Scalable Anytime Planning for Multi-Agent MDPs (Extended Abstract)

Shushman Choudhury1*, Jayesh K. Gupta2*, Mykel J. Kochenderfer1
1 Stanford University
2 Microsoft
shushman@cs.stanford.edu, jayesh.gupta@microsoft.com, mykel@stanford.edu

Abstract

We present a scalable planning algorithm for multi-agent sequential decision problems that require dynamic collaboration. Teams of agents need to coordinate decisions in many domains, but naive approaches fail due to the exponential growth of the joint action space with the number of agents. We circumvent this complexity through an anytime approach that allows us to trade computation for approximation quality and also dynamically coordinate actions. Our algorithm comprises three elements: online planning with Monte Carlo Tree Search (MCTS), factoring local agent interactions with coordination graphs, and selecting optimal joint actions with the Max-Plus method. On the benchmark SysAdmin domain with static coordination graphs, our approach achieves comparable performance with much lower computation cost than the MCTS baselines. We also introduce a multi-drone delivery domain with dynamic, i.e., state-dependent coordination graphs, and demonstrate how our approach scales to large problems on this domain that are intractable for other MCTS methods.

1 Introduction

Coordination is crucial for decision-making in multi-agent systems with a shared objective. Real-world problems like formation control [Oh et al., 2015], package delivery [Choudhury et al., 2020], and firefighting [Oliehoek et al., 2008] require a team of autonomous agents to perform a common task. Such cooperative sequential settings can be modeled as a Multi-Agent Markov Decision Process (MMDP) [Boutilier, 1996], an extension of the Markov Decision Process (MDP) [Kochenderfer, 2015]. MMDPs can be reduced to centralized single-agent MDPs with a joint action space for all agents. Such reductions often make large problems intractable because the action space grows exponentially with the number of agents. Solving independent MDPs for all agents, however, can yield arbitrarily sub-optimal behavior in problems that need coordination [Matignon et al., 2012].

Many previous MMDP approaches have tried to balance these extremes of optimality and efficiency. In the offline setting, these include ad hoc function decomposition approaches, such as Value Decomposition Networks [Sunehag et al., 2018] and QMIX [Rashid et al., 2018], or parameter sharing in decentralized policy optimization [Gupta et al., 2017]. [Guestrin et al., 2002] introduced the concept of a coordination graph to reason about joint value estimates from a factored representation, while [Kok and Vlassis, 2004] used approximations to scale these ideas to larger problems. Monte Carlo Tree Search (MCTS) [Browne et al., 2012], a common approach to online planning, has been combined with coordination graphs in Factored Value MCTS [Amato and Oliehoek, 2014]. However, Factored Value MCTS coordinates actions with an exact Variable Elimination (Var-El) step, which conflicts with the anytime nature of MCTS planning.

The key idea of this paper is to recover the anytime nature of MCTS planning for MMDPs requiring coordination and also scale to larger teams of agents. To that end, we propose combining Max-Plus action selection, introduced by [Vlassis et al., 2004], with the factored value MCTS of [Amato and Oliehoek, 2014]. We do so for many reasons. Unlike Var-El, which is exact, Max-Plus is an iterative procedure and allows for truly anytime behavior that can trade computation for approximation quality. The representation of Max-Plus is much more efficient than that of Var-El for using dynamic, i.e., state-dependent, coordination graphs (state-dependent data-structures are a key benefit of online planning for MDPs). Finally, Max-Plus can scale to much larger and denser coordination graphs than Var-El, and it can be distributed for additional scalability [Kok and Vlassis, 2005].

We present a scalable anytime MMDP planning algorithm called Factored Value MCTS with Max-Plus. On the standard SysAdmin benchmark domain [Guestrin et al., 2002], with static coordination graphs, we demonstrate that our approach performs as well as or better than Factored Value MCTS with Var-El [Amato and Oliehoek, 2014] and is much faster for the same tree search hyperparameters. We also introduce a new MMDP domain, Multi-Drone Delivery, with dynamic coordination graphs. On the second domain, we show how our approach scales to problem sizes that are entirely intractable for other MCTS variants, while also achieving better performance on smaller problem sizes. The extended version (with code) is available at https://sites.google.com/stanford.edu/fvmcts/.

