

---

# Token-Based Approach for Scalable Team Coordination

Yang Xu, Paul Scerri, Michael Lewis and Katia Sycara

**Summary.** 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. However, state of the art coordination algorithms are not capable of achieving efficient and effective coordination when a is very large. Building on recent successful token-based algorithms for task allocation and information sharing, we have developed an integrated and efficient approach to effective coordination of large scale teams. We use *tokens* to encapsulate anything that needs to be shared by the team, including information, tasks and resources. The tokens are efficiently routed through the team via the use of local decision theoretic models. Each is used to improve the routing of other tokens leading to a dramatic performance improvement when the algorithms work together. We present results from an implementation of this approach which demonstrates its ability to coordinate large teams. By comparing with the market-based approach, we demonstrated how the token-based approach has made a good trade off between team performance and communication cost.

## 1 Introduction

Efficient and flexible coordination among large numbers of robots, agents and people promises to revolutionize the achievement of complex and distributed tasks. In domains such as disaster response [13], the military [29] and business organizations [9], decentralized cooperative coordination can dramatically reduce costs and improve efficiency while lowering risks and improving safety. In these applications, a large number of heterogeneous agents need to coordinate in a dynamic, uncertain environment and adjust their activities according to the status of the team and their teammates. Typically, coordination requires tasks including plan monitoring, information delivery, role allocation and resource sharing.

Previous work on coordination has typically focused on only one specific coordination task, e.g. role allocation [24] or planning [11], precluding the use of knowledge from other aspects of coordination being used to improve the performance of that algorithm. For example, results of the task allocation process have not been used to guide resource allocation, although intuitively they will improve the search.

On the other hand, even algorithms designed for single coordination tasks do not scale well to very large teams. The major challenges include communication limitation, decentralized control and incomplete knowledge for decision support. Existing approaches, which are successful in small-team coordination when apply to large teams [14, 28], are incapable of making decisions under incomplete team knowledge with the communication limitations. Decentralized communication decision approaches require accurate models of all team members which is infeasible for agents in a large team [22, 34]. Algorithms that are scalable, often rely on swarm-like behavior that, while robust, can be very inefficient [6]. Other approaches, e.g., using an auctioneer [11], require some degree of centralization which is not always desirable. Although rapid progress has been made in developing coordination algorithms [10], teamwork algorithms that scale to large numbers of agents while remaining efficient, distributed and flexible are not yet available.

In this chapter, we present an integrated and scalable approach to coordinating a large number of heterogeneous agents. Three novel ideas underlie this approach. The first idea is to encapsulate *all* coordination interactions, including information, assignable tasks and sharable resources within *tokens*. The agent holding the token has exclusive control over whatever is represented by that token, hence tokens provide a type of access control. Agents either keep tokens or pass them to teammates. For example, an agent holding a resource token has exclusive access to the resource represented by that token and passes the token on to transfer access to that resource. The resulting movement of tokens implements the coordination by distributing information, resources and tasks with low communication overhead.

The second novel idea is for agents to use local decision theoretic models to determine when and where to pass tokens. When an agent passes a token to another agent, that exchange is used to refine local models of the team. These models are used in a decision theoretic way to determine whether and where to forward any token the agent currently holds, so as to maximize the expected utility of the team. Informally, agents will try to pass tokens to where they help team performance the most by inferring from their local models which team member will either have use for the information, resource or task represented by the token or be in the best position to know who will. A logical static network across the team, limits 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 [16], 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.

The third novel idea in this work is to leverage *all* available information for creating models of the team, specifically using the movement of one token to inform the movement of other tokens. This synergistically integrates the execution of key coordination algorithms in a way not done before. For

example, tokens representing resources useful for a particular task should be passed to the same agent as the token representing that task was. Intuitively, making use of the relationship between tokens, each coordination task becomes more efficient because it focuses its search based on the progress of other coordination tasks.

In the remainder of this chapter we describe how tokens are routed around the network to maximize the expected utility of the team. Specifically, we begin with a Markov Decision Process (MDP) model based on the full observation of team state then make a series of approximations to develop efficient, local reasoning for routing the tokens. To test our approach, there are two groups of experiments. In the first group, the results show that the local routing models lead to a dramatic improvement in coordination performance and excluding any type of token from the development of the local reasoning models decreases performance building on previously described individual token-based algorithms [24, 12, 27, 31]. In the second group of experiments, we systematically and scientifically compare our token-based coordination with the market-based coordination, a popular, centralized algorithm to find the optimal coordination solution. Our experiment results meet the hypothesis based on the nature of each approach. Auctions are focused on maximizing overall utility taking into account the *bids* of all team members [2]. Token-algorithm is focused on scalability, hence it minimizes communication, sometimes at the expense of overall utility.

## 2 Large Scale Coordination

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

### 2.1 Problem Description

Coordination is required between a team  $A = \{\alpha_1, \alpha_2, \dots, \alpha_n\}$  of agents that share a top level common goal  $G$  (as in [28]). Achieving  $G$  requires achieving a number of sub-goals  $\{g_1, g_2, \dots, g_i, \dots\}$ . When sub-goal  $g_i$  is satisfied the team receives a reward  $reward_i$ . For example, sub-goals of a high level goal to respond to a disaster might be to extinguish fires and provide medical attention to injured civilians. To satisfy sub-goals, the team follows plan templates  $Plan = \{plan_1, plan_2, \dots, plan_i, \dots\}$  represented in a library. Each template  $i$  includes four parts and is written as  $plan_i = \langle g_i, conditions_i, roles_i, reward_i \rangle$ . The first element is the sub-goal  $g_i$ ; the second is the conditions under which it is applicable,  $conditions_i = event_1 \cap event_2 \cap \dots \cap event_i$ ; the third element is the individual roles  $roles_i = \{r_1, r_2, \dots, r_k\}$  which are required to achieve  $g_i$  and the last part, the  $reward_i$  is to be received by the team on successful satisfaction of  $g_i$ . Each role

$r_i = \langle task_i, ability_i, resource_i \rangle$  is represented by its task, i.e., a description of the actual thing to be done, the capabilities required to perform that task and the resources needed to perform the role.

For example, a fire fighting template can be defined as:  $\langle plan_{fire} = (\text{Fight fire at location X}), (\text{Fire alarm at X} \cap \text{Smoke at X}), \{r_1, r_2, r_3\}, (100) \rangle$ . This template requires two conditions before it is initiated: a fire alarm and smoke. After this plan is initiated, three roles,  $\{r_1, r_2, r_3\}$  need to be assigned and a reward 100 will be credited to the team. The three roles in this template are: driving the fire truck, fighting the fire and searching for victims, i.e.,  $r_1 = \langle (\text{Driving the fire truck}), (\text{Skillful in driving truck}), (\text{Fire truck}) \rangle$ ,  $r_2 = \langle (\text{fighting the fire}), (\text{Have training in fire fighting}), (\text{Hose, water}) \rangle$  and  $r_3 = \langle (\text{Searching for victims}), (\text{None}), (\text{Breathing equipment}) \rangle$ . To perform  $r_1$ , an agent is required to be able to drive and have access to a fire truck which is an exclusive resource.

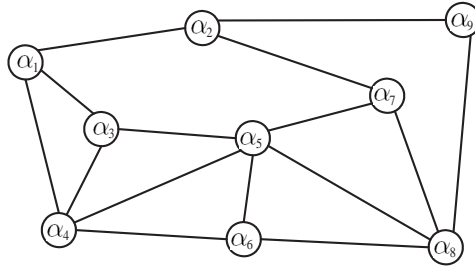
## 2.2 Acquaintance Network

It has been observed that in a human group, members typically maintain a small number of acquaintances but can rapidly transmit information to any member of the group in a series of hops, a phenomenon known as a *small world effect* [16]. The most popular manifestation of this phenomenon is the *six degrees of separation* concept [19]. Milgram concluded that there is a path of acquaintances with typical length six between any two people in the United States. By using very vague (and often incorrect) information about other members of the population, people will pass a message to someone better placed to find the intended recipient until the information reaches the desired recipient.

