Cooperative Decision Algorithm for Time Critical Assignment without Explicit Communication

Yulin Zhang, Yang Xu, Haixiao Hu

Cooperative Decision Algorithm for Time Critical Assignment without Explicit Communication

Yulin Zhang, Yang Xu, and Haixiao Hu

School of Computer Science and Engineering,
University of Electronic Science and Technology of China,
xuyang@uestc.edu.cn

Abstract. Decentralized time critical assignment is popular in the domains of military and emergency response. Considering the large communication delay, unpredictable environment and constraints, to make a rational decision in time critical context toward their common goal, agents have to jointly allocate multiple tasks based on their current status. This process can be built as a Markov Decision Model when they can share their status freely. However, it is computationally infeasible when explicit communication becomes a severe bottleneck. In this paper, by modelling the decision process as a Partially Observable Markov Decision Process for each agent and building heuristic approaches to estimate the state and reduce the action space, we apply greedy policies in our heuristic algorithms to quickly respond to time critical requirement.

Keywords: Multi-agent systems, task assignment, time critical

1 Introduction

With the progress of research in distributed artificial intelligence, cooperative agents have been applied in dynamic and complex application domains such as disaster response[1], military[2] and business organizations[3] in recent years. In those domains, task allocation becomes a basic problem that needs to be solved efficiently [3]. The basic task allocation problem is organized as follows: Given a set of agents and a set of tasks, where each agent obtains some payoff for each task, find a one-to-one assignment of agents to tasks to maximize the total payoff of all agents. This problem can be solved (near) optimally in polynomial time by centralized algorithms [4, 5], and decentralized algorithms [3, 6]. In all of these works, it is assumed that communication is available and the environment is static. However, in real domain, the communication is not available and the environment including the status of the agents and the tasks may change. For example, when a group of robots are deployed to carry out time critical tasks for Urban Search and Rescue (USAR), communication infrastructure is most likely destroyed in disaster and communication may be restricted for each rescue robot. Another example, when missiles are launched to attack hostile targets,

the missiles have to keep silent due to security reasons. In both of these domains, the environment is dynamically changing and necessary cooperative replanning or reallocation is required. Therefore, we summarize those scenarios as a group of time critical agents that they have to cooperate to optimize their common goals without explicit communication.

More specifically, time critical task assignment without explicit communication that we study can be stated as follows: Given a set of agents and a set of tasks, firstly, the capability of each agent is determined by the dynamically changing environment; Secondly, the state of task and agent may change unexpectedly; Thirdly, no explicit communication is allowed; In addition, a payoff can be obtained from a completed task. We need to find the assignment of agents to tasks within critical response time, such that the sum of payoff from all agents is maximized. To improve team performance, team states have to be well maintained from observations. This can be modelled as a Decentralized Partially Observable Markov Decision Process, which is NEXP-complete [7]. And current approaches such as [8, 9], try to solve this problem with the help of communication, which is not allowed in our problem.

In this paper, several efforts were made to solve the task assignment problem in time critical context. Firstly, since the capabilities of all agents and the statuses of their tasks are changing with Markovian property, the task assignment process can be built as a Markov Decision Model. However, when communication is not available or unacceptable in time critical context, the team state cannot be accurately observed. In this paper, we model the decision process as a Partially Observable Markov Decision Process for each agent. Secondly, the objective function in our task assignment is a step function, which makes it difficult for the agent to decide whether to jointly assign an uncompleted task to a single agent. We propose an exponential reward function to fit for it. Finally, without communication, there is only a small amount of observation, which makes the problem an open-loop control and toward an NEXP-complete problem [10]. Thus, heuristic approaches are built according to the following observations: 1. Task-related agents may get the same observation with a high probability. 2. When nothing can be referred from others, it's better to assume that their choice stays the same. With these observations, the search space can be dramatically reduced and a greedy policy is applied to help operate in time critical context.