*This paper is an extended abstract of the Best Paper at Autonomous Agents and Multi-Agent Systems (AAMAS) 2022.
2 Background: Online MMDP Planning

The goal of MDP planning is to obtain a policy, $\pi : S \to A$, that specifies what action $a$ the agent should take from its current state $s$ to maximize its value, i.e. its expected cumulative reward. We consider decision-making settings where multiple agents cooperate under a shared reward function [Boutilier, 1996], i.e., Multi-Agent MDPs or MMDPs. Here, each agent takes an individual action and the planning algorithm observes states of all agents. Both offline and online methods exist for computing policies for MMDPs [Bertsekas, 2005].

Online methods (the focus of our work) interleave planning and execution by focusing only on states that are reachable for the current state, while computing the next action to take. Monte Carlo Tree Search (MCTS) is the predominant framework for online planning and has succeeded in a variety of domains [Browne et al., 2012], including in multi-agent contexts [Nijssen and Winands, 2011; Zerbel and Yliniemi, 2019]. The anytime nature of MCTS is often used to encode such interactions [Boutilier, 1996], Multi-Agent MDPs or MMDPs. Here, each agent takes an individual action and the planning algorithm observes states of all agents. Both offline and online methods exist for computing policies for MMDPs [Bertsekas, 2005].

Online methods (the focus of our work) interleave planning and execution by focusing only on states that are reachable for the current state, while computing the next action to take. Monte Carlo Tree Search (MCTS) is the predominant framework for online planning and has succeeded in a variety of domains [Browne et al., 2012], including in multi-agent contexts [Nijssen and Winands, 2011; Zerbel and Yliniemi, 2019]. The anytime nature of MCTS is often used to encode such interactions [Boutilier, 1996], Multi-Agent MDPs or MMDPs. Here, each agent takes an individual action and the planning algorithm observes states of all agents. Both offline and online methods exist for computing policies for MMDPs [Bertsekas, 2005].

Online methods (the focus of our work) interleave planning and execution by focusing only on states that are reachable for the current state, while computing the next action to take. Monte Carlo Tree Search (MCTS) is the predominant framework for online planning and has succeeded in a variety of domains [Browne et al., 2012], including in multi-agent contexts [Nijssen and Winands, 2011; Zerbel and Yliniemi, 2019]. The anytime nature of MCTS is often used to encode such interactions [Boutilier, 1996], Multi-Agent MDPs or MMDPs. Here, each agent takes an individual action and the planning algorithm observes states of all agents. Both offline and online methods exist for computing policies for MMDPs [Bertsekas, 2005].

We now discuss our method for anytime multi-agent MDP planning with coordination graphs. Factored-Value Monte Carlo Tree Search with Max-Plus. To apply the mixture of experts optimization (discussed previously) to each node of the search tree, we must define the factored statistics to maintain for each node. Given a potentially state-dependent undirected
coordination graph (CG), $G = (V, E)$, we factor the CG-induced global payoff at the current state, $\pi$, as follows:

$$Q(\pi) = \sum_{i \in V} Q_i(a_i) + \sum_{(i,j) \in E} Q_{ij}(a_i, a_j).$$  \hspace{1cm} (1)$$

Here, $Q_{ij}$ is a local payoff function for agents $i$ and $j$ connected by edge $(i, j)$, while $Q_i$ is an individual utility function for agent $i$, if applicable to the domain. All state-dependent quantities in this section’s equations are implicitly for the current state $\pi$.

Exploiting the duality between computing the maximum a posteriori configuration in a probabilistic graphical model and the optimal joint action in a CG, [Vlassis et al., 2004] introduced the Max-Plus algorithm for computing the joint action via message passing. Each node, i.e., agent, iteratively dispatches messages to its neighbors $j \in \Gamma(i)$ in the CG. A message from agent $i$ is a scalar-valued function of the action space of receiving agent $j$, i.e.,

$$\mu_{ij}(a_j) = \max_{a_i} \left\{ Q_i(a_i) + Q_{ij}(a_i, a_j) + \sum_{k \in \Gamma(i) \setminus \{j\}} \mu_{ki}(a_i) \right\},$$  \hspace{1cm} (2)$$

where $\Gamma(i)$ is the set of neighbors of $i$. Agents exchange messages until convergence or for some number of rounds. Finally, each agent computes its optimal action individually,

$$a_i^* = \arg\max_{a_i} \left\{ Q_i(a_i) + \sum_{j \in \Gamma(i)} \mu_{ji}(a_i) \right\}$$  \hspace{1cm} (3)$$

Max-Plus is equivalent to belief propagation in graphical models [Pearl, 1989] and its time complexity scales linearly with the CG size (the number of edges); it is more suitable for real-time systems and more tractable for large numbers of agents than Var-El.

