previous work in this area has focused on biologically-plausible but ad-hoc single pheromone models, we have developed a formalism which uses multiple pheromones to guide cooperative tasks. This model bears some similarity to reinforcement learning. However, our model takes advantage of symmetries common to foraging environments which enables it to achieve much faster reward propagation than reinforcement learning does. Using this approach we demonstrate cooperative behaviors well beyond the previous ant-foraging work, including the ability to create optimal foraging paths in the presence of obstacles, to cope with dynamic environments, and to follow tours with multiple waypoints. We believe that this model may be used for more complex problems still.

1. Introduction

Social insects provide a useful example from which multi-agent research draws inspiration, because they yield sophisticated collective action arising from very large numbers of agents following relatively simpler individual behaviors. Social insect-inspired "swarm behavior" methods have found their way into clustering and foraging algorithms in multi-agent and robotics research (for example, [1, 8]). Ideas inspired from swarm behavior have also emerged in *ant-colony optimization*, a population-oriented stochastic search method which has been very successful in difficult optimization problems such as network routing [4].

For many such collective behaviors a few key communication procedures form the substrate on which the swarm relies to achieve its task. A notable example is the use of *pheromones*, chemical substances deposited by ants and similar social insects in order to mark the environment with information to assist other ants at a later time. For example, while foraging for food, an ant deposits pheromones to mark trails connecting the nest to food sources [9]. This information is later used by the ant to find its way back to the food sources, and also to recruit other ants to the foraging task. Ants use pheromones to impressive degree in foraging and other tasks. For example, leafcutter ants are able to organize trails connecting their nest with food sources located as far as hundreds of meters away [9].

We are interested in the capabilities of pheromones as a guide for artificial agents and robots. In some sense pheromones may be viewed as a mechanism for inter-agent communication that can help reduce the complexity of individual agents. But pheromones cannot be viewed as essentially a blackboard architecture. While pheromones are *global* in that they may be deposited by and read by any agent, and generally last long periods of time, they are *local* in that agents may only read or change the pheromone concentrations local to themselves in the environment.

Our model treats pheromones as utility estimates for environment states, and our agents' pheromone-depositing and decision-making functions somewhat resemble, but are not the same as, utility update and state transition functions in reinforcement learning. The model allows the use of multiple pheromones, and an agent's choice of pheromones to update or to use in decision-making is based on the agent's current internal state. In this way the agents may be seen as mobile automata responding to and updating external state in the environment.

This model is in marked contrast to nearly all the existing pheromone-based foraging literature. Previous work has eschewed any formalism, concentrating instead on approaches inspired by biology if not actually biologically plausible. Importantly, because agents generally use only one pheromone, these models assume a single pheromone in order to mark locations to food, and so must rely on an ad-hoc mechanism for getting back home (for example, compass information).

We begin with a description of previous work in simulated ant foraging and then present the new model. We then show with application of the model to foraging in environments with obstacles, with a moving food source, or that contain multiple waypoints. The paper concludes with a brief discussion of the results, and lists several directions for future work.

2. Previous Work

Artificial swarm foraging often utilizes various forms of “indirect communication”, involving the implicit transfer of information from agent to agent through modification of the world environment. Examples of indirect communication include: leaving footsteps in snow, leaving a trail of bread crumbs in order to find one’s way back home, and providing hints through the placement of objects in the environment (perhaps including the agent’s body itself).

Much of the indirect communication literature has drawn inspiration from social insects’ use of pheromones to mark trails or to recruit other agents for tasks [9]. Pheromones are chemical compounds whose presence and concentration can be sensed by fellow insects, and like many other media for indirect communication, pheromones can last a long time in the environment, though they may diffuse or evaporate. Early models of ants (for example [7]) have demonstrated that pheromones make it possible for ants to optimize trails from the nest to a food source. It has been suggested that this happens because larger amounts of pheromones can more quickly accumulate on the shorter paths, rather than on the longer ones [2, 3].

Several pheromone-based learning algorithms have been proposed for foraging problem domains. A series of reinforcement learning algorithms have adopted a fixed pheromone depositing procedure, and use current pheromone amounts as additional sensor information while exploring the space or while updating the state-action utility estimates [10, 12, 13]. Evolutionary computation techniques have also been applied to learn exploration/exploitation strategies using pheromones deposited by hardcoded mechanisms. For example, in [16, 17] evolutionary computation is used to tune an agent policy in an application involving multiple digital pheromones. A similar idea applied to network routing is presented in [19]. There have been relatively fewer attempts to actually learn the pheromone-depositing procedure itself: the most notable example is AntFarm, a system that combined communication via pheromones and evolutionary computation [5, 6]. AntFarm ants use a single pheromone to mark trails toward food sources, but use a compass to point themselves along the shortest path back to the nest.

