Multi-agent Path Planning based on MA-RRT* Fixed Nodes

Extended Abstract

Jinmingwu Jiang
Chongqing University, College of Computer Science
Chongqing, China
jiangjinmingwu@cqu.edu.cn

Kaigui Wu
Chongqing University, College of Computer Science
Chongqing, China
kaiguiwu@cqu.edu.cn

ABSTRACT

In cooperative pathfinding problems, no-conflicts paths that bring several agents from their start location to their destination need to be planned. This problem can be efficiently solved by Multi-agent RRT*(MA-RRT*) algorithm. However, the implementation of this algorithm is hindered in systems with limited memory because the number of nodes in the tree grows indefinitely as the paths get optimized. This paper proposes an improved version of MA-RRT*, called Multi-agent RRT* Fixed Nodes(MA-RRT*FN), which limits the number of nodes stored in the tree by removing the weak nodes which are not likely to reach the goal. The results show that MA-RRT*FN performs close to MA-RRT* in terms of scalability and solution quality while the memory required is much lower and fixed.

KEYWORDS

Multi-agent motion planning; Cooperative pathfinding; Collision avoidance; Path planning

ACM Reference Format:

Jinmingwu Jiang and Kaigui Wu. 2020. Multi-agent Path Planning based on MA-RRT* Fixed Nodes. In *Proc. of the 19th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2020), Auckland, New Zealand, May 9–13, 2020,* IFAAMAS, 3 pages.

1 INTRODUCTION

The problem of planning a series of routes for mobile robots to destinations and avoiding collisions can be modeled as a *cooperative pathfinding* problem. Traditionally, this problem is often simulated in highly organized environments such as grids, which include several obstacles and agents. The straightforward method to find the paths of these agents is looking for the answer in a joint configuration space which is composed of the state spaces of single agents. Such a space is typically searched using a heuristic guided function such as A*[4]. However, the problem of cooperative pathfinding is proved to be PSAPCE-hard[5].

So far, a lot of work has been proposed to solve the multi-agent path-planning problems, such as Local Repair A*(LRA*)[11] and Optimal Anytime(OA) [12], and the recently work conflict based search(CBS) [10] and its improved version [2]. There are also many attempts in using the sampling-based algorithm, such as RRT[3][6][7][8]. In [15], Čáp marries RRT* to the classical multi-agent motion-planning algorithm and proposes Multi-agent RRT* (MA-RRT*). Recently, the work [9] and [14] propose a decentralized version of MA-RRT*, which run RRT* for each agent in a pre-specified order to plan a path. However, unlike MA-RRT*, The decoupled algorithms proposed by [9] and [14] are not complete nor optimal. So far, in the field of coupled algorithms, no works outperform MA-RRT* considering the speed, optimality, completeness, and flexibility at the same time.

However, while MA-RRT* can solve the multi-agent path planning problem efficiently, the application of the MA-RRT* is hindered in embedded systems with limited memory, because as the solution gets optimized, the number of nodes in the tree grows indefinitely. The closest work to this problem is the RRT* Fixed Nodes(RRT*FN) proposed by Adiyatov[1], which only focuses on improving the memory efficiency of RRT*. Up to now, there is no prior work which limits the memory required for the MA-RRT*.

This work presents a new MA-RRT* based algorithm, the Multi-agent RRT* Fixed Nodes(MA-RRT*FN), shown in algorithm 1, which works by employing a node removal procedure to limit the maximum number of nodes in the tree, performs close to MA-RRT* in terms of scalability and solution quality while the memory required is much lower and fixed.

2 PROBLEM FORMULATION

To make a fair comparison with MA-RRT*, which is simulated on graphs, the paper tests both the two algorithms in a four-connected grid world and uses the following definition. Assuming that n agents are running on a Euclidean space, and each agent, which takes up a single cell of the grid world, has a unique start location and destination. For each timestep, all agents can move to its four neighbor cells if it is free or stay on its current location[13]. A cell is free means that it will not be occupied by an agent at the end of the timestep and does not include an obstacle. The total number of timesteps that the agent takes from its start state to the goal location is regarded as the cost of the individual agent's path. If all the agents can reach their goal without collision, then the sum of each path cost is taken as the cost of the final solution, which is the metric of solution quality.

3 THE ALGORITHM

The MA-RRT*FN initially grows the tree before the maximum number of nodes **M** is attained, after which the MA-RRT*FN removes a node that has one or no child in the tree before adding a new node. The first attempt to remove the node is during the EXTEND procedure, in which the algorithm updates the cost of nodes near the newly added node x_{new}. If a node x_{near} from X_{near} can reach a lower cost to the initial state by reconnecting to the newly added node, then the algorithm will check whether the par-

Proc. of the 19th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2020), B. An, N. Yorke-Smith, A. El Fallah Seghrouchni, G. Sukthankar (eds.), May 9–13, 2020, Auckland, New Zealand.

