A POMDP Approach to Token-Based Team Coordination

Yang Xu[+], Paul Scerri[*], Bin Yu[*], Michael Lewis[+] and Katia Sycara[*]

[+] University of Pittsburgh, [*] Carnegie Mellon University
yxu@sis.pitt.edu, {pscerri, byu}@cs.cmu.edu, ml@sis.pitt.edu, katia@cs.cmu.edu

ABSTRACT

Efficient coordination among large numbers of heterogeneous agents promises to revolutionize the way in which some complex tasks, such as responding to urban disasters can be performed. Token-based approaches have shown to be a novel and promising way for such coordination. However, previous token-based algorithms were built on heuristics and did not explicitly consider utilities related to token movements or changes in team states. In this paper we put forward an algorithm that uses team rewards to improve token routing decisions. The ideal solution of this token movement model is a centralized Markov Decision Process (MDP) with joint activity. Unfortunately, the assumptions underlying this model are not feasible for large team coordination and we have to make several approximations. First, we decentralize the centralized MDP as a set of standard MDPs with independent individual activities. Then this MDP is approximated by a Partially Observable Markov Decision Process (POMDP) because agents in a large team may not know the exact states of their teammates or that of the environment. A logical team organization is imposed to limit the token passing among one agent and its neighbors. Belief states of the POMDP model are efficiently estimated using Monte Carlo sampling process.

1. INTRODUCTION

The ability to effectively coordinate large numbers of heterogeneous agents to perform complex tasks in uncertain environments promises to change domains such as the military [23] and disaster response [7]. Effective coordination requires performing a number of functions including allocating tasks and resources, sharing information and recovering from failures.

Previous work on coordination has typically focused on only one specific coordination task, e.g., role or resource allocation [17]. Such an approach precludes the use of knowledge from one type of coordination task to improve the performance of another. For example, knowledge of an agent's task should be able to improve resource allocation. Moreover, these algorithms limited to either task or resource coordination do not scale to very large teams because of expensive requirements such as needing accurate models of all team members [15, 29]. Algorithms that are scalable, often rely on swarm-like behavior that while robust can be inefficient or lead to unpredictable system behaviors [1, 16]. Other approaches have used some degree of centralization to support decentralized activities, e.g., using an auctioneer [25]. Significant progress is needed to develop truly integrated coordination algorithms capable of large scale coordination.

We have proposed an alternative approach to coordination that simultaneously addresses the challenges of scale and of integrating information across various coordination tasks. The approach is based around the concept of a *token* and has been described in [28]. A token encapsulates anything that is shareable by the team, e.g., roles, resources and information. Team members pass tokens around ensuring that information, resource and role tokens reach those team members who need them. Different types of tokens work slightly differently. For example, an actor "remembers" the information contained on the token when it passes on an information token, but when the actor passes on a resource token, it no longer has access to the resource denoted by that token. Our previous approach [28] to where and whether to move a token used heuristic rules. These rules used only local information and information contained within the tokens. Although this approach was demonstrated to be efficient by combining information from different types of coordination to improve routing decisions, it cannot be proven to be optimal because its token routing decision is not based on the expected rewards for the global team state.

This paper develops a mathematical framework for routing tokens. The expected utility of a situation where specific actors hold specific tokens is modelled as a Markov Decision Process(MDP). The policy of this MDP specifies the way tokens should be moved between team members as the environment evolves. For example, role tokens should be passed to those capable actors who have access to required resources. The optimal policy of the MDP defines the joint movement of tokens according to joint activity of team coordination [20] for maximizing the expected utility of the whole team. Although this MDP is optimal, it requires centralized team control which is infeasible for large scale team coordination.

The first step of approximation is to decentralize the team's joint activity into a set of agents' individual activities. Thus,

AAMAS'05, July 25-29, 2005, Utrecht, Netherlands.

the centralized MDP model is decomposed into many standard MDPs whose policies specify the way each single agent should independently move the tokens it possesses, based on the global state of the team and environment. The second step of approximation arises from an agent's incapability of observing the whole team state, especially for large teams. Therefore, the individual MDPs are modelled as Partially Observable MDPs (POMDPs) for each actor of the team. In this model, agents' local views are determined by the tokens they have previously passed. From the observations of their local views, agents maintain a belief state of probability distribution of what the team state should be. We adopt the idea of Q-MDP [3] that agents choose the action to pass a token that is most likely to result in maximum expected team rewards. Notice that the actor's local estimate of state will typically not be very accurate and the optimal policy of the local Q-MDP is only an approximation of the optimal, centralized MDP. However, our previous experiments have shown that such models can dramatically improve team coordination efficiency even when they are not accurate [27].

Although this POMDP model can coordinate large teams in a sub-optimal manner, its efficiency is still low considering the computational cost. Although Q-MDP may be more precise to consider all teammates in finding the best receiver, this can result in high levels of computation particularly for a large team. Therefore, we set up a logical static network across the team, and limit agents to forwarding tokens to their neighbors in this network. As a result, an agent directly receives tokens from only a small number of neighbors in the network and can thus build better models of those agents. By ensuring that the network has a small world property [26], i.e., the distance between any two nodes in the network is small, the effect of these better models outweighs the additional number of "hops" a token might need to take to get where it is required.