Some work has demonstrated the ability of ants to dynamically react to changes in the environment. Resnick [15] demonstrates pheromone-based algorithms which enable ants to “forget” about depleted food sources and establish paths to new ones. Further investigations of this model are presented in [14].

The previous work has largely been ad-hoc: it assumes a single ant pheromone to help set up a gradient to the food source, plus some arbitrary *a priori* mechanism to return to the nest (a built-in compass, a gradient produced by the nest itself, etc.). This assumption is either stated explicitly, or it is adopted implicitly by using specific environment models and transition functions. Usually this is justified in terms of biology (ants’ use of navigational techniques, such as orientation based on position of the sun).

We are aware of two papers which do not rely on ad-hoc methods to return to the nest. The first such work [20] is similar to our own, using two pheromones to establish gradients to the nest and to a food source in the presence of an obstacle. The ants use simple pheromone addition to create trails and so the authors rely on high rates of pheromone evaporation to maintain the gradients. Another paper where foraging agents do not have ad-hoc methods for nest recovery is presented in [18]. Here, Vaughan *et al* propose agent behaviors that use *directed pheromones* (pheromones that indicate a direction), and show successful foraging in simulation and on real robots. Aside from using directed pheromones, their work also assumes that the agents can deposit pheromones at any location in the environment, even if they are located far away from it (similar to a random-access memory).

3. A Reexamination

We begin with a description of the so-called “central nest foraging” problem: the environment is a nontoroidal grid world and consists of a nest location and some N food source locations (for simplicity, our examples will assume $N = 1$). To be consistent with past literature, we will refer to our swarm agents as “ants”. Ants leave the nest in search of the food sources. From any given square, an ant may move to any eight-neighbor square which is not occupied by too many ants (we set the maximum to 10) or does not contain an obstacle. When it happens upon a food source, the ant becomes laden with food. When reaching the nest again, the ant leaves the food at the nest and begins searching again. The goal is to maximize the rate of food brought to the nest from the food sources.

The foraging task may be abstractly viewed as a sequence of two alternating tasks for each ant: start from the nest and reach the food source, and then start from the food source laden with food and reach the nest. The particular task the ant must perform depends on its internal state: either it is laden with food or it is not. We view the performance of each of these tasks as an *episode* of state transitions for the ant: where the nest is the start state and the food source is the goal state, or vice versa. The ant receives a reward at goal states; at other states it does not.

This may now be cast into a framework which strongly resembles reinforcement learning, where pheromone values are the *utility values* of various states (ant locations). The ant's *external state* s is the ant's current location. An ant receives a reward $R(s)$ for transitioning to the desired goal state (the food location or nest, depending on internal state). The utility of a state $U_p(s)$ is the concentration of a given pheromone type p at location s; when foraging, each state has two utilities, each encoded in one type of pheromone. The ant's policy $\pi(S \rightarrow A)$, which maps states to actions, may be defined to be "move towards the neighboring state s with the highest value $U_p(s)$." The foraging algorithm presented here deviates somewhat from traditional dynamic programming and reinforcement learning methods in the assumptions that it makes about the model. Here the model is presumed to be *symmetric*, meaning that if the ant can travel from state s_1 to neighboring state s_2 with some probability, then the ant can also travel back from s_2 directly to s_1 with the same probability. Symmetry is not always a valid assumption depending on the environment: but we believe it to be common in many robotics and multi-agent environments. The model is also usually, but not required to be, *deterministic*: the action an ant chooses in state s_i will always result in transitioning to a specific state s_j.

The particular choice of pheromones to update and to base transition decisions on is dependent on the ant's internal state (in this example: if it's laden with food or not). Thus an ant's policy may be viewed as an automaton which, depending on the external and internal states, plus the external state utilities, reward function, and transition model, will then transition to new internal and external states and will update utilities of the external states.

As we will discuss later, although this model bears a strong similarity with reinforcement learning models, the assumption of symmetry makes possible update rules which are much more efficient than reinforcement-learning style "backup" rules like TD(λ) or prioritized sweeping. One such efficient update rule is essentially a modified form of value iteration though its application is somewhat unusual.

