A comparison of the A\*, Ant Colony and composite search algorithms in solving the shortest path problem

By James A. Wasson

Of Southern Illinois University

March 2017

Abstract

A\* (A-star) is a well-known, heuristic, discrete, path finding algorithm that uses nodes in a graph to find the shortest distance between the starting and ending node. It is an improvement upon Dijkstra's algorithm, which was first introduced in 1956 and is the basis for most routing protocols that were in use for many years. A relatively new algorithm, Ant Colony, is based upon the movement of ants regarding the pheromone trails of previous ‘ants’ was introduced in 1992. It provides a metaheuristic, swarm type approach to finding the shortest path between two nodes. The report describes the difference between these algorithms as well as the nature of a composite A\*/Ant Colony Algorithm.

Keywords: A Star, A\*, Path Finding, Dijkstra's algorithm, Ant Colony, composite A\*/Ant Colony

A comparison of the A\*, Ant Colony and composite search algorithms in solving the shortest path problem

# Introduction

In this report, we will be discussing about A\*, it’s history, uses, and application. We will also do a quick overview of Ant Colony, a newer pathfinding algorithm. Then we will compare these algorithm’s in terms of speed, accuracy, recent improvements, and ease of implementation. After which we will discuss some of the benefits of an A\*/Ant Colony composite, and compare it with what we know about A\* and Ant Colony separately.

# A\*(A star) Overview

A\* is one of the most widely used pathfinding algorithms. Its predecessor, A1 was developed in the 1960’s as a branch of Dijkstra’s Algorithm in combination with a Greedy approach. After a few short years, A1 was developed into A2, which was then proved to be an optimal solution for discrete pathfinding problems and the term A\* was coined as stated by Harika Reddy.

## How A\* Works.

A\* uses a heuristic estimative approach to solving the shortest path problem, which means that it is not guaranteed to be optimal in all situations but it is good enough for most uses (Reddy 11). A\* uses a pointed approach to finding the shortest path, where Dijkstra’s Algorithm can find the shortest path to all nodes in a system, A\* finds the shortest path only to one as explained by Amit Patel. It is important to note however that for A\* to be a good approximation of the shortest path we need to know the estimated direction and distance of the node you wish to find. It uses this information to find the least cost path, using its heuristic algorithm to determine what the next lowest cost node is (Reddy 7). The Better the estimates, the closer to optimal the solution becomes.

## A\* Applications.

A\* can be used for a wide range of pathfinding problems, including edge finding problems, travelling salesman problem, also one major application of A\* is Open Shortest Path First which is implemented in internet trafficking (Reddy 12). It is also useful in natural language processing, which just shows how diverse of an algorithm A\* is (Klein). It’s uses can be expanded with minor effort and because it is such a simple nature of the algorithm these expansions can apply to an expansive problem set.

## Recent Improvements on A\*.

The number of recent improvements on A\* is quite small because it has been proven to be the best admissible case in the shortest path problem. There are, however, several off-shoot Algorithms that make use of weighting, bound relaxation, and using more than one heuristic algorithm at different points which give it dynamicity (Sun 105, 106). The most common of all changes is the weighting of A\*; effectively decreasing the weight creates a broader search field which allows for us to solve much more complex problems at a greater speed than with the standard A\*, but results in a loss of overall accuracy (Sun 109).

# Ant Colony Overview

Ant Colony is a meta-heuristic search algorithm that was based upon a concept of how ants locating food using pheromone trails. The Ant System, as it was called, was published in 1996 by Marco Dorigo as his doctorate thesis. The basis for the algorithm is that each ant acts individual of the others and implements its own heuristic algorithm which in turn impacts other ants.

## How Ant Colony Works.

Ant colony uses a swarm intelligence system which interprets pheromone trail’s, as stated in the previous paragraph, laid buy individual ants as its main method for solving the path finding problem (Zhao 1). It works on the principal of pheromone evaporation, over time the pheromones become less potent. Each ant can smell these pheromone levels; the higher the level of pheromones on a specific trail, the higher the chance of the ant to choose that trail. It is vital however that pheromone evaporation works in such a way as to not cause the ants to get stuck in a loop. For example, take a starting and ending node on a graph, where all the ants are released at random. After each ant chooses its path it will eventually make its way through all nodes until it reaches the ending node, then it tries to make its way back to the starting node. This should produce an admissibly short path (Dynamic Fuzzy Logic-Ant Colony…).

## Ant Colony Applications.

Ant Colony has a wide range of uses from the travelling sales man problem and shortest path problem to image processing and protein folding (Blum). This system has such a wide range of uses is due partly because of how the ant heuristic algorithm can be tweaked to allow for more flexibility, which in turn allows the user to implement the algorithm on very different and difficult problems. However, is should be noted that Ant Colony is ideal for the shortest path problem, which was its original use (Colorni).

