# Rapid Randomized Restarts for Multi-Agent Path Finding: Preliminary Results

Extended Abstract

Liron Cohen
Sven Koenig
T. K. Satish Kumar
University of Southern California

Glenn Wagner
Howie Choset
Carnegie Mellon University

David Chan
Nathan Sturtevant
University of Denver

## ABSTRACT

Multi-Agent Path Finding (MAPF) is an NP-hard problem with many real-world applications. However, existing MAPF solvers are deterministic and perform poorly on MAPF instances where many agents interfere with each other in a small region of space. In this paper, we enhance MAPF solvers with randomization and observe that their runtimes can exhibit heavy-tailed distributions. This insight leads us to develop simple Rapid Randomized Restart (RRR) strategies with the intuition that multiple short runs will have a better chance of solving such MAPF instances than one long run with the same runtime limit. Our contribution is to show experimentally that the same RRR strategy indeed boosts the performance of two state-of-the-art MAPF solvers, namely M* and ECBS.

**ACM Reference Format:**
Liron Cohen Sven Koenig T. K. Satish Kumar, Glenn Wagner Howie Choset, and David Chan Nathan Sturtevant. 2018. Rapid Randomized Restarts for Multi-Agent Path Finding: Preliminary Results. In *Proc. of the 17th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2018), Stockholm, Sweden, July 10–15, 2018,* IFAAMAS, 3 pages.

## 1 INTRODUCTION AND BACKGROUND

The runtimes of search algorithms can exhibit heavy-tailed distributions [6]. These distributions have tails that decay according to a power law. Their means, variances or other higher moments may not be finite due to the large probability mass in their tails that allows outliers to have a substantial impact on their moments [4]. Heavy-tailed distributions in the runtimes of randomized search algorithms can be exploited with Rapid Randomized Restart (RRR) strategies [5]. RRR strategies generally exploit the property that, given hard instance of an NP-hard problem, multiple short runs have a better chance of solving it than one long run with the same runtime limit.

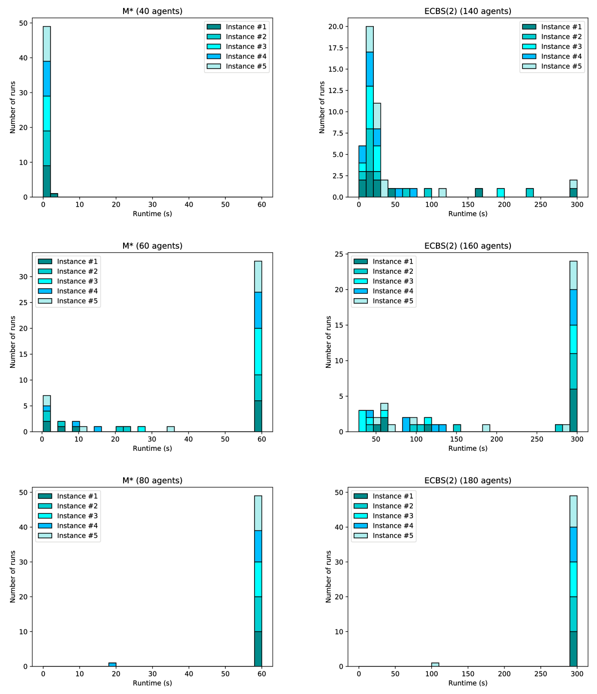
We apply this idea to the Multi-Agent Path Finding (MAPF) problem, which is defined as follows: Given a graph and a set of agents with unique start and goal vertices each, find collision-free paths for all agents from their respective start vertices to their respective goal vertices. The agents traverse edges in discrete time steps with the possibility of waiting at vertices. Minimizing the solution cost given by the sum of travel times of the agents along their paths is NP-hard [8]. We enhance two state-of-the-art search-based MAPF solvers, namely M* [7] and ECBS [1], by replacing some of the arbitrary, but deterministic, decisions that shape their search trees with random ones. Our contribution is to show experimentally that the distribution of runtimes can be heavy-tailed and that the same RRR strategy indeed boosts the performance of M* and ECBS.

Contact author: `lironcoh@usc.edu`

*Proc. of the 17th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2018), M. Dastani, G. Sukthankar, E. André, S. Koenig (eds.), July 10–15, 2018, Stockholm, Sweden.* © 2018 International Foundation for Autonomous Agents and Multiagent Systems (www.ifaamas.org). All rights reserved.