Imagine, as is shown in Figure 1 that the ant has just transitioned from location s to location s', and at s' the ant now has available actions $a \in A$. The model provides the probabilities $T(s', a, s'')$ that performing a in s' will transition to s''. Let γ be a learning rate constant between 0 and 1. The update rule would then be:

$$U_p(s') = R(s') + \gamma \max_{a \in A} \sum_{s''} T(s', a, s'') U_p(s'')$$

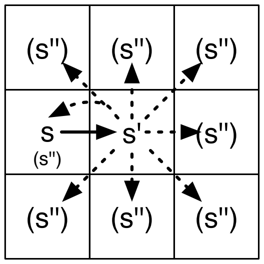

Figure 1. State transitions for a simple two-dimensional discrete grid environment. The ant was just in state *s*, and has just transitioned to state *s'*. From there the ant may transition to any state *s''*∈*S''*. Note that *s* is also in *S''*.

If the model is deterministic, which we assume here but is not required, then all values of $T(s', a, s'')$ are 1 or 0 and we may instead think of a set S'' as consisting of available (neighboring) states to which the ant may transition from s':

$$U_p(s') = R(s') + \gamma \max_{s'' \in S''} U_p(s'') \tag{1}$$

The value of γ can be constant, but we chose to make it smaller for diagonal transitions than for other ones to reflect the slightly longer distance when moving diagonally.

The ant then must choose an action to perform. With some probability[1], ants choose a completely random action: this ensures that all states in the environment are sooner or later visited. In the rest of the cases, the ant transitions to the state with the highest concentration of a given pheromone:

$$s' = \operatorname*{argmax}_{s'' \in S''} U_p(s'') \tag{2}$$

However, the ant will usually also update *different* pheromones than the one being used for transitioning. Thus while the ant is following one gradient, at the same time it may be building up different gradients.

An example here may be useful in showing how these rules produce a trail between a nest and a food source of unknown location. In this example, we will use two pheromones: p_{food} increases with proximity to the food source, and p_{nest} increases with proximity to the nest. An ant may be in one of two internal states: either it is looking for food or it is returning to the nest laden with food. When the ant reaches its goal, it receives a positive reward; after updating the pheromone at the goal state, the ant then

1 We used exploration rate probabilities of 10% in most stationary environments, but as high as 30% or even 40% in the dynamic environments.

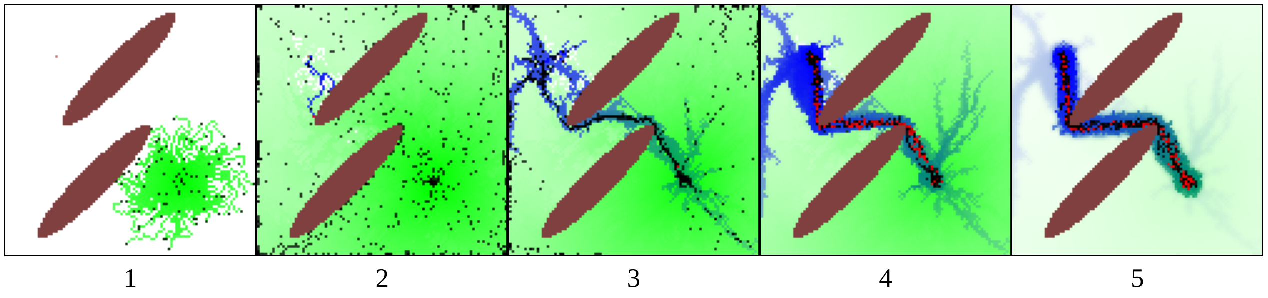

Figure 2. Foraging sequence with two obstacles. Nest is in bottom right quadrant, and food source is in top left quadrant. (1) Ants leave the nest while depositing the to-nest pheromone. (2) Ants discover the food source, and begin return to nest along the to-nest pheromone while depositing the to-food pheromone. (3) Trail is established (4) All ants are now engaged. (5) Ants perform trail optimization.

switches to the alternative internal state. As the ants start from the nest, they also receive an initial maximal reward to update p_{nest}.

The ants set up gradients to the nest and to the food in the following way. If it is searching, then an ant updates (deposits) p_{nest} using Equation 1; but transitions using a gradient towards food, that is, using p_{food} with Equation 2. The ant updates the nest pheromone because it is essentially building a gradient towards from the nest while searching for the food.

