AAAI Technical Report FS-12-04
Human Control of Bioinspired Swarms

# AntBeePath: A Hybrid Bio-Inspired Algorithm for Path Determination

**João Paulo Lamartin, Joberto S.B. Martins**
Salvador University - UNIFACS
jplamartin@gmail.com, joberto.martins@gmail.com

**Abstract**

AntBeePath is a hybrid bio-inspired algorithm based on the behavior of ants and honeybees aimed at the resolution of the problem of finding the shortest paths for a given network topology. The algorithm, in brief, combines the pheromone release mechanism of existing Ant Colony Optimization (ACO) algorithms with a new bio-inspired mechanism based on the recruitment strategy of bees. Three versions of the algorithm were developed incrementally. Proof-of-concept results indicate that the AntBeePath Decay Hybrid Chain version is more efficient than the other developed versions and, beyond that, presented an improved performance in relation to an equivalent ACO algorithm. The results suggest that a hybrid algorithm, combining the ant's pheromone release with the new bio-inspired mechanism of bee recruitment along with a stagnation control mechanism can result in a new bio-inspired algorithm for path determination with improved characteristics.

## Introduction

The increasing complexity of computational systems has led computer scientists and engineers to search for inspiration in different areas. Mathematics and physics have always been intrinsically connected to computing, however, in the last few years, a different area has been gaining influence: biology.

Defined as the science that studies life and the living organisms, their structure, growth, inner workings, reproduction, origin, evolution, distribution, as well as the relationships between the environment and themselves, (Houaiss 2009) biology can be seen by some as the opposite of technology, but nothing could be farther from the truth. In fact, biological systems, after billions of years of evolution, have succeeded in solving problems that still afflict many current computational systems.



The basic idea behind the area that has been called bio-inspired computing is to search for parallels between natural and computational systems, and to attempt to apply the solutions found by nature in computing problems.

The purpose of this paper is to present an algorithm that combines the characteristics of two natural species in the resolution of the problem of finding paths in computer networks. The two species selected were the ant and the honey bee, each for a particular reason.

## Ant Colonies and Path Determination – Basic Algorithmic Aspects

At first sight, an ant colony may seem to be a simple system without much sophistication. A closer inspection will reveal that these minute insects are masters in the art of the search. When it leaves the colony in search of food, the ant moves in a random direction and leaves in its path a trail of pheromone. When the ant finds a source of food, it returns through its path and releases more pheromone. When the path of a second ant crosses with the trail laid by the first, there is a probability "p" that this second ant will follow the path, and if that does happen, the route will be even more attractive to other ants, in other words: the degree of interest (attractability) of the path is directly proportional to the amount of times the ants have traveled through it.

This pattern of behavior is called stigmergy and it is characterized by a modification to the environment that affects and stimulates other agents to perform an action that reinforces the previous one and that ultimately leads to a cooperative, albeit unconscious, effort (Grasse 1986).

Besides stigmergy, one of the most interesting algorithmic characteristics of the search performed by the ants is that, even after the establishment of a route that has been followed by hundreds of ants, when a new ant reaches this trail, the probability of following it is not 100%. It can still decide to pursue a different and new path. This means that ants are always searching for alternate routes.

Another important aspect about the behavior of the ants is that they tend to find the shortest path between the colony and the source of food. To understand why, consider the following scenario: An ant finds a source of food and returns to the colony. The path it traveled from the colony to the food was not the shortest, because when it was faced with an obstacle on the middle of the route, having to choose between two different possibilities (up or down) it randomly selected the longest (Fig. 1).


Figure 1. Ant establishing longest path and two ants establishing different paths

Later on, two ants come to the same bifurcation. One decides to follow the trail left by the first ant (going down) but the second decides to try the other one (going up) and by doing so ends up forming a new and shorter route.