2 Problem Statement

In this section, we provide a time critical assignment problem without explicit communication for cooperative teams.

2.1 Task Assignment Problem

We focus on a cooperative time critical agent team $A = \{a_1, a_2, ..., a_i, ...\}$, such as a group of robots or UAVs, coordinating to carry out a set of tasks $T = \{t_1, t_2, ..., t_j, ...\}$. Each task consists of two parts, $t_j =< \chi_j, \Phi_j >$. χ_j is the

payoff to complete t_j, indicating its importance. Φ_j describes t_j's workload, which represents the total amount of work required to complete t_j. Before the task is executed, there is a preparation stage for the agent to prepare for the execution or heading for the target location. During this preparation stage, the potential contribution of the agents to their tasks may change, since their states dynamically change. To obtain the highest payoff with limited agents, we should dynamically assign all the tasks during this stage.

At time t, the potential contribution of a_i to t_j is described as $c(a_i, t_j)^t$, which is dynamically updated as the situation between a_i and t_j changes, according to the domain knowledge. For example, the capability of the rescue robot is related to its remaining energy, holding resources and even distance to its target location. When the robot is of low power, its capability to carry out a long-distance task is very small. If the resource in the robot is going to run out, it is less capable to carry out a resource-consuming mission. After agent a_i is assigned with task $\tau_{a_i}(t)$, all agents assigned with task t_j is summarized as $\Gamma_j(t)$. Thus, if the total contribution $\psi_j(t)$ of all assigned agents for task t_j meets its workload, that is,

$$\psi_j(t) = \sum_{a_i \in \Gamma_j(t)} c(a_i, t_j)^t \geqslant \Phi_j$$

t_j can be completed and $Completed(t_j) = 1$; Otherwise, $Completed(t_j) = 0$.

During the preparation stage, we need to find the best strategy π^* as following to get the highest payoff.

$$\pi^*(t) = argmax_{Jt\_\tau(A,T,t)} \sum_{t_j \epsilon T} \chi_j \cdot Completed(t_j) \tag{1}$$

where $Jt\_\tau(A, T, t)$ is the task allocated for all agents in this team, $Jt\_\tau(A, T, t) = \bigcup_{a_i \in A} \tau_{a_i}(t)$.

2.2 Time Critical Assignment without Explicit Communication

In a time critical system, a tiny delay may lead to disasters. For example, when a UAV in fast mode finds an obstacle in front, the decision about how to change its orbit should be made immediately, or it will be destroyed before it changes its orbit. Therefore, the task assignment should be carried out in a time critical context and no deliberation model should be allowed. In addition, when the time critical system is performing a task far away from its base, communication is extremely unreliable and of long delay. To operate in a time critical context, no explicit communication could be relied on. Without explicit communication, due to the limited sensing capabilities, the agents cannot fully observe the team state but can only get partial observations. Therefore, the agents have to reason from its partial observations to make rational decisions.

3 Cooperative Task Assignment Algorithm

In order to obtain the highest payoff, each agent has to dynamically update its task according to the changing team state, including the capabilities of all agents and the status of each task. Since the task of each agent is only determined by the latest team state rather than the historical ones, the task assignment process can be built as a Markov Decision Model when they can share their status freely. Unfortunately, in time critical context, the agents cannot share the state through unreliable communication with large delay. Each agent can only estimate the team state from the ambient data of its own observation. Due to the limitation of its observation range, the observation obtained by the agent is incomplete and team state is uncertain. Therefore, in this paper, a Partially Observable Markov Decision Process model is built for each agent to reason from its partial observations. Since small amount of observation makes the POMDP problem even more difficult, heuristic approaches are proposed to solve this problem.

