A Quality Diversity Approach to Automatically Generate Multi-Agent Path Finding Benchmark Maps (Extended Abstract)

Cheng Qian, Yulun Zhang, Jiaoyang Li

Robotics Institute, Carnegie Mellon University
chengqia@andrew.cmu.edu, {yulunzhang, jiaoyangli}@cmu.edu

Introduction

The recent advancements in Multi-Agent Path Finding (MAPF) have facilitated the implementation of multi-robot systems that can manage hundreds of robots. While studying MAPF algorithms, much of the focus has been on developing novel planners that coordinate the agents smoothly. Typically, these planners are tested on fixed or human-designed benchmark maps (Stern et al. 2019). However, such fixed benchmark maps have several problems. First, these maps may not cover all the potential failure modes of the planners, making the benchmarking insufficient. Second, these maps may favor certain categories of planners, making the comparison of different planners unfair. Third, despite the large number of benchmark maps available, researchers are less pruned to test their planners on all available maps for intractable computational cost.

Meanwhile, a recent work (Zhang et al. 2023b) has used a map generation technique based on quality diversity (QD) algorithms to improve the efficiency of multi-robot systems by optimizing the layout of the maps. Different from single-objective optimization algorithms, QD algorithms generate a collection of high-quality solutions in an *archive* by optimizing a given objective function while diversifying a set of given diversity measure functions. The archive is a discretized measure space storing high-quality solutions in each individual cell. Another work (Zhang et al. 2023a) further uses neural cellular automata (NCA) to generate maps with patterns. NCA applies convolutional neural networks (CNN) to represent the rules of cellular automata and iteratively update each grid cell based on the state of its neighbors. In this paper, we intend to adapt the same map generation technique to MAPF algorithms with an alternative goal of automatically generating diverse and targeted benchmarking maps for different MAPF planners. Our preliminary results on EECBS (Li, Ruml, and Koenig 2021), a state-of-the-art bounded-suboptimal MAPF algorithm, exhibit some interesting results, including the divergence in behavior between two maps with similar structures and common patterns that present challenges for EECBS.

Approach

In our map generation technique, we evenly discretize the measure space (i.e., the space includes all measure pairs) into 6,400 cells. For each cell, we aim to find the best map which maximizes our objective function. We use the QD algorithm CMA-ME (Fontaine et al. 2020) to search for and update a diverse collection of NCA generators with the objective and diversity measures computed from EECBS to generate diverse maps with patterns.

We start by sampling a batch of 100 parameter vectors from a multivariate Gaussian distribution with mean 0 and sigma 0.2, which forms 100 NCA generators. Each NCA generator has 3 convolutional layers with kernel size 3×3 and 1730 parameters. Each NCA generator will generate one map based on a fixed seed map, leading to 100 maps. After repairing maps to enforce connectivity using a Mixed-Integer Linear Programming solver, we evaluate each map by running EECBS 5 times, each with randomly generated start and goal locations of 150 agents, a given cutoff time of 5 seconds, and a suboptimality bound of 1.2. We calculate the average objective, namely CPU runtime, and measures, namely map entropy (Zhang et al. 2023a) and standard deviation of betweenness connectivity (BC) (Ewing et al. 2022). To compute the map entropy, we define a tile pattern as an arbitrary arrangement of a 2×2 grid, count the occurrence of the same tile patterns in the given map, and then normalize the counts to form a tile distribution. The map entropy is the entropy of the tile distribution. To compute the standard deviation of BC, we search for the shortest path between each possible start and goal location in the map and then calculate the standard deviation among usage of all tiles.

Preliminary Results

Our generated maps (shown in Figure 1) consist of empty spaces (white) and obstacles (black) with size 32×32 and exactly 20% obstacles. Figure 1 shows six generated maps, together with the tile-usages (the frequency of each tile being used in the EECBS solution) of two of them.

The two maps with tile-usage plots on the left ((a) and (b)) reveal the most interesting comparisons. Qualitatively, they have similar stylistic features in terms of the obstacle distribution, the shape of obstacle components, and the spread of empty space. Quantitatively, they demonstrate similar map entropy around 0.5 and similar standard deviation of BC

