Using Complex Network Effects for Communication Decisions in Large Multi-robot Teams

Yang Xu, Xuemei Hu, Yan Li, Dong Li, and Mengjun Yang
School of Computer Science and Engineering
University of Electronic Science and Technology of China
Chengdu, Sichuan, P.R.China
xuyang@uestc.edu.cn

ABSTRACT

Sharing information is critical to multi-robot team coordination when robots are widely deployed in a dynamic and partially observable environment. To be efficient, robots should balance well between broadcasting information and reserving limited bandwidth so that only the right information should be broadcast to the interested receivers. Robots' communication decision is normally modeled as a multi-agent decision theoretical problem. However, when the team expands to very large, the solution is classified as NEXP-COMPLETE. In this paper, in addition to building heuristic approaches to solve the decision theoretical problem based on the information context to be broadcast, we put forward a novel context-free decision model that allows fast communication decision by considering complex network attributes in large teams. Similar to human society, information should be broadcast if the action can make a good information coverage in the team. We analyze how complex network attributes can improve communication in a broadcast network. By putting forward a heuristic model to estimate those complex network attributes from robots' local view, we can build decision models either from robots' experiences or from their local incoming communications. Finally, we incorporate our algorithm in well-known information sharing algorithms and the results manifest the feasibility of our design.

Categories and Subject Descriptors

I.2.11 [**Artificial Intelligence**]: Distributed Artificial Intelligence -*Intelligent Agents, Multiagent Systems.*

Keywords

Complex network; Information Sharing; Coordination;

1. INTRODUCTION

Coordinating large groups of robots or unmanned vehicles in a dynamic and partially observable environment is compelling in various domains such as urban search and rescue [1], planet exploration [2] and military operations [3]. In such applications, information sharing is necessary because robots have to share their observations and intentions so that their mutual beliefs as well as joint activities towards the best team performance can be reached [4]. In a mobile robot team, robots use wireless connections to broadcast information to their neighbors. But when the team scales up, communication bandwidth becomes a serve bottleneck. Robots then are required to balance well between broadcasting valuable information and reserving limited bandwidth. Their decision is hard because robots in this scenario are normally highly distributed and have only a partial view of the team. Without a complete knowledge on what the others know, robots may not be able to make rational decisions [5].

The information sharing problem on how to decide to broadcast valuable information in large robot teams has been intensively studied in recent years. In the computer network research community, researchers have focused on how to avoid redundant information coverage overlay no matter what the information is. Typically, Scalable Broadcast Algorithm (SBA) [6] shares information and avoids robots rebroadcasting if most of their neighbors already have it. Low-Energy Adaptive Clustering Hierarchy (LEACH) [7] uses static hierarchical protocol and its improved protocol [13] combines clustering with predefined forming chains to proactively share information. Sensor Protocols for Information via Negotiation (SPIN) [8] is a reactive protocol which avoids redundant data transmission by meta-data negotiation between neighbors. Robots only need to forward data to the neighbors who need it. But if no request is received, robots will prefer to not sending the information.

Information sharing research in AI community starts from the view of how the information to be shared can help the robot team improve its performance [5]. Therefore, information sharing is modeled as a decision theoretical problem that, for each specific piece of information, robots evaluate the utilities if it were broadcast. When the utility of broadcasting it is beyond the cost, the robot will broadcast. However, with the partial observable capability of each robot, the information sharing utility calculation is a typical DEC-POMDP problem with the expand of team scale [9]. Extra efforts have been done on building heuristic algorithms to solve this intrinsic NEXP-COMPLETE problem in each specific coordination domain, such as myopic decision model [10], and State-Connection-Reward matrix-based model [11]. Although their performances are proven to be good, they are usually hard to be optimized.

In this paper, addition to the information sharing algorithm design in previous studies, we explore how the complex network effects can be used in the robot communication decision, and put forward an interesting approach to help lo-

Appears in: *Alessio Lomuscio, Paul Scerri, Ana Bazzan, and Michael Huhns (eds.), Proceedings of the 13th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2014), May 5-9, 2014, Paris, France.*
Copyright © 2014, International Foundation for Autonomous Agents and Multiagent Systems (www.ifaamas.org). All rights reserved.

cal robots improve their information sharing overlap while reducing communication cost on redundant message transmissions. Since a large group of robots build a complex social network as human society does, our key idea comes from a comparison of how distributed robots and human beings share information. Although a human being does not gain a complete view of his society, he can make rational decisions based on a partial understanding of his social network structure. For example, with an understanding of Milgram's six-degree separation theory, although people try passing information randomly, they will stop sending it after a limited relays [12]. As another example, without understanding who will be interested in a specific piece of information, people prefer to disseminate it to whom has more friends [18]. Therefore, in human society, information sharing can be done without precise calculations of information utility or its coverage on the team.

Next, we analyse the characters of broadcasting and explore how their complex attributes can be helpful. There are three key observations. First, by evaluating the average distance, robots can infer the information coverage from the length of information path so as to decide wether it should be rebroadcast. Second, robots with higher connections will be helpful for information dissemination. Third, the high betweenness robots are more likely to be involved in information transmission than the robots in a cluster do.

According to the observations, our second contribution comes to a combined updating function to estimate the complex network attribute values solely from robots' local view. In the very beginning when robots get very few messages, robots can learn the initiated network attributes from the introduction phase and make a good use of those values for their rebroadcasting decisions. Later, when robots get enough messages from the team, they can effectively infer the network attributes from their received information. Based on the complex network updating algorithm, we are able to build an integrated algorithm for information rebroadcast decision making from the estimated complex attribute values. In the last part, this algorithm has been incorporated into several classic information sharing approaches such as Flooding [21], SBA [6], Dominate Pruning [19] and heuristic myopic algorithm [10]. The experimental results illustrate that by taking the advantage of complex attribute evaluation, our design uniformly helps the algorithms to improve their sharing performances.