Another disadvantage of the POMDP model for coordinating large teams is the need to consider a large number of possible team states to choose the optimal action according to Q-MDP. To make this tractable we decrease the number of possible team states via a Monte Carlo sampling process. The POMDP then computes the optimal action given each of the states represented by the sampling and the actor chooses an action based on the relative weights of the sampling and the actions suggested by the Q-MDP. Given the restricted set of actions available to an actor, e.g., tokens it currently possesses, this Q-MDP can be efficiently computed, especially when the state space is sufficiently abstracted.

2. JOINT INTENTION TEAM COORDINATION

In this section, we provide a detailed model of coordination problem for the multiagent team.

The team coordination problem is defined as: There is a large team of agents $A = \{a_1, a_2, \ldots, a_{|A|}\}$ (a_i represents a specific agent i). They share a top level common goal G. Based on the joint intentions framework [20], G can be realized by achieving a set of joint intention sub-goals $\{g_1, g_2, ..., g_i, ...\}$ in multiple team intention solutions (TIS). Each of the TIS solutions j for g_i is denoted as a tuple $tis_{i,j} =< g_i, p_{i,j}, q_{i,j}, reward_i >$, where

- g_i is the subgoal.
- $p_{i,j} = (event^1_{i,j}, event^2_{i,j}, ...)$ is all the preconditions to activate the solution.
- $q_{i,j} = (role^1_{i,j}, role^2_{i,j}, ...)$ denotes all the joint activities required a single team member to take after $tis_{i,j}$ is activated.
- $reward_i$ will be credited when sub-goal g_i is satisfied.

We define $INF = \{event^1_{1,1}, event^2_{1,1}, ..., event^k_{i,j}, ...\}$ as the set of all possible domain events; $ROLE = \{role^1_{1,1}, role^2_{1,1}, ..., role^k_{i,j}, ...\}$ is the set of potential available activities and $RESOURCE = \{res_1, res_2, ...res_k, ...\}$ is all available exclusive resources in team A.

The capability of agent a_i to perform $role^k_{i,j}$ is mapped as a quantitatively value given by: $Cap(a_i, role^k_{i,j}) \rightarrow [0, 1]$. We also write the resource requirements for a_i to perform $role^k_{i,j}$ as $RequireRes(a_i, role^k_{i,j}) \subseteq RESOURCE$ and the resources that are currently available for a_i as $AvailableRes(a_i) \subseteq RESOURCE$. Whether agent a_i is able to perform $role^k_{i,j}$ depends on its capability and available resources. Formally, the requirements are:

$$perform(a_i, role^k_{i,j}) = ((Cap(a_i, role^k_{i,j}) > 0) \wedge (RequireRes(a_i, role^k_{i,j}) \subseteq AvailableRes(a_i)))$$

For any agent $a_i \in A$, we have $Assign(role^k_{i,j}) = a_i$ if $perform(a_i, role^k_{i,j}) = 1$; otherwise, $Assign(role^k_{i,j}) = Null$. Moreover, each resource and task are exclusive to be shared, and for $\forall a_i, a_j \in A$ we require $AvailableRes(a_i) \cap AvailableRes(a_j) = \phi$. Similarly, for tasks, $perform(a_i, role^k_{i,j}) \wedge perform(a_j, role^k_{i,j}) = 0$.

Based on this joint intention coordination model, team coordination is defined as: $\Xi = INF \cup ROLE \cup RESOURCE$ and has been segmented into pieces $tc \in \Xi$ which is either a precondition, a joint activity or an indispensable resource. By allocating those coordinations to specific agents, i.e., by fusing all the preconditions to a single agent and allocating roles to capable agents with required resources, TIS will be activated and implemented. Therefore, G will be achieved.

The objective of team coordination is to achieve the top level goal G or as many as the sub-goals g_i. Suppose in a period of time, domain events sensed by A are written as $\Upsilon \subseteq INF$. Only part of TIS will be activated by Υ, which is written as $TisValid \subseteq TIS$. Therefore, $\forall tis_{i,j} \in TisValid$, $tis_{i,j}.p_{i,j} \subset \Upsilon$. Then $Complete(tis_{i,j}) = 1$ if $\forall r \in tis_{i,j}.q_{i,j}$, $assign(r) \neq NULL$. This equation requires all its joint activities are assigned to one of the team members. The objective function is to get rewards from implementing all the roles $r \in tis_{i,j}.q_{i,j}$, where $tis_{i,j} \in TisValid$.

Before we discuss how the reward for each role is defined, two important parameters are introduced beforehand.

- $UsefulInf(r) \subseteq INF$ defines all the useful domain events which are not required but helpful for performing r, e.g., knowing which street has been blocked is helpful for performing the role of driving a fire trunk
- $UsefulRes(r) \subseteq RESOURCE$ defines all the non-requested but helpful resources to perform r.

We define the reward for allocating a task r as:

$$U(r) =$$
$$(Cap(Assign(r), r) + 1) \times reward_i \times Complete(tis_{i,j})$$
$$\times (\frac{|UsefulInf(r) \cap AvailableInf(Assign(r))|}{|UsefulInf(r)|} + 1)$$
$$\times (\frac{|UsefulRes(r) \cap AvailableRes(Assign(r))|}{|UsefulRes(r)|} + 1)$$

In this formula, $r \in tis_{i,j}.q_{i,j}$ and $AvailableInf(Assign(r)) \subseteq INF$ are all the domain events previously known by the agent who is performing r. The function shows that the reward of performing a role r depends on $reward_i$ and whether $tis_{i,j}$ is realized. Moreover it is to assign r to an agent who is more capable of performing that role and holding more useful resources and knowing more helpful domain events. Then team coordination objective function is defined as:

$$maximize(\sum_{r \in ValidRoles} U(r) \times d^{|t(r)|})$$

where $ValidRoles = \bigcup_{tis \in TisValid} tis.q_{i,j}$, d is an discount factor and $t(r)$ is the time point when r is allocated.