Algorithm 1 MA-RRT*FN

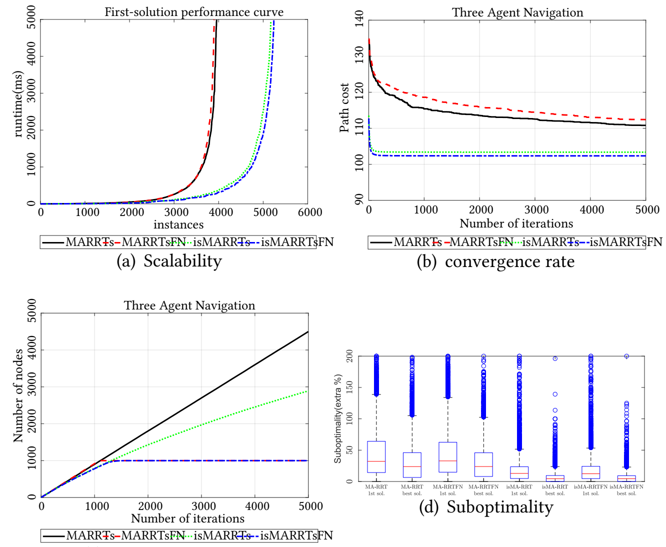
$V \leftarrow \{x_{init}\}$; $E \leftarrow \emptyset$;
while not interrupted **do**
  if $M = NodesInTree(v)$ **then**
    $(V_{old}, E_{old}) \leftarrow (V, E)$
  end if
  $T \leftarrow (V, E)$;
  $x_{rand} \leftarrow SAMPLE$;
  $(V, E) \leftarrow EXTEND(T, x_{rand})$;
  if $M > NodesInTree(v)$ **then**
    $(V, E) \leftarrow ForceRemoval(V, E)$;
  end if
  if $No\ ForceRemovalPerformed()$ **then**
    $(V, E) \leftarrow RestoreTree()$;
  end if
end while

ent of this node has only one child and whether the number of nodes in the tree reaches M. If so, x_{near} will be rewired as a child of x_{new}, and the parent of x_{near} will be deleted. If none of the nodes in the near domain of x_{new} has only one child, then the *ForcedRemoval* procedure will be employed, which searches the entire tree, except the x_{new} and the goal node, to find the nodes without children and deletes one randomly[1]. In case no nodes are deleted in *EXTEND* and *ForceRemoval* function, x_{new} is removed from the tree. The performance of MA-RRT*FN can be improved by frequently sampling the regions that are more likely to have high-quality solutions. This improved version is called informed sampling MA-RRT*FN(isMA-RRT*FN).

4 EXPERIMENTS AND RESULTS

The paper compares the capability of the MA-RRT*, MA-RRT*FN, isMA-RRT* and isMA-RRT*FN in terms of scalability, solution quality and memory cost. All experiments are performed on *matlab 2018a64-bit* in a common program framework and tested on *intel core i7 8700k* 3.7 GHz CPU.

To compare the scalability and suboptimality, the paper sets the problem instance set as follows: The agents run in a grid-like square-shaped world, where each agent occupies a single cell. Ten percent of the grids are removed to represent obstacles or barriers. A unique start location and destination are selected randomly for every agent. The problem instances set varies in the following two parameters: The grid sizes: 10x10, 30x30, 50x50, 70x70,90x90 and the number of agents: 1, 2, 3, 4, 5, 6, 7, 8, 9, 10. Each combination contains 120 random instances. Thus, in total, there are 6000 random instances. Particularly, to compare the memory cost and convergence rate, for clarity, this paper sets another experiment which fixes the two parameters: the grid sizes: 50x50 and the number of agents: 3, and randomly sets 120 instances to qualitatively show the memory needed and convergence rate of all algorithms. All algorithms are implemented on the same instance set when comparing the same capabilities. For comparing scalability and suboptimality, the runtime of each instance is limited to 5 seconds, and the maximum number of nodes of MA-RRT*FN and isMA-RRT*FN is set to 200. For comparing the memory cost and convergence



(a) Scalability (b) convergence rate (c) Memory required (d) Suboptimality

Figure 1: experiment results.

rate, the maximum number of iterations of each instance is limited to 5000, and the maximum number of nodes of MA-RRT*FN and isMA-RRT*FN is set to 1000. The results are plotted in Figure 1. The suboptimality is calculated by the following formula:

$$\text{suboptimality} = \left(\frac{\text{the cost of returned solution}}{\text{the cost of optimal solution}} - 1 \right) \cdot 100.$$