3.1 Partially Observable Markov Decision Process Model for Cooperative Task Assignment

The basic decision model for the task assignment process of each agent can be written as a POMDP $< S, Jt\_act, T, \Theta, O, U >$. S is the state space and its value at time t is defined as $s(t)$, Jt_act is the action space of all agents, $T : S \times A \rightarrow S$, is the transition function that describes the resulting state $s(t+1) \in S$ when executing $Jt\_\tau(A, T, t) \in Jt\_act$ in $s(t)$. Θ and O are the observations and their observation function respectively to indicate the possible states under a given observation. $R : S \rightarrow \mathbb{R}$ defines the reward of being in a specific state. This model can be applied to each agent.

In this case, the team state $s(t)$ is modelled as the capabilities of all agents and the assigned agents for each task at time $t-1$:

$$S(t) = \cup_{t_j \in T}(\Gamma_j(t-1) \cup_{a_i \in A} c(a_i, t_j)^t)$$

After assigning $Jt\_\tau(A, T, t)$ at state $s(t)$, the capability of each agent is transited according to the domain knowledge: $T : c(a_i, t_j)^t \times \tau_{a_i}(t) \rightarrow c(a_i, t_j)^{t+1}$. For example, when a rescue robot changes its task, it will change its direction to head for its new target location and improve its capability to accomplish this task in time.

However, each agent cannot fully observe the whole team state but can only get partial observations $\theta(t) \in \Theta$, such as the status of the task, the state of itself and other agents. For example, when the rescue robot is approaching its target location, it can sense the topography of the target area and its remaining energy. From the observation, the belief state $b(t)$ could be derived according to the observation function: $O : \Theta \rightarrow B$. For each state $s(t) \in b(t)$, we know the perceptual distribution $Pr(s(t)|\theta(t))$, which describes the likelihood of the team being in the state of $s(t)$ under observation $\theta(t)$. For example, when a robot observes that the traffic to the target location becomes extremely tougher,

which will consumes more energy to reach the target location and indicates a capability drop of this agent, others with the same task may also observe the same capability drop with a high probability.

To find the optimal assignment, the key is to maximize the objective function [1]. However, the objective function is a step function, which generates reward only when the task is able to be completed and makes it difficult to decide whether to assign an uncompleted task to a single agent. Hence we propose an exponential reward function to fit for the objective function. And the exponential reward function should satisfy the following two properties: Firstly, for each task t_j, with all the potential agents and their contributions, the potential progress $\psi_j(t)$ of t_j is the sum of all contributions from its assigned agents.

$$\psi_j(t) = \sum_{a_i \in \Gamma_j(t-1)} c(a_i, t_j)^t$$

If $\psi_j(t)$ equals to zero, which means that there is no potential progress for task t_j and the team obtains no reward from task t_j. Secondly, when the task receives enough progress to be completed, the team gets the whole priority of the task as its reward. Considering these two properties of the objective function, the exponential reward function is built as follows:

$$R(\psi_j(t)) = \frac{\chi_j}{e^{\Phi_j} - 1}(e^{\psi_j(t)} - 1)$$

And the utility of $s(t)$ is the total reward from all tasks.

$$U(S(t)) = \sum_{t_j \in T} R(\psi_j(t))$$

In this model, the agents coordinate to find the best strategy π^* to maximize the expected utility.

$$\pi^*(t) = \arg \max_{Jt\_\tau(A,T,t)} \sum_{s(t+1) \in b(t+1)} Pr(s(t+1)|\theta(t+1)) \times U(s(t+1))$$

Without communication, each agent can only get a small amount of observation, which makes this problem an open-loop control and toward an NEXP-complete problem, according to the study of [10].

3.2 Heuristic Approach

In this section, we build heuristic approaches according to the following efforts. Firstly, we assume that an optimal assignment is initialized for each agent offline. Secondly, since agents act in the same environment, task-related agents may obtain the same observation with a high probability. Hence the belief of all task-related agents may change and their tasks may be reassigned. Finally, when we cannot infer anything about others, since the state is Markovian, we can assume that their choice stays the same to keep synchronization of all agents. Thus, the search space can be dramatically reduced and a greedy policy can be applied for each agent in a time critical context.

