Complex Adaptive Systems: Energy, Information and Intelligence: Papers from the 2011 AAAI Fall Symposium (FS-11-03)

Ant Colony Optimization in a Changing Environment

John Seymour, Joseph Tuzo, Marie desJardins
University of Maryland Baltimore County
Department of Computer Science and Electrical Engineering
1000 Hilltop Circle, Baltimore MD, 21250
{seymour1, tuzoj1, mariedj}@umbc.edu

Abstract

Ant colony optimization (ACO) algorithms are computational problem-solving methods that are inspired by the complex behaviors of ant colonies; specifically, the ways in which ants interact with each other and their environment to optimize the overall performance of the ant colony. Our eventual goal is to develop and experiment with ACO methods that can more effectively adapt to dynamically changing environments and problems. We describe biological ant systems and the dynamics of their environments and behaviors. We then introduce a family of dynamic ACO algorithms that can handle dynamic modifications of their inputs. We report empirical results, showing that dynamic ACO algorithms can effectively adapt to time-varying environments.

Introduction

Ant colonies are a common example of parallelism and emergent behavior. Each ant can remember only a small amount of information, and can only utilize a small number of simple rules: they can transmit and receive pheromone inputs from the environment; they can move in the direction that contains the most pheromones; and they can retrace their steps back to the ant colony. Despite this individual simplicity, the colony as a whole exhibits complex behavior.

Ants coordinate themselves by modifying their environment through pheromones, leading to parallelism and self-organization. While looking for food, ants detect pheromones to find the direction of a previously found food source. When ants find a food source, they take some food and retrace their paths back to the nest, emitting pheromones along the way. In this way, ants self-organize to find near-optimal solutions for finding food, choosing roles, and allocating resources. The ants exhibit adaptation by changing the path, if the path becomes an inefficient way for the ants to reach the food source. Ants exhibit adaptation when a food source is depleted: as the result of changes to the environment, they are able to respond by searching for a new food source.

Tipping points can arise in an ant system from both the pheromones and the number of ants taking a path. If the pheromones dissipate too quickly, then the ants are unable to establish a stable trail, and if they have to leave the path because of some obstruction, the path may have disappeared by the time they circumnavigate the obstruction. On the other hand, if the pheromones dissipate too slowly, then multiple problems arise as well. A depleted food source still attracts ants even if the pheromones do not decrease. Also, even if a shorter path to the food source is presented to the ants, the ants will not change to the shorter path until the pheromone level on the new path increases enough to justify switching over. If the pheromones do not diffuse quickly, then the longer path continues to receive pheromones, while the shorter path receives very few. A similar tipping point comes from the number of ants taking a particular path, and the length of that path. If only a few ants use a long path, then the pheromones will diffuse before other ants can discover it. A final complexity of the system is that the system itself is designed as a feedback loop. Good paths to food receive more pheromones, and in turn, more ants are convinced to use that path again. Bad paths receive fewer pheromones because they take longer to travel, so other ants are not convinced to use the path.

Ant colony optimization (ACO) algorithms have been invented as attempts to model these complex behaviors in order to solve computational problems. ACO algorithms typically simulate ants traveling along the edges of a graph and spreading pheromones, based on some measure (e.g., length of path). The paths that are chosen and found to be better than others are examined more closely, whereas paths found to be worse than others are discarded. Depending on the ACO variant, the process is iterated until an acceptable path is found, or until some predefined level of confidence that the path is the optimal path is achieved.

The goal of our research is to explore the following questions:

1. Do the ant simulations used in ACO algorithms accurately portray how real ants solve problems?
2. Can the ant simulations used in ACO algorithms be made to adapt to changing environmental factors?
3. Can manipulating the ability to change these environmental factors provide a substantial increase in the long-term efficiency of ACO algorithms?
4. Can allowing "artificial ants" in ACO algorithms manipulate their own behaviors decrease the time necessary to

converge to a solution, without having a great impact on the space or conceptual complexity of the algorithm?