3.1 UCB Exploration with Max-Plus

The key implementation issue for extending MCTS to factored value functions and coordination graphs is that of action exploration as per the Upper Confidence Bound (UCB) strategy. In the Var-El case, [Amato and Oliehoek, 2014] added the exploration bonus using component-wise statistics during each elimination step. We cannot apply this strategy with Max-Plus. The node and edge exploration strategies we have defined here are heuristic choices that we make and evaluate empirically through an ablation. Our exploration strategies differ from the component-wise exploration in the previous work that uses Var-El, because we do not consider cliques/components in the CG, only nodes and edges.

4 Experiments and Results

We used cumulative discounted return as the primary metric to evaluate our approach. Factored Value MCTS with Max-Plus (FV-MCTS-MP). Our most relevant baseline is Factored Value MCTS with Variable Elimination (FV-MCTS-Var-El). We also compared against standard MCTS (with no factorization), independent Q-learning (IQL), and a random policy. Besides performance, we also measured the MCTS computation time with increasing problem size. The extended conference version [Choudhury et al., 2021] examines the effect of different hyperparameters on both variants of FV-MCTS and of different action exploration schemes on FV-MCTS-MP. Both of our experimental domains represent a range of MMDP problems and underlying CGs. All implementations and simulations are in Julia [Bezanson et al., 2017].

4.1 SysAdmin Domain

Our first domain is a standard MMDP benchmark: SysAdmin [Guestrin et al., 2003]. Each agent $i$ represents a machine in a network with two state variables: Status $S_i \in \{\text{GOOD}, \text{FAULTY}, \text{DEAD}\}$, and Load $L_i \in \{\text{IDLE, LOADED, SUCCESS}\}$. A DEAD machine increases the probability that its neighbor also dies. The system gets a reward of 1 if a process terminates successfully, processes take longer when status is FAULTY, and a DEAD machine loses the process. Each agent must decide whether to reboot its machine, in which case the Status becomes GOOD and any running process is lost. The discount factor, $\gamma$ used in all the experiments is 0.9. All evaluations have been averaged over 40 runs.

For all SysAdmin topologies, we varied the number of machines (agents) and compared the performance of all methods...
Figure 2: On SysAdmin topologies: Ring (left), Star (middle), Ring-of-Rings (right), FV-MCTS-MP performs as well as or slightly better than other baselines. NaN indicates that the algorithm runs out of memory.

Figure 3: Runtime comparisons (lower is better) on SysAdmin with Ring (top) and Star (bottom) topologies.

in Figure 2. With fewer agents, all MCTS methods perform similarly to each other. However, Factored Value MCTS methods are able to scale with increasing number of agents. Both FV-MCTS variants perform comparably on larger problems with a slight edge for MaxPlus variant on ring-of-ring topology. However, as shown in Figure 3, FV-MCTS-Var-El is consistently much slower than FV-MCTS-MP, e.g., taking approximately 35s versus 16s for 32 agents on a single-threaded implementation in the ring topology.

4.2 Multi-Drone Delivery Domain
We introduce an MMDP domain that simulates a team of delivery drones navigating a shared operation space to reach their specified goal regions. Our domain requires multiple drones assigned to the same goal to arrive within a short time window of each other. The MMDP is episodic and terminates only when all drones have reached their goal locations. Unlike typical MMDP domains used in prior work, Multi-Drone Delivery encode dynamic or state-dependent coordination graphs; any two drones benefit from coordination only when they are close to each other. Therefore, at the current joint state, we assign a CG edge between any two drones whose mutual distance is lower than a resolution-dependent threshold.

As with SysAdmin, we varied the number of drones (agents) and compared against all baselines in Figure 4. Given the large action space per-agent, neither naive MCTS nor FV-MCTS-Var-El was able to scale to larger number of agents. Var-El’s restriction to static CGs leads to slightly worse performance with 8 agents while FV-MCTS-MP is able to successfully solve tasks even with 32 agents. Moreover, even on the 8 agent problem, FV-MCTS-MP is much faster, taking on average approximately 1s instead of 40s for FV-MCTS-Var-El for the same tree search hyperparameters.

5 Conclusion
We introduced a scalable online planning algorithm for Multi-Agent MDPs with dynamic coordination graphs. Our approach uses Max-Plus for action coordination, in contrast to the previous approach that used Variable Elimination. Over the SysAdmin and Multi-Drone Delivery domains, we demonstrated that our approach performs as well as baselines on static CGs, outperforms them significantly on dynamic CGs, and is far more computationally efficient (enabling online MMDP planning on previously intractable problems).
References