State estimation. In a decentralized task assignment domain, several agents may be assigned to the same task. When there is an unexpected accident happening to the task, all agents assigned to the task tend to get the same observation. For example, when a group of robots are assigned to carry out the same task $\mathfrak{t}$, if robot α observes that the condition of the task is more difficult than it expected, its capability $\widetilde{c}(\alpha, \mathfrak{t})^t$ will drop and other task-related agents may also observe the same disturbance, which drives them to reassign their tasks. Thus, the state can be estimated with all the possible observations from all task-related agents.

$$\widetilde{s}(t) = s(t-1) \cup_{a_i \in \Gamma_{\mathfrak{t}}(t-1)} \widetilde{c}(a_i, \mathfrak{t})^t$$

where $\mathfrak{t}$ is the task that causes the disturbance and $\widetilde{c}(a_i, \mathfrak{t})^t$ is the estimated capability from the observations.

Action space reduction. In state estimation, task-related agents in $\Gamma_{\mathfrak{t}}$ may get the same observation and their states can be easily estimated. But without communication, we know nothing about other agents, which do not share the same task and their states cannot be derived. Hence there are many possible states for these agents, which make it difficult to reason about their states within critical response time. When we know nothing about other agents and what they know, since the state is Markovian, it is reasonable to assume that their state changes regularly and their choices stay the same. That is, the joint action of the agents in $A - \Gamma_{\mathfrak{t}}$, whose state cannot be derived, is estimated from that of last assignment.

$$\widetilde{Jt\_\tau}(A - \Gamma_{\mathfrak{t}}, T, t) = Jt\_\tau(A - \Gamma_{\mathfrak{t}}, T, t-1)$$

Thus, only the joint action $Jt\_\tau(\Gamma_{\mathfrak{t}}, T, t)$ of the task related agents in $\Gamma_{\mathfrak{t}}$ has to be reallocated, which dramatically reduces the action space. With these two parts, the joint action of the team is approximated:

$$Jt\_\tau(A, T, t) = Jt\_\tau(\Gamma_{\mathfrak{t}}, T, t) \cup \widetilde{Jt\_\tau}(A - \Gamma_{\mathfrak{t}}, T, t)$$

To react in a time critical context, no deliberation model could be used. Therefore, we use greedy policy to find a good joint action $Jt\_\tau(\Gamma_{\mathfrak{t}}, T, t)$ for the task-related agents.

Each agent in the team runs algorithm 1. In this algorithm, when the agent gets an observation (line 2), it triggers a task assignment process. For the agents that have different tasks (line 4), it assumes that their tasks keep the same (line 5). According to the domain knowledge, their contributions to the tasks can be updated (line 7). Thus the total potential contribution each task receives could be summarized (line 8). For the task-related agents (line 10), the contribution to each task can be updated from the observation (line 11). To obtain the highest priority and respond within critical time, a greedy policy is conducted to find the task with max reward for each agent. Considering the task may be over completed, the reward has to be calculated in two cases (line 14-19 and line 20-26). When the task is not over completed, the reward is the additional utility

after adding current agent (line 15). If the reward is the maximum one, record the task and its reward (line 16-17). When the task is over completed after adding current agent (line 20-21), the reward is the additional utility of the task before it exceeds its priority (line 21). And the same is done to record the maximum reward (line 22-23). Finally, find the task with maximum reward and update its status (line 28).