2. INFORMATION SHARING PROBLEM IN LARGE MUTI-ROBOT TEAMS

Over an ad-hoc wireless network created by the robots themselves, when a team member gets some information, the sharing problem is to decide whether it should be rebroadcast. With a dynamically changing team states such as the information distribution and network connection, robots must balance well between providing useful information and reserving network bandwidth. Specially, the robot team $R = \{r_1, r_2, ...r_j, ...\}$ is composed of a set of distributed robots and $N(t) = \bigcup_{r \in R} n(r,t)$ defines the network, where $n(r,t)$ is defined as all robots who can receive a broadcast message from robot r at time t. In short, we call the robots of $n(r,t)$ as the neighbors of r at that time. In addition, we defined r's two-hops neighbors $n^2(r,t)$ as $\{\forall r_j \in n(r_k,t) | \forall r_k \in n(r,t)\}$.

The information I considered to be broadcast is encapsulated as a communication message m, and a message m is normally described as the following data structure: $m =< ID, I, Path >$. In this data structure, a message is composed of at least three parts: the identification of the message as ID, the context of message containing the information I, and the path of message m that records the sequence of robots m transmitted. Specially, $m.Path.getFirst()$ returns the first robot in the queue, which discovers the information and is its source. When a robot r decides to broadcast m, it will append itself at the end of $m.path$ before broadcasting it.

As robots use broadcast to share information, all their neighbors at that time may be able to get the message. To avoid creating intensive redundant communication, with the message with the same $m.ID$, robots are allowed to broadcast the message at most *once* at its first time of getting the message. Therefore, we will have:

Lemma: If robots broadcast as flooding, $m.Path$ records the shortest distance between any pairs of robots in the path.

Proof: Suppose not. [We take the negation of the lemma and suppose it to be true.] As shown in Figure 1, we suppose A and B are two robots recorded in the path of m, $\exists$ a shorter distance between A and B. Then B will receive another message called m' from the shorter path, which is a copy of m broadcast by A. Broadcast at the same time from A, m' is passed in the shorter path and reached B earlier. Because the message with same content is only allowed to broadcast once by each robot, the latter message m will be dropped and B will not be recorded in $m.path$, which contradicts the supposition that B is recorded in m. [Hence, the supposition is false and the lemma is true.]

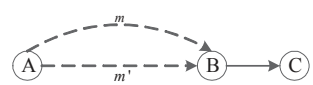


Figure 1: The proven of shortest distance

3. COMPLEX NETWORK EFFECTS IN INFORMATION BROADCASTING

In computer network research, the objective of context-free information sharing is to reach a good information overlay so that the robots interested in the information can get it, but avoid redundant information dissemination. In this paper, before introducing our algorithm, we explore how complex network effects can be used to infer robots' information sharing coverage. The key observation is that complex network effect can help robots make context free information sharing decisions without knowing the complete distribution of the other robots.

3.1 Average distance

Distances between nodes is the primary attribute evaluated in the complex network research. Starting from Milgram's six degree separation theory [12], we have learned that the average distance in a complex network is far less than the size of the network. In some cases, the average distance has been fixed to an extreme small number no matter the size of the network and this scenario is called a "Small

world effect" [14]. In a typical distributed robot team, although it may be not a typical small world network, its average distance is bounded. In most cases, when a message is broadcast for several times, the coverage of this message hardly rises and it is gradually stabilized, a phenomenon called full coverage hops. We hypothesize that the full coverage hops is linear to the average distance of the network. If the function exists, we can use it to find the full converge hops and avoid redundant broadcasting. In order to manifest how the information coverage is related to average distance, we set up a simple simulation and deploy some robot teams where the network generates by distance. In this fixed scene, we perform eight groups experiments with different team sizes (from 300 robots to 1000) which means different densities. Initially, a piece of information is broadcast from a random robot. At each time step, each message for each robot can be broadcast at most once. Therefore, within a few time steps, the information coverages are shown in Figure 2.

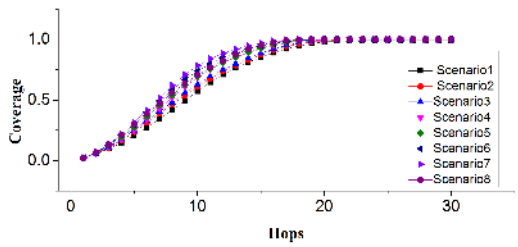


Figure 2: Information coverage by average distance

According to the experimental result, we list the average distances and full coverage hops of each scenario in Table 1. As shown in the result, there is always an optimal step to broadcast the message such that the robot team is just fully covered by the information. In addition, according to the Figure 2 we find out that the full coverage hop is about twice average distance of the network in our simulations. Therefore, if a robot knows the average distance, it can use this property to reduce broadcasting and improve the efficiency by inferring the full coverage distance with an offline learned linear function.

Table 1: Average distance and full coverage hops.

Scenario	Team size	Average degree	Average distance	Coverage on Avg_dist	Full coverage hops	Bias to Avg_dist×2
1	300	6.9	9.8	0.572	20	+2.0%
2	400	8.5	9.4	0.608	19	+1.1%
3	500	11.12	9.0	0.558	18	0.0%
4	600	12.43	8.6	0.608	18	+4.4%
5	700	15.09	8.2	0.542	17	+3.5%
6	800	17.46	8.1	0.584	17	+4.7%
7	900	19.42	7.9	0.619	16	+1.3%
8	1000	21.72	8.0	0.600	16	0.0%

3.2 Degree distribution

The distribution of the robots' connection degree is similar to human's social connections, called "Matthew Effect". Therefore, if a robot has more connections, it has more ways to disseminate information. The efficiency of sharing information can be improved by using those "high degree nodes" to broadcast. For example, in a scale free network, hubs can promote the formation of a team, and make sure agents with different abilities to cooperate with others to reach a common goal [17]. Some related work shows that in a multi-robot system which is based on scale free network or random network for peer to peer information sharing, if there are some nodes that can act as hubs, they can significantly cut down the average distance of the network [20].