## 2 RANDOMIZED MAPF SOLVERS

M* is an optimal A*-based MAPF solver that uses subdimensional expansion to initially create a one-dimensional search space embedded in the joint configuration space of the multi-agent system. In



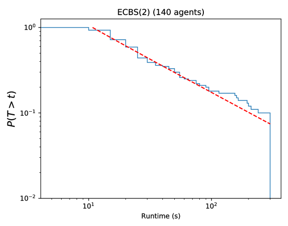


**Figure 1: Distribution of runtimes. Top three rows: Runtimes before, in and after the phase transition (see Section 5), respectively. Each of these figures shows the runtimes of 10 runs of a randomized solver for each of 5 MAPF instances. Right: Heavy-tailed distribution of runtimes for a 140 agent MAPF instance.**

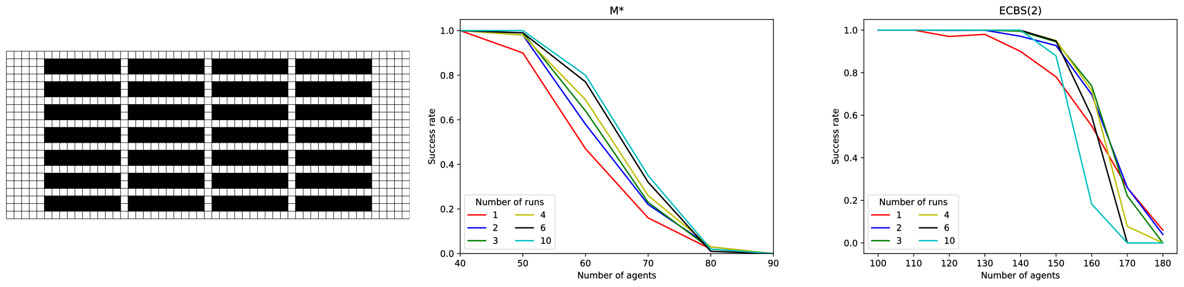


**Figure 2: Left: Automated warehousing domain. Center and Right: Success rates of RRR strategies for M* and ECBS(2).**

case of collisions, it increases the dimensionality of the search space locally to ensure that an alternative path can be found. The way in which M* breaks ties among search nodes with equal $g$-values has a significant effect on its runtime because one node may lead to a complete resolution of a collision while another node may either fail to fully resolve the collision or even lead to new collisions. The expansion order of nodes with equal $g$-values depends on the order in which they were inserted into the OPEN list which, in turn, depends on the labeling of the agents (that is, the order in which they are considered). Therefore, we randomized the labeling of the agents.

ECBS(2) is a bounded-suboptimal A*-based MAPF solver (that is, produces solutions whose costs are within a factor 2 of minimum) and is thus able to solve larger MAPF instances than optimal MAPF solvers like M*. It, too, avoids operating in the joint configuration space but does so using a two-level search. The low-level search plans paths for all agents from their respective start vertices to their respective goal vertices that satisfy a given set of constraints. The high-level search performs a search on a constraint tree whose nodes contains paths for all agents. It imposes more and more constraints on the paths during the search until the paths no longer result in collisions. The collisions in the root node of the constraint tree have a significant effect on the runtime of ECBS(2) because the shape and size of its constraint tree heavily depend on them. The collisions depend on the labeling of the agents since the low-level search plans paths for the agents in the order of their labeling and always tries to avoid collisions with agents whose paths it has already planned. Therefore, we again randomized the labeling of the agents.

## 3 EXPERIMENTAL SETUP

We performed experiments on a cluster of 38 Amazon EC2 c4.xlarge instances running on Intel CPUs E5-2666 v3 @ 2.90GHz with 4 VCPUs (2 physical cores) and 7.5GB RAM per instance. ECBS(2) used 2 workers per instance and a 5 minute runtime limit per run. M* used 1 worker per instance and a 1 minute runtime limit per run (due to the high memory consumption of our M* implementation). We averaged over 100 randomly generated MAPF instances in the automated warehousing domain from Figure 2 (left). We varied the number of agents in increments of 10. Half of the agents were assigned a random start vertex in the left open space and a random goal vertex in the right open space, and vice-versa for the other half of the agents. The resulting MAPF instances are generally considered to be a hard to solve for MAPF solvers because many agents interfere with each other in the narrow passageways [2].