Previous Work

The first well known experiment with ants on finding the shortest path is the so-called "double-bridge" experiment (Deneubourg et al. 1990; Goss et al. 1989; Dorigo and Stutzle 2004). This experiment has two different setups. In the first, a colony of ants is presented with two paths of equal length to a food source. After a sufficient amount of time has passed, the ants converge to one path. Each path is chosen, over a number of trials, about 50 percent of the time. In the second setup, two paths to a food source are presented to the ant colony, but one is twice as long as the other. After a sufficient amount of time, all of the ants converge to the shorter of the two paths. The original double-bridge experiment introduced the notion of simulating an ant colony to solve the shortest path problem, and was the basis for the ACO algorithms that followed.

The first ACO algorithm, Ant System, was designed as a simple genetic algorithm (Dorigo 1992; Dorigo, Maniezzo, and Colorni 1996). A population of "artificial ants" attempted to solve the problem individually, and after each generation of artificial ants, pheromones of solutions were updated proportional to their quality. Since then, many extensions have been made to the Ant System. For example, the Elitist Ant System strongly weights the best solution found so far, resulting in a substantial speedup in converging to a solution (Dorigo, Maniezzo, and Colorni 1996).

Some research has been published on the effectiveness of ants and ACO algorithms to adapt to dynamic modification of data at run time (Guntsch, Middendorf, and Schmeck 2001). For real ants, several major environmental changes can affect the colony's efficiency as they search for a food source; two of the most significant changes are the main food source changing location and new paths being created. These two factors can be transformed into two computational problems: querying a program for solutions for multiple, similar problems in succession, and modifying the input data during a query to the program. The ability for artificial ants to adapt to this dynamic modification of data is the major incentive for using ACO algorithms over algorithms such as Dijkstra's Shortest Path algorithm. In instances such as routing network traffic, it is imperative that the algorithm adapt to the changing data and update the solutions accordingly (Dorigo, Birattar, and Stutzle 2006). One ACO algorithm used in industry today for such a purpose is AntNet, which is a major area of research for ACO algorithms (B. Baran 2001; Dhillon and Vanmieghem 2007).

However, ACO algorithms greatly simplify an ant system. Real ants face numerous other obstacles, such as changes in weather and seasonal variations. Real ants adapt to these conditions in various ways; for example, wood ants use information from the preceding years to decide which places to search first for food after winter (Mabelis 1978). This is analogous to a property of the problems in which ACO algorithms are used: previously found solutions may be applicable to future problems. Even if the best found solution to this problem is a sub-optimal solution to the next problem, another problem may arise at a later time for which this solution will again be optimal. If the algorithm utilizes a memory technique to store previous solutions, the algorithm requires more storage for these previous solutions. There is a tradeoff between how detailed and accuracy of the simulation, and how complex the ACO algorithms are to implement and maintain.

Approach

Our goal is to show the effects of an ACO algorithm that allows agents to modify their own behaviors in response to a change in input data. To do this, we first created a NetLogo model to simulate both double-bridge experiments (Wilensky 1999), expanding the model to include changing the input data during runtime of an ACO algorithm. We also made the assumption that the user would know when the data had been altered, and modified the ACO algorithm to allow ants to probabilistically choose their direction based both on the pheromone content of the path and on how much time elapsed since the data was changed. Finally, we included an exploration parameter that permits each ant to choose a sub-optimal path with some probability, enabling adaptation in dynamic situations.

In the first double-bridge scenario, there are only two possible paths to the food; both of which are of equal length. In the second scenario, there are still only two paths to take to the food, but one path is substantially longer than the other.

In our model, ants looking for food choose their next step probabilistically, biased in favor of paths with higher pheromone content, and then push their current location onto the stack. The probability that an ant will take a particular path is given by the amount of scent on that path divided by the total scent. Once the ant finds a food source, it retraces its steps back to the colony by use of the stack, while dropping a scent to entice other ants to follow its trail. The amount of scent dropped is inversely proportional to the length of the path found by the ant. These ant behaviors are similar to those defined in Ant System (Dorigo, Maniezzo, and Colorni 1996). Any possible loops are removed from the stack as the ants move from one node to another, rather than when the ants returned to the colony. This is a slight modification of the Simple-ACO algorithm, as defined by Dorigo and Stutzle (2004).