Inspired by such social networks, we arrange the team as an *acquaintance network*. The *acquaintance* is a graph  $G = (A, N)$ , where A is the team of agents and N is the set of links between agents. Specifically, for  $\alpha_i, \alpha_j \in A$ ,  $\langle \alpha_i, \alpha_j \rangle \in N$  denotes that  $\alpha_i$  and  $\alpha_j$  are acquaintances and are able to exchange tokens directly.  $n(\alpha)$  is defined as all the acquaintances of agent  $\alpha$ . Note that  $n(\alpha) \ll |A|$ . We additionally require that the acquaintance network be a *small world network*, which means that a relatively small number of links separate any two agents in comparison to a regular grid network. Previous work has shown that such networks lead to better performance of token-based algorithms [31]. A subset of a typical acquaintance 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.

## 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 tasks [31, 24]. *Control*



**Fig. 1.** An example of a subset of a typical acquaintance network.

*information* is included in the token to help the agents determine what to do within the token. For example, for information sharing the control information is the number of "hops" a token can move before stopping [31]. For task allocation, the control information is the minimum capability an agent must have to accept a task [24]. However, while these algorithms share the important common feature of being based on tokens, they operate separately. In this chapter, we generalize and integrate token-based approaches to make a complete approach to coordination.

Let  $\Gamma = \{\Delta_1, \Delta_2, \dots, \Delta_m\}$  be all types of coordination tokens. These tokens can be classified into three basic types: information tokens, roles tokens and resources tokens. Each token  $\Delta_i$  is defined as a tuple with four elements:  $\Delta_i = \langle Type, Coordination, Path, Threshold \rangle$ .  $Type = \{inf, role, res\}$  denotes the type of coordination; it contains information, role or resource, respectively.  $Coordination$  captures the specific coordination element represented by this token. In the case of an information token, it is the information to be shared. In the case of a resource token, it is a description of the resource to which this token grants exclusive access. In the case of a role token, it is a description of the task for which the acceptor of this token is responsible.  $Path$  records the route the token has taken through the network.  $\Delta.path$  is also used as stop condition for information and role tokens when  $|\Delta.path| > TTL$  where  $TTL$  is empirically set to be the maximum number of "hops" that  $\Delta$  is allowed to be passed.  $Threshold$  generalizes the *control information* for resource and role tokens but is not required for information tokens. An agent may keep a resource if its need for that resource is greater than the token's *threshold*. Determining an agent's requirement for a resource is outside of the scope of this chapter. While an agent holds a resource token  $\Delta$ ,  $\Delta.threshold$  slowly increases up to some maximum. When the token is passed,  $\Delta.threshold$  is decreased to avoid the token being passed indefinitely. This mechanism ensures that resources can flow through the team resource. For example, to coordinate a team of unmanned aerial vehicles (UAVs) holding tokens representing airspace, might be forced to relinquish tokens to the longest held regions as their thresholds increased unless the airspace was crit-

ical. Similarly, a role token  $\Delta$  will be accepted by an agent whose capability is greater than  $\Delta.threshold$  and its *threshold* will be decreased if it has not been accepted. The role allocation algorithm is described in detailed in [24].

## 4 Routing Tokens

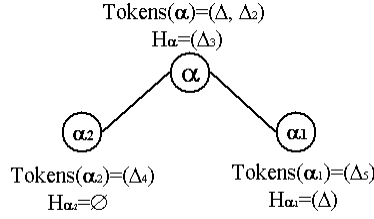
Token-based coordination is a process by which agents attempt to maximize the overall team reward by moving tokens around the team. If an agent were to know the exact state of the team, it could use an MDP to determine the expected utility maximizing way to move tokens. Unfortunately, it is infeasible for an agent to know the complete state, however, [22] it is illustrative to look at how tokens would be passed if it were feasible. Then, by dividing the monolithic joint activity into a set of actions that can be taken by individual agents, we can decentralize the token routing process where distributed agents, in parallel, make independent decisions of where to pass the tokens they currently hold. Thus, we effectively break a large coordination problem into many small ones.

### 4.1 MDP Model for Complete Team State

The basic decision model of agent  $\alpha$  for a token  $\Delta$  can be written as an MDP  $\langle S, Action_\alpha, T, R \rangle$ .  $S$  is the state space and its specific value in time  $t$  defined as  $s(t)$ ,  $Action_\alpha$  is the action space of  $\alpha$ ,  $T : S \times A \rightarrow S$ , is the transition function that describes the resulting state  $s(t+1) \in S$  when executing  $\chi \in Action_\alpha$  in  $s(t)$ .  $R : S \rightarrow \mathbb{R}$  defines the instantaneous reward for being in a specific state. This model can be applied to any agent and any token.

In this case, the state  $s(t)$  at time  $t$  is modelled as the locations of all tokens across the team and is written as:  $s(t) = \langle \langle Tokens(\alpha, t), H_\alpha(t) \rangle, \langle Tokens(\alpha_1, t), H_{\alpha_1}(t) \rangle, \langle Tokens(\alpha_2, t), H_{\alpha_2}(t) \rangle, \dots \rangle$ .  $Tokens(\alpha, t)$  are all the tokens currently held by  $\alpha$  and  $H_\alpha(t)$  records all the incoming and out-going tokens of  $\alpha$  before  $t$ . For notation convenience, we write  $Tokens(\alpha, t)$  as  $Tokens(\alpha)$  and  $H_\alpha(t)$  as  $H_\alpha$  when there is no ambiguity. Figure 2 shows an example of specific team state  $s(t) = \langle \langle Tokens(\alpha), H_\alpha \rangle, \langle Tokens(\alpha_1), H_{\alpha_1} \rangle, \langle Tokens(\alpha_2), H_{\alpha_2} \rangle \rangle$ , where  $Tokens(\alpha) = \{\Delta, \Delta_2\}$ ,  $H_\alpha = \{\Delta_3\}$ ,  $Tokens(\alpha_1) = \{\Delta_5\}$ ,  $H_{\alpha_1} = \{\Delta\}$ , and  $Tokens(\alpha_2) = \{\Delta_4\}$ ,  $H_{\alpha_2} = \emptyset$ . Since the tokens represent resources, roles and information,  $s(t)$  unambiguously defines who is doing what, with what resources and what information.

$Action_\alpha : S \rightarrow (n(\alpha) \cup \alpha)$  is to move  $\Delta$  to one of  $n(\alpha)$  or keep it for itself. For notation convenience,  $\chi \in Action_\alpha$  can be written as  $move(\Delta, b)$  where  $b \in (n(\alpha) \cup \alpha)$ . Note, keeping a token for itself applies when the agent accepts the role or requires the resource or information has propagated sufficiently for



**Fig. 2.** An example showing part of a team. Agent  $\alpha$  holds  $\Delta$ ,  $\Delta_2$  and has previously had  $\Delta_3$ ;  $\alpha_2$  holds  $\Delta_4$  while  $\alpha_1$  holds  $\Delta_5$  and has previously had  $\Delta$ .

across the team. In general, we define a function  $Acceptable(\alpha, \Delta)$  to determine whether  $\Delta$  should be kept by agent  $\alpha$ .

$R(s(t)) > 0$  when at  $s(t)$ , a sub-goal  $g_i$  are achieved. The team will be credited an instant rewards value of  $R(s(t)) = reward_i$ .

The utility of state  $S$  under a policy  $\pi$  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

$$\pi^*(s(t)) = argmax_{\chi \in Action_\alpha} v^*(s(t+1))$$

By value iteration,  $v^*(s(t)) = argmax_{\chi \in Action_\alpha} [R(s(t)) - commcost + d \times v^*(s(t+1))]$ . This policy tells the agent where to move resources information and roles to maximize the team's expected utility.