3. TOKENS FOR COORDINATION

Token-based algorithms for specific tasks have been developed by us and others and have been shown to be effective for specific coordination tasks [27, 17]. However, while these algorithms share the important common feature of being based on tokens, they operate separately. In this paper, we generalize and integrate token-based approaches to make a complete approach to coordination.

We define $Token$ as a data structure for communication messages that encapsulates everything that can be shared in team A. The structure of any token Δ_j is $\Delta_j =< tc, path >$, where $tc \in \Xi$ is a piece of coordination information. According to the nature of tc, we prescribe that Δ_j cannot be duplicated or resented. When an agent is holding Δ_j, it takes the control of $\Delta_j.tc$. The agent will release $\Delta_j.tc$ if Δ_j is passed. To require the uniqueness of coordination in the team, we require $\forall i, j, i \neq j, \Delta_i.tc \neq \Delta_j.tc$. Specially, if $\Delta_j.tc \in INFO$, we call Δ_j as information token and to be clear in presentation, write as Δ_j^I, role token if $\Delta.tc \in ROLE$, write as Δ_j^R or resource token if $\Delta.tc \in RESOURCE$ and write as Δ_j^S. $\Delta_j.path$ records the sequence of agent where Δ_j has been passed.

The basic algorithm for token routing is as Algorithm 1: At each time point, agent a_i will wait for incoming tokens defined as $Tokens(a_i)$ (line 2); For each incoming token Δ_j, it will decide whether this token is acceptable (line 4); If it is, a_i will keep it (line 5) and will try to pass it to the one of its teammates called Next (line 8) otherwise; before passing it, agent a_i will add itself in the token's path (line 7).

Algorithm 1: Decision process for agent a_i to pass incoming tokens

```
1: while true do
2:   Tokens(a_i) ← getToken(sender);
3:   for all Δ_j ∈ Tokens(a_i) do
4:     if Acceptable(a_i, Δ_j) then
5:       Keep(Δ_j)
6:     else
7:       Append(self, Δ_j.path);
8:       RouteToken(Next, Δ_j);
9:     end if
10:  end for
11: end while
```

4. MDP MODEL WITH JOINT ACTIVITY

The issue discussed in the rest of this paper is how to design the algorithm for $RouteToken(Next, \Delta_j)$ if agent a_i does not wish to keep it. Based on our joint intention model, tokens have to be moved to agents who are prepared to receive them, e.g., two events of hearing the fire alarm and seeing a smoke in the same building should be delivered to a single agent who can active a TIS of fire fighting based on those information; the driving a fire truck role needs to be routed to an agent who is capable of driving and a resource token for a fire truck needs to reach the same agent.

The general model for team coordination is a centralized Markov Decision Process (MDP) with joint activity and it is a tuple: $< A, S, \Theta, T, R >$. A is the team to be coordinated; S is the state space and the specific state in time t is defined as $s(t)$; Θ is the joint action space of team A; $T : S \times \Theta \rightarrow S$, is the transition function that describes the resulting state $s(t+1) \in S$ when executing $\theta(t) \in \Theta$ in $s(t)$. $R : S \rightarrow \mathbb{R}$ defines the instantaneous reward for being in a specific state.

In this case, $s(t)$ is modelled as how the exclusive coordination Ξ is distributed in A:

$$s(t) =<< Hold(\Delta_1, t), Hold(\Delta_2, t), Hold(\Delta_3, t), ... >,$$
$$< Know(\Delta_1^I, t), Know(\Delta_2^I, t), Know(\Delta_3^I, t), ... >>$$

where $TOKEN = \{\Delta_1, \Delta_2, \Delta_3, ...\}$ and $INF = \{\Delta_1^I.tc, \Delta_2^I.tc, \Delta_3^I.tc, ...\}$. $s(t)$ includes two parts:

- $Hold(\Delta_i, t) \in A$ directly describes where a token Δ_i is being hold by one of its team member in A at t, e.g., $Hold(\Delta_i, t) = a_j$.
- $Know(\Delta_i^I, t) \subseteq A$ denotes that information token Δ_i^I is known by a few number of team A, e.g., $Know(\Delta_i^I) = \{a_i, a_j, a_k\}$.

Since the tokens represent resources, roles and information, $s(t)$ unambiguously defines who is doing what, with what resources and what information. An initial state $s(0)$ denotes the initial team state, e.g., agents in the team have nothing to do and no environment change is detected.

Team action Θ is a joint activity where team members jointly move tokens they are holding. The joint team action at time point t is defined as $\theta(t) \in \Theta$ which represents all the actions that team members are doing: $\theta(t) = \rho_1(t) \wedge \rho_2(t) \wedge ... \wedge \rho_{|A|}(t)$, where $\rho_i(t)$ is agent a_i's action at time point t. If holds a token, the available action of each agent a_i is to keep it or pass it to any other teammates, $\rho_i(t) = \{\rho_i^{a_i}, \rho_i^c | \forall c \in A, c \neq a_i\}$ where $\rho_i^{a_i}$ denotes a_i will keep the token and ρ_i^c denotes a_i will pass it to a team member c. For convenience, we write $\theta(t)$ as θ and $\rho_i(t)$ as ρ_i when there is no ambiguity.

