# Multi-Robot Cooperation Using the Ant Algorithm with Variable Pheromone Placement

**Eric Borzello[i] and Laurence D. Merkle[ii]**

## Abstract

**The Ant Algorithm was created by examining real life ant colonies and developing an algorithm to use the concept of "stigmergy" to approach multi-robot problems with distributed control. As robots work on tasks, more robots attempt difficult tasks. Task deadlock occurs when robots attempt impossible tasks indefinitely. Previous research avoids task deadlock through adaptive attenuation factors. This research investigates avoidance of task deadlock through variable pheromone placement. Results of computational experiments are presented demonstrating the increased effectiveness of the new algorithm.**

**Key Words: multi-robot system, cooperation, ant algorithm, variable pheromone placement**

## I. Introduction

There are many applications of multi-robot systems, such as exploring unknown areas, cleaning a house, and automated laboratory systems. Though these applications vary in their specifics, most of them share a key feature – it would be best if the robots in the system could collaborate with each other without any outside human intervention.

A difficulty facing multi-robot system implementations is avoiding situations in which all tasks are completed successfully, but the number of robots working on each task is subobtimal. If too few robots are working then their effort is wasted until more robots arrive. If too many robots are working then the effort of the excess robots is wasted. In either case, the wasted effort implies wasted time steps.

In 2003, Ding Yingying, He Yan, and Jian Jingping [5] applied the ant algorithm discussed by Dorigo and DiCaro [3] to multi-robot system design. The algorithm assigns more robots to more difficult tasks, and fewer robots to easier tasks. This allows the number of robots actually doing useful work to increase, in the sense that the number of time steps required to complete all of the possible tasks decreases. Yingying, et al. modify Dorigo and DiCaro's algorithm by adding an adaptive attenuation constant to eliminate task deadlock. They present results of experiments in which this modification decreases the time that robots spend going back to impossible tasks instead of seeking out and performing tasks they can accomplish.

This research examines a further modification of the ant algorithm for multi-robot systems that uses a novel technique of placing pheromone on difficult tasks. This technique causes robots to place more pheromone on tasks that are near completion, making the robots more likely to choose tasks that can be completed soon. It is demonstrated experimentally that effectiveness is improved as a result.

The remainder of this paper is organized as follows. Section II gives an overview of the ant algorithm in general and a description of the specific version used by Yingying, et al. Next, Section III describes the version of the ant algorithm developed in this research. Section IV describes the software system used as a simulation tool. Finally, Section V presents the results of computational experiments evaluating the effectiveness of the proposed algorithm.

## II. Background

Systems composed of multiple robots have several key features that make them desirable for many applications. First, presence of more than one robot allows robots to cooperate in order to achieve goals too difficult for any one robot alone. Furthermore, the presence of multiple robots allows for relative autonomy from human intervention, since many such systems can lose several robots without severe loss of functionality. In contrast, a system of a single robot is completely disabled if that robot is lost.

As a result of these features, many real world problems are approached relatively easily by systems of collaborating robots. Consequently, multiple robot systems are a field of significant interest in the area of artificial intelligence.

Another topic in artificial intelligence is ant algorithms. An ant algorithm is any algorithm in which the individual agents use decision making processes that are in some sense similar to those used by real ants. This general approach has been applied with some success to the traveling salesman problem [1], the single machine job scheduling problem [1], dynamic vehicle routing [2], image rendering [4], and numerous other domains.

Details of ant algorithms vary, but many share core tenets. First, agents mark tasks by placing pheromone. Also, agents are more likely to choose tasks with more pheromone. Thus, agents attract each other by placing pheromone. This indirect communication method, called "stigmergy," is surprisingly effective.

The basic process of this algorithm is as follows. When the robots enter a new area, they have no knowledge of the locations or difficulty of any tasks in that region, or of the positions of other robots. Each robot's motion is determined independently by a random walk, and the robots do not directly communicate with one another. When a robot reaches a task, it attempts that task.

If the robot makes progress on the task, the robot continues until the task is complete. On the other hand, if the robot is unable to make progress, it adds pheromone to

the task. The pheromone is added on a "blackboard" that can be read from and written to by any of the robots.

Robots looking for tasks check to see if any tasks are indicated by pheromone on the blackboard. If such tasks exist, the robot will choose one of them rather than continuing its random search. If there is more than one task on the blackboard, then the probability that the robot chooses task *i*, given there are *k* tasks total, is

$$p_i(t) = \frac{\mu^{select_i}\tau_i(t)\eta_i(t)}{\sum_{s=1}^{k}\mu^{select_i}\tau_s(t)\eta_s(t)}$$, where $\tau_i(t)$ is the amount of pheromone on task *i* at time *t*, $\mu \in (0,1)$ is the attenuation rate, $select_i$ is the number of times the particular robot has tried and failed task *i*, and $\eta_i(t)$ is the heuristic value associated with task *i*. For the experiments in Yingying, et al. $\eta_i(t) = 1$.

