Evolved Response Thresholds Generalize Across Problem Instances for a Deterministic-Response Multiagent System

H. David Mathias
University of Wisconsin - La Crosse
La Crosse, Wisconsin, USA
dmathias@uwlax.edu

Annie S. Wu
University of Central Florida
Orlando, Florida, USA
aswu@cs.ucf.edu

Daniel Dang
Whitman College
Walla Walla, Washington, USA
dangd@whitman.edu

ABSTRACT

In this work, we use a multiobjective genetic algorithm to evolve agent response thresholds for a decentralized swarm and demonstrate that swarms with evolved thresholds outperform swarms with thresholds set using other methods. In addition, we provide evidence that the effectiveness of evolved thresholds is due in part to the evolutionary process being able to find, not just good distributions of thresholds for a given task across all agents, but also good combinations of thresholds over all tasks for individual agents. Finally, we show that thresholds evolved for some problem instances can effectively generalize to other problem instances with very different task demands.

CCS CONCEPTS

• **Computing methodologies** → **Multi-agent systems**; **Genetic algorithms**.

ACM Reference Format:
H. David Mathias, Annie S. Wu, and Daniel Dang. 2021. Evolved Response Thresholds Generalize Across Problem Instances for a Deterministic-Response Multiagent System. In *2021 Genetic and Evolutionary Computation Conference Companion (GECCO '21 Companion), July 10–14, 2021, Lille, France.* ACM, New York, NY, USA, 2 pages. https://doi.org/10.1145/3449726.3459522

Swarms are large scale decentralized multi-agent systems in which the component agents work collectively towards one or more common goals. Division of labor in the absence of centralized control of individual behaviors is a difficult problem and response thresholds are one approach that is successful in achieving decentralized coordination. In the response threshold model, individuals sense external stimuli and act if the stimulus exceeds a threshold value. Inter-individual variation, differences in when and how individuals respond to task demands, is an important mechanism for effective division of labor. In particular, variability in response thresholds serves to desynchronize activations by individuals, preventing the swarm from responding in lockstep.

Previous work finds that, when agent thresholds are static, a basic uniform distribution of threshold values provides the best goal achievement. Dynamic thresholds allow swarms to tailor threshold distributions to specific problem demands but have, in practice, been shown to have difficulty re-adapting to new task demands once converged. We explore *a priori* evolution of thresholds rather than real-time adaptation. Genetic algorithms (GAs) are known for their ability to find effective solutions to computationally expensive, high-dimensional problems. We use a multi-objective genetic algorithm to evolve agent response thresholds for a decentralized swarm.

Our testbed is a collective control problem in which a swarm collectively tracks the movement of a target in a 2-D plane [**?**]. Each agent in the swarm selects to push in one of the four compass directions based on its observation of the positional difference between the target and the tracker. Tracker movement is generated by aggregating the actions of all active agents. In each timestep, the position deviation in each direction represents a task demand, and the tracker movement generated by agent actions represents agent response to task demand. Performance is evaluated according to three goals: (1) minimize the average position difference, per time step, between the target location and the tracker location, (2) minimize the difference between total distance traveled by target and the total distance traveled by the tracker, and (3) minimize the average number of task switches experienced by agents.

Our genetic algorithm is based on NSGA-II and uses three optimization objectives, each aligned with one of the criteria above. Each individual in the GA population represents a complete swarm consisting of 50 agents. The genome consists of 200 real-valued numbers, 4 for each of the 50 agents represented by an individual. The values for an agent represent the 4 task thresholds for that agent, each a real value in $[0, 1]$. Thus, in aggregate the genome is: $[\theta_{i,\mathrm{D}}]$, $\forall\ \mathrm{D} \in \{\texttt{NORTH, EAST, SOUTH, WEST}\}$ and $\forall\ i \in [0, 49]$. Individuals are initialized with random values generated uniformly in $[0, 1]$. Uniform crossover exchanges threshold values and mutation alters each threshold value with a fixed probability. The fitness of a GA individual is determined by running the tracking simulation using a swarm with agents set to the encoded threshold values. For deterministic paths, evaluation consists of a single simulation while for random paths we run three simulations and average the results.

We evolve and test response thresholds for six target paths: `circle`, `diamond`, `random`, `s-curve`, `square`, and `zigzag` [**?**]. These paths offer a wide range of balance of task demands and frequency and magnitude of changes in task demands. For each target path, we perform 32 runs of the genetic algorithm and choose one individual from front 0 of each run. We use these individuals for testing on the path used for evolution as well as all other paths. For each test, we perform 30 runs of the simulation, averaging position difference, path length difference and number of task switches across the runs.