When the ant is laden with food, its usage of pheromones flip-flops. Now, Equation 1 uses p_{food}, and Equation 2 uses p_{nest}. This causes the ant to follow the gradient back to the nest, while simultaneously depositing a gradient towards the food source for other ants to follow. Figure 2 shows the process of discovery, trail development, and optimization in this algorithm.

There are other utility-update possibilities. Equation 3 below, which bears a similarity to those used in TD-learning, is also fairly effective:

$$U_p(s') = \max(U_p(s'), (1-\alpha)U_p(s') + \alpha(R(s') + \gamma U_p(s)) \quad (3)$$

γ and α are learning rate constants between 0 and 1. While this equation is not as efficient as Equation 1, it has the benefit of not requiring the ant to know any state utility values except for s – which it has just visited – and s' – where it is currently located. This equation appears backwards when compared to traditional TD-learning equations, which update $U(s)$ based on $U(s')$ rather than the other way around. It is the symmetry of the model that enables us to use such "backward" looking algorithms: the ant can now draw a gradient as it moves *away* from the reinforcement rather than having to build one up through repeated moves *towards* the expected reinforcement. This results in a dramatic improvement in efficiency, as it avoids the need for repeated backups.

In this sense we think of Equation 3 as a "forward updating" algorithm, rather than the "backup updating" of reinforcement learning. Interestingly, Equation 1 automatically performs *both* forms of updating, which is why we believe it to be more efficient than Equation 3.

Domain-specific Improvements In some cases (including foraging) ants may reasonably build up *both* gradients at the same time, one using the algorithms discussed earlier, and the other using "backup updating" as in reinforcement learning. For example, while searching for food the ant can build up a gradient to the nest using the algorithms discussed earlier; and build up a gradient to the nest using backup.

As it does both forward updating *and* backup updating, Equation 1 may be used to update both pheromones in question. But when it comes to using Equation 3, things are more complicated. Suppose the ant has just left the nest, and it is currently searching for the food source. When transitioning from state s to s', we can use Equation 3 to update the amounts of nest pheromones at state s' based on the amount of nest pheromone at state s. As for the food pheromones, we can use the a similar formula obtained by swapping s and s':

$$U_p(s) = \max(U_p(s), (1-\alpha)U_p(s) + \alpha(R(s) + \gamma U_p(s')) \quad (4)$$

This equation is essentially a slight modification of TD-learning. The need for this separate equation is due to the fact that while Equation 1 uses the utilities of all nearby states to update the utility of a single state, Equations 3 and 4 only use a single sampled state for this update. In our previous example, while searching for the food source, the ant transitioned from state s to s'. Hence it is likely that on av-