As the path chosen by the second ant is shorter than the first, it will return to the colony before the other one (Denny, Wright, and Grief 2001). As the path can be traveled faster, the tendency is that the amount of pheromone will increase at a faster rate in the shorter route and as such, it should become the dominant route. In this way, given enough time, the ants tend to converge to the shortest routes.

The ant colonies present a number of properties that are desirable to computational systems. First of all, the ants represent distributed agents deprived of a central controlling unit. Secondly, they have the capability of not only finding routes, but of finding the shortest routes and do so not by exchanging messages but through modifications to their environment. Thirdly, even though they already have an established route, they keep searching for new and better routes.

## Bee Colonies and Path Determination – Basic Algorithmic Aspects

Unlike ants, bees don't use pheromone, instead, they adopt a more direct and complex communication mechanism, the dance. Just like the ant, the bee leaves the colony in a random direction, however, when it finds the food, it returns directly to the colony and communicates to its peers its distance and direction. This is done through a series of movements similar to a dance. The bee positions itself in such a way that it forms an angle between its body on the colony and the sun. The angle communicates the direction of the food and the duration of the dance informs the distance (Nagpal 2010).

The bee colonies present a number of algorithmic properties that are interesting to computational systems. Just like ants, the bees are distributed agents deprived of a central controlling unit. Secondly, they have the capability of finding routes, but most importantly they communicate specific parameters associated with the route such as its direction, distance and quality. Thirdly, they are able to recruit other agents to pursue informed routes.

## Previous Research on ACO and BCA

The ant colonies capability to discover shortest routes between the nest and the source of food, using a pheromone release mechanism was submitted to a process of reverse engineering and used, among other possibilities, as an instrument in the resolution of optimization problems, such as the traveling salesman problem (Dorigo and Gambardella 1997). The solutions proposed in the literature are, in most cases, variations of the concept originally suggested by Dorigo, et al, in the piece in which they presented the first algorithmic solution to the Traveling Salesman Problem inspired by the behavior of an ant colony, the Ant System (Dorigo, Colorni and Maniezzo 1991). The expression Ant Colony Optimization (ACO) serves as an umbrella for the algorithms inspired by ant colonies.

The Bee Colony Algorithms are an area of Swarm Intelligence which studies bee's behavioral patterns, as well as similar types of insects, and attempts to apply their basic principles to challenges faced by man (Lemmens 2006).

The first Bee Colony Algorithm proposed in the literature that offers something related to a computational perspective was BS, the Bee System (Lucic and Teodorovic 2003). Just like the Ant System, BS was proposed as a way to solve Traveling Salesman Problems. As far as this paper is concerned, the specific aspect of bee behavior to be incorporated into the AntBeePath algorithm proposed here is the bee's ability to dance and recruit more bees.

## AntBeePath and Hybrid Algorithms

Ant-Bee Routing (Rahmatizadeh, Shah-Hosseini and Torkaman 2009) is an existing hybrid ACO/BCA algorithm. It differs from AntBeePath in the way the bio-inspired mechanism mapping was conceived and, also, in respect to some implemented functionalities. Firstly, AntBeePath searches for all the shortest paths from a central node (departure point) to every other node in a given topology, much like a path discovery mechanism available in some routing protocols. The assumption is basically to determine a basic construct - path determination – that can be used, for instance, for routing purposes. AntBeePath assumes that another higher level abstraction or functionality will use this basic construct for routing purposes or any other applications need requiring "path determination". Ant-Bee Routing uses a distributed approach (in a sense breaking the bi-

ological analogy) and launches agents from every node to destination nodes selected according to traffic. Another bio-inspired mechanism difference is that the agents in Ant-Bee Routing are not truly hybrid. In effect, when an ant reaches its destination it dies and a bee is generated. In AntBeePath the agent is truly hybrid, simulating the behavior of both ant and bee simultaneously.

## The AntBeePath Algorithm

The AntBeePath algorithm aims to combine some algorithmic aspects of the behavior of ants and bees in order to propose a new algorithm (resolution method) for the problem of determining best paths in a network of nodes. The criteria used to measure the quality of a path was distance, so the shorter the path, the better it is considered. In practical terms, computer networks are the main target for the new proposed algorithm.