We define a matrix  $V$  where each element  $V[s(t), b] = R(s(t)) - commcost + d \times v^*(s(t+1))$  when  $\chi = move(\Delta, b)$ . Then  $V[b]$  represents the expected utilities vector for  $\alpha$  to send token  $\Delta$  to  $b$  at each different state  $s(t)$ .

## 4.2 Local POMDP Model

Knowing the complete team state is only feasible for small teams. In large teams, agents must make token coordination decisions based on a more limited view of the team. Thus the reasoning must be modelled as a Partially Observable Markov Decision Process (POMDP). Standard POMDP techniques such as [16] and [15] could be used to solve the POMDP to determine optimal token routing. However, for fast routing of tokens, while this local POMDP does tell the agent the optimal action, the computational complexity is still too high for practical applications. However, the POMDP model does provide important hints for how to do a heuristic approach.

The POMDP model is defined as  $\langle S, Action_\alpha, T, \Theta_\alpha, O, R \rangle$ . In this case, the observations of agent  $\alpha$  are defined as  $\Theta_\alpha = \langle Tokens(\alpha, t), H_\alpha(t) \rangle$  to

include not only the tokens the agent currently holds but also all the previously incoming and out-going tokens (in  $H_\alpha(t)$ ). The observation function is defined as  $O : \Theta_\alpha \times S \rightarrow \Omega_\alpha$ . *Belief state*  $\Omega_\alpha$  is a discrete probability distribution vector over the team state  $s(t)$  inferred from current local state  $\Theta_\alpha$ . For example, if  $S = \{s_1, s_2, s_3\}$  and  $\Omega_\alpha = [0.6, 0.2, 0.2]$ ,  $\alpha$  estimates that the probability of  $s(t)$  being  $s_1$  is 0.6 and being  $s_2$  and  $s_3$  are 0.2.

One way of solving a POMDP is via a Q-MDP [15]. Agent  $\alpha$  makes use of  $V$ , see above, to calculate the expected reward vector  $EU(\Omega_\alpha) = \Omega_\alpha \times V$ . For example, if  $\alpha$  has acquaintances  $b$ ,  $c$ , and  $d$  and  $EU(\Omega_\alpha) = [5, 10, 6, 4]$ , then  $EU(\Omega_\alpha, b) = 10$  represents the expected utility to send  $\Delta$  to  $b$  according to the Q-MDP. The locally-optimal policy  $\pi^{**}(\Omega_\alpha)$  is  $\operatorname{argmax}_{\chi \in \text{Action}_\alpha} EU(\Omega_\alpha, c)$ . This is the action the agent should take to maximize expected utility, given that it has an incomplete view of the team state. As in the previous example, passing  $\Delta$  to  $b$  is the best choice because  $EU(\Omega_\alpha, b) = 10$  is the maximum value of  $EU(\Omega_\alpha)$ .

## 5 Local Heuristic Approach

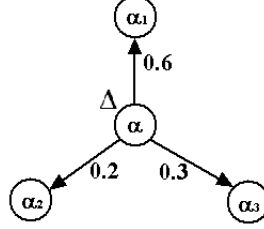
In this section, we provide a heuristic approach for token-based team coordination inspired by the local POMDP. The resulting approach allows fast, efficient routing decisions, without requiring accurate knowledge of the complete state. In the next section, we show that this approach is effective for improving token routing.

### 5.1 Local Model

$P_\alpha$  is the decision matrix agent  $\alpha$  uses to decide where to move tokens. Each row  $P_\alpha[\Delta]$  in  $P_\alpha$  represents a vector that determines the decision where to pass a token  $\Delta$  to one of its acquaintances. Specifically, each value  $P_\alpha[\Delta, b] \rightarrow [0, 1]$ ,  $b \in n(\alpha)$  represents  $\alpha$ 's decision that the probability of passing token  $\Delta$  to an acquaintance  $b$  would be the action that maximize team reward. Then our policy  $\pi^{***}$  for this local model is to choose action  $\chi$  to  $\operatorname{argmax}_{\chi \in \text{Action}_\alpha} P_\alpha[\Delta, c]$  where  $\chi = \text{move}(\Delta, c)$ . Figure 3 shows an example where  $P_\alpha[\Delta] = [0.6, 0.1, 0.3]$  and agent  $\alpha$  has three acquaintances  $\alpha_1, \alpha_2, \alpha_3$ .  $P_\alpha[\Delta, \alpha_1] = 0.6$ ,  $P_\alpha[\Delta, \alpha_2] = 0.1$ ,  $P_\alpha[\Delta, \alpha_3] = 0.3$  and  $\pi^{***}$  will choose the action  $\text{move}(\Delta, \alpha_1)$  to pass  $\Delta$  to  $\alpha_1$ . The key to this distributed reasoning lies in how the probability model  $P_\alpha$  for each agent  $\alpha$  is updated. If the action indicated by  $P_\alpha$  matches the optimal policy  $\pi^*$  from the MDP model, then the team will act optimally.

Initially, agents do not know where to send tokens, but as tokens are received, a model can be developed and better routing decisions made. That is, the model,  $P_\alpha$  is based on the accumulated information provided by the receipt of previous tokens. For example, when an agent sends a role to an acquaintance that has previously rejected a similar role, the team is potentially hurt because

this acquaintance is likely to reject this role too and thus communication bandwidth has been unnecessarily wasted.



**Fig. 3.** Agent  $\alpha$ 's local model for where to send  $\Delta$ . The probability that  $\alpha_1$  is the best to send  $\Delta$  to is 0.6 and  $\alpha$  will pass  $\Delta$  to  $\alpha_1$  according to  $\pi^{***}$

From this view point,  $P_\alpha$  can only depend on  $\alpha$ 's history of received tokens,  $H_\alpha$ . The update function  $Update(P_\alpha[\Delta], \Delta_i)$  for  $P_\alpha[\Delta]$  defines the calculation of the probability vector of where to send  $\Delta$  based on previously received token  $\Delta_i$  in  $H_\alpha$ . It will be explained in detail in next section.

*Algorithm 1* shows the reasoning of agent  $\alpha$  when it receives incoming tokens from its acquaintances via function  $getToken(sender)$  (line 2). For each incoming token  $\Delta$ , function  $Acceptable(\alpha, \Delta)$  determines whether the token will be kept by  $\alpha$  (line 4). When a resource is kept, its threshold is raised (line 6). If  $\alpha$  decides to pass  $\Delta$ , it will add itself to the path of  $\Delta$  (Line 9) and  $Update(P_\alpha[\Delta], \Delta_i)$  will update how to send  $\Delta$  according to each previously received token  $\Delta_i$  in  $\alpha$ 's history (line 11). If  $\Delta$  is a resource or role token, its threshold will be decreased (line 14). Then  $\alpha$  will choose the best acquaintance to pass the token to according to  $P_\alpha[\Delta]$  (line 16) and record  $\Delta$  in its history,  $H_\alpha$  (line 18).

Algorithm 1: Decision process for agent  $\alpha$  to pass incoming tokens

```

1: while true do
2:    $Tokens(\alpha) \leftarrow getToken(sender)$ ;
3:   for all  $\Delta \in Tokens(\alpha)$  do
4:     if  $Acceptable(\alpha, \Delta)$  then
5:       if  $\Delta.type == Res$  then
6:          $Increase(\Delta.threshold)$ ;
7:       end if
8:     else
9:        $Append(self, \Delta.path)$ ;
10:    for all  $\Delta_i \in H_\alpha$  do
11:       $Update(P_\alpha[\Delta], \Delta_i)$ ;
12:    end for
13:    if  $(\Delta.type == Res) || (\Delta.type == Role)$  then

```

```

14:         Decrease( $\Delta.threshold$ );
15:     end if
16:      $acquaintance \leftarrow Choose(P_\alpha[\Delta])$ 
17:     Send( $acquaintance, \Delta$ );
18:     AddtoHistory( $\Delta$ );
19: end if
20: end for
21: end while

```

## 5.2 Model Update Function

The effectiveness of the token-based approach depends on how well agents maintain their local models so that tokens are routed to where they lead to the highest gain in expected reward. In this section, we describe an algorithm to update the localized decision model by utilizing previously received tokens. The key is to make use of relationships between tokens, which we refer to as *relevance*.