$R(s(t)) > 0$ when at $s(t)$, a sub-goals $tis_{i,j}$ are achieved. Team will be credited an instant rewards value of

$$R(s(t)) = \sum_{r \in tis_{i,j}.q_{i,j}} U(r)$$

.

The utility of state S under a policy π is defined as

$$v^{\pi}(s) = \sum_{t=0:\infty} (d^t \times R(s(t)) - t \times commcost)$$

where *commcost* is the communication cost and $d < 1$ is a predefined discount factor. $v^*(s)$ allows the agent to select actions according to optimal policy

$$\pi^*(s(t)) = argmax_{\theta \in \Theta} v^*(s(t+1))$$

By value iteration, $v^*(s(t)) = argmax_{\theta \in \Theta}[R(s(t)) - commcost + d \times v^*(s(t+1))]$. Policy π^* tells the team how to control all the agents to move tokens to maximize the team's expected utility.

5. TOWARD APPROXIMATE SOLUTION FOR LARGE TEAM COORDINATION

The MDP model with joint activity is infeasible for large scale teams because an agent can neither get an exact model of S nor know precisely what the other teammates' intentions are. Moreover, the coordination is decentralized. Team members have to coordinate tokens with their own local knowledge in parallel, which is in most cases, an incomplete view of $s(t)$ at any time.

In this section we describe in greater detail the four approximations needed to convert an optimal but infeasible solution for token-based coordination of large scale teams to a computationally tractable approximation solution. First, large team coordination is decentralized and joint activity MDP are approximated as MDPs with individual activities. Second, the MDPs are approximated by POMDPs to avoid requiring complete observations to the team state when the team becomes large. Third, token passing decisions are restricted to a small number of "neighboring" agents. This is based on the assumption that free communication is not available between any two team members in large scale teams. Fourth, a Monte Carlo Sampling process is used to decrease the number of possible states considered by the POMDP which potentially a fast search for the suboptimal action to route tokens.

5.1 Decentralized MDP for token routing

As the first step, we divide the monolithic joint activity into a set of actions that can be taken by individual agents. In this way, The token routing process is decentralized where distributed agents, in parallel, make independent decisions about where to pass the tokens they currently hold. Thus, we effectively break a large coordination problem into many small ones. Then the MDP model of a single agent a_i making token routing decision is a tuple as $< a_i, S, \Theta_i, T', R >$. This model can be applied to any other agents in the team.

The major difference between this MDP model and the joint activity model in previous section is that Θ is replaced with Θ_i. Θ_i is all the available individual activity for agent a_i. For each time t, $\rho_i(t) \in \Theta_i$ has been defined in previous section. Then transition function $T' : S \times \Theta_i \rightarrow S$, team state S will be transited according to Θ_i rather than Θ.

Now the individual optimal policy $\pi^{**}(s(t))$ is defined as:

$$\pi^{**}(s(t)) = argmax_{\rho_i \in \Theta_i} v_I^*(s(t+1))$$

By value iteration,

$$v_I^*(s(t)) = argmax_{\rho_i \in \Theta_i}[R(s(t)) - commcost + d \times v_I^*(s(t+1))]$$

$v_I^*(s(t))$ is the rewards according to individual optimal policy π^{**} and will tell agent a_i to choose action to global state with the maximum expected rewards. In practice, we always choose a relatively small value for d. In this way, the agent can only choose optimal actions at states which are close to states with instant rewards while most of the other states are with a little expected rewards. This helps the MDP to find the optimal action rapidly by only considering a few potential next states with prominent expected rewards. On the other hand, in large team coordination, the number of these candidate next states may still be extremely high.

5.2 Partial Observation Markov Decision Process

Token-based coordination is a process by which agents attempt to maximize the overall team reward by moving tokens around the team. If an agent were to know the exact state of the team, it could use an MDP to determine the expected utility maximizing way to move tokens. Unfortunately, it is infeasible for an agent to know the complete state. [15] it is illustrative to look at how tokens would be passed if it were feasible. In this section, we did the second step of approximation, where agents do not have complete map of team state $s(t)$ and have to make their decision according to its local state. This model is defined as $< S, L, \Theta_i, Z, O, T', R >$.

L maintains the local model of agent a_i and at each time t, it is defined as $l_{a_i}(t) =< tokens(a_i), h_{a_i}(t) >$, where history $h_{a_i}(t)$ includes all the tokens agent a_i previously passe. if $a_i \in \Delta.path$, $\Delta \in h_{a_i}$. As defined in section 3, $tokens(a_i)$ are all the tokens currently hold by a_i.

Observation function is $O : L \rightarrow Z$ where Z maintains an local observation from L. Each observation at time t is $z_{a_i}(t)$ and it will include two parts:

$z_{a_i}(t) =$

$<< PrevHold(\Delta_1, \Delta_1.path), PrevHold(\Delta_2, \Delta_2.path), \ldots, >,$

$< PrevKnow(\Delta_1^I, \Delta_1^I.path), PrevKnow(\Delta_2^I, \Delta_2^I.path), \ldots >>$

$\forall \Delta_j \in h_{a_i}(t)$, $PrevHold(\Delta_j, \Delta_j.path)$ denotes that a_i has observed that all the agents in $\Delta_j.path$ have previously hold Δ_j. In the other part, $\forall \Delta_j^I$, $\Delta_j^I.tc \in INF$ and $PrevKnow(\Delta_j^I,$
$\Delta_j^I.path)$ denotes that the information Δ_j^I encapsulated has been known by the agents that Δ_j^I has previously reached.

