Multiagent Monte Carlo Tree Search

Extended Abstract

Nicholas Zerbel
Oregon State University
Corvallis, Oregon, USA
zerbeln@oregonstate.edu

Logan Yliniemi*
Amazon Robotics
Boston, Massachusetts, USA
loganyli@amazon.com

ABSTRACT

Monte Carlo Tree Search (MCTS) is a best-first search which is efficient in large search spaces and is effective at balancing exploration versus exploitation. In this work, we introduce a novel extension for MCTS, called Multiagent Monte Carlo Tree Search (MAMCTS), which pairs MCTS with difference evaluations. We demonstrate the performance of MAMCTS in a cooperative, multiagent path-planning domain called Multiagent Gridworld. We show that MAMCTS using difference evaluations outperforms MAMCTS using local rewards by up to 31.4% and MAMCTS using the global reward by up to 88.9% for a system with 1,000 agents.

KEYWORDS

Multiagent Learning, Difference Evaluations; Monte Carlo Tree Search

ACM Reference Format:
Nicholas Zerbel and Logan Yliniemi. 2019. Multiagent Monte Carlo Tree Search. In *Proc. of the 18th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2019), Montreal, Canada, May 13–17, 2019,* IFAAMAS, 3 pages.

1 INTRODUCTION

Monte Carlo Tree Search (MCTS) is a best-first search which uses Monte Carlo methods to probabilistically sample actions in a given domain [2]. MCTS gained a lot of interest after being used to develop game players for the game of *Go* which could compete with some of the best human players on smaller *Go* boards [2, 6]. Since then, MCTS has been used heavily in planning problems and computerized games such as *Poker*, *Settlers of Catan*, *Civilization*, and *Shogi* [3, 8–11].

MCTS has several qualities which make it ideal for application within multiagent systems. For example, MCTS does not require heuristics derived using domain-specific knowledge to find near-optimal solutions [2]. MCTS is also very efficient at searching through large search spaces while effectively balancing exploration versus exploitation within the search space [2]. This gives MCTS an advantage over other approaches, such as evolutionary algorithms, which can be computationally expensive due to the extremely large search space for agent policies within multiagent systems [1, 4, 7].

Although there are many works demonstrating the efficacy of MCTS in multiagent environments, such as multiplayer games, most of the literature is focused on developing strong players in competitive multiagent systems. In this work, we propose a novel extension to MCTS for cooperative multiagent systems, called Multiagent Monte Carlo Tree Search (MAMCTS), which pairs MCTS with difference evaluations. We show that MAMCTS-based agents are able to learn effective coordination strategies in a multiagent path-planning domain called Multiagent Gridworld (MAG). We also show that MAMCTS using difference rewards outperforms MAMCTS using local rewards by up to 31.1% and MAMCTS using the global reward by up to 88.9%.

*Work for this project was completed prior to joining Amazon Robotics.

Proc. of the 18th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2019), N. Agmon, M. E. Taylor, E. Elkind, M. Veloso (eds.), May 13–17, 2019, Montreal, Canada.

2 MULTIAGENT MONTE CARLO TREE SEARCH

In a cooperative multiagent system, individual agents must adopt individual policies which allow them to coordinate effectively with other agents in the system. Multiagent Monte Carlo Tree Search (MAMCTS) pairs MCTS with difference evaluations which allows agents to efficiently search through the space of possible policies by focusing on those policies which positively contribute to the overall system. MAMCTS is described in more detail by Algorithm 1. Steps 2-8 describe a standard MCTS which involves selection, expansion, rollout, and back-propagation. These steps are done at the individual agent level. In steps 9-12, each agent selects the best policy found so far from within their search tree and takes the actions stored within that policy. In steps 13-16, each policy is evaluated using difference evaluations, and each agent updates their search tree using the calculated difference reward before resetting and continuing to the next episode.