1 RESULTS

Effect of Evolved Thresholds: Figure 1 compares the average observed position difference and the average task switches across

GECCO '21 Companion, July 10–14, 2021, Lille, France

ACM ISBN 978-1-4503-8351-6/21/07.
https://doi.org/10.1145/3449726.3459522

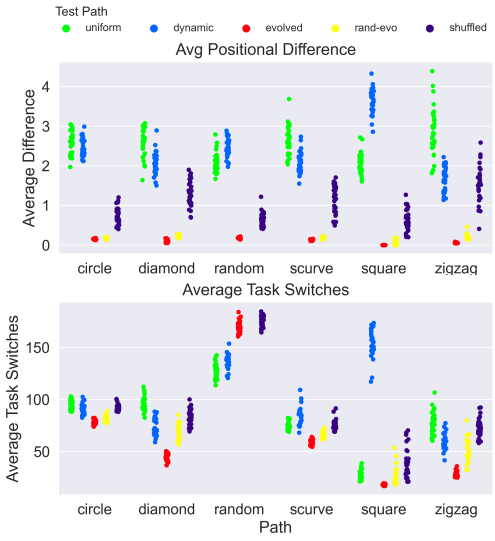

Figure 1: Comparison of uniform, dynamic, and evolved response threshold performance by path.

different threshold types on the paths tested. The x-axis consists of six groups, one for each target path. Within each group, we show data points for uniform thresholds, dynamic thresholds, thresholds evolved for that target path, thresholds evolved for `random` paths but tested on the target path, and thresholds that are evolved then shuffled among agents. In the group for `random`, we omit the fourth data since they duplicate the third. The y-axis represents average positional difference. Each data point represents 30 runs of the simulation for one of the 32 runs of the genetic algorithm. Results for tracker path length are similar.

The top plot shows that evolved thresholds significantly outperform both uniform thresholds and dynamic thresholds. Dynamism is beneficial for some paths but not for others. This may be due to the effects of the positive feedback loop on dynamic thresholds, causing them to migrate to sink states. One possible advantage of dynamic over evolved thresholds is that they adapt in real-time to any path, perhaps making them more general. The rand-evo data demonstrate that this is not the case. Using thresholds evolved for `random` we find excellent performance for all paths suggesting that evolved thresholds can generalize.

To gauge the importance of the relative threshold values each agent possesses, we randomly shuffle the evolved threshold values for all agents, maintaining the distribution of thesholds within each task but disrupting the relative values across tasks for each agent. The performance of the `shuffled` swarm is noticeably worse than the evolved swarm but still better than a static uniform threshold distribution. These results suggest that the relative threshold values possessed by each agent are relevant and that specialization and division of labor may be more subtle than individual agents simply favoring one single task.

The bottom plot shows average task switches per agent for 30 runs with each of the 32 GA runs for all six paths. Improvement ranges from modest for `square`, for which specialization is unnecessary due to demand for only one task in any timestep, to substantial

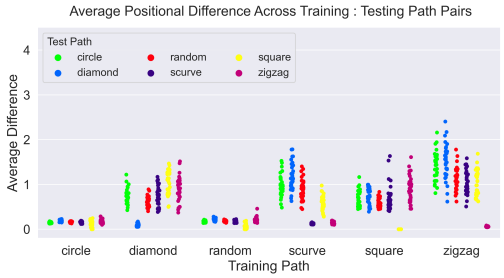

Figure 2: Generalization of thresholds evolved for six paths. `circle` and `random` provide the best generalization.

for `diamond` and `zigzag`, which share the properties of long periods of unchanging task demand and demand for multiple tasks in all timesteps. `random` is an outlier with an increase in task switches for evolved thresholds, which we cannot yet explain.

Generalizing across paths: Figure 2 depicts the degree of generalization for evolved thresholds; how well do swarms evolved for one problem perform on a different problem? The x-axis indicates the problem on which a swarm is trained and the y-axis indicates the average positional difference. Within each training path, we show the data points for each test path in a different color. These results show that `circle` and `random` generalize well to all other paths. By comparison, the other paths do not generalize well, with the exception of `s-curve` generalizing well to `zigzag`. We note, however, that all of these generalization results are comparable or better than the performance of static uniformly distributed thresholds shown in the top plot of Figure 1. Closer examination reveals that thresholds evolved for fitness determined by a single revolution around a circular path is sufficient to generalize into good performance on all other paths tested, consistently producing average positional difference of less than 0.3.

These results suggest two necessary and sufficient features for a universal training instance: sufficient demand for all tasks, and a wide range of simultaneous demand levels to allow thresholds to evolve to reasonably address any balance of demand between multiple tasks. It is important to note that this does not require that the training instance includes every possible turn or curve that might be encountered in testing. A simple circle in which demand changes in the same way, from one timestep to the next throughout a run, is universal. There are no left turns, no sharp turns and no straight lines. The number of timesteps during which there is demand for only one task is extremely small as these occur only when the target and tracker are at the same x or y location but are not co-located. This result, provides hope that simple universal training instances may exist for other problem domains.

ACKNOWLEDGEMENTS

This work was supported by the National Science Foundation under Grant No. IIS1816777.

REFERENCES

[] Annie S. Wu, H. David Mathias, Joseph P. Giordano, and Arjun Pherwani. 2021. *Collective control as a decentralized task allocation testbed.* Technical Report CS-TR-21-01. University of Central Florida.