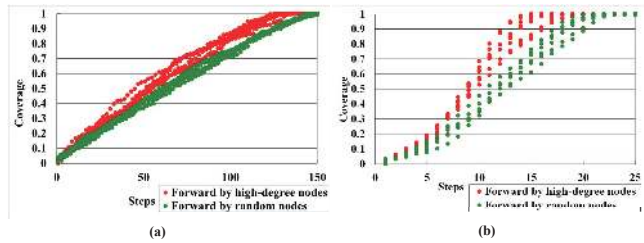


Figure 3: High degree nodes improve the information coverage

Similar to human society, to spread information quickly, people often choose the person who has more friends to push the information. In order to explore the capability of the high degree robots to improve the information coverage, we build simulation to deploy 500-robot team in seven different network deployments, where networks are generated by distance. Initially, a piece of information starts from a random robot. We use a function $Choose(methods, numbers)$ to determine the way to send information and the number of receivers. We choose one receiver in Figure 3(a) and two receivers in Figure 3(b). And the red scatter plots represent the information coverage that robots always forward messages to high degree nodes and the green scattered plots represent the coverage that information is randomly forwarded. As shown in Figure 3, although the information coverage keeps increasing in each setting, the way of forwarding to high degree nodes diffuses information much quicker than the random way. Therefore, high degree robots are more helpful to spread information.

3.3 Clustering and betweenness centrality

In a large robot team, robots tend to form a cluster, where the robots inside are densely connected. Those robots from outside are sparsely connected. Between the clusters, the robots are of high betweenness centralities. Betweenness centrality was introduced to human communication by Bavelas [15] [16], and he concluded that centrality plays an important role in group efficiency. Betweenness centrality of a node is one of the measures, which addresses the importance of a node as intermediary in the interaction between the others. Similar to human society, we hypothesize that in a typical distributed robot team, the robot with high betweenness centrality as the source node to broadcast a message, can help to speed up the spreading of this information. To manifest how cluster and betweenness centrality help to speed up the information diffusion, we simulate a team of robots around a lake as Figure 4(a) shows. Blocked by the lake, robots in each direction form a cluster. We assume that each cluster has 60 robots. In the lake, there are 5 robots connecting all the clusters around the lake.

As shown in Figure 4(a), in the simulation, we deploy three scenarios with 2 clusters (A and B), 3 clusters (A, B and C) and 4 clusters (A, B, C and D). In each scenario, we select a robot as a given information source that starts to

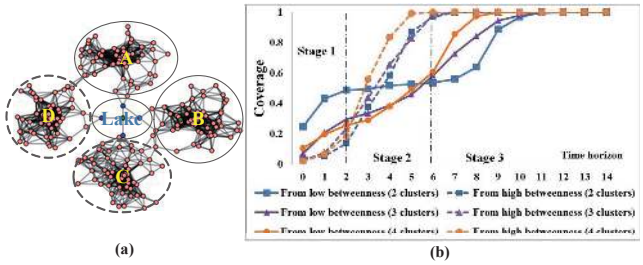


Figure 4: High betweenness helps to the information coverage

disseminate information but there are two cases. In the first case, the robot selected is one with the high betweenness that is located in the lake to connect clusters. In the other case, the robot selected is one with low betweenness that is located within a given cluster. We investigate in each case all the scenarios of the information coverage of the team (Y-axis) with the progress of time horizon (X-axis). Figure 4(b) shows the results with 2, 3 and 4 clusters, which are marked with blue, purple and orange lines respectively. In each scenario, the result that information is broadcast from low betweenness is shown in solid line, and the result that information is broadcast from high betweenness is shown in dashed line. Clearly, from Figure 4(b) we can see that in all the scenarios, the time horizon is divided into three stages. In the first stage, information starts from a low betweenness robot covers the team quicker than information from high betweenness robots. The reason is that the low betweenness robot located in a cluster can quickly cover the information within its cluster, while the high betweenness robots located in the lake can only cover very few robots in the early stage. In the second stage, while information starting from the low betweenness begins to cut across the lake and their information coverage goes into a bottleneck, the information starting from the high betweenness begins to be broadcast within all the clusters and reaches the full coverage very quickly. In the last stage, information starting from low betweenness begins to be broadcast in other clusters to reach full coverage, while the information from the high betweenness has already done that. Therefore, high betweenness centrality robots are helpful to speed up the information diffusion.

4. LOCAL OBSERVATION TO COMPLEX NETWORK ATTRIBUTES

As robots in a large team can only obtain a partial view of the team, in this section we analyze and build a local observation algorithm to help robots maintain a valid observation of the complex network attributes of the team so that they can be used to help robots' local information sharing decisions. The key of the observation function comes as two basic investigations.

- Robots can maintain complex network attributes if they previously gain a complete view of the network within a time interval. Therefore, they can use this knowledge to infer their current connections.
- Robots can effectively evaluate and update the network attributes solely from their received messages.

Therefore, in the very beginning when robots get very few messages, robots can learn the initiated network attributes from the introduction phase and make a good use of those values for their rebroadcasting decisions. Later, when robots get enough messages from the team, they can effectively infer the network attributes from their received information.