We adopt a standard POMDP technique called Q-MDP [3, 9] and use it to solve the POMDP to determine optimal token routing. In this solution, a_i's individual belief $b_{a_i}(t)$ is defined as a set of possible team state $b_{a_i}(t) \subseteq S$. This denotes that the agent a_i believes the previously possible team state is directly from its observation $z_{a_i}(t)$.

The mapping function is defined as

$$z_{a_i}(t) \rightarrow b_{a_i}(t)$$

It is a peer to peer function and will exclude all the $s(t) \in S$ that is compatible with $z_{a_i}(t)$. For example, a_i observes that Δ_j is held by a_i and all the states that denote a_i is not holding Δ_j will be excluded.

For each state $s(t) \in b_{a_i}(t)$, we supposed, for each agent a_i, we know the perceptual distribution $Pr(s(t)|z_{a_i}(t))$, which describes the likelihood of the team being in the state of $s(t)$ when its observation is $z_{a_i}(t)$ and $\sum_{s(t) \in b_{a_i}} Pr(s(t)|z_{a_i}(t)) =$ 1. Initially we have $Pr(z_{a_i}(t)|s(t)) = \frac{1}{|b_{a_i}(t)|}$ when none of

the $s(t) \in b_{a_i}(t)$ is prominent. Then we can calculate the expected reward of each observation $z_{a_i}(t)$ as:

$$\begin{aligned} R'(z_{a_i}(t)) &= R'(b_{a_i}(t)) \\ &= \sum_{s(t)\in b_{a_i}} Pr(s(t)) \times (v_I^*(s(t))) \\ &= \sum_{s(t)\in b_{a_i}} Pr(s(t)|z_{a_i}(t)) \times (v_I^*(s(t))) \end{aligned}$$

Although the transition function T' is the same as previous MDP for decentralized model, the local policy under POMDP π_P^* is to select the action Θ_i to

$$\begin{aligned} &argmax_{\rho_{a_i}(t)} R'(z_{a_i}(t+1)) \\ &= argmax_{\rho_{a_i}(t)} R'(b_{a_i}(t+1)) \\ &= argmax_{\rho_{a_i}(t)} \sum_{s(t+1)\in b_{a_i}(t+1)} Pr(s(t+1)|z_{a_i}(t+1)) \\ &\times v_I^*(s(t+1)) \end{aligned}$$

This formula is based on the assumption that we can online or off-line learn from the policy π^{**} in the previous decentralized MDP model and know the optimal team reward $v_I^*(s(t))$ for each state $s(t)$. Then Q-MDP will always choose an activity to pass a token to a team member who is the most likely maximize the team rewards from agent a_i's local observation $z_{a_i}(t)$.

5.3 Associate Network

Although the Q-MDP approach can theoretically be computed to solve our POMDP problem, a major difficulty for system design is that Q-MDP has to consider every teammate as a potential receiver for every incoming token. This will result a very high volume of computations. Moreover, in some likely application domains free communications may not be available between all pairs of agents due to bandwidth, proximity, or other limitations. One solution is to limit the teammates eligible to receive a token. Therefore, in this section, we impose a logical team organization as an *associate network* where each agent is able to pass a token to only a few of its teammates.

The *associate network* is an undirected graph $G = (A, N)$, where A is the team of agents and N is the set of associate relationship between two agents. For any two agents $a_i, a_j \in A$, $< a_i, a_j > \in N$ denotes that a_i and a_j are associates and are able to exchange tokens directly. Specifically, $n(a_i)$ is defined as all the associates of agent a_i and $|n(a_i)| << |A|$. A subset of a typical associate network for a large team is shown as Figure 1. In the Figure, each node represents a team member and when pairs of agents are connected by a line, they can exchange tokens with each other directly.

We rewrite this POMDP model as $< S, L, \Theta_i^*, Z, O, T'', R >$ where $T'' : S \times \Theta_i^* \to S$. But the major difference is that the available action $\rho_{a_i}^*(t) \in \Theta_i^*$ of each agent a_i is changed to keep it or pass it to one of its associates as

$$\rho_{a_i}^*(t) = \{\rho_i^{a_i}, \rho_i^c | \forall c \in n(a_i)\}$$

.

Then the optimal policy π_P^{**} under this associate network is to

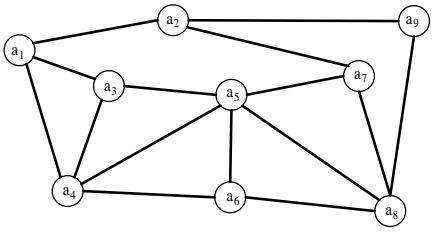

Figure 1: An example of a subset of a typical associate network.

$$\begin{aligned} &argmax_{\rho_{a_i}^*(t)} R'(z_{a_i}(t+1)) \\ &= argmax_{\rho_{a_i}^*(t)} \sum_{s(t+1)\in b_{a_i}(t+1)} Pr(s(t+1)|z_{a_i}(t+1)) \\ &\times (v_I^*(s(t+1))) \end{aligned}$$

