Alex Pendell

CIS 421 – Artificial Intelligence

Assignment 4 – ACO

November 8th, 2019

**I. CHANGING THE VARIABLES**

The three variables used in the assignment were ALPHA, BETA, and RHO; representing the heuristic weight, pheromone weight, and trail persistence.

Overall, altering any of these variables did not seem to change much at all. It is suspected that this has to do with my algorithm seemingly failing to converge at an acceptable rate.

**II. COMPARISON TO A\***

The fundamental difference between A\* and ACO can be seen in the results (especially mine). That is, it’s plain to see that ACO is a probabilistic search. The ants will sometimes take paths that aren’t necessarily optimal, but that’s the point of probabilistic searches.

However, it’s important for these ants to sometimes make “bad decisions”. This is what keeps the algorithm from getting locked into a certain path and getting stuck at a local maximum. Since The algorithm has not meta-information about the path it’s taking, encouraging exploration is critical.

The ACO algorithm encourages exploration via an evaporation function. Once the ants have chosen their solutions, they lay pheromone on the paths they’ve taken to mark them as successful so that other ants may be coaxed into taking similar paths. The evaporation function “evaporates” some of the pheromone on the path so that the ants don’t get railroaded into one successful path.

The evaporation can take place before or after the pheromone has been laid. By the nature of the evaporation function (it evaporates higher pheromone edges more aggressively), evaporating the pheromone *after* the pheromone has been laid promotes more exploration while evaporating *before* promotes convergence time.

Comparing to A\*, the ACO algorithm seems inefficient in a plethora of areas. Outside the technicalities of the algorithm, the complexity of the ACO algorithm is much higher than that of A\* in terms of implementation and programming. This only adds room for error.

**III. WHICH TO PICK?**

The nice thing about ACO is that it requires almost zero meta data about the environment it’s operating in. Whereas with A\*, it needed a multitude of measurements in order to operate optimally.

Furthermore, I theorize that ACO operates best under dynamic environments. Since we are releasing wave after wave of ants with each one generating their own solutions, if something happens to the environment in between cycles, the ACO will find new optimal paths to its solution. Looking over the A\* algorithm once again, I’m not convinced that A\* could even detect changes in its environment without a serious overhaul; when choosing the next node, it has to trust that the state of the environment has not changed, otherwise all its heuristic values would be moot, and would need to be recalculated.

Finally, I think ACO would out-perform A\* in the case of multiple unknown destinations. ACO makes no assumptions about the locations of its goal, so it will wander probabilistically until a goal is reached, and then the ants will favor the trail to that goal. However, since it’s probabilistic, the discovery of new goals is still possible, and encouraged through the evaporation function.

**IV. CONCLUSION**

In conclusion, my thoughts are that while ACO is a much more difficult algorithm to grasp and implement, the tradeoff is that it seems more widely applicable as far as environments are concerned. If you have access to a lot of information and can make decisions about path finding using this information, A\* will probably be a better choice due to ease of implementation and completeness. But if you can make almost zero assumptions about the environment you’re operating in, ACO seems to be a safer choice. This coupled with its self-adaptability makes ACO a very, very charming pathfinding algorithm.