4.1 Complex network attributes maintenance

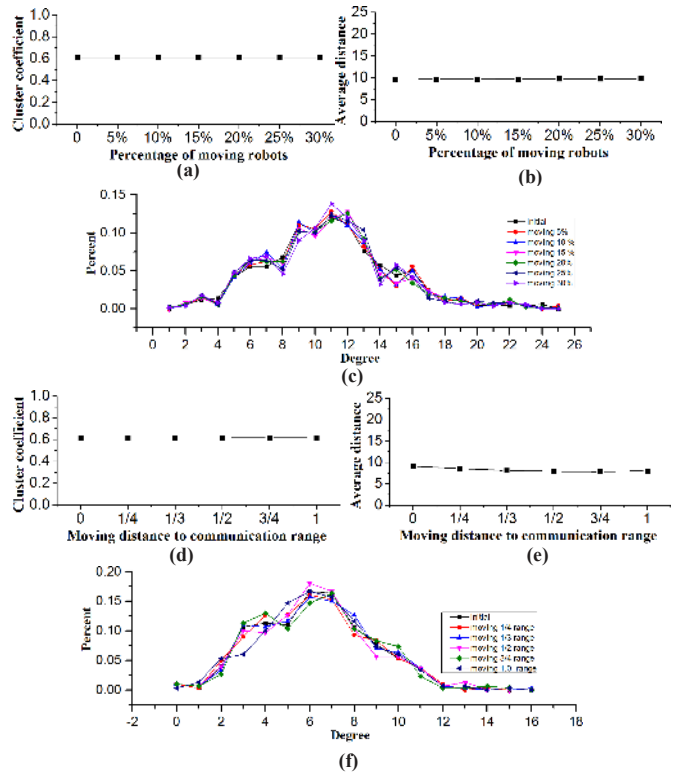


Figure 5: Complex network attributes in a dynamic robot team

Complex network attributes maintenance is based on the assumption that the variance of the complex network attributes in a large robot team is little in a short time due to robots' movements being far slower than their communication rate. Therefore, decentralized robots can effectively maintain the attributes if they previously gained a complete view of the network. For example, robots can gain a complete view of the network with a two-hop neighborhood knowledge obtained by exchanging the adjacent node lists with their neighbors. The two-hop introduction is normally used in network initiation, which is to help robots get a complete view of their connections with some centralized census approach [19]. To verify the assumption, we investigate the real multi-robotic applications. Although the team dynamically changes, not all the robots move in each time, and not all the moving robots change the network due to their limitations on moving range, task location as well as formation in a short time. In this scenario, we build two brief simulations where a 300-robot team was deployed in a given area and the network was generated according to the distance between robots. In the first simulation, we assumed in a short time interval, only a set of robots randomly move in a quarter of their communication range, and the moving

percentages are 5%, 10%, 15%, 20%, 30%. In the second simulation, 10% of robots move and their moving distances are 1/4, 1/3, 1/2, 3/4 and 1 of their communication range. Figure 5 shows our simulation results. We can see that in Figure 5(a)(d) network clustering coefficients in both experiments keep around 0.6, while in Figure 5(b)(e), the average distance keeps around 9.2. Figure 5(c)(f) illustrate the degree distribution of the network after robots' movements in each experiment. The simulations manifest that the complex network attributes change very little, even when 30% robots have moved or the moving distance has reached its communication range. Therefore, consistent with our hypothesis, once robots get a connection map, they may be able to use the map to infer their future state of complex network attributes within a short time interval, especially in the stage of the robots starting to work.

4.2 Complex network attribute updating

Complex network attribute updating function evaluates robots' local complex network attributes solely based on the massages they have received. With the time passed by, although their network keeps changing, robots will gain more and more information to coordinate their joint activities [5]. With the rich information base (IB), robots are able to infer their local complex network attributes from their local views, which are built from their previously received messages.

4.2.1 Local network attribute model

Before the updating algorithm is introduced, we define the robot's local network attribute model. Since robots cannot gain a complete view of the team, their local model is slightly different from the complex network model of the whole network which has to be centrally counted. The local network attribute model for a robot r is composed of three parts: $< r.avg_dist, r.degree_distribution, r.freq_nodes >$

r.avg_dist defines robot r's local view of the average distance of the network. The robot always assumes that it is the center of the network and the information source is randomly scattered in the network. Therefore, the average distance of the network from r's view is the average path length of the messages it has received. The full coverage distance is normally linear to its local average distance and according to section 3.1, it is twice of $r.avg_dist$.

$$r.avg_dist = \frac{\sum_{m \in r.IB} m.path.length}{|r.IB|}$$

r.degree_distribution defines robot r's view of its neighbor's degree distribution. Since r cannot observe the whole team, it cannot observe the network degree distribution. But it can easily understand the degree for a given neighbor r_i by count its 2 hops neighbors:

$$\{m.path.get2ndLast() | \forall m \in IB, m.path.getLast() = r_i\}$$

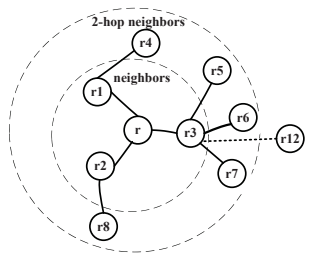
where $m.path.get2ndLast()$ popups the second last robot of the message which comes to r from neighbor r_i.

r.freq_nodes defines robot r's local view of high frequent nodes which are not within its 2-hop neighbors. From the analysis in section 3.2 and 3.3, they could be high degree nodes within a cluster or the high betweenness nodes in the robot team. As r cannot get a good view of what they are, those high frequent nodes can help to diffuse information. The frequent nodes can be found from the received information and each frequent node will be recorded as a data structure

$$< ID, frequency, \{..., neighbor_j, ...\} >$$

ID and $frequency$ denote the ID of the frequent node and its frequency that appeared in the received information. After these, there is a neighbor list, called $neighbor_list$, and each $neighbor_j$ denotes that the neighbor of robot r who sends a piece of information to r that was broadcast by the frequent node. Please note that although robot only uses the first few high frequency nodes in $r.freq_nodes$ for their decision, $r.freq_nodes$ is required to be long enough because some low frequency nodes may become high frequent by extensive messages. They still can be sorted and recorded in $r.freq_nodes$.