Deciding where to send one token based on the receipt of another relies on knowing something about the relationship between the tokens. We quantify this relationship as the *Relevance* and define the relationship between tokens  $\Delta_i$  and  $\Delta_j$  as  $Rel(\Delta_i, \Delta_j)$ .  $Rel(\Delta_i, \Delta_j) > 1$  indicates that an agent with use for  $\Delta_i$  will often also have use for  $\Delta_j$ , while  $Rel(\Delta_i, \Delta_j) < 1$  indicates that an agent, which has use for  $\Delta_i$  is unlikely to have use for  $\Delta_j$ . If  $Rel(\Delta_i, \Delta_j) = 1$  then nothing can be inferred. Details about how *relevance* is computed to ensure appropriate behavior will be explained in the next section. The update function of  $P_\alpha[\Delta_j]$  according to  $H_\alpha$ , written as  $Update(P_\alpha[\Delta_j], \Delta_i)$  where  $\Delta_i \in H_\alpha$  is found by using Bayes' Rule as follows:

$$\forall b \in n(\alpha), \forall \Delta_i \in H_\alpha, d = first(n(a), \Delta_i.path)$$

$$Update(P_\alpha[\Delta_j, b], \Delta_i) = \begin{cases} P_\alpha[\Delta_j, b] \times Rel(\Delta_i, \Delta_j) & \text{if } \Delta_i \neq \Delta_j, b = d \\ P_\alpha[\Delta_j, b] & \text{if } \Delta_i \neq \Delta_j, b \neq d \\ P_\alpha[\Delta_j, b] \times \varepsilon & \text{if } \Delta_i = \Delta_j, b \in \Delta_j.path \cap n(\alpha) \end{cases}$$

where  $Update(P_\alpha[\Delta_j, b], \Delta_i)$  is to update the  $P_\alpha[\Delta_j, b]$  in  $P_\alpha[\Delta_j]$  according to  $\Delta_i$  and  $first(n(a), \Delta_i.path)$  extracts from the recorded path of the token the acquaintance of agent  $\alpha$  that had the token  $\Delta_i$  earliest. The first case in this function is the most important. The probability that the sender of previous token  $\Delta_i$  is the best agent to receive the token  $\Delta_j$  is updated according to  $Rel(\Delta_i, \Delta_j)$ . The second case in the equation changes the probability of sending that token to agents other than the sender in a way that ensures the subsequent normalization has the desired effect. Finally, the third case encodes the idea that  $\alpha$  should typically not pass a token back from where it came.  $P_\alpha[\Delta_j]$  is subsequently normalized to ensure that  $\sum_{b \in n(\alpha)} P_\alpha[\Delta_j, b] = 1$ .

To see how the updating function works, consider the following example. Supposed agent  $\alpha$  has five acquaintances  $\{a, b, c, d, e\}$  and  $P_\alpha[\Delta_j] =$

$[0.1, 0.4, 0.2, 0.2, 0.1]$ . Moreover,  $H_\alpha = \{\Delta_i, \Delta_k\}$ ,  $rel(\Delta_i, \Delta_j) = 1.2$  and  $rel(\Delta_k, \Delta_j) = 0.4$ .  $\Delta_i.path = \{b, ..\}$ ;  $\Delta_k.path = \{c, ..\}$ ;  $\Delta_j.path = \{e, ..\}$ . If currently  $\alpha$  holds  $\Delta_j$ , by applying our updating function to  $P_\alpha[\Delta_j]$ , we get the result as  $P_\alpha[\Delta_j] = [0.12, 0.56, 0.09, 0.23, \varepsilon]$  and  $\Delta_j$  will be most likely passed to acquaintance  $b$ .

### 5.3 Token Similarity

When an agent receives two tokens that are relevant to one another they are more likely to be usable in concert to obtain a reward for the team. While infeasibly complex, the POMDP model can suggest how relevance should be defined. If the local policy  $\pi^{***}$  always matches with  $\pi^{**}$ ,  $P_\alpha[\Delta]$  will be the normalization of  $EU(\Omega_\alpha)$ .

$$\forall b \in n(\alpha), P_\alpha[\Delta, b] = \frac{EU(\Omega_\alpha, b)}{\sum_{c \in N(\alpha)} EU(\Omega_\alpha, c)}$$

That is, the largest expected utility for sending a token to an acquaintance should result the highest probability. Following the previous example, if  $EU(\Omega_\alpha) = [5, 10, 6, 4]$ , then for optimal behavior  $P_\alpha[\Delta] = [0.2, 0.4, 0.24, 0.16]$ .

Now we are in a position to see how the receipt of a token affects the locally optimal policy for routing token  $\Delta$  and hence determine how to compute relevance. Suppose that the state estimation of agent  $\alpha$  just before a token  $\Delta_{pre}$  arrives is  $\Omega_\alpha$  while after it arrives, the state estimation is changed to  $\Omega'_\alpha$  because  $\alpha$  gains additional knowledge from this token. Thus, according to Q-MDP, before  $\Delta_{pre}$  is received, the expected reward of  $\alpha$  is  $EU(\Omega_\alpha) = \Omega_\alpha \times V$  while after the arrival of  $\Delta_{pre}$ ,  $EU(\Omega'_\alpha) = \Omega'_\alpha \times V$ . Moreover, agent  $\alpha$ 's local model will also be updated according to  $\Delta_{pre}$ . Suppose  $\Delta_{pre}$  comes from acquaintance  $b$  and  $P_\alpha[\Delta, b]$  is the probability that  $\alpha$  will send  $\Delta$  to  $b$  before  $\Delta_{pre}$  comes while  $P'_\alpha[\Delta, b]$  is the updated probability after the arrival of  $\Delta_{pre}$ . According to our assumption that policy  $\pi^{***}$  (according to the  $P_\alpha$  model) and  $\pi^{**}$  (according to the POMDP model) will choose the same action which is to send  $\Delta$  to  $b$ . Thus, we have

$$\frac{P'_\alpha[\Delta, b]}{P_\alpha[\Delta, b]} = \frac{EU(\Omega'_\alpha, b)}{EU(\Omega_\alpha, b)} = \frac{[\Omega'_\alpha \times V]_b}{[\Omega_\alpha \times V]_b}$$

Where  $[\Omega'_\alpha \times V]_b$  is the value of the component in vector of  $[\Omega'_\alpha \times V]$  according to acquaintance  $b$ . It is the same vector as  $EU(\Omega'_\alpha, b)$ .

According to the update function  $Update(P_\alpha[\Delta, b], \Delta_{pre})$ , we should get  $P'_\alpha[\Delta, b] = Rel(\Delta, \Delta_{pre}) \times P_\alpha[\Delta, b]$ . Thus, the relationship between  $Rel$  and our POMDP model is:

$$Rel(\Delta, \Delta_{pre}) = \frac{[\Omega'_\alpha \times V]_b}{[\Omega_\alpha \times V]_b}$$

From this equation, we can conclude that a received token changes the agent’s estimation of the probability distribution of the team’s state, which in turn directly influences the decision of where to send related tokens. If we know a little bit of how the probability distribution changes for an agent after it has passed a token, we can use this to predict how this agent updates its distributed decision model and therefore define the relevance between tokens. A heuristic that captures this relationship will approximate the locally optimal policy and hence lead to good behavior. In this chapter, we simply estimate this value based on the similarity between tokens. Intuitively, if two tokens are similar, receiving one token allows an agent to update its estimation of the team state and infer where to pass the similar tokens. For example, receiving a role token from a particular acquaintance, tells the agent that it is relatively less likely that similar role tokens will be accepted in the part of the network accessible via that acquaintance; receiving an information token with information about Pittsburgh tells the agent that some agents in that part of the network must currently be in Pittsburgh.