In this research, the hybrid agents are suggestively called "antBees". AntBee agents combine the capability of ants to communicate through the release of pheromone (stigmergy) and the dancing ability of the bee, in which it communicates to its peers the path it has just discovered.

A hybrid antBee agent is basically represented by:

- State
- VisitedNodes
- distanceCovered
- electedDistance

AntBee's state describes the situation in which an agent is found in a given moment. "AntBees Visited Nodes", as suggested, indicates the list of nodes that the agent has already visited. The "distanceCovered" parameter indicates the number of nodes the agent has already visited and, finally, the "electedDistance" parameter is a pseudo-random number generated for each agent that specifies the maximum distance an agent can travel in order to limit the scope of the search.

The AntBeePath algorithm execution traverses a set of antBee agent states as illustrated in Figure 02.

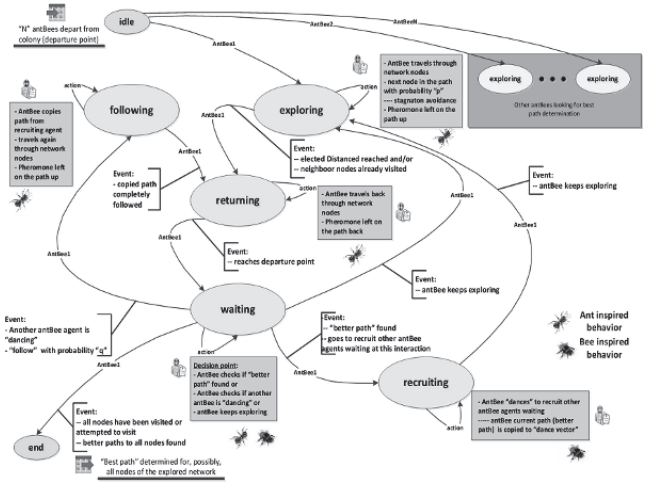


Figure 2. AntBee Agent States

At initialization, every antBee agent leaves the colony (departure point) and begins to explore the network. Following that, an antBee agent can be in any of the following five states at any given moment: exploring, returning, waiting, recruiting and following.

Exploring: It represents the state in which antBee agents explore the network. The antBee agent first searches the nodes that are reachable from its current position. From this set of reachable nodes, the agent selects one of them based on a weighted equation. The probability "p" of choosing the next node from the set of reachable ones is a function of the amount of pheromone present in the links between the nodes. This relationship is not 1 to 1, to avoid the phenomenon known as "stagnation", where the agents cease to pursue alternate routes in the presence of a dominant path. Every antBee can only move one node per iteration and it leaves a pheromone trail along its path. The agent continues in the exploring state until one of two events happen: either it will have exhausted all nodes in its current path (it reaches a point where all reachable nodes have already been visited) or the "electedDistance" will be equal to the "distanceCovered meaning that the agent has traveled its maximum allowed distance. At this point the antBees state changes to "returning".

Returning: At this state, antBee agent returns to the colony reversing the path traveled during the exploring state. Once again, the agent releases pheromone along its path. For every iteration, the agent goes back one node. AntBee agents continue in this state until it ultimately returns to the colony. When it reaches the colony, its state changes to "waiting".

Waiting: State in which an antBee agent finds itself after it returns to the colony. At this point there are three distinct courses of action: 1.) If the AntBee agent discovers that its recently traveled path is better than the current path stored in the colonies table it changes then his state to "recruiting". 2.) In case another antBee agent is currently "dancing" (bee behavior) there is a probability "q" of being recruited by this agent and proceeds to the "following" state. 3.) AntBee agent resumes the exploration by going to the "exploring" state.