## 4 DISTRIBUTION OF RUNTIMES

Figure 1 suggests that the resulting runtimes can indeed exhibit heavy-tailed distributions, as has been suggested before [3]. We validated experimentally that the distribution of runtimes is heavy-tailed by using the Pareto-Levy distribution, defined as:

$$P(T > t) = \begin{cases} 1 & \text{if } t < t_{min} \\ \left(\frac{t_{min}}{t}\right)^{\alpha} & \text{otherwise,} \end{cases}$$

where $t_{min} > 0$ is the minimum possible value of $T$ and $0 < \alpha < 2$. Figure 1 also shows the log-log plot of $P(T > t)$ (in blue) for a MAPF instance with 140 agents. Here, $T$ is a random variable representing the runtime of the MAPF solver. We computed the slope of the approximately linear decay to provide an estimate of $\alpha$. We used an external library to fit the data and show the fitted curve in red (corresponding to $\alpha = 1.771$ and $t_{min} = 10.759$s). The distribution is heavy-tailed since $\alpha < 2$. (In fact, its variance is not finite.)

## 5 SUCCESS RATES OF RAPID RANDOM RESTARTS

Figures 2 (center) and (right) report the success rates of M* and ECBS(2) (that is, the percentage of MAPF instances solved within the runtime limit) for different numbers of runs and increasing numbers of agents. Two runs, for example, mean that the MAPF solvers processed each instance twice, each time with half of the runtime limit. A sharp decline in the success rate is characteristic of a phase transition. MAPF instances on its left side require little coordination among the agents, which means that a MAPF solver can easily recover from bad decisions in the search process and likely solve them within the runtime limit. MAPF instances on its right side require a significant amount of coordination among the agents, which means that many of them are not solvable within the runtime limit. MAPF instances in the phase transition require a critical amount of coordination and can therefore serve as good test cases for comparing MAPF solvers.

Our RRR strategy generally boosts the success rates of both MAPF solvers. However, simply increasing the number of runs does not always increase their success rates since they cannot find solutions arbitrarily fast and the runtime limit thus cannot be arbitrarily short. Smaller numbers of runs can result in higher success rates for larger numbers of agents.

**Acknowledgements:** The research at the University of Southern California was supported by the National Science Foundation (NSF) under grant numbers 1724392, 1409987 and 1319966 as well as a gift from Amazon. The research at Carnegie Mellon University was supported by a gift from Foxconn. The research at the University of Denver was supported in part by NSF IUCRC grant number 1439693.

## REFERENCES

[1] Max Barer, Guni Sharon, Roni Stern, and Ariel Felner. 2014. Suboptimal Variants of the Conflict-Based Search Algorithm for the Multi-Agent Pathfinding Problem. In *Proceedings of the 7th Annual Symposium on Combinatorial Search*.

[2] Liron Cohen, Tansel Uras, and Sven Koenig. 2015. Feasibility Study: Using Highways for Bounded-Suboptimal Multi-Agent Path Finding. In *Proceedings of the 8th Annual Symposium on Combinatorial Search*.

[3] Liron Cohen, Tansel Uras, Satish Kumar, Hong Xu, Nora Ayanian, and Sven Koenig. 2016. Improved Solvers for Bounded-Suboptimal Multi-Agent Path Finding. In *Proceedings of the 6th International Joint Conference on Artificial Intelligence*.

[4] Sergey Foss, Dmitry Korshunov, and Stan Zachary. 2013. *An Introduction to Heavy-Tailed and Subexponential Distributions* (2nd ed.). Springer.

[5] Carla P. Gomes, Bart Selman, Nuno Crato, and Henry Kautz. 2000. Heavy-Tailed Phenomena in Satisfiability and Constraint Satisfaction Problems. *Journal of Automated Reasoning* 24, 1-2 (2000), 67–100.

[6] Richard A. Valenzano, Nathan R. Sturtevant, Jonathan Schaeffer, Karen Buro, and Akihiro Kishimoto. 2010. Simultaneously Searching with Multiple Settings: An Alternative to Parameter Tuning for Suboptimal Single-Agent Search Algorithms. In *Proceedings of the 20th International Conference on Automated Planning and Scheduling*.

[7] Glenn Wagner. 2015. *Subdimensional Expansion: A Framework for Computationally Tractable Multirobot Path Planning*. Ph.D. Dissertation. Carnegie Mellon University.

[8] Jingjin Yu and Steven M. LaValle. 2013. Structure and Intractability of Optimal Multi-Robot Path Planning on Graphs. In *Proceedings of the 27th AAAI Conference on Artificial Intelligence*.