Algorithm 1: Multiagent Monte Carlo Tree Search

```
1:  for each Episode do
2:    for each Agent do
3:      Selection
4:      Check for terminal node
5:      Expand tree if selected node is non-terminal
6:      Rollout using default, random policy (π_D)
7:      Back-Propagation
8:    end for
9:    for each Agent do
10:     Select best policy from tree
11:     Do actions in best policy
12:   end for
13:   for each Agent do
14:     Evaluate policy and update tree
15:   end for
16:   Reset agents to initial state
17: end for
```

3 EXPERIMENTAL SETUP

We test MAMCTS in a cooperative, multiagent path-planning domain known as Multiagent Gridworld (MAG). MAG is a discrete environment in which agents can take one action per turn from among the available actions: up, down, left, right, or null. The outer boundaries of MAG are treated as an impassable wall. Any agent attempting to move beyond these boundaries is "bumped" back onto the grid. In MAG there are n agents and n goals. Each goal is worth 100 points when captured; however, a reward is only received for the first agent to capture a goal. Once an agent captures a goal, it stays at the goal and does not continue to move. Therefore, the optimal solution is for each agent to travel to a different goal and capture it.

In this work we compare the performance of MAMCTS using difference evaluations with MAMCTS using local rewards (which is equivalent to only using MCTS) and MAMCTS using the global reward for policy evaluations. All of the tests are run on a $100x100$ MAG starting with a (200 agent, 200 goal) system and then scaling up to a (1,000 agent, 1,000 goal) system. Data is collected over 30 statistical runs. Error bars are included in the plots; however, the error bars are smaller than the data point markers used in the plot. All error bars report the standard error of the mean.

4 RESULTS AND DISCUSSION

In the $100x100$ MAG, we test the scalability and performance of MAMCTS by steadily increasing the number of agents and goals within the system. Starting with 200 agents and 200 goals, we record the percentage of goals captured by the agents after 500 episodes of searching. These results are reported by Figure 1 which show the quantity of agents and goals in the system and the percentage of goals captured when each agent uses their best policy. The average goal capture rate is approximately 79.2% for MAMCTS using difference evaluations, 38.6% for MAMCTS using the global reward, and 61.7% for MAMCTS using local rewards. The results presented in Figure 1 show that the percentage of goals captured improves as more agents and goals are added into the system. At 200 agents and 200 goals, the goals are more widely dispersed throughout the world. We believe this makes it difficult for agents to discover policies leading to goals which are further away. When we add more goals into the system, there is a higher goal density making it easier for agents to discover policies leading to goal states even though there are also more agents in the system.

For the system containing 1,000 agents and 1,000 goals, we record the system score at the end of each search episode. This allows us to assess agent learning by tracking the improvement in the system score each episode as seen in Figure 2. By comparing the final system reward of MAMCTS using difference rewards to the final system reward achieved by MAMCTS using local rewards, we see that MAMCTS using difference evaluations outperforms MAMCTS using local rewards by 31.4%. Additionally, we see that MAMCTS using difference evaluations outperforms MAMCTS using the global reward by 88.9%. Using the global reward to update agent search trees, we see that the system score converges to a low value very quickly and never improves. Interestingly, the agents using MAMCTS with local rewards achieve a decent system-level performance. However, this performance eventually converges to

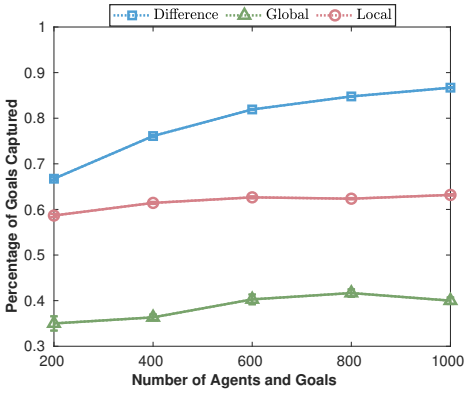

Figure 1: Percentage of goals captured by agents for increasing numbers of agents and goals on a 100x100 MAG.