The similarities between tokens come from the *coordination* they carry and the calculation depends on the domain knowledge of applications. We assume that, from  $\Delta_i.coordination$  and  $\Delta_j.coordination$ , we can deduce the similarity between two tokens as  $sim(\Delta_i, \Delta_j)$ .  $sim(\Delta_i, \Delta_j) > 1$  if  $\Delta_i$  and  $\Delta_j$  are a pair of similar tokens. For example, if two tokens both reference Pittsburgh, we deem them similar because both are involved with the same location; we deem two tokens which require driving a specific machine as similar because they need the same kind of capacity; two tokens that both are preconditions of the same plan would also be considered similar.

We distinguish the relationship between relevance and similarity of two tokens as positively related or negatively related. For two similar tokens  $\Delta_i$  and  $\Delta_j$ , if an agent previously received a token from an acquaintance and would prefer to send a similar token to that acquaintance, similar tokens are positively related to each other and  $Rel(\Delta_i, \Delta_j) = sim(\Delta_i, \Delta_j)$ . Otherwise, if this agent is less likely to send the similar token to that acquaintance similar tokens are negatively related to each other, so  $Rel(\Delta_i, \Delta_j) = \frac{1}{sim(\Delta_i, \Delta_j)}$ .

The similarity between different types of tokens potentially influences agents’ estimation in different ways. As we have shown in the previous example, receipt of role tokens discourages sending similar tokens to agents along the role tokens’ paths because the previous token senders refused the role token and are incapable of accepting the role, therefore are less likely to be interested in the information, tasks or resources that similar tokens carry. Thus, a previous role token is negatively related to its similar tokens. Similarly, receipt of an information token will indicate that agents along the information tokens’ paths are more likely to work on things related to that information and are interested in other similar tokens. Hence, a previous information token is positively related to its similar tokens.

If the *threshold* of a resource token  $\Delta_i$  is greater than its initial value (*init*) upon arrival to current agent, this means that the resource has been

used by the agents previously holding  $\Delta_i$  and that those agents are potentially engaged in tasks requiring the resource. Therefore, if the current agent gets similar tokens, it will be more likely to send them to the part of the network where the previous token has been passed. In this case, the previous resource token is positively related to similar tokens. Alternatively, if  $\Delta_i.threshold$  is lower than its initial value (*init*), it means that agents passing the token did not need it. In such a case, the previous resource token is negatively related to similar tokens.

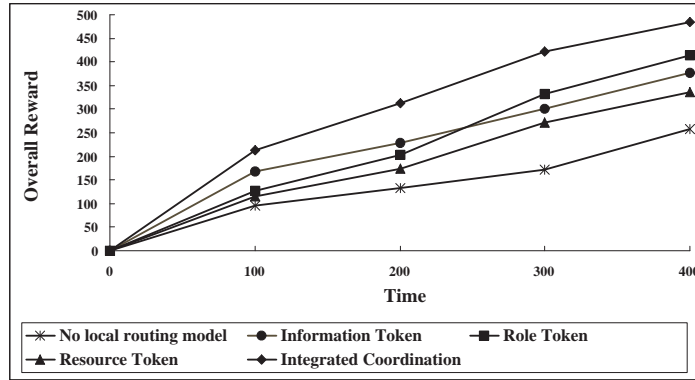
Supposing  $\Delta_i$  is a previously received token, we can summarize the calculation of  $Rel(\Delta_i, \Delta_j)$  according to  $sim(\Delta_i, \Delta_j)$ . No matter what  $\Delta_j.Type$  is, this function only depends on the type of previously incoming token:

$$Rel(\Delta_i, \Delta_j) = \begin{cases} sim(\Delta_i, \Delta_j) & \text{if } \Delta_i.Type = inf \\ sim(\Delta_i, \Delta_j) & \text{if } \Delta_i.Type = res, \Delta_i.threshold > init \\ \frac{1}{sim(\Delta_i, \Delta_j)} & \text{if } \Delta_i.Type = role \\ \frac{1}{sim(\Delta_i, \Delta_j)} & \text{if } \Delta_i.Type = res, \Delta_i.threshold < init \end{cases}$$

## 6 Evaluation

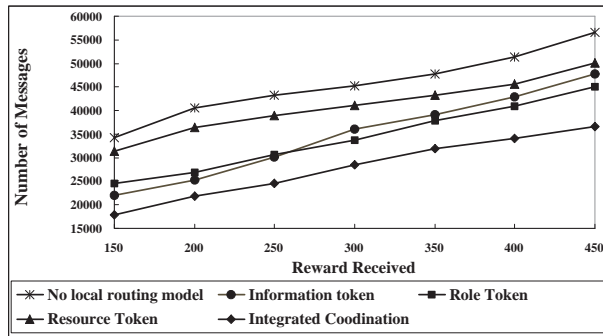
In this section, we describe an empirical evaluation of our approach. Tests were conducted using an abstract simulation called CoordSim [33] configured to simulate a group of 400 distributed UAVs searching a hostile area. The network topology was that of a small world network where each UAV had, on average, four acquaintances. Simulating automatic detection rates, 200 pieces of information were randomly sensed by UAVs and passed around the team. Fifty plans instances, each with four independent preconditions, were given to the team. After a plan was initiated, tokens for the four roles needed to realize the plan were circulated through the acquaintance network. To accept a role an agent must be close to the region the role requires and have access to resource tokens for airspace at the role allocation. Airspace over the hostile area was divided into fifty regions. Each of these regions was duplicated in three resource tokens allowing a maximum of three UAVs to simultaneously access that airspace. Each UAV needed to obtain the resource for the region related to its task before they could be performed. If all four roles of a plan were successfully executed, a reward of 10 units was credited to the team. A maximum reward of 500 units (10 units x 50 plan instances) was possible. Results for each experiment are based on one hundred trials.

The first experiment investigated the algorithm's performance in enhancing overall team reward. Reward obtained was recorded for each tick of the simulation which corresponded to the time taken for a token to move from agent to agent. Five configurations of the algorithm were compared. In the first configuration, agents passed tokens randomly if they did not keep them. In the next three configurations, local reasoning model updating is applied to



**Fig. 4.** The team gets more reward (y-axis) over time (x-axis) when all token types are used to update local models.

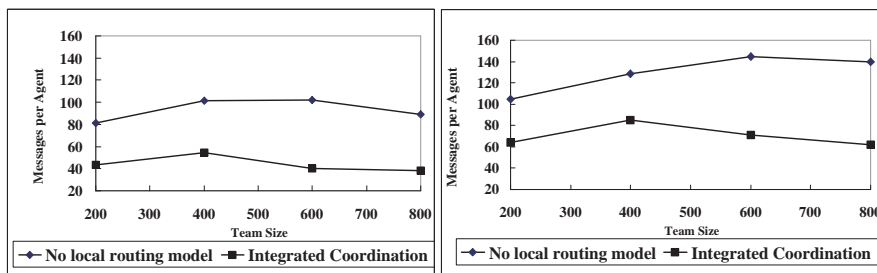
only one type of token, i.e., information, resource or role, with no updating by the other two types. The fifth configuration provided integrated coordination using tokens of each type to update agents' local model for routing tokens. The results are shown in Figure 4. Use of any previous tokens for token routing improved team reward. This benefit was most pronounced when all token types were used. In fact reward was almost doubled over random token movement. Notice that early on using plan instantiation alone was more effective than using other types of tokens but later role allocation tokens were the most effective type. We hypothesize that as roles are allocated they become most useful but before then it is critical to know who is initiating plans.



**Fig. 5.** The team needs to send less messages (y-axis) to coordinate when all token types are used to create local models.