It can be seen from Figure 1(a) that MA-RRT* resolves 66% of the instances, MA-RRT*FN 65%, isMA-RRT* 86% and isMA-RRT*FN 87%, from the problem instance set. Figure 1(d) shows that MA-RRT*FN and isMA-RRT*FN have a similar suboptimality to MA-RRT* and isMA-RRT*FN. Figure 1(b) shows that the MA-RRT*FN has a similar convergence rate to MA-RRT* while its number of nodes in the tree is much less, as shown in Figure 1(c), memory required for MA-RRT* grows linearly with the iterations increase, while the number of nodes stored in MA-RRT*FN is lower and fixed. The results in Figure 1(b) and Figure 1(c) also indicate that the isMA-RRT*FN performs well than isMA-RRT* concerning the convergence rate and memory requirement. Finally, MA-RRT* is proved to be convergent in [15]. Although the experiment results strongly imply that the MA-RRT*FN and isMA-RRT*FN also have the theoretical guarantee of converging to the optimal path, the optimality of MA-RRT*FN and isMA-RRT*FN remains to be proved.

5 CONCLUSIONS

This paper proposes MA-RRT*FN, an improved version of MA-RRT* that has lower demands in the memory requirements. The experiment results show that the MA-RRT*FN performs as well as MA-RRT* in terms of scalability, solution quality and convergence rate while its memory required is much lower and fixed. Besides, its improved version, isMA-RRT*FN, has a better convergence rate and scalability than isMA-RRT*. In the future, we will continue to improve the convergence rate of MA-RRT*FN.

REFERENCES

[1] Olzhas Adiyatov and Huseyin Atakan Varol. [n. d.]. Rapidly-Exploring Random Tree Based Memory Efficient Motion Planning. In *2013 IEEE International Conference on Mechatronics and Automation* (2013). IEEE, 354–359.

[2] Anton Andreychuk, Konstantin Yakovlev, Dor Atzmon, and Roni Stern. [n. d.]. Multi-Agent Pathfinding (MAPF) with Continuous Time. ([n. d.]).

[3] Dave Ferguson and Anthony Stentz. [n. d.]. Anytime Rrts. In *2006 IEEE/RSJ International Conference on Intelligent Robots and Systems* (2006). IEEE, 5369–5375.

[4] Peter E Hart, Nils J Nilsson, and Bertram Raphael. [n. d.]. A Formal Basis for the Heuristic Determination of Minimum Cost Paths. 4, 2 ([n. d.]), 100–107.

[5] John E Hopcroft, Jacob Theodore Schwartz, and Micha Sharir. [n. d.]. On the Complexity of Motion Planning for Multiple Independent Objects; PSPACE-Hardness of the" Warehouseman's Problem". 3, 4 ([n. d.]), 76–88.

[6] Shotaro Kamio and Hitoshi Iba. [n. d.]. Random Sampling Algorithm for Multi-Agent Cooperation Planning. In *2005 IEEE/RSJ International Conference on Intelligent Robots and Systems* (2005). IEEE, 1265–1270.

[7] Steven M LaValle and James J Kuffner Jr. [n. d.]. Randomized Kinodynamic Planning. 20, 5 ([n. d.]), 378–400.

[8] Michael Otte and Nikolaus Correll. [n. d.]. Any-Com Multi-Robot Path-Planning with Dynamic Teams: Multi-Robot Coordination under Communication Constraints. In *Experimental Robotics* (2014). Springer, 743–757.

[9] Matteo Ragaglia, Maria Prandini, and Luca Bascetta. [n. d.]. Multi-Agent Poli-Rrt. In *International Workshop on Modelling and Simulation for Autonomous Systems* (2016). Springer, 261–270.

[10] Guni Sharon, Roni Stern, Ariel Felner, and Nathan R Sturtevant. [n. d.]. Conflict-Based Search for Optimal Multi-Agent Pathfinding. 219 ([n. d.]), 40–66.

[11] David Silver. [n. d.]. Cooperative Pathfinding. 1 ([n. d.]), 117–122.

[12] Trevor Scott Standley. [n. d.]. Finding Optimal Solutions to Cooperative Pathfinding Problems. In *Twenty-Fourth AAAI Conference on Artificial Intelligence* (2010).

[13] Trevor Scott Standley and Richard Korf. [n. d.]. Complete Algorithms for Cooperative Pathfinding Problems. In *Twenty-Second International Joint Conference on Artificial Intelligence* (2011).

[14] Paolo Verbari, Luca Bascetta, and Maria Prandini. [n. d.]. Multi-Agent Trajectory Planning: A Decentralized Iterative Algorithm Based on Single-Agent Dynamic RRT. In *2019 American Control Conference (ACC)* (2019). IEEE, 1977–1982.

[15] Michal Čáp, Peter Novák, JiYí Vokrínek, and Michal Pěchouček. [n. d.]. Multi-Agent RRT: Sampling-Based Cooperative Pathfinding. In *Proceedings of the 2013 International Conference on Autonomous Agents and Multi-Agent Systems* (2013). International Foundation for Autonomous Agents and Multiagent Systems, 1263–1264.