## III. Ant Algorithm Multi-Robot Cooperation

The goal of this research is to improve the effectiveness of the ant algorithm for multi-robot systems proposed by Yingying, et al. The system differs from theirs in the mechanism by which heuristics are incorporated into the algorithm. Specifically, in the system proposed by Yingying, et al., the amount of pheromone placed by a robot on a task in a time step is a constant, while in this research it varies heuristically depending on the difficulty of the task relative to the total strength of the robots currently working on the task. The rule used to determine the amount of pheromone placed is: $\Delta\tau = \omega + (1-\omega) * (F_c / F_n)$, where $\omega \in (0,1]$, $F_n$ is the amount of total force needed by the robots to push the ball, and $F_c$ is the amount of force currently exerted by the robots pushing on the ball.

The above has the consequence that more pheromone is placed on balls that already have nearly enough force exerted on them than is placed on balls that need much more force. The goal is to have nearly complete tasks completed sooner so that the robots who are working on them can instead work on other tasks.

Furthermore, the value of $\Delta\tau$ is computed using only the weight of the ball and the total amount of effort being put forth by the robots currently working on the task, which is local information that is available to the robot placing the pheromone. However, the robots that are selecting tasks use only the pheromone amounts and positions, which is information stored in the blackboard. This allows the robots to make informed decisions without needing a priori knowledge about the task or locations of the other robots.

Three details of the pheromone placement function are noteworthy. First, when $\omega = 1$ this function corresponds exactly to the function used by Yingying, et al. Also, the probability of a robot choosing a particular task is determined by the amount of pheromone placed on that task relative to the total amount of pheromone placed, rather than by the absolute amount placed on that task. Finally, $\Delta\tau \in (0,1]$, because if $F_n \le F_c$ then no pheromone will placed (recall that pheromone is only placed on a task when a robot fails to complete it).

## IV. Simulation

This section describes a simulation environment used to evaluate the algorithm discussed above. In this environment, robots have the characteristics discussed in Ding, et al. Specifically, they "know" their current position and the position of the goal, they "sense" robots and the boundary of the environment within their range of perception, they detect and push balls, and they place pheromone on the blackboard. Collisions are not modeled.

The environment is initialized by placing *x* robots in random positions, as well as *y* balls and one goal in specified positions. After initialization, the behavior of robot *i* is determined by its state, which is either *Wandering*, *Proceeding*, or *Working*, and whether or not there is at least one task *s* for which the following condition holds.

$$\phi(i,s) \equiv \left[\mu^{select_i}\tau_s(t) < 0.01\right] \wedge \left[\tau_s(t) > 0\right]$$

- If robot *i* is in the *Wandering* state and $\phi(i,s)$ becomes true for one or more tasks, then it chooses one of those tasks as described above and enters the *Proceeding* state with the corresponding ball as its destination. If $\phi(i,s)$ is false for all tasks *s*, and the robot discovers a ball *b*, then it enters the *Proceeding* state with ball *b* as its destination. Otherwise, the robot takes a step and makes a small random change in direction. Ignoring tasks for which $\phi(i,s)$ is false prevents robots from cycling between impossible tasks.
- Robots in the *Proceeding* state take steps in the direction of their destinations until either they reach the destination or the corresponding task is completed. In the former case, they enter the *Working* state, while in the latter case they enter the *Wandering* state.
- Robots in the *Working* state attempt to move their ball towards the goal. If they are not able to do so, they increment a counter. If the counter reaches a limit (15 in this case), they add pheromone to their task and enter the *Wandering* state. They also enter the *Wandering* state if they are able to push their ball to the goal.

## V. Results

The scenarios used in these experiments are as follows, where *y*w* indicates *y* balls, each weighing *w* units: Scenario A: 1*15, 1*5, 2*3; Scenario B: 1*15, 1*5, 2*3, 1*11; Scenario C: 1*15, 1*5, 2*3, 1*11, 1*8; Scenario D: 1*15, 1*5, 2*3, 2*11, 1*8; Scenario E: 10*15, 65*5, 25*3;

Scenario F: 50*15, 100*5, 50*3. Scenario A corresponds to the experiments performed by Yingying, et al. [5]. Also, as mentioned in Section III, the special case in which ω = 1 corresponds to their algorithm. Each simulation ends when the *y* tasks consisting of pushing the balls to the goal are all completed or determined to be impossible.