Algorithm 1 Cooperative Decision Algorithm for Time Critical Assignment

1: **for** at time step t **do**
2: $Observation \leftarrow getObservation$;
3: init array $\psi(t)$ with zero;
4: **for** all $\alpha \in A - \Gamma_j(t)$ **do**
5: $\tau_\alpha(t) \leftarrow \tau_\alpha(t-1)$;
6: $task \leftarrow \tau_\alpha(t)$;
7: update $c(\alpha, task)^t$ according to domain knowledge;
8: $\psi_{task}(t) \leftarrow \psi_{task}(t) + c(\alpha, task)^t$;
9: **end for**
10: **for** all $\alpha \in \Gamma_j(t)$ **do**
11: update $c(\alpha,)^t$ according to domain knowledge from $Observation$
12: $max_r \leftarrow 0$;
13: **for** all $task \in T$ **do**
14: **if** $\psi_{task}(t) + c(\alpha, task)^t \leq \Phi_{task}$ **then**
15: **if** $R(\psi_{task}(t) + c(\alpha, task)^t) - R(\psi_{task}(t)) > max_r$ **then**
16: $max_r \leftarrow R(\psi_{task}(t) + c(\alpha, task)^t) - R(\psi_{task}(t))$;
17: $\tau_\alpha(t) \leftarrow task$;
18: **end if**
19: **else**
20: **if** $\psi_{task}(t) \leq \Phi_{task}$ **then**
21: **if** $\chi_{task} - R(\psi_{task}(t)) > max_r$ **then**
22: $max_r \leftarrow \chi_{task} - R(\psi_{task}(t)$;
23: $\tau_\alpha(t) \leftarrow task$;
24: **end if**
25: **end if**
26: **end if**
27: **end for**
28: $\psi_{\tau_\alpha(t)}(t) \leftarrow \psi_{\tau_\alpha(t)}(t) + c(\alpha, \tau_\alpha(t))^t$;
29: **end for**
30: **end for**

4 Experiment Setup and Result

In this section, we present the empirical evaluations of our approach in the cooperative task assignment. In our experiment, a group of agents are deployed from the same base station to carry out several tasks, which takes the agents about 40 time steps to reach the target location. Each task is initialized with

different priorities and workloads. For the sake of simplicity, we initiate agents' capabilities for all tasks to 1. The objective is to maximize the priorities of all tasks accomplished by the agent group. Initially, the agents do not have accurate knowledge about all the tasks and the situation may change. When the agent is heading for the target location, its capabilities can be found changed since the situation is different.

In order to investigate the capability of adapting to the dynamics and uncertainties, we compare our approach (*dynamic approach*) with traditional static assignment policy (*static approach*), which initially assigns the optimal policy to each agent offline. Since the agents are initialized with identical capabilities, 0-1 Integer Programming could be used to find the initial optimal policy.

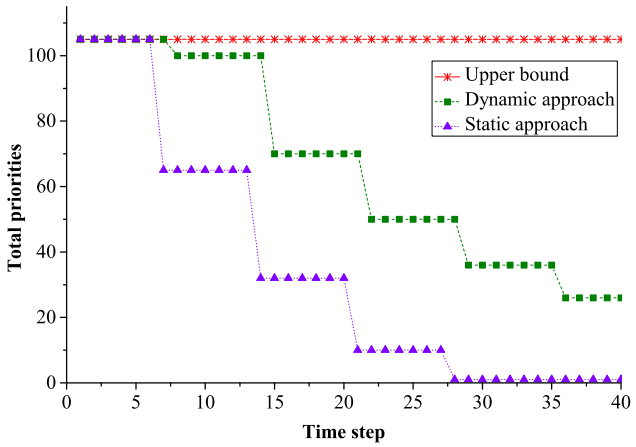
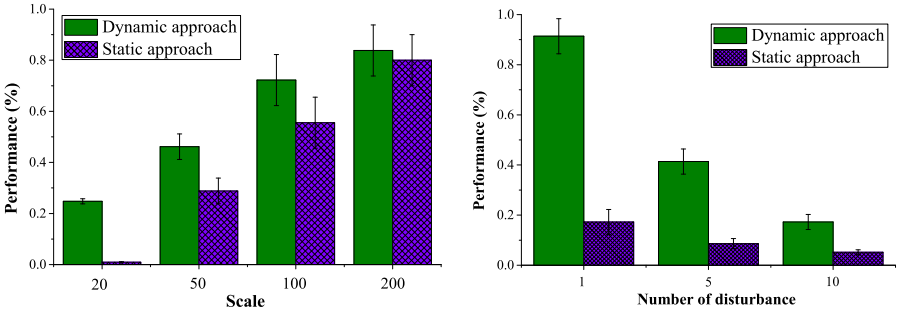