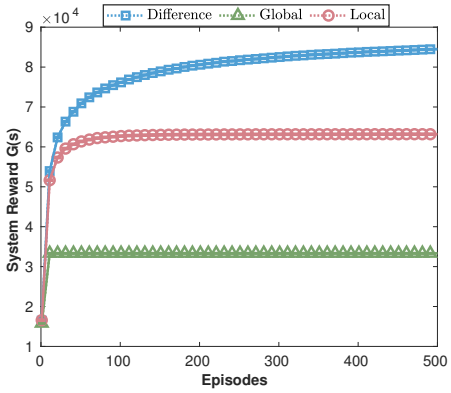

Figure 2: Learning curves for 1,000 agent and 1,000 goal system comparing MAMCTS using difference evaluations, local reward, and the global reward for policy evaluation.

a final value after which no further improvement occurs. Using difference evaluations, we see that system performance is not only better overall, but that system performance continues to improve by the 500th episode. These results are consistent with examples from the literature comparing difference evaluations with local rewards and global rewards [5, 12, 13].

5 CONCLUSIONS

In this work we introduced a novel enhancement to MCTS for cooperative multiagent systems called Multiagent Monte Carlo Tree Search. MAMCTS pairs MCTS with difference evaluations which enables agents to learn effective coordination strategies in loosely coupled, cooperative multiagent environments such as MAG. In MAG, we demonstrated that MAMCTS using difference rewards outperforms MAMCTS using the global reward by up to 88.9% and MAMCTS using local rewards by up to 31.4%. Finally, we demonstrated the impressive scalability of MAMCTS by achieving near-optimal performance in a $100x100$ MAG with up to 1,000 agents.

REFERENCES

[1] Adrian Agogino and Kagan Tumer. 2008. Efficient evaluation functions for evolving coordination. *Evolutionary Computation* 16, 2 (2008), 257–288.

[2] Cameron B Browne, Edward Powley, Daniel Whitehouse, Simon M Lucas, Peter I Cowling, Philipp Rohlfshagen, Stephen Tavener, Diego Perez, Spyridon Samothrakis, and Simon Colton. 2012. A survey of monte carlo tree search methods. *IEEE Transactions on Computational Intelligence and AI in Games* 4, 1 (2012), 1–43.

[3] Guillaume Chaslot, Sander Bakkes, Istvan Szita, and Pieter Spronck. 2008. Monte-Carlo Tree Search: A New Framework for Game AI.. In *Artificial Intelligence for Interactive Digital Entertainment*. AAAI.

[4] Mitchell Colby, Jen Jen Chung, and Kagan Tumer. 2015. Implicit adaptive multi-robot coordination in dynamic environments. In *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 5168–5173.

[5] Sam Devlin, Logan Yliniemi, Daniel Kudenko, and Kagan Tumer. 2014. Potential-based difference rewards for multiagent reinforcement learning. In *Proceedings of the 2014 international conference on Autonomous Agents and Multiagent Systems*. International Foundation for Autonomous Agents and Multiagent Systems, 165–172.

[6] Sylvain Gelly and David Silver. 2008. Achieving Master Level Play in 9 x 9 Computer Go.. In *Proceedings of the 2008 AAAI Conference*, Vol. 8. 1537–1540.

[7] Liviu Panait and Sean Luke. 2005. Cooperative multi-agent learning: The state of the art. *Autonomous Agents and Multiagent Systems* 11, 3 (2005), 387–434.

[8] Marc Ponsen, Geert Gerritsen, and Guillaume Chaslot. 2010. Integrating opponent models with monte-carlo tree search in poker. In *Workshops at the Twenty-Fourth AAAI Conference on Artificial Intelligence*.

[9] David Robles, Philipp Rohlfshagen, and Simon M Lucas. 2011. Learning non-random moves for playing Othello: Improving Monte Carlo tree search. In *2011 IEEE Conference on Computational Intelligence and Games (CIG'11)*. IEEE, 305–312.

[10] Yoshikuni Sato, Daisuke Takahashi, and Reijer Grimbergen. 2010. A shogi program based on monte-carlo tree search. *International Computer Games Association Journal* 33, 2 (2010), 80–92.

[11] István Szita, Guillaume Chaslot, and Pieter Spronck. 2009. Monte-carlo tree search in settlers of catan. In *Advances in Computer Games*. Springer, 21–32.

[12] Michael Wooldridge. 2009. *An introduction to multiagent systems*. John Wiley & Sons.

[13] Logan Yliniemi and Kagan Tumer. 2016. Multi-objective multiagent credit assignment in reinforcement learning and NSGA-II. *Soft Computing* 20, 10 (2016), 3869–3887.