To manifest the model, considering the following example shown in figure 6. For robot r's IB, there are five pieces of information m_1, m_3, m_3, m_4, m_5, the paths are shown in Figure 6(b). Then, $r.avg_dist = 2.2$, $r.degree_distribution =< \{r_1, < r_4 >\}, \{r_2, < r_8 >\}, \{r_3, < r_5, r_6, r_7 >\} >$, and Figure 6(c) shows $r.freq_nodes$.



(a) Robot r' s IB

$m_1.path = r_{13} \rightarrow r_4 \rightarrow r_1$
$m_2.path = r_5 \rightarrow r_3$
$m_3.path = r_{12} \rightarrow r_{11} \rightarrow r_6 \rightarrow r_3$
$m_4.path = r_{12} \rightarrow r_{10} \rightarrow r_7 \rightarrow r_3$
$m_5.path = r_9 \rightarrow r_8 \rightarrow r_2$

(b) Messages' Paths

frequent node	frequency	neighbor
r_3	3	r_5, r_6, r_7
r_{12}	2	r_{10}, r_{11}

(c) r.freq_nodes

frequent node	frequency	neighbor
r_3	4	r_5, r_6, r_7
r_{12}	4	r_{10}, r_{11}, r_{14}

(d) r.freq_nodes aftering updating

Figure 6: An example of local model updating function

4.2.2 Local network attribute updating

A robot r can build it's local network attribute model from its previous received information, and for each piece of incoming message, it can be used to update this model. For an incoming message m, the local updating function for robot r is described as algorithm 1.

In this algorithm, robot r will firstly update its local average distance (line 1). Next, it searches on m's path to see its neighbor and its 2-hop neighbor (line 2-3). If the neighbor has been in r's neighbor list, it will check whether the $2HopNgb$ is in the neighbor's 2-hop neighbor set (line 4-5). Otherwise, they should be updated into r's local neighborhood model (line 6-10). Thirdly, to update the high frequency nodes, each node in $m.path$ should be used (line 12). If the nodes have been in the high frequency node list, its frequency as well as its connection neighbor list should be updated (line 13-17). Otherwise, the node should be added to the frequency node list and its frequency and connection neighbor list should be initiated (line 19-21). After r updates all the frequent nodes, the frequent nodes list need to be sorted according to the frequency (line 24). Then message m will be added into r's information base (line 25).

Considering the example above, if a new message m_6 comes and $m_6.path =< r_{12} \rightarrow r_{14} \rightarrow r_6 \rightarrow r_3 >$, after updating we will have $r.avg_dist = 2.3$. But $r.freq_nodes$

Algorithm 1 $localUpdating(r, m)$

$r.avg_dist \leftarrow \frac{r.avg_dist \times |r.IB| + m.path.length}{|r.IB|+1}$;
$Ngb \leftarrow m.path.getLast()$;
$2HopNgb \leftarrow m.path.get2ndLast()$;
if $Ngb \in r.neighbors$ **then**
  if $2HopNgb \notin r.neighbors[Ngb].2hopsNeighbor$ **then**
    $r.neighbors[Ngb].2hopsNeighbor.add(2HopNgb)$;
  end if
else
  $r.neighbors.add(Ngb)$;
  $r.neighbors[Ngb].2hopsNeighbor.add(2HopNgb)$;
end if
for all $r_j \in m.path$ **do**
  if $r.freq_nodes.search(r_j) = true$ **then**
    $r.freq_nodes[r_j].frequenecy++$;
    if $Ngb \notin r.freq_nodes[r_j].neighbor_list$ **then**
      $r.freq_nodes[r_j].neighbor_list.add(Ngb)$;
    end if
  else
    $r.freq_nodes.add(r_j)$;
    $r.freq_nodes[r_j].frequenecy \leftarrow 1$;
    $r.freq_nodes[r_j].neighbor_list, add(Ngb)$;
  end if
end for
$r.freq_nodes.sort()$;
$r.IB.add(m)$;

and $r.degree_distribution$ model remain, as shown in Figure 6(d). Note that, a key advantage of the design is that there is a reinforced effect because the better the local model is, the right information will be broadcast. In addition, robots gain more valuable information to polish their local models to make right decisions.

5. INFORMATION SHARING ALGORITHM WITH COMPLEX NETWORK EFFECTS

As explained, the information sharing problem for a distributed robot is to decide whether a newly received message m should be rebroadcast. When robots can maintain their local complex network attribute model either from complex network attribute maintenance function in section 4.1 or complex network attribute updating algorithm in section 4.2, it can be used for their sharing decision. According to the analysis in section 3, the key lies on how complex network denotes the information coverage. In summary, when robot r well maintains its $< avg_dist, degree_distribution, freq_nodes >$ model, it can infer that:

The hops on full information coverage is linear to the average distance of the team. When the length of path is more than the full coverage hops, it should be stopped to be rebroadcast. Therefore, we can define a threshold: $threshold_{avg_dist} = k \times r.avg_dist$, where k is a predefined parameter and most times, $k = 2$ as the experimental result in Table 1. r should less likely rebroadcast m when $m.path.length$ exceeds the threshold.

Higher degree nodes help to achieve information diffusion. Therefore, if r finds that its high degree neighbors are not covered, it is more like to rebroadcast m.

High frequency nodes which consist of either high degree nodes within a cluster or high betweenness nodes, help to sharing information. Therefore, if robot r finds that its high frequency nodes are not covered by m, it is more likely to rebroadcast m. However, there are two cases. First, although the frequency nodes are not in $m.path$, if most neighbors in frequency nodes' lists are covered, r cannot infer that they are not covered because the broadcasts to them may have been sent without knowing by r. On the other hand, if the neighbors on the frequency nodes' lists are not covered, m is more likely rebroadcast.