5.4 Monte Carlo Sampling

Our Q-MDP solution is still inefficient because before agents can send out any tokens, they must search and calculate all the possible team states consistent with their belief states. Monte Carlo sampling [21] is helpful in reducing this search by sampling only a small number of possible states from agents' beliefs. Suppose each agent a_i takes M samples. At time t, each sampling k will randomly sample from a possible belief state $b_{a_i}(t)$ as $s^k(t)$. These particles will provide an estimation of $b_{a_i}(t)$ as $b'_{a_i}(t) = \{s^1(t), s^2(t), ..., s^M(t)\}$. Then $R'(z_{a_i}(t))$ is replaced with

$$R'(z'_{a_i}(t)) = R'(b'_{a_i}(t)) = \sum_{1:M} Pr^*(s^i(t)|z_{a_i}(t)) \times (v_I^*(s^i(t)))$$

where each $Pr^*(s^i(t)|z_{a_i}(t)), 1 \le i \le M$, is normalized from $\{Pr(s^1(t)|z_{a_i}(t)), Pr(s^2(t)|z_{a_i}(t)), ..., Pr(s^M(t)|z_{a_i}(t))\}$. If $Pr(s^i(t)|z_{a_i}(t)) = \frac{1}{|b_{a_i}(t)|}$, then $Pr^*(s^i(t)|z_{a_i}(t)) = \frac{1}{M}$.

The sub-optimal policy π_P^{***} now is to:

$$\begin{aligned} &argmax_{\rho_{a_i}^*(t)} R'(z'_{a_i}(t+1)) = \\ &argmax_{\rho_{a_i}^*(t)} \sum_{i=1:M} Pr^*(s^i(t+1)|z_{a_i}(t+1)) \times v_I^*(s^i(t+1)) \end{aligned}$$

This policy will not choose the optimal action to route a token by computing all the possible team states in the belief state $b_{a_i}(t)$ but $b'_{a_i}(t)$ which is an approximation from M sample states.

Now we can summary our token coordination algorithm for agent a_i to route Δ_j as algorithm 2. In the initial time, agent a_i didn't get any token and its history is empty (line 2). After that, agent's history will record each incoming token Δ_j (line 4) and its observation $z_{a_i}(t)$ will be updated according to its local model $l_{a_i}(t)$ (line 5). For each available act_j for agent a_i, it will result $z_{a_i}(t)$ to $z_{a_i}^j(t+1)$ (line 7) and new belief state $b_{a_i}^j(t+1)$ (line 8). By applying a Monte Carlo Sampling process, we sample M states from belief $b_{a_i}^j(t+1)$ (line 9) and calculate $\sum_{i=1:M} Pr^*(s^i(t+1)|z_{a_i}(t+1)) \times v_I^*(s^i(t+1))$ (line 10). Then the policy is to choose the best action according to π_P^{***} (line 12).

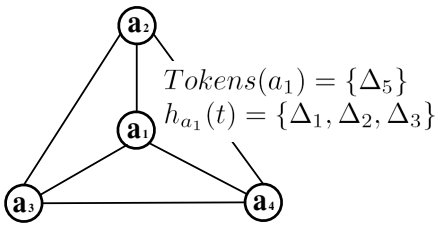

Figure 2: An example of token routing algorithm for team $\{a_1, a_2, a_3, a_4\}$.

Algorithm 2: $RoutToken(Next, \Delta_j)$

if $t == 0$ **then**
  $h_{a_i}(0) = \Phi$;
else
  $h_{a_i}(t) \leftarrow \Delta_j$;
  Update $z_{a_i}(t)$ according to $l_{a_i}(t)$;
  for all $act_j \in \rho^*_{a_i}(t)$ **do**
    $z_{a_i}(t) \times act_j \rightarrow z^j_{a_i}(t+1)$;
    $z^j_{a_i}(t+1) \rightarrow b^j_{a_i}(t+1)$;
    Randomly sample M states from $b^j_{a_i}(t+1)$;
    Calculate $\sum_{i=1:M} Pr^*(s^i(t+1)|z_{a_i}(t+1)) \times v^*_I(s^i(t+1))$
  end for
  Choose $Next$ according to π^{***}_P;
end if

To provide a numerical example of this algorithm, we suppose there is a multiagent team as shown in figure 2 and $n(a_1) = \{a_2, a_3, a_4\}$. At time t, $l_{a_i}(t) =< \{\Delta_5\}, \{\Delta_1, \Delta_2, \Delta_3\} >$; $\Delta_1 =< event_1, \{a_2\} >$; $\Delta_2 =< event_2, \{a_2\} >$; $\Delta_3 =< res_1, \phi\} >$; $\Delta_5 =< event_1, \{a_3, a_4\} >$. The TIS related with $\Delta_1, \Delta_2, \Delta_3, \Delta_5$ is $tis_{1,1} =< g_1, \{event_1, event_2, event_3\}, \{role_1, role_2\}, 100 >$. Moreover, Agent a_1 will test 2 samples at each time, and we supposed that $\forall i = 1, 2, Pr^*(s^i(t+1)|z_{a_i}(t+1)) = 0.5$.

Then $z_{a_1}(t) =<< PrevHold(\Delta_1, \{a_2\}), PrevHold(\Delta_2, \{a_2\}), PrevHold(\Delta_3, \phi\}, PrevHold(\Delta_5, \{a_3, a_4\} >, < PrevKnow(\Delta_1, \{a_2\}), PrevKnow(\Delta_2, \{a_2\}, PrevKnow(\Delta_5, \{a_3, a_4\} >$. Since $n(a_1) = \{a_2, a_3, a_4\}$, $\rho^*_{a_1}(t) = \{\rho^{a_1}_1, \rho^{a_2}_1, \rho^{a_3}_1, \rho^{a_4}_1\}$.