Results are given in Table 1, averaged over 50,000 simulations per scenario, with independent starting locations for each simulation. For the examples used in these experiments, smaller ω values generally increase effectiveness. This suggests that the heuristic strategy proposed in this research is useful for this class of problem.

For Scenarios A-D, the simulation lengths have markedly bimodal distributions. In order to gain insight into the cause of this phenomenon, each simulation was categorized as corresponding to either a "Good" solution (small number of time steps) or a "Bad" solution (large number of time steps), and the results for the two categories were analyzed separately. Results for the "Good" solutions (given in Table 2) strongly suggest that near-optimal solutions can be found using a wide range of values for P. Results for the "Bad" solutions (given in Table 3) indicate that when the algorithm does not find near-optimal solutions, lower ω -values tend to result in better solutions.

For Scenarios E and F, in which the number of robots is substantially less than the number of tasks, the distributions of the time steps required per simulation are not noticeably bimodal. It is strongly suspected that the solutions found are relatively far from optimal, regardless of the P-value. However, smaller ω-values again result in greater effectiveness, with the differences being more significant than those for the easier scenarios.

## REFERENCES


[1] Eric Bonabeau, Marco Dorigo, and Guy Theraulaz, Swarm Intelligence: From Natural to Artificial Systems. (New York: Oxford University Press, 1999)

[2] Darren M. Chitty and Marcel L. Hernandez, "A Hybrid Ant Colony Optimization Technique for Dynamic Vehicle Routing," *Proceedings, 2004 Genetic and Evolutionary Computation Conference*, Seattle, WA, June 2004, 48-59.

[3] Marco Dorigo and Gianni DiCaro, "The Ant Colony Optimization Meta-Heuristic", *New Ideas in Optimization* 1999. (New York: McGraw-Hill)

[4] Yann Semet, Una-May O'Reilly and Fredo Durand, "An Interactive Artificial Ant Approach to Non-photorealistic Rendering," *Proceedings, 2004 Genetic and Evolutionary Computation Conference*, Seattle, WA, June 2004, 188-200.

[5] Ding Yingying, He Yan, and Jiang Jingping, "Multi-Robot Cooperation Method Based on the Ant Algorithm", *Proceedings, 2003 IEEE Swarm Intelligence Symposium,* Indianaoplis, IN, April 24-26, 2003, 14-18.


**Table 1. Average Time Steps to Complete Simulation**

| Scenario | ω Value | | | | | |
|---|---|---|---|---|---|---|
| | 0.025 | 0.05 | 0.1 | 0.2 | 0.5 | 1.0 |
| A | 388 | 394 | 394 | 393 | 393 | 401 |
| B | 584 | 579 | 587 | 589 | 588 | 604 |
| C | 685 | 679 | 680 | 681 | 683 | 695 |
| D | 707 | 709 | 707 | 709 | 710 | 721 |
| E | 9561 | 9590 | 9604 | 9634 | 9656 | 9874 |
| F | 41958 | 42118 | 42229 | 42491 | 42609 | 43726 |

**Table 2. Average Time Steps to Complete Simulations of "Good" Solutions**

| Scenario | ω Value | | | | | |
|---|---|---|---|---|---|---|
| | 0.025 | 0.05 | 0.1 | 0.2 | 0.5 | 1.0 |
| A | 68 | 68 | 68 | 69 | 69 | 69 |
| B | 107 | 106 | 106 | 107 | 106 | 106 |
| C | 139 | 139 | 138 | 138 | 138 | 138 |
| D | 167 | 167 | 166 | 166 | 167 | 167 |

**Table 3. Average Time Steps to Complete Simulations of "Bad" Solutions**

| Scenario | ω Value | | | | | |
|---|---|---|---|---|---|---|
| | 0.025 | 0.05 | 0.1 | 0.2 | 0.5 | 1.0 |
| A | 851 | 852 | 852 | 853 | 860 | 874 |
| B | 877 | 878 | 879 | 880 | 887 | 901 |
| C | 902 | 902 | 903 | 903 | 911 | 924 |
| D | 932 | 933 | 933 | 934 | 941 | 954 |


[i] Rose-Hulman Institute of Technology, Terre Haute, IN, 47803, Eric.Borzello@Rose-Hulman.edu.

[ii] Rose-Hulman Institute of Technology, Terre Haute, IN, 47803, Laurence.D.Merkle@Rose-Hulman.edu.