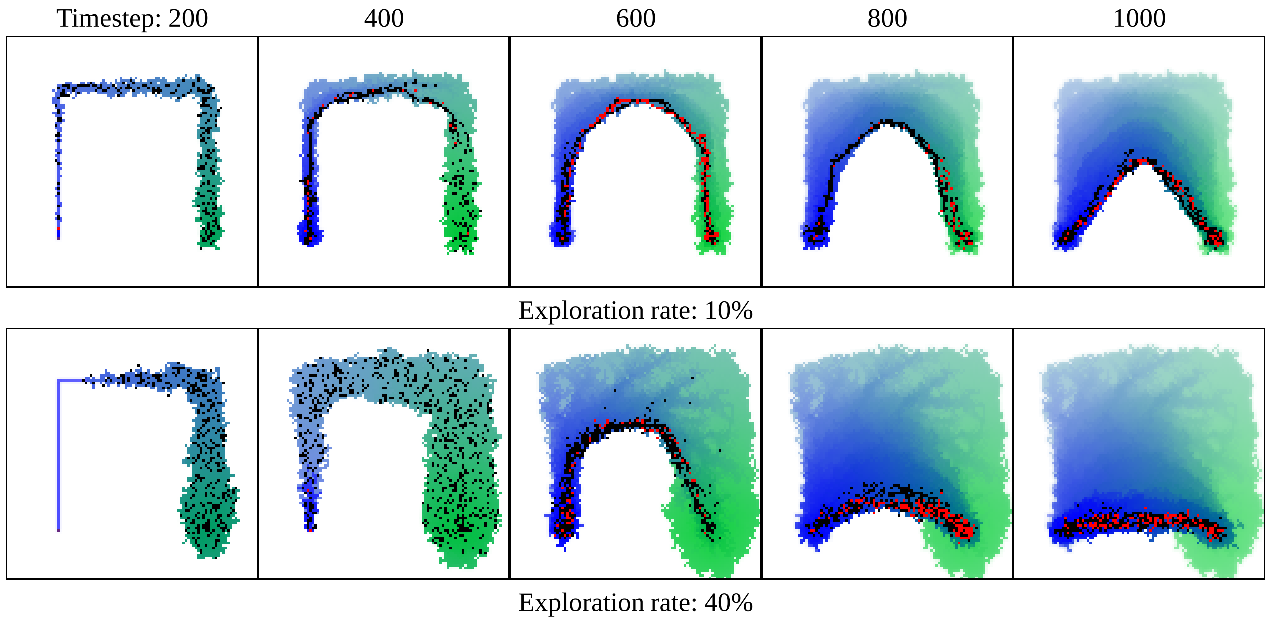

Figure 3. Foraging sequence with predefined suboptimal path, showing local path optimization. With higher exploration rate, the foraging trail is thicker, because many ants slightly deviate from it. In the end, more exploration helps straighten the path quicker to an optimal one.

erage state s is closer to the nest, while s' is closer to the food source. Therefore Equation 3 should propagate utility from s' to s, while Equation 4 should propagate utility from s to s'.

4. Experiments

Using these update and transition rules we have produced self-optimizing trail behaviors in domains which have (1) Obstacles, (2) Dynamically changing food or nest locations, and (3) Multiple waypoints. Experiments were performed on the MASON multi-agent simulator [11].

Obstacles and Trail Optimization Figure 2 presents an example of foraging trails using Equation 1. The nest is located in the bottom-right area of the environment, and two obstacles separate it from the food source. As the ants explore, they find the food source and start marking trails connecting it to the nest. Later, all ants are attracted to this marked foraging trail, and they are all repeatedly move between the nest and the food source. Ants will often discover alternative routes, such as all the way over the top obstacle, but will eventually discard these routes in favor of the optimal path.

The "fingers" appearing to emanate from the nest and the food source are harmless artifacts of the square environment and our inclusion of "backup updating" as a domain-specific improvement. If an ant continues to explore, he may never find any pheromone and may continue to explore outward. In our square environment such ants usually find themselves stuck near corners. To compensate for this, after some period we caused the ants to become "bored" and follow their own gradient back to their start location to try again. As the ants return, and discover a working pheromone trail, they would perform sample backups near the nest along the trail, building up the "fingers".

For the experiments presented in Figure 3, we used pheromones to mark a clearly suboptimal foraging trail. The ants use it to initially locate the food source, but later straighten it until optimal. This shows that the foraging behaviors proposed in this paper lead to an additional emergent collective hill-climbing process to improve the foraging trail. Note that the higher the exploration rate, the more rapidly the ants optimize the trail.

Adaptation to Dynamic Changes Dynamism requires that the ants are capable of forgetting old information. This can be achieved through gradual pheromone evaporation and diffusion in combination with the algorithms described earlier. Figure 4 shows the application of the two foraging algorithms (one using Equation 1, and the other using Equations 3 and 4). The nest is located in the lower area of the map, while the food source moves from right to left in the upper area. As expected, the algorithm using Equation 1 is able to propagate the reinforcement information much better, and its foraging trail is closer to optimality. The algo-