If agent a_1 takes $\rho^{a_1}_1$, $z_{a_1}(t+1) =<< PrevHold(\Delta_1, \{a_2\}), PrevHold(\Delta_2, \{a_2\}, PrevHold(\Delta_3, \phi\}, PrevHold(\Delta_5, \{a_3, a_4, a_1\} >, < PrevKnow(\Delta_1, \{a_2\}), PrevKnow(\Delta_2, \{a_2\}), PrevKnow(\Delta_5, \{a_3, a_4, a_1\}) >$. By sampling from $b_{a_1}(t+1)$, suppose that $s^1(t+1) =<< a_3, a_4, a_4, a_1 >, < \{a_2\}, \{a_2, a_3\}, \{a_3, a_4, a_1\} >>$ and $s^2(t+1) =<< a_3, a_2, a_4, a_1 >, < \{a_2\}, \{a_2\}, \{a_3, a_4, a_1\} >>$ and we have known $v^*_I(s^1(t+1)) = 25$ and $v^*_I(s^2(t+1)) = 15$. Then $R'(z'_{a_i}(t+1)) = 20$.

If agent a_1 takes $\rho^{a_2}_1$, $z_{a_1}(t+1) =<< PrevHold(\Delta_1, \{a_2\}), PrevHold(\Delta_2, \{a_2\}), PrevHold(\Delta_3, \phi\}), PrevHold(\Delta_5, \{a_3, a_4, a_2\}) >, < PrevKnow(\Delta_1, \{a_2\}), PrevKnow(\Delta_2, \{a_2\}), PrevKnow(\Delta_5, \{a_3, a_4, a_2\}) >$. By sampling from $b_{a_1}(t+1)$, suppose that $s^1(t+1) =<< a_3, a_4, a_4, a_2 >, < \{a_2\}, \{a_2, a_3\}, \{a_3, a_4, a_2\} >>$ and $s^2(t+1) =<< a_3, a_2, a_2, a_2 >, < \{a_2, a_3\}, \{a_2, a_4\}, \{a_3, a_4, a_2\} >>$ and we have known $v^*_I(s^1(t+1)) = 35$ and $v^*_I(s^2(t+1)) = 35$. Then $R'(z'_{a_i}(t+1)) = 35$.

For the limited length for this paper, we omit the following calculation and the results are: if agent a_1 takes $\rho^{a_3}_1$, $R'(z'_{a_i}(t+1)) = 15$; if agent a_1 takes $\rho^{a_4}_1$, $R'(z'_{a_i}(t+1)) = 22$. Obviously, a_1 will choose to pass Δ_5 to a_2 to get the maximum expected reward.

6. RELATED WORK

Multiagent coordination is an extensively studied area of multiagent systems, but most of the existing work does not scale well to very large teams. Distributed constraint-based algorithms[10, 11] have high communication requirements that get dramatically worse as the team size increased. Combinatorial auctions[6] have an exponential number of possible combinations of bids, and frequently use centralized auctioneers that can become severe bottlenecks. Swarm-inspired approaches[4] have been used for large-scale coordination, but the behavior can be inefficient.

Recent work focusing on scalable coordination[19] illustrates that exponential search spaces, excessive communication demands, localized views, and incomplete information of agents pose major problems for large scale systems. Initial work on token-based approaches promises a way to address these challenges. Large scale coordination in the GPGP/TAEMS framework[8] was demonstrated using a token-based algorithm[24]. The effectiveness of large-scale, token-based coordination has also been demonstrated in the Machinetta proxy architecture[18] for task allocation[17] and information sharing[27].

Decision theoretic approaches, such as MDPs and POMDPs, have been used for team coordination[5, 15, 29]. Because centralized control is frequently not possible, the team coordination problem is recast as a communication problem[15]. However, such approaches are known to be intractable in general[15], and the presence of free communication at every time step transform a multiagent POMDP into a more tractable single agent POMDP[12]. Q-MDP has been approved to be an efficient solution for this problem [2, 3]. Thrun solved the POMDP problem in robot navigation by using Monte Carlo filter [21] or particle filters [22]. [13] synthesized a distributed POMDP problem for multiagent teamwork with a distributed constraint optimization (DCOP) algorithm. [14] designed a fast online POMDP algorithm for robot coordination to explore complex environment by using state compression via factor POMDP.

7. CONCLUSION

In this paper we have described a formal mathematical decision model of a token based team coordination algorithm. We have shown how the joint activity MDP model which only fits small teams, can be approximated through decentralized MDP and POMDPs to coordinate large scale teams. This model allows us to explicitly reason about the movement of tokens around the team. Moreover, further approximating by associate network and Monte Carlo sampling process can enhance the efficiency of our algorithm to find the optimal policy.

While the work presented here represents a step forward, it also points to significant challenges that we plan to address in the near future. First, experiments and high level realistic domains simulations, i.e., urban search and rescue response and unmanned aerial vehicle applications will be run to explore the performance characteristics and limitations of our approach. Moreover, we will examine how the associate network topologies such as small world network

and scale free network will influence the token routing behavior.

Acknowledgements

This research was supported by AFOSR under grant No. F49620-01-1-0542 and by AFRL/MNK grant No. F08630-03-1-0005.

8. REFERENCES

[1] W. Agassounon and A. Martinoli. Efficiency and robustness of threshold-based distributed allocation algorithms in multiagent systems. In *Proceedings of AAMAS'02*, 2002.