## Recent Improvements on Ant Colony.

Several improvements have recently been made on the base Ant Colony System, most of which either involve changing the pheromone trails or increasing the sensitivity of the heuristic algorithms of the ants. One of these improved algorithms is the Min-Max Ant Colony which has a set cap on the amount of pheromone in a set area, and when it reaches that level it updates the cap on the pheromone level and saves the “best so far” path. This is a way to build parts of the shortest path which can be complied together later (Blum). The Min-Max Ant Colony is one of the most successful branches of Ant Colony and it experiences widespread use (Blum).

# Comparison of A\* and Ant Colony

In the previous paragraphs, we’ve seen what A\* and Ant Colony algorithms are, how they work, their uses, and some recent improvements. This information will carry over to the coming section in which we will discuss similarities and differences between A\* and Ant Colony in the areas of speed, future possibility for improvement, accuracy, and ease of implementation. From this comparison, we will try to draw a notable result in respect to either A\* and Ant Colony.

## Speed of path finding A\* compared to Ant Colony.

A\* is consider to be one of the fastest path finding algorithms that are admissible, as Harika Reddy wrote in his report Path Finding - Dijkstra’s and A\* Algorithm’s,

*“uniform-cost search and pure heuristic*

*search to effectively compute optimal solutions.”*

Ant Colony on the other hand is having difficulty finding a good algorithmic process to speed up path finding (Santoso). One study showed that for the travelling sales man problem; which is like the shortest path problem; gave the times to find the shortest path on a graph of 30 nodes as A\*: 4 milliseconds – Ant Colony: 10 milliseconds. If the nodes are increased to 100 the time for A\*: 80 milliseconds – Ant Colony: Undefined, meaning it couldn’t come to a solution in a reasonable amount of time.

**Result:** Based on the information presented directly above we can conclude that A\* is the faster shortest pathfinding algorithm.

## Recent Improvements in A\* compared to Ant Colony.

A\* is a simple algorithm that has existed for a long time. With that said, it is difficult to find recent advancements in the A\* algorithm. Which either indicates that it has lost its appeal for study or that it is at its best possible solution. Whereas Any Colony is new and being applied to a multitude of new fields and problems. While Ant Colony slightly more complicated, it is still easily understandable and usable in a variety of unique applications.

**Result:** The above leads up to conclude that Ant Colony will continue to grow and improve where A\* may stay static.

## Ease of Implementation between A\* and Ant Colony.

A\* uses weighted graphs and an array of nodes it is currently examining, this can cause issues with memory space, but overall it uses simple ideas to create a very optimized algorithm (Reddy). Ant Colony on the other hand is slightly more complicated, because it implements a meta heuristic algorithm. Which means that each ant implements its own algorithm, and in comparison, A\* only implements one (Colorni 2).

**Result:** A\* is easier to implement, but that doesn’t mean it is easier to implement for all cases, Ant Colony can provide a lot more flew ability in implementation.

## Accuracy of A\* contrasting Ant Colony.

Ant Colony can produce a perfect response the shortest path problem, but the more complicated the graph given the higher the processing time to achieve the shortest path. In contrast, A\* can produce a given result in O(log(x)) time, which is a very surprising result (Santoso 1-3). However, Ant Colony doesn’t need estimators to find the shortest path which in some ways puts it a step above A\*.

**Result:** So, while both can produce a near perfect response, A\* is much faster and is more stable than Ant Colony (Santoso 8).

# A\*/Ant Colony Composite

In this section, we will discuss the possible benefits of a composite A\* and Ant Colony denoted by A\*/AC. Much like A\* and how it combined the greedy approach with Dijkstra’s algorithm, we will see if we can combine the best aspects of A\* and Ant Colony. How this composite works is by A\* increasing the amount of pheromone released when an ant takes a route that is pointed towards the end goal, thus increasing the efficiency of the ant system (Salehinejad 3).

## Improvements of A\*/AC over Ant Colony.

While A\*/AC keeps the same structure as Ant Colony, by using the A\* algorithm as a pointed approach to finding the best pathway. You can see that (Figure 1) A\*/AC has a lower cost over traditional Ant Colony, giving it some of the speed and accuracy afforded by A\*. It reaches a stable system faster than the traditional Ant Colony System outlined by Salehinejad in his article Combined A\*-Ants Algorithm: A New Multi-Parameter Vehicle Navigation Scheme (3).