Experimental Design

We performed four experiments: for each of the two double-bridge scenarios, we simulated modification of data at runtime in two different ways. In each of these four experiments, we both tested how the original ACO algorithm would react to the modification, and gave an example of how our dynamic ACO algorithm in which ants can modify their behavior would react to the modification.

The *change-path* method simulates dynamic modification of data through direct modification of the path lengths. If the paths are originally equal, this technique simulates the introduction of a new, shorter path by changing the path that the ants did not choose to be shorter than the original path. If the paths are not equal, it swaps the lengths of the paths,

so that the ants are now on the substantially longer path. The *change-food* method simulates change in search parameters through deletion of the food source and introduction of a new food source at a different location. The food source is moved to the midpoint of the path that the ants did not choose. Both of these scenarios are invoked during run-time, so that ants are forced to adapt to the changing conditions. More specifically, they are called after the ants converge to a path.

The exploration probability for the ants in the original scenarios is 0.1 (that is, with probability 0.1, the ants will choose a random path (with each path being equally probable), and with probability 0.9, they will choose a path with probability proportional to the pheromone level on each path). In the modified experiment, when one of the two change methods is invoked, the willingness to deviate is set to 1.0, and then decreases every time the ant reaches the food source. This behavior is intended to model the ant's increased willingness to deviate from the path to explore other potential solutions when the solution is known to have substantially changed. In our dynamic ACO algorithm, the exploration probability is initially set to 0.1, but decreases by .002 to a minimum of .01 every time the ant reaches the food source.

To test the ability for ants to adapt to the modified conditions, the amount of pheromones on each path was recorded. The ants were said to converge to one of the two paths when the probability of choosing one path is 10 times as high as that of the other path. After convergence, one of the change methods was invoked.

A test is considered to be *successful* if the amount of pheromones decreases on the no-longer-optimal path, and the amount of pheromones on the new, optimal path eventually increases above the pheromone level on the old path.

We also compared the time taken to adapt in an ongoing simulation to that of the simulation beginning with the environmental changes already in effect, to determine the relative efficiency of the ACO algorithm as an underlying strategy. Finally, we recorded the number of ticks required to converge before the modification, and the number of ticks required for the ants to discover and converge to the optimum path after the data was modified.

Results

Figures 1-8 illustrate some sample runs of this experiment. Each figure shows lines: one for the average scent on the upper path and one for the average scent on the lower path.

Figures 1-4 show some results of the original ACO algorithm. Figure 1 shows the results of the first double-bridge experiment, in which each path was the same length and the path lengths were directly modified. The ants completely converged to a path after 3500 steps, and the change-path function was invoked. After 100 more steps, the ants completely converged to the opposite path, which was now optimal. Figure 2 shows the results of the first double-bridge experiment, in which each path was the same length and the search parameters were modified. The ants completely converged to a path after 5000 steps, and the change-food function was invoked. After 5000 more steps, the ants still had

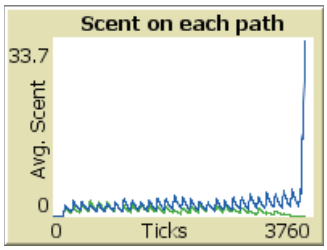

Figure 1: A sample trial of the first double-bridge experiment with the original ACO algorithm using the Change Path function at tick 3500 and running the simulation for 100 more ticks

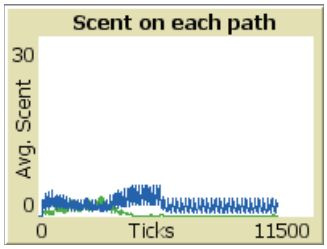

Figure 2: A sample trial of the first double-bridge experiment with the original ACO algorithm using the Change Food function at tick 5000 and running the simulation for 5000 more ticks