Recruiting: AntBee agent compares the path it just traveled to node "N" to the current path stored in the colony's table. If it discovers that its recently traveled path is better (shortest) than the current colony's best path, it "dances" to recruit more agents to repeat path. Only antBee agents that are in the colony at the same moment can be recruited. This state is followed by "exploring".

Following: State in which an antBee agent is recruited by another agent to follow its path. This is done by copying the list of visitedNodes. The recruited antBee agent follows this acquired path releasing pheromone over it and when it reaches its destination its state changes to "returning".

## Implementation

In brief, the algorithm's objective is to find out a set of best paths from the colony (agents departure point) to all remaining nodes of the explored network. The stop criteria adopted in the proof-of-concept to conclude antBeePath's execution and determine the best possible paths for the network is based on the following assumption: all the best paths are already known, having been previously calculated and informed to the algorithm. At every cycle of execution, the current best paths table discovered by the colony is compared to this known best path's table, when both are identical the algorithm stops. If this convergence has not occurred after a specified interval, it is assumed that the algorithm either has stagnated or simply is not efficient enough. Efficiency, for the purpose of this proof-of-concept evaluation, is defined in terms of the number of algorithm iterations and resulting execution time. Whether the algorithm's effectiveness and efficiency are directly related to the number of agents involved in network exploration was also investigated and is discussed briefly in the following sections.

The first algorithm developed was the "Hybrid antBeePath" (basic version). This version is characterized by combing the ant's pheromone release with the bee's recruitment strategy for best path determination. Being true to the biological analogy, only the final destinations are taken into account, the paths to the intermediary nodes are not updated in the colony's table. This is as if a bee had returned to the colony and informed the distance to a source of food while disregarding intermediary points.

A second version named "Chain antBeePath" was developed incrementally. Chain antBeePath is a variation of the basic hybrid antBeePath in which the knowledge acquired in terms of the paths is fully reflected in the algorithm best paths (routes) database. In other words, once a best path to the final destination is found, not only this destination is updated on the colony's table but also all intermediary nodes are updated against the colony's best path table. This is an enhancement to the basic biological model. This is as if a bee returned to the colony and informed the distance not only to the source of food but to all intermediary points between the colony and food.

A third version, named "Decay antBeePath" was also created to deal with the problem of stagnation. Stagnation is a state (operational) situation where the algorithm is stuck because the agents cease to pursue alternate routes. As such, the algorithm either becomes unable to converge and reach its objective or does it in a way that compromises performance. The stagnation problem is, in general, a possible state condition of the algorithm since route/path decisions are taken by agents based on a probability draw and, as such, it can lead to this non convergence scenario. The solution proposed to was to restart the pheromone table at given intervals. As such, a "Decay Chain antBeePath" version was implemented and evaluated in order to identify/determine the suitable intervals for stagnation prevention and, beyond that, compare the overall algorithm behavior in relation to the previous ones.

## Proof-of-Concept

Following the definition of antBeePath's variations and their effective implementation, a set of simulations were developed in order to investigate the basic characteristics of the proposed algorithms as a basic proof-of-concept. As such, the idea was to validate the basic assumption that the hybrid behavior of ants and bees, when incorporated into a unique algorithm, could efficiently determine paths in a network for diverse purposes (routing, messaging passing, etc.). In other words, the proof-of-concept tried to identify the antBeePath algorithm variations that, potentially, find all the best routes in the fastest possible way and try to cross-relate their resulting behavior.

The selected simulation topology chosen for the antBeePath algorithm variations testing was the Japanese backbone NTTnet (figure 3) which, in effect, is a prevalent option adopted in the literature for bio-inspired algorithms.