Algorithm 2 $BooleanlocalDecision(r, m)$

1: $decision \leftarrow false$;
2: $threshold_{avg_dist} \leftarrow k \times r.avg_dis$;
3: $degree \leftarrow 0$;
4: $fullDegree \leftarrow 0$;
5: **for all** $r_j \in r.neighbors$ **do**
6: **if** $r_j \notin m.path$ **then**
7: $degree += r.neighbors[r_j].2hopsNeighbor.length$;
8: **end if**
9: $fullDegree += r.neighbors[r_j].2hopsNeighbor.length$;
10: **end for**
11: $accessList \leftarrow select(r.freq_nodes, M)$;
12: **for all** $freq_node \in accessList$; **do**
13: **if** $freq_node \in m.path$ **then**
14: $accessList.delete(freq_node)$;
15: **end if**
16: **for all** $acessNgb \in freq_node.neighbor_list$ **do**
17: **if** $acessNgb \in m.path$ **then**
18: $accessList.delete(freq_node)$;
19: **end if**
20: **end for**
21: **end for**
22: $degree2 \leftarrow 0$;
23: $unvistedNgb \leftarrow r.neighbors$
24: **for all** $freq_node \in accessList$; **do**
25: **for all** $acessNgb \in freq_node.neighbor_list$ **do**
26: $unvistedNgb.delete(acessNgb)$;
27: **end for**
28: **end for**
29: $unvistedNgb \leftarrow r.neighbors - unvistedNgb$
30: **for all** $r_j \in unvistedNgb$ **do**
31: $degree2 += r.neighbors[r_j].2hopsNeighbor.length$;
32: **end for**
33: $Pr = \alpha(1 - \frac{m.path.length}{threshold_{avg_dist}}) + \beta\frac{degree}{fullDegree} + \gamma\frac{degree2}{fullDegree}$
34: $decision \leftarrow lottery(Pr)$;
35: $return(decision)$;

Algorithm 2 briefly describes the local decision process of robot r for each piece of incoming message m. Robot r first evaluates the threshold based on its local average distance (line 2). Next, r will search its neighbors to find who is not covered by m (line 5, 6) and count the uncovered neighbors' degree for their importance (line 7-9). Thirdly, robot r selects the top M high frequency nodes as the $accessList$ (line 11). By using $accessList$, r can find (1) whether the high frequency nodes has been covered by m and (2) the neighbor of r who connects the high frequency nodes have been covered (line 12-21). Then, by deleting those covered high frequency nodes, the robot counts the neighbors that connect potential uncovered high frequency nodes (line 22-32). Line 33 combines all the three factors: threshold of

Table 2: The performace in different time steps

Time	Statics	FLOOD		SBA		Dominate Pruning		Myopic	
		Original	Heuristic	Original	Heuristic	Original	Heuristic	Original	Heuristic
t=5	Utility	18.66%	17.55%	20.98%	19.87%	20.35%	17.45%	18.94%	17.43%
	Cost	487.5	392	602.8	442.1	531.4	329.2	507.6	385.1
	Improved	16.96%		29.14%		38.42%		21.30%	
t=10	Utility	40.85%	35.98%	43.76%	35.57%	42.27%	34.93%	41.17%	34.43%
	Cost	2088.3	1529.2	2220.4	1550.6	2207.4	1355.2	2156.6	1534
	Improved	20.28%		16.40%		34.60%		17.57%	
t=15	Utility	73.21%	69.02%	76.25%	70.15%	75.40%	65.11%	71.24%	69.16%
	Cost	3632.6	2621.5	3618.5	2648.4	3670.2	2419.8	3649.6	2719.2
	Improved	30.64%	0.00%	25.70%	0.00%	30.97%		30.30%	
t=20	Utility	96.44%	88.88%	97.78%	91.66%	98.04%	88.74%	96.73%	91.12%
	Cost	4677.2	3270.9	4464.3	3212.7	4704	3119.8	4692.6	3325.9
	Improved	31.79%		30.26%		36.48%		32.91%	
t=25	Utility	99.34%	93.03%	98.86%	93.01%	99.33%	92.05%	99.14%	95.78%
	Cost	4791	3393.5	4486	3277.9	4791	3206.9	4791	3448.3
	Improved	32.21%		28.76%		38.45%		34.23%	

Table 3: The experimental results for different density

Team size in (600*600)	Statics	FLOOD		SBA		Dominate Pruning		Myopic	
		Original	Heuristic	Original	Heuristic	Original	Heuristic	Original	Heuristic
300	Utility	8.71%	4.25%	10.60%	2.83%	9.72%	3.18%	9.15%	2.41%
	Cost	114.4	3.9	71.2	3.5	100.6	3.6	121.3	3.3
	Improved	1331.30%		443.12%		814.23%		868.15%	
500	Utility	99.56%	93.90%	99.56%	94.20%	99.54%	90.78%	99.56%	94.56%
	Cost	4791	3436.8	4485.2	3324.6	4791	3207.6	4791	3379.6
	Improved	31.50%		27.65%		36.22%		34.64%	
800	Utility	100.00%	97.14%	100.00%	97.50%	100.00%	100.00%	100.00%	100.00%
	Cost	11452	10017	11195	10031	11452	9694	11452	10153
	Improved	11.06%		8.81%		18.14%		12.79%	
1000	Utility	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%
	Cost	17388	15275	17112	15190	17388	14658	17388	15287
	Improved	13.83%		12.65%		18.63%		13.74%	
1500	Utility	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%
	Cost	38652	33080	38360	33091	38652	32823	38652	33196
	Improved	16.85%		15.92%		17.76%		16.44%	