The second experiment investigated the effect of our algorithm on communication. In this experiment we compared the number of messages need for a team to gain a particular level of reward. A message was credited to each transfer of a token from an agent to its acquaintance. The same five configurations (random, three with coordination for single token-types, and integrated) were employed. As shown in Figure 5 configurations using token coordination algorithm performed better. They used fewer communications to attain the same level of reward as the random configuration. Once again, complete integrated token routing was superior to the partial token-based algorithm in attaining the same results with substantially fewer messages. Notice also that the total number of references is quite low, even for a large team.

The third experiment examined in more detail the scalability of our algorithm to larger teams. In this experiment teams of 200 to 800 agents were run under conditions otherwise identical to the first two experiments, however, only two configurations were used: no use of previous tokens to improve agents' local reasoning model versus use of all types of previous tokens to improve agents' local reasoning model to routing tokens. Performance was measured using *average number of messages per agent*. In Figure 11, the top configuration is to get a team reward of 200 units while the bottom one is to get a reward of 400. Our results show that integrated token-based routing produced lower message overload under all conditions. For both 200 and 400 reward levels observed message overhead was lower for teams of 800 agents using the integrated algorithm than for 200 agent teams using the random one.



**Fig. 6.** The average number of messages per agent (y-axis) stays relatively constant as the team size is increased (x-axis). This holds when tokens are used for integrated model and when they are not.

## 7 Comparing with Market-based Coordination

In addition to testing the token-based approach itself, we compare the token-based coordination with the market-based coordination, a popular centralized algorithm. Although this algorithm is good at finding the optimal coordination solution, it ignores the communication costs, which are critical to large teams. The objective for this comparison is to verify how the token-based coordination approach makes the trade off between team utilities and communication cost.

Our implementation of market-based approach was based on TraderBots [8]. One agent acts as auctioneer and both tasks and resources are treated as merchandize. Agents bid for either single items or combinatorial sets of items in order to maximize their own utilities. The auctioneer maximizes its utility by “selling” their “merchandize”. Because of the centralized position of the auctioneer, it develops a complete knowledge of how agents will use a task or resource if allocated and the auctioneer can perform assignments that maximize the team utility. Notice that several constraints also apply to this approach. The auction should last for a fixed period of time and early determination is infeasible; Agents are allowed to bid for resources after tasks have been allocated. Moreover, to prevent deadlock in resource allocation, agents are only allowed to bid for resources for their *first* pending task.

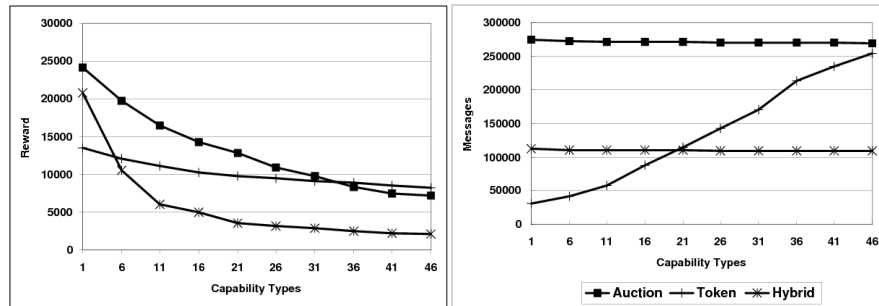
We designed a hybrid approach which is to combine the two different approaches. In the hybrid approach, the auctioneer algorithm runs exactly as before, except that instead of broadcasting announcements for auctions an *auction token* is created. Each auction token is allowed to exist from the start of the auction to the end of the auction. The auctioneer has a probabilistic model of the team state, just as all agents do in the token-based approach. The auction token is then intelligently routed to the agents most likely to be able to submit the best bids. The token stops moving after the auction it presents is closing or has visited a fixed number of teammates.

In those experiments, we only focused the comparison on task and resource allocation which can be performed by both token and auction. The basic experiment settings are configured as follows. There are 100 agents to perform 50 tasks with 50 resources. Each task requires only one resource which was interchangeable with four others. In the default setup, there is only one type of capability required and all agents have non-zero value for this capability, i.e., all agents are at least somewhat capable of all tasks. Auctions are held open for 40 time steps, and the task tokens and resource tokens are allowed to move unless accepted. The initial threshold on a task token is 100, meaning that the task will not be accepted by an agent until it can get a reward of more than 100 by performing this task. We measured two key statistics required to support or refute our hypothesis about the algorithms. “Reward” is the sum of reward received by each agent. “Messages” is the number of times agents communicated, either between themselves or with the auctioneer. The “messages” count indicates messages sent to perform sensor fusion, plan

initiation and information sharing. Simulation runs for 2000 time steps. The experiment results below are based on 100 runs.

### 7.1 Heterogeneous Team

In the first experiment, we examined team performance by varying team composition and the capabilities required to perform tasks. For example, in an emergency response experiment some agents might only be able to fight fires while others could only provide medical treatment. As capabilities grew more varied fewer agents were available to perform particular tasks. In this experiment, we varied the number of capabilities from 1 to 46 where in the most heterogeneous condition, only two agents on average are capable to performing a task.



**Fig. 7.** The average reward for heterogeneous teams dramatically decreases in auction and hybrid approaches. Token-based approach maintains constant reward but requires more messages

The experimental results in Figure 7 show that for heterogeneous teams, auction and hybrid approaches earn less reward as the team becomes more heterogeneous because there are fewer agents able to compete for the more specialized tasks. The advantages of team wide maximization of utility by the auctioneer decrease as there are progressively fewer feasible alternative bids. In contrast, reward for the token-based approach remain almost flat with increasing specialization. We propose two reasons. One is that token-based approach greedily finds a reasonable solution rather than searching for the optimal.

The other reason is that by passing a higher number of tokens around the network and making use of the relevance between them, the intelligent routing algorithm gains a better knowledge of how to route tokens. This results in a higher number of messages but an equal amount of reward.

## 7.2 Time Critical Tasks

In the second experiment, we investigated team performance when many tasks needed to be performed within a short period of time. To increase their reward, teams were required to perform tasks and allocate resources as rapidly as possible. In this study we varied the number of tasks the teams were required to finish from 20 to 182. After 2000 time steps, the accumulated reward and message count were recorded as shown in Figure 8.

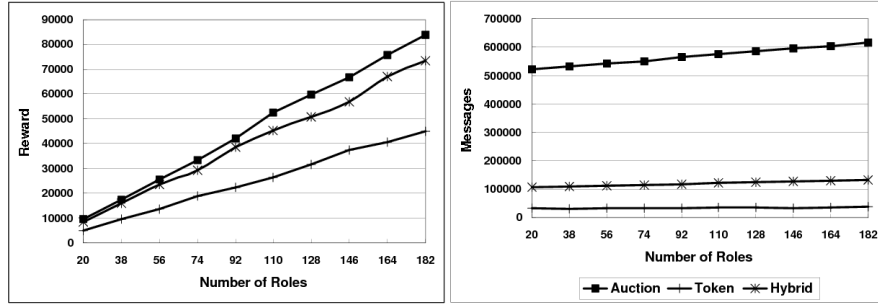


Fig. 8. Reward and messages increase dramatically with the number of tasks.

All three approaches performed more tasks in order to get higher reward. As expected, the auction approach attained higher reward than the hybrid or token-based approaches. Considering both reward and messages, however, the hybrid approach performs well by almost matching the reward obtained by the auction at just a quarter of the communication cost. The reason the hybrid approach achieves such good performance with so little communication overhead is that the intelligent routing algorithm limits communication to a small number of agents while high bidders must always be informed in auctions.

## 7.3 Competitive Resources

The third experiment used 200 tasks each requiring an average of four resources with no interchange possibilities. As available resources are increased from 4 to 40, competition for them declines and they become less likely to be a bottleneck.

The experiment was stopped after 1000 time steps. Figure 9 shows that the reward for the auction based approach increased rapidly with increases in resources. Both the token-based and hybrid approaches remained flat with the token-based approach earning the highest reward at all levels of scarcity while the hybrid approach yielded very limited reward.