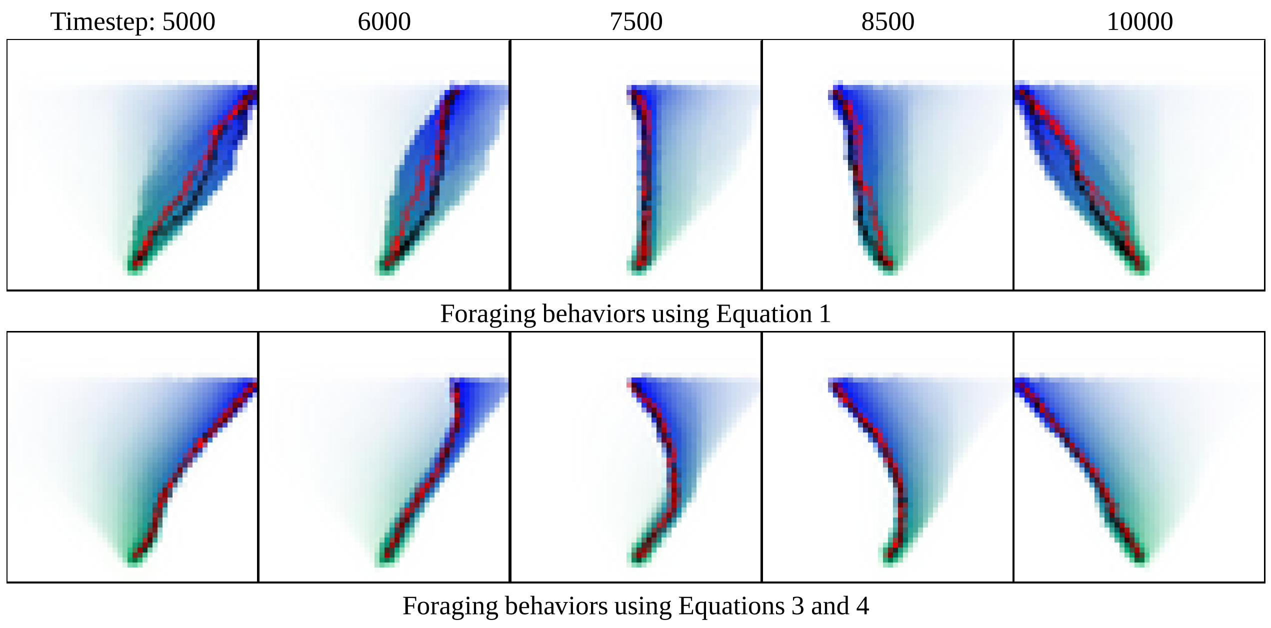

Figure 4. Foraging from a moving food source: the nest is located in the lower area, while the food source moves from right to left in the upper area of the environment.

rithm using Equations 3 and 4 is slower in propagating the information, and its foraging trail is suboptimally curved.

In Figure 5 we also added an obstacle to the dynamic environment. The nest is in the center and the food source moves around the nest clockwise in a wide circle. Interestingly, the ants stretch well around the obstacle before finally adopting the straighter path: the problem stems from the fact that the evaporation and diffusion of pheromones have affected the encoded utilities to such a degree that ants believe the straight trail to the nest is longer than the bent one they currently follow. Once the difference in approximated utility for the two paths becomes similar, ants start to explore the alternative, and immediately discover that its utility is higher than expected. This leads to all ants rapidly adopting the new foraging trail.

Multiple Waypoints The inclusion of internal states in our procedure allows for more sophisticated behaviors than just there-and-back-again foraging trails. For example, with additional states and pheromones, ants are able to achieve tours between multiple waypoints, including ones with intersecting paths. Figure 6 shows three-waypoint, four-waypoint, and five-waypoint tours with intersections. This more complex task is achieved through a series of automata rules which can be described as follows. Each ant has as internal state a value N indicating the most recent waypoint it has visited, and a boolean flag F telling it whether or not to return to waypoint N upon discovery of the next waypoint M in the sequence. N starts at 0 for all ants. When an ant is at waypoint N, it determines if there are any pheromones for waypoint M in the immediate vicinity. If so, it sets F to false and follows those pheromones. If not, it begins searching for waypoint M, setting F to true, and when it has discovered waypoint M, it follows the N pheromone back again, while painting the M pheromone. An ant that has just visited N and is currently searching for M updates both (and only) the N and M pheromones. In essence, when ants discover a trail to the next waypoint, they paint a trail back to the previous waypoint for the others to follow, if one has not already been painted.

5. Conclusions and Future Work

Pheromones are an attractive model for large-scale collective communication and coordination, and one which we think deserves to be studied more in a multiagent engineering context. This paper introduced a novel foraging strategy where agents (ants) use pheromones to actively mark how to reach the nest and the food source. In a manner similar to reinforcement learning, we view ant locations as external states and ant pheromone concentrations as the utilities of those states. Ant behaviors are essentially automata with internal state which perform actions and deposit pheromones based on reward, internal state, and pheromone values in the immediate neighborhood. With the addition of pheromone evaporation and diffusion we are able to achieve trail optimization in the presence of obstacles, dynamic adaptation