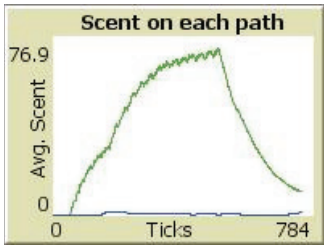

Figure 3: A sample trial of the second double-bridge experiment with the original ACO algorithm using the Change Path function at tick 500 and running the simulation for 250 more ticks

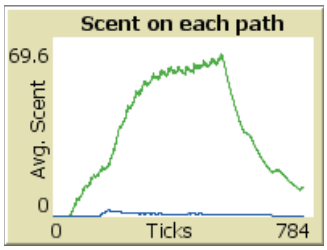

Figure 4: A sample trial of the second double-bridge experiment with the original ACO algorithm using the Change Food function at tick 500 and running the simulation for 250 more ticks

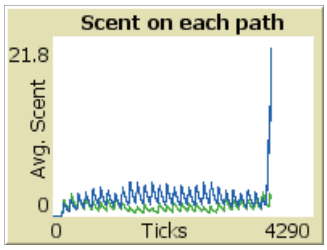

Figure 5: A sample trial of the first double-bridge experiment with the dynamic ACO algorithm using the Change Path function at tick 3500 and running the simulation for 100 more ticks, with ants being able to modify their behaviors

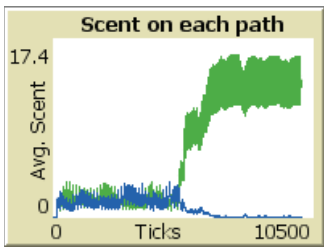

Figure 6: A sample trial of the first double-bridge experiment with the dynamic ACO algorithm using the Change Food function at tick 5000 and running the simulation for 5000 more ticks, with ants being able to modify their behaviors

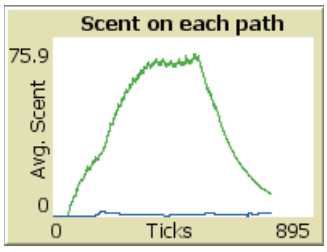

Figure 7: A sample trial of the second double-bridge experiment with the dynamic ACO algorithm using the Change Path function at tick 500 and running the simulation for 250 more ticks, with ants being able to modify their behaviors

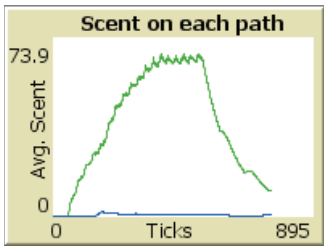

Figure 8: A sample trial of the second double-bridge experiment with the dynamic ACO algorithm using the Change Food function at tick 500 and running the simulation for 250 more ticks, with ants being able to modify their behaviors

not discovered the optimal path. Figure 3 shows the results of the second double-bridge experiment, in which one path was significantly longer than the other and the path lengths were directly modified. The ants completely converged to a path after 500 steps, and the change-path function was invoked. After 250 more steps, the ants began to converge to the opposite path, which was now optimal. Figure 4 shows the results of the second double-bridge experiment, in which one path was significantly longer than the other and the path lengths were directly modified. The ants completely converged to a path after 500 steps, and the change-food function was invoked. After 250 more steps, the ants had not yet begun to converge to the opposite path, which was now optimal.

Figures 5-8 show some results of our dynamic ACO algorithm. Figure 5 shows the results of the first double-bridge experiment, in which each path was the same length and the path lengths were directly modified, but with the ants having the ability to modify their behaviors. The ants had not yet converged to a path after 3500 steps, and the change-path function was invoked. After 100 more steps, the ants completely converged to the optimal path. Figure 6 shows the results of the first double-bridge experiment, in which each path was the same length and the search parameters were modified, but with the ants having the ability to modify their behaviors. The ants had not yet converged to a path after 5000 steps, and the change-food function was invoked. After 5000 more steps, the ants completely converged to the optimal path. Figure 7 shows the results of the second double-bridge experiment, in which one path was significantly longer than the other and the path lengths were directly modified, but with the ants having the ability to modify their behaviors. The ants completely converged to a path after 500 steps, and the change-path function was invoked. After 250 more steps, the ants began to converge to the opposite path, which was now optimal. Figure 8 shows the results of the second double-bridge experiment, in which one path was significantly longer than the other and the path lengths were directly modified, but with the ants having the ability to modify their behaviors. The ants completely converged to a path after 500 steps, and the change-food function was invoked. After 250 more steps, the ants began to converge to the opposite path, which was now optimal.