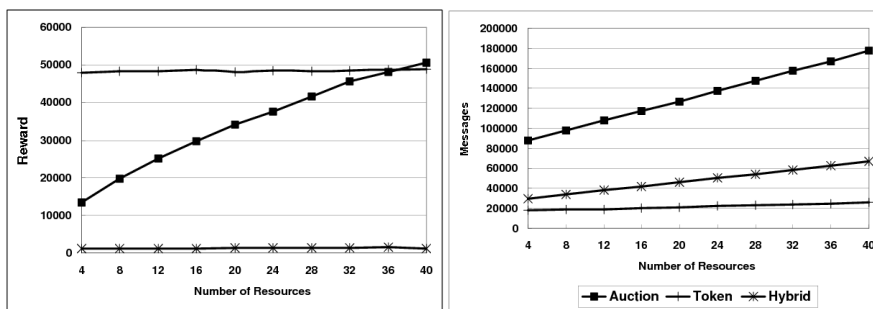


Fig. 9. Reward decreases with scarcity in auction approach and is very low in hybrid approach, but with token-based approach is uniformly high

We hypothesized that because resource contention in this experiment was high the centralized control of the auction and hybrid approaches would often force agents to either bid for all four resources together or miss the task while the distributed token-based approach weakened this constraint. If our hypothesis were true, auction and hybrid approaches would get more reward if we either increased the simulation length or reduced the length of the auction to weaken the constraint. Figure 10 shows the effect of shortening auction length from 40 to 20 steps (a) and increasing session length from 2000 and 4000 steps (b). The token-based approach continues to produce its constant level of reward while the hybrid approach obtain very little higher. The auction-based approach, however, improves with increasing resources exceeding the token-based approach at most levels in the two alternative experiments.

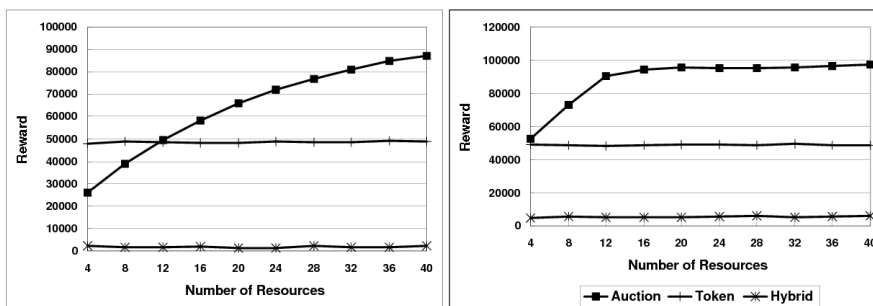


Fig. 10. Auction approach gets more reward when auction last length is 20 than it is 40 (a) and when experiment lasts from 2000 time points to 4000 (b).

#### 7.4 Interchangeable Resources

In the fourth experiment, there were 100 tasks each requiring three resources. The number of interchangeable resources was varied from one to five. Experiments were stopped at 1000 steps. Results are shown in Figure 11. Interchangeable resources did not help the token-based approach, helped the auction-based approach very little but substantially increased reward for the hybrid approach. We contend that three required resources for each task is a high constraint for a centralized auction. The constraint has been weakened in the market-based approach because this experiment lasts long enough for auctioneer to search bids and maximize the reward. In contrast, this constraint is higher in the hybrid approach because within a limited number of moves, all resource tokens for a role are required to visit the agent who has this task pending. This is also a reason why the hybrid approach gained such a low reward. Moreover, interchangeable resources led to dramatic increases in the number of messages for auction based approach because every agent could participate in every resource auction leading to the submission of a large number of multiple bids. For example, when interchangeable resources are 5, an agent should submit  $5^3$  resource bids.

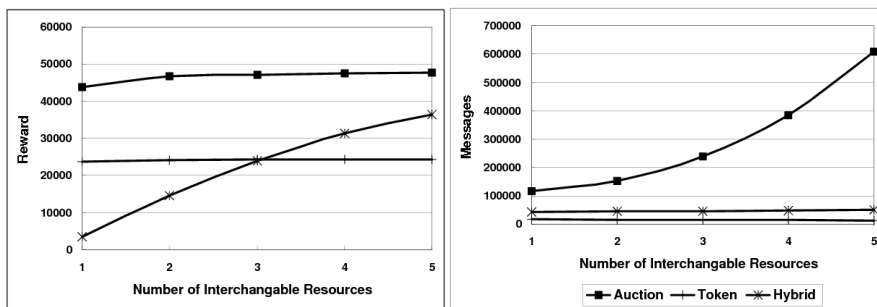
#### 7.5 Conclusion

The experiment results above support our hypothesis. Auctions are focused on maximizing overall utility by taking into account the *bids* of all team members [2], but at the expense of higher communications. Token-algorithms are well focused on scalability and make the trade off between team performance and communication. Therefore, sometimes at the expense of overall utility. More subtly, the performance advantage of an auction should be most pronounced when small changes in allocations lead to big differences in performance, i.e., typically highly constrained cases, while the token algorithms should maximize their communication advantage when the probabilistic models they rely on are most advantageous, i.e., weakly constrained cases.

The hybrid algorithm only performs well under restricted circumstances. If the problem is so tightly constrained that the auctions need to see many bids to make good allocations then using tokens to solicit bids only adds overhead. Conversely, if the coordination is so underconstrained that the tokens can reliably and accurately target the best agents, then the auction only adds unnecessary overhead.

## 8 Related Work

Multi-agent coordination is an extensively studied area of multi-agent systems, but most of the existing work does not scale well to very large teams. STEAM combined the joint intention model and the shared plan model and developed



**Fig. 11.** Reward for hybrid approach increases rapidly with more interchangeable resources

a general model of teamwork for persistent coordination in an uncertain, complex, and dynamic environment [28]. Unfortunately, its coordination requires a precise model of each individual agent. Distributed constraint-based algorithms [17, 20] have high communication requirements that get dramatically worse as the team size is increased. Swarm-inspired approaches [6] have been used for large-scale coordination, but the behavior can be inefficient.

Centralized coordination approaches typically make improper assumptions for scalable team coordination: complete communication with central agent, e.g., message board [5], or central information agents which have complete knowledge of the team [7]. Combinatorial auctions [11] have an exponential number of possible combinations of bids, and frequently use centralized auctioneers that can become severe bottlenecks. Generalized Partial Global Planning (GPGP) and its associated hierarchical task network representation TAEMS (Framework for Task Analysis, Environment Modelling, and Simulation) are focused on optimization to maximize the overall team utility. Unfortunately, this approach requires a centralized reasoning framework which is computationally hard to coordinate a moderate or scalable team [14].

Decision theoretic approaches, such as MDPs and POMDPs, have been used for team coordination, but MDP model requires agents have a complete view of team states. Decentralized POMDP model is more realistic for handling uncertainty in scalable teamwork; however Bernstein provided the mathematical evidence that decentralized decisions to find the optimal joint policy is NEXT-complete [4]. Xuan and Pynadath [10, 22, 34] recast the decentralized POMDP as a communication problem [22] by considering that sending messages to teammates with a belief that they did not observe, the team can get higher reward. But such approaches are known to be intractable in general [22], and so are of limited use in large scale applications.

The token-based approach was first introduced from networking design [1]. This idea was introduced to multi-agent coordination research by Wagner [27]. He renamed tokens as “keys” and applied them to the coordination of

dynamic readiness and repair service in aircraft simulation. Recent work focusing on scalable coordination [21] 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. The effectiveness of large-scale, token-based coordination has also been demonstrated in the Machinetta proxy architecture [25] for task allocation [24] and information sharing [31].

Research on social networks began in physics [1, 16]. [18] showed that social network structures in team formation can dramatically affect team abilities to complete cooperative tasks. In particular, using scale-free network structures for agent teams facilitates team formation by balancing the number of skill-constrained paths available in the agent organization with the effects of potential blocking. [12] compared the merits of small world networks and scale-free networks in the application of emergent coordination.