[2] G. Apostolikas and S. Tzafestas. Improved Qmdp policy for partially observable Markov Decision Processs in large domain: Embedding exploration dynamics. In *Intelligent Automation and Soft Computing*, 2004.

[3] A. Cassandra, M. Littman and N. Zhang Incremental pruning: a simple, fast, exact method for partially observable Markov decision processes. *Proceedings of the Thirteenth Annual Conference on Uncertainty in Artificial Intelligence*, 1997.

[4] V. Cicirello and S. Smith. Wasp nests for self-configurable factories. In *Proceedings of the Fifth International Conference on Autonomous Agents*, 2001.

[5] C. Goldman and S. Zilberstein. Optimizing information exchange in cooperative multi-agent systems. In *Proceedings of AAMAS'03*, 2003.

[6] L. Hunsberger and B. Grosz. A combinatorial auction for collaborative planning. In *Proceedings of the Fourth International Conference on Multi-Agent Systems*, pages 151 – 158, 2000.

[7] H. Kitano, S. Tadokoro, I. Noda, H. Matsubara, S. Takahashi, A. Shinjoh and S. Shimada. Robocup rescue: Search and rescue in large-scale disasters as a domain for autonomous agents research. In *IEEE International Conference on Systems, Man and Cybernetics*, volume VI, pages 739–743, 1999.

[8] V. Lesser, K. Decker, T. Wagner, N. Carver, A. Garvey, B. Horling, D. Neiman, R. Podorozhny, M. NagendraPrasad, A. Raja, R. Vincent, P. Xuan and X. Zhang. Evolution of the GPGP/TEAMS domain-independent coordination framework. *Autonomous Agents and Multi-Agent Systems*, 9, 2004.

[9] M. Littman, A. Cassandra, and L. Kaelbling. Learning policies for partially observable environments: scaling up. In *International Conference on Machine Learning*, 1995.

[10] R. Mailler and V. Lesser. Solving distributed constraint optimization problems using cooperative mediation. In *Proceedings of AAMAS'04*, 2004.

[11] P. Modi, W. Shen, M. Tambe and M. Yokoo. An asynchronous complete method for distributed constraint optimization. In *Proceedings of AAMAS'03*, 2003.

[12] R. Nair, M. Roth, M. Yokoo and M. Tambe. Communication for improving policy computation in distributed POMDPs. In *Proceedings of AAMAS'04*, 2004.

[13] R. Nair, P. Varakantham, M. Tambe and M. Yokoo. Networked distributed POMDPs: A synthesis of distributed constraint optimization and POMDPs. In *Proceedings of the Twentieth National Conference on Artificial Intelligence*, 2005.

[14] S. Paquet, L. Tobin and B. Chaib-draa An Online POMDP Algorithm for Complex Multiagent Environments In *Proceedings of AAMAS'05*, Forthcoming, 2005.

[15] D. Pynadath and M. Tambe. The communicative multiagent team decision problem: analyzing teamwork theories and models. In *Journal of AI research*, 2002.

[16] P. Sander, D. Peleshchuk and B. Grosz. A scalable, distributed algorithm for efficient task allocation. In *Proceedings of AAMAS'02*, 2002.

[17] P. Scerri, A. Farinelli, S. Okamoto and M. Tambe. Allocating tasks in extreme teams. In *Proceedings of AAMAS'05*, Forthcoming, 2005.

[18] P. Scerri, D. Pynadath, L. Johnson, P. Rosenbloom, N. Schurr, M. Shi and M. Tambe. A prototype infrastructure for distributed robot-agent-person teams. In *Proceedings of AAMAS'03*, 2003.

[19] P. Scerri, R. Vincent and R. Mailler. Comparing three approaches to large scale coordination. In *AAMAS'04 Workshop on Challenges in the Coordination of Large Scale MultiAgent Systems*, 2004.

[20] M. Tambe. Towards flexible teamwork. *Journal of AI research*, 7: 83–124, 1997.

[21] S. Thrun. Monte carlo POMDPs. In *Proceedings of NIPS*, 2000.

[22] S. Thrun. Particle filters in robotics. In *Proceedings of the 17th Annual Conference on Uncertainty in AI*, 2002.

[23] T. Wagner, J. Phelps, V. Guralnik and R. VanRiper. Coordinators: Coordination managers for first responders. In *Proceedings of AAMAS'04*, 2004.

[24] T. Wagner, V. Guralnik and J. Phelps. A key-based coordination algorithm for dynamic readiness and repair service coordination. In *Proceedings of AAMAS'03*, 2003.

[25] W. Walsh and M. Wellman. A market protocol for decentralized task allocation. In *Third International Conference on Multi-Agent Systems*, pages 325–332, 1998.

[26] D. Watts and S. Strogatz. Collective dynamics of small world networks. *Nature*, 393: 440–442, 1998.

[27] Y. Xu, M. Lewis, K. Sycara and P. Scerri. Information sharing in large scale teams. In *AAMAS'04 Workshop on Challenges in Coordination of Large Scale MultiAgent Systems*, 2004.

[28] Y. Xu, P. Scerri, B. Yu, S. Okamoto, M. Lewis, and K. Sycara. An integrated token-based algorithm for scalable coordination. In *Proceedings of AAMAS'05*, Forthcoming, 2005.

[29] P. Xuan, V. Lesser and S. Zilberstein. Communication decisions in multi-agent cooperation: Model and experiments. In *Proceedings of the Fifth International Conference on Autonomous Agents*, 2001.