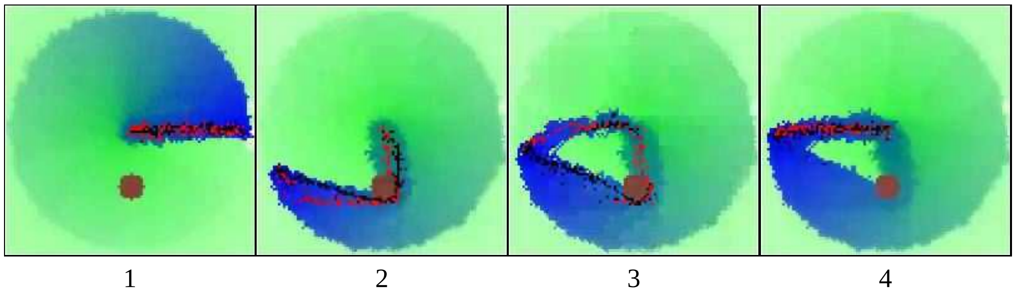

Figure 5. The "Ant Clock": trail optimization in presence of a dynamically changing food source and an obstacle. Nest is in the center. (1) Food source is at 3 o'clock position, moving clockwise. (2) Food source has moved to 7 o'clock position, and ants are bending around obstacle to reach it. (3) Ants achieve better trail. (4) Food source at 9 o'clock position.

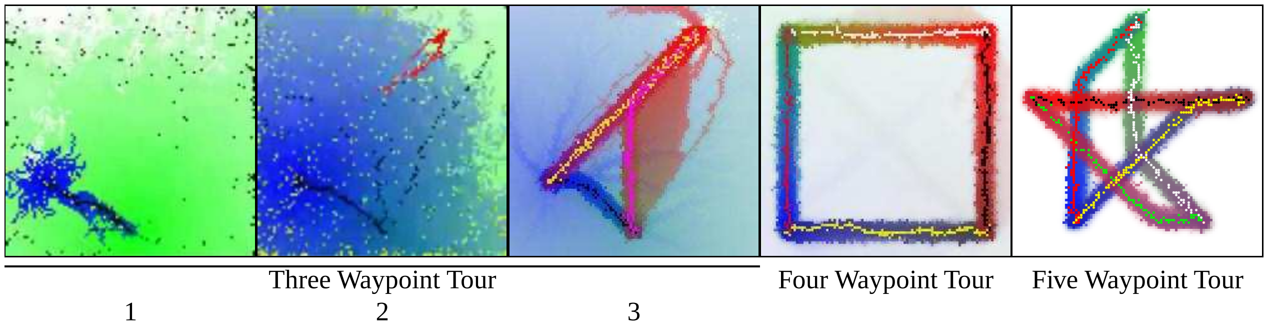

Figure 6. Tours with waypoints. Each path between two waypoints requires one unique pheromone and two unique internal states. For the three-waypoint tour: (1) First segment of tour is established. (2) Second waypoint (top right) is discovered and next segment is begun. (3) Trails are optimized.

to changing environments, and complex N-waypoint tours rather than simple there-and-back-again foraging paths. We believe both the formalism and the results to be a significant improvement over the existing literature.

Furthermore, we strongly believe that this "pheromone automata" model may produce yet more surprises: can artificial ants collectively solve maze or other search problems? Ferry objects around the environment? What is possible with the addition of local signaling or other communication modes? We also plan to extend this model to applicability to real robots as future work, in the form of movable beacons.

Acknowledgments

The authors would like to thank Claudio Cioffi-Revilla and the George Mason University Center for Social Complexity for supporting this work in part.

References

[1] R. Beckers, O. E. Holland, and J.-L. Deneubourg. From local actions to global tasks: Stigmergy and collective robotics. In *Artificial Life IV: Proceedings of the International Workshop on the Synthesis and Simulation of Living Systems , third edition*. MIT Press, 1994.

[2] E. Bonabeau. Marginally Stable Swarms are Flexible and Efficient. *Phys. I France*, pages 309–320, 1996.

[3] E. Bonabeau and F. Cogne. Oscillation-Enhanced Adaptability in the Vicinity of a Biffurcation: The Example of Foraging in Ants. In P. Maes, M. Mataric, J. Meyer, J. Pollack, and S. Wilson, editors, *Proceedings of the Fourth International Conference on Simulation of Adaptive Behavior: From Animals to Animats 4*, pages 537–544. MIT Press, 1996.

[4] E. Bonabeau, M. Dorigo, and G. Theraulaz. *Swarm Intelligence: From Natural to Artificial Systems.* Santa Fe Institute Studies in the Sciences of Complexity. Oxford University Press, 1999.