## 9 Conclusion and Future Work

This chapter presented a novel integrated token-based algorithm for team coordination. By utilizing relationships between tokens we were able to use the execution of one type of coordination algorithm to improve the performance of the others. Our experiments show that our approach is scalable and efficient. This approach opens the possibility to develop a range of new executing applications of heterogeneous agents not possible with existing approaches. By comparing with the market-based approach, we showed that token-based approach makes a good trade off between quality of allocation and use of communication. In some specific circumstances, its performance may outperform the auction which tries to find the optimal solution.

While the results presented here represent a step forward, they also point to significant challenges and exciting questions. We plan to address some of these issues in the near future. From a scientific perspective, the effect of the underlying acquaintance network on the coordination is clearly important but poorly understood. We will investigate the effect of the properties of the network on the team's performance, specifically looking at how the small worlds nature of the network is important. This work represented a novel attempt at integrating coordination algorithms into a unified approach and showing how by working together the overall performance can be improved. However, the individual algorithms were designed without thought to future integration. A key question is whether knowing that we will be integrating the token algorithms allows us to build algorithms that work better with other algorithms. Finally, but critically, we will use the token-based approach in more realistic domains to understand its utility in the real world. Specifically, we are currently developing large scale coordination applications for rescue response and unmanned aerial vehicle applications.

## Acknowledgements

This research has been sponsored in part by AFRL/MNK Grant F08630-03-1-0005 and AFOSR Grant F49620-01-1-0542.

## References

1. Token ring access method. In *IEEE. 802.5*.
2. N. Kalra, B. Dias, R. Zlot and A. Stentz, Market-based multirobot coordination: A survey and analysis. Technical report, CMU-RI-TR-05-13, Robotics Institute, Carnegie Mellon University, 2005.
3. A. Barabasi and E. Bonabeau, Scale free networks, *Scientific American*, pp. 60–69, May 2003.
4. D. S. Bernstein, S. Zilberstein and N. Immerman, The complexity of decentralized control of markov decision processes, In *Proceedings of the Sixteenth Conference on Uncertainty in Artificial Intelligence (UAI-00)*, pp. 32–37, 2000.
5. M. H. Burstein and D. E. Diller, A framework for dynamic information flow in mixed-initiative human/agent organizations, *Applied Intelligence on Agents and Process Management*, 2004.
6. V. A. Cicirello and S. F. Smith, Wasp nests for self-configurable factories, in *Proceedings of the Fifth International Conference on Autonomous Agents*, 2001.
7. K. Decker, K. Sycara, A. Pannu and M. Williamson, Designing behaviors for information agents, in *Procs. of the First International Conference on Autonomous Agents*, 1997.
8. B. Dias and A. Stentz, Traderbots: A market-based approach for resource, role, and task allocation in multirobot coordination, Technical report, CMU-RI-TR-03-19, Robotics Institute, Carnegie Mellon University, 2003.
9. D. Goldberg, V. Cicirello, M. B. Dias, R. Simmons, S. Smith and A. Stentz, Market-based multi-robot planning in a distributed layered architecture, in *Multi-Robot Systems: From Swarms to Intelligent Automata: Proceedings from the 2003 International Workshop on Multi-Robot Systems*, vol. 2, pp. 27–38, Kluwer Academic Publishers, 2003.
10. C. Goldman and S. Zilberstein, Optimizing information exchange in cooperative multi-agent systems, in *Proceedings of the Second International Joint Conference on Autonomous Agents and Multiagent Systems*, 2003.
11. L. Hunsberger and B. Grosz, A combinatorial auction for collaborative planning, In *Proceedings of the Fourth International Conference on Multi-Agent Systems*, pp. 151–158, 2000.
12. J. Pujol J. Delgado and R. Sanguesa, Emergence of coordination in scale-free networks, *Web Intelligence and Agent Systems*, pp. 131–138.
13. H. Kitano, S. Tadokoro, I. Noda, H. Matsubara, T. Takahashi, A. Shinjoh and S. Shimada, Robocup rescue: Search and rescue in large-scale disasters as a domain for autonomous agents research, in *Proc. 1999 IEEE Intl. Conf. on Systems, Man and Cybernetics*, vol. VI, pp. 739–743, October 1999.
14. 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. Q. Zhang, Evolution of the gpgp/taems domain-independent coordination framework, *Autonomous Agents and Multi-Agent Systems*, 9(1), 2004.

15. M. Littman, A. Cassandra and L. Kaelbling, Learning policies for partially observable environments: scaling up, in *International Conference on Machine Learning*, 1995.
16. M. L. Littman, Probabilistic propositional planning: Representations and complexity, in *Proceedings of the 14th National Conference on Artificial Intelligence (AAAI-97)*, pp. 748–761, 1997.
17. R. Mailler and V. Lesser, Solving distributed constraint optimization problems using cooperative mediation, in *AAMAS'04*, 2004.
18. J. Simmons, M. E. Gaston and M. desJardins, Agent-organized networks for dynamic team formation, in *2005 International Conference on Autonomous Agents and Multi-Agent Systems*, 2005.
19. S. Milgram, The small world problem. in *Psychology Today*, 22, pp. 61–67, 1967.
20. P. J. Modi, W. Shen, M. Tambe and M. Yokoo, An asynchronous complete method for distributed constraint optimization, in *Proceedings of Autonomous Agents and Multi-Agent Systems*, 2003.
21. R. Vincent, P. Scerri and R. Mailler, Comparing three approaches to large scale coordination, in *Proceedings of AAMAS'04 Workshop on Challenges in the Coordination of Large Scale MultiAgent Systems*, 2004.
22. D. Pynadath and M. Tambe, Multiagent teamwork: Analyzing the optimality and complexity of key theories and models, in *First International Joint Conference on Autonomous Agents and Multi-Agent Systems*, 2002.
23. T. Sandholm, Algorithm for optimal winner determination in combinatorial auctions, In *Artificial Intelligence*, 135, 2002.
24. P. Scerri, A. Farinelli, S. Okamoto, and M. Tambe, Allocating tasks in extreme teams, in *Proceedings of Fourth International Joint Conference on Autonomous Agents and Multiagent Systems*, 2005.
25. P. Scerri, D. V. Pynadath, L. Johnson, P. Rosenbloom, N. Schurr, M Si and M. Tambe, A prototype infrastructure for distributed robot-agent-person teams, in *The Second International Joint Conference on Autonomous Agents and Multiagent Systems*, 2003.
26. P. Scerri, Yang Xu, E. Liao, J. Lai and K. Sycara, Scaling teamwork to very large teams, in *Proceedings of AAMAS'04*, 2004.
27. V. Guralnik T. Wagner and J. Phelps, A key-based coordination algorithm for dynamic readiness and repair service coordination, in *Proceedings of Second International Joint Conference on Autonomous Agents and Multiagent Systems*, 2003.
28. M. Tambe, Towards flexible teamwork, *Journal of Artificial Intelligence Research*, 7, pp. 83–124, 1997.
29. A. Vick, R. M. Moore, B. R. Pirnie, and J. Stillion, *Aerospace Operations Against Elusive Ground Targets*, RAND Documents, 2001.
30. D. Watts and S. Strogatz, Collective dynamics of small world networks, *Nature*, 393, pp. 440–442, 1998.
31. Y. Xu, M. Lewis, K. Sycara and P. Scerri, Information sharing in very large teams, in *In AAMAS'04 Workshop on Challenges in Coordination of Large Scale MultiAgent Systems*, 2004.
32. Y. Xu, P. Scerri, B. Yu, S. Okamoto, K. Sycara and M. Lewis, An integrated token-based algorithm for scalable coordination, in *Proceedings of Fourth International Joint Conference on Autonomous Agents and Multiagent Systems*, 2005.

33. Y. Xu, P. Scerri, M. Lewis and K. Sycara, Comparing Market and Token-Based Coordination. in *Proceedings of Fifth International Joint Conference on Autonomous Agents and Multiagent Systems*, 2006.
34. 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.