Figures 9-12 show the results of our analysis of the original ACO algorithm, with respect to our four scenarios.

Figure 9 shows the results of the experiment in which the lengths are originally the same, but then the path lengths are changed. For this experiment, the average number of ticks required for the ants to first converge on a path was 1528.11 with standard deviation 1379.97, whereas the number of ticks required for the ants to converge to the new path was 93.13 with standard deviation 51.22. Therefore, we believe that the experiment was successful and that the ants were able to adapt to this particular scenario.

Figure 10 represents the results of the scenario in which the lengths are originally the same, but then the food position is changed. For this experiment, the average number of ticks required for the ants to first converge on a path was 1692.72 with standard deviation 1706.84, whereas the num-

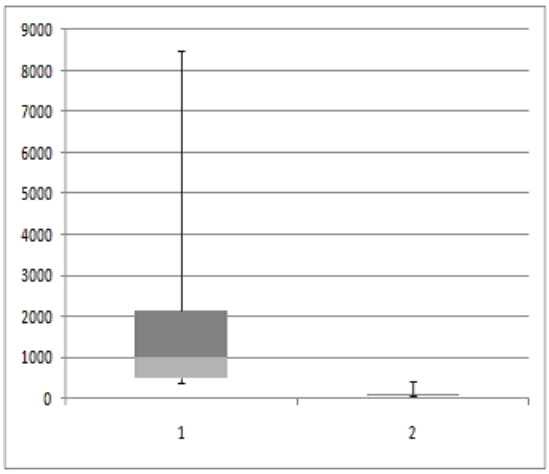

Figure 9: Box plots, including maximum (top of bar), second quartile (dark gray box region), third quartile (light gray box region), and mininum (bottom of bar), representing results of 100 trial runs for the experiment in which the lengths are originally the same, but then the path lengths are changed. Box plot 1 represents the number of ticks required to converge before modification, plot 2 represents the number of ticks required to converge after modification.

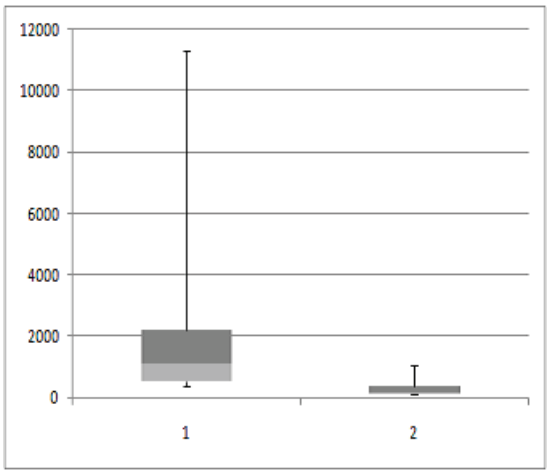

Figure 10: Box plots, including maximum (top of bar), second quartile (dark gray box region), third quartile (light gray box region), and mininum (bottom of bar), representing results of 100 trial runs for the experiment in which the lengths are originally the same, but then the food location is changed. Box plot 1 represents the number of ticks required to converge before modification, plot 2 represents the number of ticks required to converge after modification.

ber of ticks required for the ants to converge to the new path was 266.7 with standard deviation 179.32. Therefore, we believe that the experiment was successful and that the ants were able to adapt to this particular scenario.