[5] R. J. Collins and D. R. Jefferson. An artificial neural network representation for artificial organisms. In H.-P. Schwefel and R. Männer, editors, *Parallel problem solving from nature: 1st Workshop, PPSN I*, pages 259–263, Berlin, 1991. Springer.

[6] R. J. Collins and D. R. Jefferson. Antfarm : Towards simulated evolution. In C. G. Langton, C. Taylor, J. D. Farmer, and S. Rasmussen, editors, *Artificial Life II*, pages 579–601. Addison-Wesley, Redwood City, CA, 1992.

[7] J. L. Deneubourg, S. Aron, S. Goss, and J. M. Pasteels. The Self-Organizing Exploratory Pattern of the Argentine Ant. *Insect Behavior*, 3:159–168, 1990.

[8] J. L. Deneubourg, S. Goss, N. Franks, A. Sendova-Franks, C. Detrain, and L. Chretien. The dynamics of collective sorting: Robot-like ants and ant-like robots. In *From Animals to Animats: Proceedings of the First International Conference on Simulation of Adaptive Behavior*, pages 356–363. MIT Press, 1991.

[9] B. Hölldobler and E. O. Wilson. *The Ants*. Harvard University Press, 1990.

[10] L. R. Leerink, S. R. Schultz, and M. A. Jabri. A reinforcement learning exploration strategy based on ant foraging mechanisms. In *Proceedings of the Sixth Australian Conference on Neural Networks*, Sydney, Australia, 1995.

[11] S. Luke, G. C. Balan, L. A. Panait, C. Cioffi-Revilla, and S. Paus. MASON: a Java multi-agent simulation library. In *Proceedings of Agent 2003 Conference on Challenges in Social Simulation*, 2003.

[12] N. Monekosso, P. Remagnino, and A. Szarowicz. An improved q-learning algorithm using synthetic pheromones. In E. N. B. Dunin-Keplicz, editor, *From Theory to Practice in Multi-Agent Systems, Second International Workshop of Central and Eastern Europe on Multi-Agent Systems, CEEMAS 2001 Cracow, Poland, September 26-29, 2001. Revised Papers*, Lecture Notes in Artificial Intelligence LNAI-2296. Springer-Verlag, 2002.

[13] N. D. Monekosso and P. Remagnino. Phe-Q: a pheromone based q-learning. In *Australian Joint Conference on Artificial Intelligence*, pages 345–355, 2001.

[14] M. Nakamura and K. Kurumatani. Formation mechanism of pheromone pattern and control of foraging behavior in an ant colony model. In C. G. Langton and K. Shimohara, editors, *Artificial Life V: Proceedings of the Fifth International Workshop on the Synthesis and Simulation of Living Systems*, pages 67–76. MIT Press, 1997.

[15] M. Resnick. *Turtles, Termites and Traffic Jams*. MIT Press, 1994.

[16] J. Sauter, R. S. Matthews, H. V. D. Parunak, and S. Brueckner. Evolving adaptive pheromone path planning mechanisms. In *Proceedings of First International Joint Conference on Autonomous Agents and Multi-Agent Systems (AAMAS-02)*, pages 434–440, 2002.

[17] J. Sauter, H. Van Dyke Parunak, S. Brueckner, and R. Matthews. Tuning synthetic pheromones with evolutionary computing. In R. E. Smith, C. Bonacina, C. Hoile, and P. Marrow, editors, *Evolutionary Computation and Multi-Agent Systems (ECOMAS)*, pages 321–324, San Francisco, California, USA, 7 2001.

[18] R. T. Vaughan, K. Støy, G. S. Sukhatme, and M. J. Mataric. Whistling in the dark: Cooperative trail following in uncertain localization space. In C. Sierra, M. Gini, and J. S. Rosenschein, editors, *Proceedings of the Fourth International Conference on Autonomous Agents*, pages 187–194. ACM Press, 2000.

[19] T. White, B. Pagurek, and F. Oppacher. ASGA: improving the ant system by integration with genetic algorithms. In J. R. Koza, W. Banzhaf, K. Chellapilla, K. Deb, M. Dorigo, D. B. Fogel, M. H. Garzon, D. E. Goldberg, H. Iba, and R. Riolo, editors, *Genetic Programming 1998: Proceedings of the Third Annual Conference*, pages 610–617, University of Wisconsin, Madison, Wisconsin, USA, 22-25 1998. Morgan Kaufmann.

[20] M. Wodrich and G. Bilchev. Cooperative distributed search: The ants' way. *Control and Cybernetics*, 26, 1997.