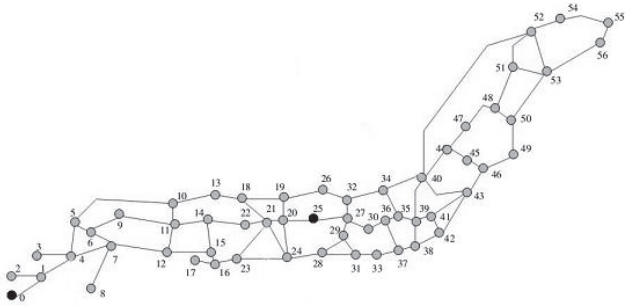

Figure 3. AntBeePath simulation topology – NTTnet

Simulations considered, in terms of the proof-of-concept, whether the placement of the colony could affect the performance of the algorithm. As such, two nodes were tested as the colony, the starting point from where antBee agents depart:

- Node 0: a choice that potentially explores network traversing;
- Node 25: a choice that, as a central node, potentially explores the algorithm behavior in relation to network coverage.

The performance of the algorithm was evaluated in terms of two parameters: number of iterations and execution time. The number of iterations is the most accurate in terms of the simulation runs because it effectively provides a measure and/or indication of the convergence time required by the algorithm to find out the best route(s). The execution time is highly dependent on the available hard-

ware and, realistically, was considered as an indication of the feasibility of the algorithms implementation on current and common machines (PCs). Pragmatically, it provides an indication about "how executable" would be the algorithm in terms of its application in different areas.

The simulations were conducted with a wide variation in the number of agents (from 1.000 to 16.000, through increments of 100) exploring the network. These values were selected for two different reasons. Firstly, they are used in actual ant's research (Ratnieks 2008) and, as such, can facilitate result analysis in relation to other existing solutions. Secondly, there is also a need to identify what is the minimum number of agents that would eventually drive the algorithm to a stagnation state incapable of achieving its best paths purpose.

The results for the Hybrid antBeePath, Chain antBeePath and Decay Chain antBeePath are illustrated in figure 4.

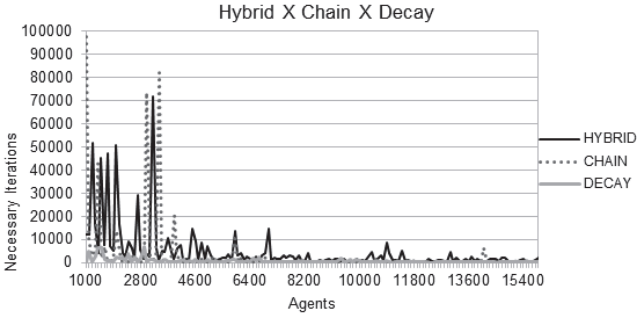


Figure 4. Comparison between the three versions

In brief, the Hybrid antBeePath algorithm was the least efficient in terms of iterations and execution time. The algorithm reached stabilization at around 7.500 agents and stagnated occasionally with the number of agents between 1.000 and 2.000, it also initially converged slower than the other versions.

The Chain antBeePath variation presented an important improvement in terms of efficiency in relation to the previous one. As illustrated in figure 4, stabilization occurs sooner, in this case, it becomes stable after 6.000 agents.

In terms of the Decay antBeePath variation we observe an additional efficiency improvement, it converges much faster than the other versions, it is also more stable and its performance tends to increase as more agents are added.

One interesting aspect of this algorithm is that it can easily be reverted to a simple Ant Colony Optimization algorithm, by the mere suppression of the bee behavior. This allowed for a comparison between something that would be very similar to a common ACO algorithm and the AntBeePath. The results of the comparison indicate that the Decay AntBeePath algorithm was more efficient than the simpler alternative. Both the number of iterations and time is increased when the bee's dance component of the algorithm is removed from the code, in other words, Decay AntBeePath performs better than the alternatives.