Figure 11 represents the results of the scenario in which the lengths are originally substantially different, but then the path lengths are changed. For this experiment, the average number of ticks required for the ants to first converge on a path was 57.01 with standard deviation 3.06, whereas the number of ticks required for the ants to converge to the new path was 197.30 with standard deviation 48.18. It is useful to notice that although the time taken to converge to the new path is greater than that to the original path, the average time taken to converge to the new path is similar to the average time taken to converge to the new path in each of the other scenarios. Therefore, we believe that the experiment was successful and that the ants were still able to adapt to this particular scenario.

Figure 12 represents the results of the scenario in which the lengths are originally substantially different, but then the food location is changed. For this experiment, the average number of ticks required for the ants to first converge on a path was 56.98 with standard deviation 2.71, whereas the number of ticks required for the ants to converge to the new path was 1081.36 with standard deviation 433.53. It is useful to notice that although the time taken to converge to the new path is much greater than that to the original path. The new, optimal path lengths are almost equal, and are similar to the lengths for the first two scenarios. Therefore, we believe that even if this was an unexpected consequence, the experiment was successful and that the ants were still able to adapt to this particular scenario.

The first interesting point raised by the results is that when the two paths are equal, the ACO algorithm took a long time converge to a path; when the two paths differed significantly in length, the ants had a relatively easy time figuring out which path was optimal. This may be due to a tipping point in the algorithm, and is unexpected.

The second interesting point is the behavior of the system after modification of the data. The two scenarios of experiment two were treated similarly by the ACO algorithm. There was an immediate drop in the amount of pheromone, and the beginning of an upward trend for the new optimal path can be seen in Figures 3, 7, and 8. However, in Experiment 1, Scenario 2, the new optimal path received no pheromones, whereas the other figures for Experiment 1 show a spike in pheromones. We believe this to be due to random fluctuations, and that more ants diverged from the path in Figures 1, 5, and 6, whereas fewer ants diverged from the path in Figure 2. If the simulation were run for a sufficient amount of time, it is expected that the plots would eventually look similar. However, we notice from Figures 5 and 6 that, with the modification, ants were more quickly and more often able to discern the optimal path after the data had changed.

Conclusions and Future Work

Our original hypotheses, in reference to our research questions, were:

1. The ant simulations used in ACO algorithms simplify, but overall accurately portray how real ants solve problems.
2. The ant simulations used in ACO algorithms can be made to adapt to changing environmental factors.

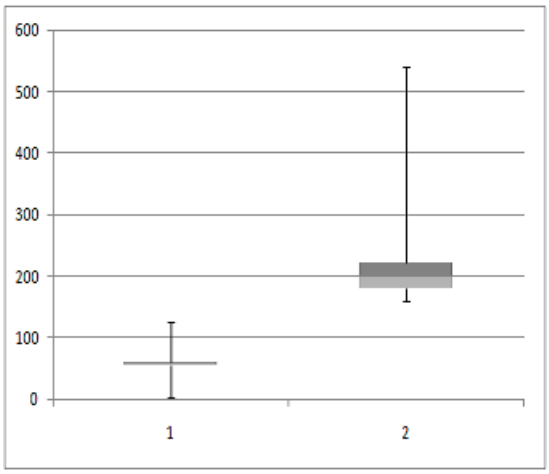

Figure 11: Box plots, including maximum (top of bar), second quartile (dark gray box region), third quartile (light gray box region), and mininum (bottom of bar), representing results of 100 trial runs for the experiment in which the lengths are originally substantially different, but then the path lengths are swapped. Box plot 1 represents the number of ticks required to converge before modification, plot 2 represents the number of ticks required to converge after modification.