Fig. 1. The experimental results of 20 robots to carry out 20 tasks

To explore the performance of our approach, we initialized 20 agents to carry out 20 tasks with different priorities, and compared our approach with *static approach* and the upper bound. During this experiment, 5 tasks change their resource requirement at different time. Thus, when the agents approach their target locations, they will find their capabilities changed. According to the experimental result in Figure 1, both the total priorities of *dynamic approach* and *static approach* drop when there are unexpected disturbances. However, since *dynamic approach* reasons from the observations and reassigns its task, there is less performance loss in *dynamic approach* than in *static approach*.

In order to investigate the scalability, we vary the number of agents and tasks as 20, 50, 100, and 200 in the experiment in Figure 2(a), and also conduct the experiment with different numbers of disturbances as shown in Figure 2(b). The performance is presented as the ratio of the obtained priorities to the upper bound. From Figure 2(a), when the numbers of agents and tasks scale up,



(a) Results of different scales

(b) Results of different number of disturbances

Fig. 2. The experimental results of different scales and disturbances

given number of disturbances can only influence a smaller part of tasks. Thus, these disturbances have smaller influences on the team performance when the team and tasks scale up. But *dynamic approach* always performs better than *static approach*. When the number of disturbances increases as shown in Figure 2(b), more tasks are influenced and the team performance drops. Because of the reassigning capabilities, *dynamic approach* still outperforms the static one.

5 Conclusion

In this paper, we presented a multi-agent decision algorithm for time critical assignment without communication. In this context, each agent can only obtain a partial observation of the team state and cannot communicate with each other. To reason with partial observations, we built a POMDP model for each agent and proposed heuristic approaches to estimate the state and reduce action space. Our experiments show that our algorithm is feasible to dynamically adjust to disturbances and can be applied in time critical assignment without communication.

References

1. Scerri P., Kannan B., Velagapudi P., Macarthur K., Stone P., et al. Flood disaster mitigation: A real-world challenge problem for multi-agent unmanned surface vehicles. Advanced Agent Technology, p. 252C269, 2012.
2. Day M. Multi-agent task negotiation among UAVs to defend against swarm attacks. PhD thesis, Naval Postgraduate School, Monterey, California. 2012.
3. Luo L., Chakraborty N, Sycara K.: A distributed algorithm for constrained multi-robot task assignment for grouped tasks. Robotics Institute. p. 1077, 2012
4. Kuhn HW.: The hungarian method for the assignment problem. Naval Research Logistics, 52(1):p. 7C21, 2005.

5. Burkard R., DellAmico M., Martello S.: Assignment problems. Cambridge University Press, 2012
6. Bertsekas D. P.: The auction algorithm: A distributed relaxation method for the assignment problem. Annals of operations research14(1):105C123, 1988.
7. Bernstein D. S., Zilberstein S., Immerman N.: The complexity of decentralized control of Markov decision processes. Proceedings of the Sixteenth conference on Uncertainty in artificial intelligence, p. 32-37, 2000.
8. Nair R., Varakantham P., Tambe M., Yokoo M.: Networked distributed pomdps: A synthesis of distributed constraint optimization and pomdps. AAAI, vol. 20, p 133, 2005.
9. Zhang C., Lesser V.: Coordinated multi-agent reinforcement learning in networked distributed pomdps. AAAI, p. 764C770, 2011.
10. Papadimitriou C. H., Tsitsiklis J. N.: The complexity of Markov decision processes. Mathematics of operations research. p. 441-450, 1987.