## Improvements of A\*/AC over A\*.

A\* is considered a fast and safe approach to solving the shortest path problem. So, what are the possible benefits can be gained by using A\*/AC? Well the biggest benefit is the diversity of Ant Colony. A\* runs at fast speeds, but lacks the ability to be applied directly to every complex problem, whereas Ant Colony can tackle complex issues such as protein folding, as discussed above. A\*/AC combines both the speed of A\* with the ability to solve complex, multifaceted issues of Ant Colony.

# Conclusion

## Summary.

A\* and Ant Colony are both pathfinding algorithms that use different means to reach the same goal. A\* is much faster, and much easier to implement than its counterpart Ant Colony. However, Ant Colony in some cases can be easier to implement, and has a much more diverse problem base than A\*. While A\* and Ant Colony are both accurate, it comes down to the speed at which that accuracy can be implement, and so A\* comes ahead as the faster.

## Results.

Based upon the research in the above sections, we can conclude, that if possible you should implement A\*; as it is the faster at producing an admissible result. If A\* is impossible to implement, you should try to implement an A\*/AC Composite, as it keeps its mailability while remaining fast and accurate.

Works Cited

Blum, Christian. "Ant Colony Optimization: Introduction and Recent Trends." Physics of Life Reviews 2.4 (2005): 353-73. Web. 30 Apr. 2017.

Colorni, Alberto, Marco Dorigo, and Vittorio Maniezzo. "An investigation of some properties of an "Ant algorithm"." Proceedings of the Parallel Problem Solving from Nature Conference (1992): n. pag. Web. 30 Apr. 2017.

Dorigo, Marco , V. Maniezzo, and A. Colorni. "Ant System: Optimization by a Colony of Cooperating Agents." IEEE (1996): n. pag. Web. 30 Apr. 2017.

Dramski, Mariusz. “A comparison between Dijkstra algorithm and simplified ant colony optimization in navigation”, Scientific Journals Maritime University of Szczecin, 2012. Web. 30 Apr. 2017.

Klein, Dan, and Christopher D. Manning. "A\* Parsing: Fast Exact Viterbi Parse Selection."Proceedings of HLT-NAACL (203): n. pag. Web. 30 Apr. 2017.

Patel, Amit. "Introduction to A\*   from Red Blob Games." Introduction to A\*. N.p., June 2016. Web. 30 Apr. 2017.

Ramteke, Shraddha, and Deepty Dubey, Mrs. "Performance Analysis of A\*Algorithm and Ant Colony Optimization Algorithm for Solving Traveling Salesman Problem." International Journal on Recent and Innovation Trends in Computing and Communication 3.2 (2015 02): n. pag. Web. 30 Apr. 2017.

Reddy, Harika. "Path Finding - Dijkstra’s and A\* Algorithm’s." (n.d.): n. pag.http://cs.indstate.edu. 13 Dec. 2013. Web. 30 Apr. 2017.

Sachithraj T., Piruthiviraj P., and Preeta Sharan. "Comparison of Dijkstra Algorithm With Ant Colony Optimization Algorithm Using Random Topology In All-Optical Network Using Rwa." International Journal of Research in Engineering and Technology 04.17 (2015): 66-69. Web. 30 Apr. 2017.

Salehinejad, Hojjat, Hossein Nezamabadi-pour, Saeid Saryazdi and Fereydoun Farrahi Moghaddam. “Combined A\*-Ants Algorithm: A New Multi-Parameter Vehicle Navigation Scheme”, Iranian Conference on Electrical Engineering, 2008. Web. 30 Apr. 2017.

Salehinejad, Hojjat, Siamak Talebi. "Dynamic Fuzzy Logic-Ant Colony System-Based Route Selection System." Applied Computational Intelligence and Soft Computing 2010 (2010): 1-13. Web. 30 Apr. 2017.

Santoso, Leo Willyanto, Alexander Setiawan, and Andre K. Prajogo. "Performance Analysis of Dijkstra, A\* and Ant Algorithm for Finding Optimal Path Case Study: Surabaya City Map." Makassar International Conference on Electrical Engineering and Informatics (2010): n. pag. Web. 30 Apr. 2017.

Stützle, Thomas, Manuel López-Ibáñez, and Marco Dorigo. "A Concise Overview of Applications of Ant Colony Optimization." Wiley Online Library (2011): n. pag. Web. 30 Apr. 2017.

Sun, Dapeng, and Min Li. "Evaluation Function Optimization of A-Star Algorithm in Optimal Path Selection." Revista Tecnica De La Facultad De Ingenieria Universidad Del Zulia (2016): n. pag. Web. 30 Apr. 2017.

Zhao, Wei, Xingsheng Cai, and Ying Lan. "A New Ant Colony Algorithm for Solving Traveling Salesman Problem." 2012 International Conference on Computer Science and Electronics Engineering (2012): n. pag. Web. 30 Apr. 2017.

Figures

Figure 1 (Salehinejad 6)