Table 4: The experimental results for scalability

Team size & area size	Statics	FLOOD		SBA		Dominate Pruning		Myopic	
		Original	Heuristic	Original	Heuristic	Original	Heuristic	Original	Heuristic
300 & 450*450	Utility	100.00%	80.44%	100.00%	80.66%	100.00%	78.65%	100.00%	80.70%
	Cost	2839	1960.3	2732	1938.7	2839	1815.3	2839	1981.2
	Improved	16.49%		13.67%		23.00%		15.64%	
500 & 600*600	Utility	99.56%	93.90%	99.56%	94.20%	99.54%	90.78%	99.56%	94.56%
	Cost	4791	3436.8	4485.2	3324.6	4791	3207.6	4791	3379.6
	Improved	31.50%		27.65%		36.22%		34.64%	
800 & 750*750	Utility	100.00%	93.80%	100.00%	93.47%	100.00%	92.39%	100.00%	95.02%
	Cost	7605	5122.5	7267.5	4921.1	7605	4814.6	7605	5134.4
	Improved	39.25%		38.03%		45.94%		40.74%	
1000 & 850*850	Utility	99.58%	93.40%	99.74%	93.29%	99.65%	91.60%	99.63%	93.41%
	Cost	9209	6474.7	8823.2	6305.5	9209	6088.9	9209	6487.5
	Improved	33.40%		30.88%		39.03%		33.10%	
1500 & 1050*1050	Utility	100.00%	90.92%	100.00%	91.27%	100.00%	87.22%	100.00%	90.09%
	Cost	13894	9178.7	13301	9031.2	13894	8638.5	13894	9165.5
	Improved	37.63%		34.43%		40.28%		36.56%	

Table 5: The experimental results in dynamic networks

Move Ratio	Statics	FLOOD		SBA		Dominate Pruning		Myopic	
		Original	Heuristic	Original	Heuristic	Original	Heuristic	Original	Heuristic
0%	Utility	100.00%	90.75%	100.00%	94.59%	100.00%	92.14%	100.00%	93.34%
	Cost	4698	3412.3	4478.4	3433.7	4698	3271.9	4698	3473.8
	Improved	24.94%		23.37%		32.30%		26.23%	
5%	Utility	100.00%	90.83%	99.13%	91.88%	99.57%	91.51%	99.78%	93.07%
	Cost	4706.7	3387.1	4471.5	3287.6	4673.9	3223.1	4705	3443.2
	Improved	26.22%		26.06%		33.27%		27.46%	
10%	Utility	100.00%	92.60%	99.77%	89.97%	100.00%	75.22%	100.00%	87.13%
	Cost	4577	3197.2	4293.6	3052.7	4561.4	2469.8	4576.9	3117.8
	Improved	32.56%		26.83%		38.92%		27.91%	
15%	Utility	100.00%	81.87%	98.96%	88.72%	99.79%	81.14%	99.55%	92.92%
	Cost	4562.7	2943.1	4254	3052	4603.3	2716.4	4603.9	3332.5
	Improved	26.92%		24.96%		37.79%		28.95%	
20%	Utility	99.57%	90.92%	99.35%	85.11%	99.14%	65.78%	100.00%	91.43%
	Cost	4599.6	3286	4304	2868.1	4565.8	2202.3	4563.8	3136.7
	Improved	27.82%		28.56%		37.56%		33.03%	

average distance, uncovered high degree neighbors and uncovered high frequency nodes and produces the probability whether m should be rebroadcast. Note that α, β, γ are predefined importance factors. Finally, the robot decides to perform broadcast by lottery based on the computed probability (line 34-35).

6. SIMULATION AND RESULTS

In this section, we present an abstract simulation test bed to manifest our design. The basic setting is to simulate a group of 500 robots deployed and coordinated in a given 600×600 units square. For each robot, the wireless communication range r is 45. In this deployment, the average degree of each robot is 9, and their average distance is close to 10. However, for the random deployment, a few of the robots may be isolated. In the simulation, one robot is randomly chosen as the source node to broadcast an important message m, and before it is broadcast, we randomly choose 10% robots in the team as sink robots who are interested in it and each of those robots will first broadcast one piece of related message to express their interests in m. Each piece of related message will be allowed to rebroadcast TTL hops and TTL follows Poisson distribution: $P(TTL = k) = \frac{e^{-\lambda}\lambda^k}{k!}$. Where $\lambda = 15$ is the control parameter. Next, the message m is broadcast and robots are required to balance well between sharing m to sink robots and avoiding communication cost.

In our experiments, we implemented four classic information sharing algorithms (labeled as "Original"): Flooding [21], SBA [6], Dominate Pruning [19] and myopic algorithm [10]. In addition, for each algorithm, we integrate our heuristic local updating and decision model (labeled as "Heuristic") to help improve their performance. The experiments are evaluated with three performance categories:

Utility is the summary of the information utility of the team in each experiment. A given amount of utility can be credited to the team when m reaches a sink robot;

Cost records the summary of the communication cost in broadcasting m around the team. When a robot broadcasts m, one sending cost and one receiving cost per neighbor will be counted;

Improved reflects the improvement of robots' trade off between broadcasting m to gain higher *utility* and minimizing *cost* to stop rebroadcasting. By integrating our algorithm into those "original" algorithms, this value is computed as

$$Improved = \frac{Utility_O/Utility_H}{Cost_O/Cost_H} - 1$$

where $Utility_O$ and $Cost_O$ represent the $Utility$ and $cost$ with "original" algorithms, while $Utility_H$ and $Cost_H$ are those with our "heuristic" algorithm.