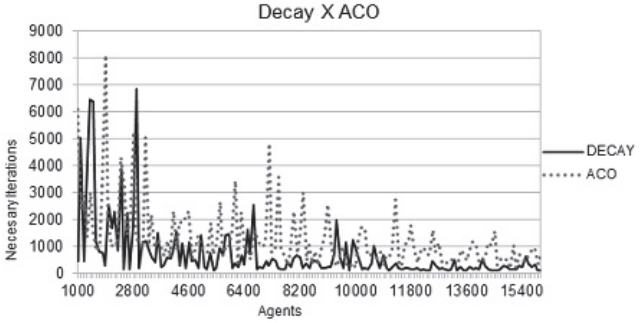


Figure 5. Comparison between Decay AntBeePath and ACO equivalent

The placement of the colony affected the performance of every variation of the AntBee Path Algorithm. In most cases, placing the colony on node 25 caused the algorithm to converge sooner. Figure 5 demonstrates the difference between the two nodes for the final and best version of the AntBee Path Algorithm.

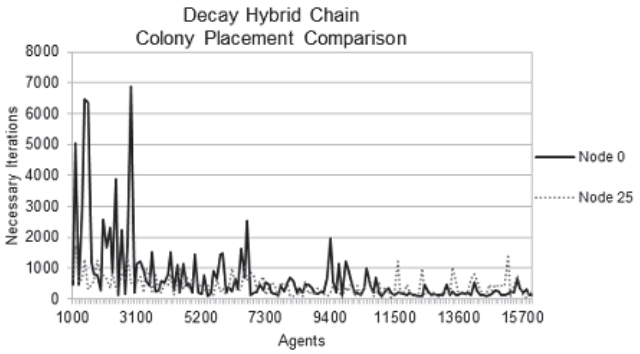


Figure 6. Colony Placement Comparison

## Final Considerations

In most cases, the different versions of the algorithm converged to find all the shortest routes within the specified time frame (with the exception of the basic Hybrid version using node 25 as the colony that stagnated on eight different moments). This does not mean that the algorithm will always find all the shortest paths in any given network topology. What it does mean is that the algorithm tends to converge, and the probability of success tends to approach 100% as time increases.

After experimenting with different versions and colony placement variations the best performing version was the Decay AntBeePath. Its superiority here is due to the fact that it is the version that consistently finds all the shortest paths while increasing the number of agents and decreasing the number of necessary iterations.

## References


Houaiss, 2009. A. Novo Dicionário Houaiss da Lingua Portuguesa. Ed. Objetiva.

Grassé, P.P. 1986. La reconstruction du nid et les coordinations inter-individuelles chez Bellicoitermes natalenis et Cubitermes sp. La theorie de la stigmergie: Essai d'interpretation des termites constructeurs. Insect Societies 6, 41-83.

Denny, A.J., Wright, J. and Grief, B. 2001. Foraging efficiency in the wood ant, Formica rufa: is time of the essence in trail following? Animal Behaviour 61, 139–146.

Nagpal, S. 2010. Honeybees' dance inspires more efficient Internet servers. TopNews. <http://www.topnews.in/honeybees-dance-inspires-more-efficient-internet-servers-26403>.

Dorigo, M. and Gambardella, L.M. 1997. Ant colonies for the traveling salesman problem. IRIDIA, Université Libre de Bruxelles. IEEE Transactions on Evolutionary Computation, 53–66.

Dorigo, M., Colorni, A. and Maniezzo, V. 1991. Ant System: An Autocatalytic Optimizing Process. Technical report, Politecnico di Milano.

Lemmens, N.P.P.M. 2006. To bee or not to bee: A comparative study in swarm intelligence. Maastricht University. Maastricht, Holland.

Lucic, P. and Teodorovic, D. 2003. Computing with Bees: Attacking Complex Transportation Engineering Problems. International Journal on Artificial Intelligence Tools 12, 375-394. Ed. World Scientific Publishing Company.

Rahmatizadeh, S., Shah-Hosseini and H., Torkaman, H. 2009. The Ant-Bee Routing Algorithm: A New Agent Based Nature-Inspired Routing Algorithm. Journal of Applied Sciences 9, 983-987.

Ratnieks, F.L.W. 2008. Biomimicry: Further Insights from Ant Colonies? Bio-Inspired Computing and Communication. 58-66.