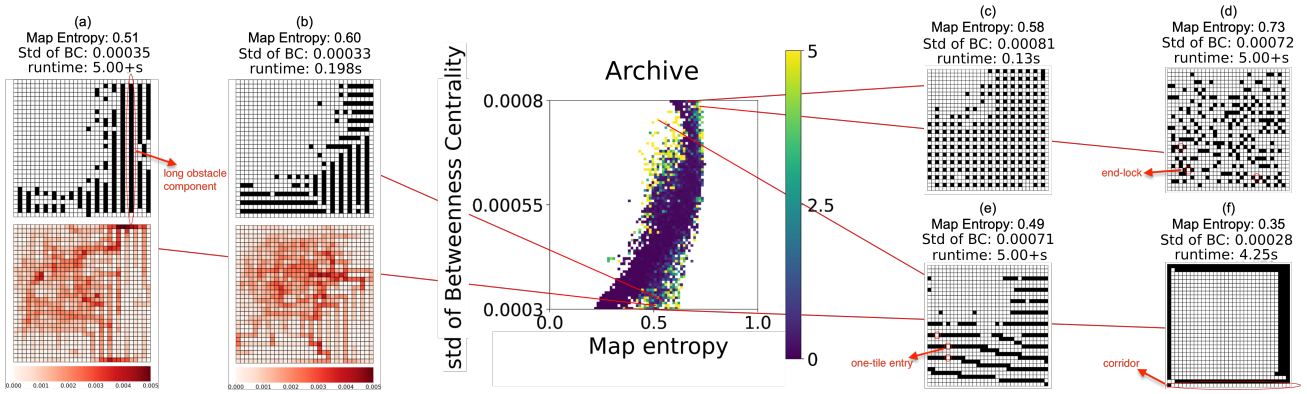

Figure 1: An archive of diverse generated maps with color indicating average runtime and axis indicating different values of the diversity measures along with the tile-usages.

around 0.0003. However, the behaviors of EECBS on these two maps are drastically different. Upon running EECBS 400 times on two maps, map (a) has a success rate of 43.75% with an average runtime of 3.2 seconds, while map (b) has a 92.00% success rate with an average runtime of 0.8 seconds. Moreover, by conducting a two-sample T-test with $\alpha = 0.05$ and $df = 798$, we observe a statistically significant difference in the average runtime of EECBS between map (a) and map (b) ($t > 1.963$). From the tile-usage plots of the two maps, we discover that for map (a), more traffic congestion is concentrated at the top entry of the long obstacle components. However, for map (b), congestion is evenly distributed around the primary entries of obstacle components. Therefore, we conjecture that while both maps have multiple corridors in parallel, shorter corridors in map (b) lead to more empty space within the interior of the map, causing less congestion and resulting in faster runtime.

Maps (c) and (d) are two maps with high standard deviations of BC and relatively high map entropy (close to or over 0.6). With a higher map entropy, map (d) manifests a more stochastic pattern of obstacles, leading to increased end-lock areas which cause much congestion. However, for map (c), with evenly distributed obstacles, there exists more evenly distributed empty space for agents to resolve collisions all over the map, leading to fast runtime for EECBS.

Map (e) is a map with high standard deviation of BC and relatively low map entropy (below 0.5). Despite its simple pattern, the presence of numerous one-tile entries and corridors leads to high runtime. The same effect applies to map (f). Its simple pattern leads to low map entropy. However, there exists a one-tile entry and a one-tile-wide long corridor which makes it the only map with high runtime among all maps with low standard deviation of BC and low map entropy. With one-tile entries and corridors, low standard deviation of BC and large portion of empty space cannot be seen as indications of the difficulty of maps.

Conclusion and Future Work

Experimental Findings. We found that with carefully designed objectives and diversity measures, it is feasible to combine the QD algorithm and NCA to generate diverse benchmark maps for MAPF algorithms. In our experiment, we observe significant divergence in the behavior of EECBS between two similar structured maps. This potentially indicates the impact of the length of the corridors on EECBS's runtime. Also, based on high runtime cases in low entropy scenarios, we verify that long corridors and one-tile entries are the main reasons for congestion.

Future Work. Our work serves as the preliminary results of automatically generating benchmark maps for MAPF, yielding many future directions. Firstly, we aim to generate a set of diverse benchmark maps that will sufficiently cover the failure modes of a given MAPF algorithm. Secondly, we seek to establish unbiased sets of benchmark maps with suitable objectives and measures to make fair comparisons between different MAPF algorithms. Thirdly, we will endeavor to limit the number of maps in the benchmark set to optimize the computational cost of benchmarking.

References

Ewing, E.; Ren, J.; Kansara, D.; Sathiyanarayanan, V.; and Ayanian, N. 2022. Betweenness Centrality in Multi-Agent Path Finding. In *AAMAS*, 400–408.

Fontaine, M. C.; Togelius, J.; Nikolaidis, S.; and Hoover, A. K. 2020. Covariance Matrix Adaptation for the Rapid Illumination of Behavior Space. In *GECCO*, 94–102.

Li, J.; Ruml, W.; and Koenig, S. 2021. EECBS: A Bounded-Suboptimal Search for Multi-Agent Path Finding. In *AAAI*, 12353–12362.

Stern, R.; Sturtevant, N. R.; Felner, A.; Koenig, S.; Ma, H.; Walker, T. T.; Li, J.; Atzmon, D.; Cohen, L.; Kumar, T. K. S.; Barták, R.; and Boyarski, E. 2019. Multi-Agent Pathfinding: Definitions, Variants, and Benchmarks. In *SoCS*, 151–159.

Zhang, Y.; Fontaine, M. C.; Bhatt, V.; Nikolaidis, S.; and Li, J. 2023a. Arbitrarily Scalable Environment Generators via Neural Cellular Automata. In *NeurIPS*, 57212–57225.

Zhang, Y.; Fontaine, M. C.; Bhatt, V.; Nikolaidis, S.; and Li, J. 2023b. Multi-Robot Coordination and Layout Design for Automated Warehousing. In *IJCAI*, 5503–5511.