In the following experiments, we set $\alpha = 0.5$, $\beta = 0.2$ and $\gamma = 0.3$ in our heuristic algorithm and each experimental result is based on 100 runs. Our experimental results are presented with five tables. Table 2 presents the performances of the 500-robot team in different time steps. Table 3 presents the performances on different robot density, where each team is consisted of 300, 500, 800, 1000 and 1500 robots. Since the team size varies and the teams are limited in the same size of square, their densities are different. Dif-

Table 6: The "*Improved*" results for different TTL

λ	FLOOD	SBA	Dominate Pruning	Myopic
9	28.97%	24.52%	42.87%	34.95%
12	27.50%	27.11%	39.59%	32.46%
15	32.21%	28.76%	38.45%	34.23%
18	33.66%	29.12%	34.78%	31.81%
21	30.91%	26.64%	35.84%	33.55%

ferent with Table 3, we present the performances in Table 4 to verify the scalability. Although similar teams with 300, 500, 800, 1000 and 1500 robots are deployed, with different size of squares, the robot teams maintain "flat" network density. Table 5 shows how the 500-robot team performs when its network dynamic changes, and in this experiment, we set 0% to 20% robots in the team to dynamically move. Table 7 in short display the results when the related messages are broadcasted with different TTLs, which is controlled by λ.

All the experimental results in those tables, as we expected, uniformly manifest that the "heuristic" model by integrating our complex network attribute based local decision model helps robots to significantly improve the balance between *Utility* and *Cost*. Although this model in most cases gains less *utility*, it is able to cut off most unnecessary communications so that effective information sharing in a large robot team can be reached. Specially, As shown in the Table 3, when the team only has 300 robots, many robots are isolated and our local model helps a lot to understand the network and improves their performances.

7. CONCLUSION

In this paper, by leveraging how complex network effects can be used for multi-robot information sharing, we presented a novel context-free decision approach. By building robots' local decision model consisting of their local average distance, degree distribution and high frequency node estimations, they are able to dynamically update and make rational decisions. There are three key advantages in the approach. First, the decision model is light weight and allows robots to make fast decision in large teams. Second, it is a reinforcement model because the better incoming information the better local decision model and this model can promote information sharing to improve their knowledge. Thirdly, this approach is compatible with any existing information sharing algorithms and our experiments also manifested that it can be integrated and help those algorithms work better.

8. ACKNOWLEDGEMENT

This research has been sponsored by NSFC 61370151, 60905042, 61202211, National Science and Technology Support Program Foundation of China 2012BAI22B05 and Central University Basic Research Funds Foundation of China ZYGX2011X013.

9. REFERENCES

[1] P. Scerri, P. Velagapudi, B. Kannan and A. Valada. Real-world testing of a multi-robot team. *The 11th International Joint Conference on Autonomous Agents and Multiagent Systems*, 2012.

[2] S. Smith, M. Schwager, etc. Persistent robotic tasks: Monitoring and sweeping in changing environments. *IEEE Transactions on Robotics*, 28(2), 2012.

[3] R. Giachetti, V. Marcelli, etc. An agent-based simulation model of human-robot team performance in military environments. *Systems Engineering*, 2013.

[4] M. Roth, R. Simmons and M. Veloso. What to communicate? execution time decision in multi-agent POMDPs. *The 8th International Symposium on Distributed Autonomous Robotic Systems*, 2006.

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

[6] W. Peng and X. Lu. On the reduction of broadcast redundancy in mobile ad hoc networks. *Mobile and Ad Hoc Networking and Computing*, 2002.

[7] W. Heinzelman, A. Chandrakasan, H. Balakrishnan. Energy-efficient communication protocol for wireless microsensor networks. *the IEEE 33rd Annual Hawaii International Conference on System Sciences*, 2000.

[8] W. Heinzelman, J. Kulik, and H. Balakrishnan. Adaptive protocols for information dissemination in wireless sensor networks. *5th International Conference on Mobile Computing and Networking*, 1999.

[9] C. Goldman and S. Zilberstein. Optimizing information exchange in cooperative multi-agent systems. *the 2nd International Joint Conference on Autonomous Agents and Multiagent Systems*, 2003.

[10] R. Becker and A. Carlin. Analyzing Myopic Approaches for Multi-Agent Communication. *Computational Intelligence*, 25(1), 2009.

[11] L. Zhu, Y. Xu and P. Scerri. An information sharing algorithm for large dynamic mobile multi-agent teams. *the 11th International Conference on Autonomous Agents and Multiagent Systems*, 2012.

[12] S. Milgram. The small world problem. *Psychology Today*, 2(1), 1967.

[13] H. Karl and A. Willig. *Protocols and architectures for wireless sensor networks*. Wiley, 2007.

[14] M. Kuperman, G. Abramson. Small world effect in an epidemiological model. *Physical Review Letters*, 2001.

[15] A. Bavelas. A mathematical model for group structures. *Human Organization*, 7(3), 1948.

[16] A. Bavelas. Communication patterns in task-oriented groups. *Journal Acoustical Society of America*, 1950.

[17] O. Sen and S. Sandip. Effects of social network topology and options on norm emergence. *Coordination, Organizations, Institutions and Norms in Agent Systems V*, 2010

[18] F. Fu and C. Hauert. Reputation-based partner choice promotes cooperation in social networks. *Physical Review E*, 2008

[19] H. Lim and C. Kim. Multicast tree construction and flooding in wireless ad hoc networks. *Workshop on Modeling, Analysis and Simulation Mobile System*, 2000.

[20] P. Crucitti and V. Latora. Efficiency of scale-free networks: error and attack tolerance. *Physica A: Statistical Mechanics and its Applications, 320*, 2003.

[21] W. Chen, R. Guha, etc. A survey and challenges in routing and data dissemination in vehicular ad hoc networks. *Wireless Communications and Mobile Computing*, 11(7), 2011