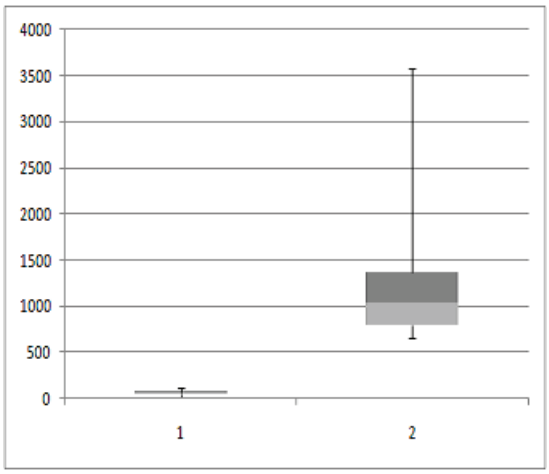

Figure 12: Box plots, including maximum (top of bar), second quartile (dark gray box region), third quartile (light gray box region), and mininum (bottom of bar), representing results of 100 trial runs for the experiment in which the lengths are originally substantially different, but then the food location is changed. Box plot 1 represents the number of ticks required to converge before modification, plot 2 represents the number of ticks required to converge after modification.

3. Manipulating the ability to change these environmental factors provides a substantial increase in the long-term efficiency of ACO algorithms.
4. Allowing "artificial ants" of ACO algorithms to manipulate their own behaviors decreases the time necessary to converge to a solution, without having a significant impact on the space or conceptual complexity of the algorithm.

We adopted slightly more complex rules by which the ants act, which resulted in two major findings. First, this modification allows ACO algorithms to more accurately model ants in the real world. Next, this modification also increases the ACO algorithm's computational efficiency in solving a shortest path problem, though not as much as expected. By allowing ants to adapt their own behaviors in order to understand when a solution will dramatically change, a reduction in the time taken to find a solution may be generated. This is most clearly demonstrated in Figure 6.

Our reproduction had similar results to the original double-bridge experiment, adding confidence that the simulation was valid. One major concern about the validity of the experiment is the amount of time it took for the ants to converge upon a solution if the two paths were equal; in the model, it took 2000 steps before one path was used significantly more than the other. However, this demonstrates a weakness in the original ACO algorithms and a strength in our modification of the ACO algorithm.

This study also suggests some possibilities for future work in ACO algorithms. Other modifications to ACO algorithms may also produce substantial consequences. As stated previously, the simplification of ant systems into an ACO algorithm overlooks several existing similarities between "real ants" and "artificial ants." Other possible relationships between real ants and simulations of ants should be explored in order for us to gain a more accurate understanding of both the Dynamic Shortest Path Problem and the ACO algorithms.

References

B. Baran. 2001. Improved AntNet Routing. *ACM SIGCOMM Computer Communication Review* 31:42–48.

Deneubourg, J.; Beckers, R.; Goss, S.; and Pasteels, J. 1990. Collective Decision Making through Food Recruitment. *Insectes Sociaux* 37:258–267.

Dhillon, S., and Vanmieghem, P. 2007. Performance Analysis of the AntNet Algorithm. *Computer Networks* 51:2104–2125.

Dorigo, M., and Stutzle, T. 2004. *Ant Colony Optimization*. MIT Press.

Dorigo, M.; Birattar, M.; and Stutzle, T. 2006. Ant Colony Optimization. *IEEE Computational Intelligence Magazine* 1:28–39.

Dorigo, M.; Maniezzo, V.; and Colorni, A. 1996. Ant System: Optimization by a Colony of Cooperating Agents. *IEEE Transactions on Systems, Man, and Cybernetics* 26:29–41.

Dorigo, M. 1992. *Optimization, Learning and Natural Algorithms*. Ph.D. Dissertation, Politecnico di Milano, Italy, [in Italian].

Goss, S.; Aron, S.; Deneubourg, J.; and Pasteels, J. 1989. Self-Organized Shortcuts in the Argentine Ant. *Naturwissenschaften* 76:579–581.

Guntsch, M.; Middendorf, M.; and Schmeck, H. 2001. An ant colony optimization approach to dynamic TSP.

Mabelis, A. 1978. Wood Ant Wars: the Relationship Between Aggression and Predation in the Red Wood Ant (Formica Polyctena Forst). *Netherlands Journal of Zoology* 29:451–620.

Wilensky, U. 1999. NetLogo. Center for Connected Learning and Computer-Based Modeling, Northwestern University, available at http://ccl.northwestern.edu/netlogo/.