Self-Organization through Bottom-up Coalition Formation*

Mark Sims
Dept. of Computer Science
University of Massachusetts
Amherst, MA 01003 USA
msims@cs.umass.edu

Claudia V. Goldman
Dept. of Computer Science
University of Massachusetts
Amherst, MA 01003 USA
clag@cs.umass.edu

Victor Lesser
Dept. of Computer Science
University of Massachusetts
Amherst, MA 01003 USA
lesser@cs.umass.edu

ABSTRACT

We present a distributed approach to self-organization in a distributed sensor network. The agents in the system use a series of negotiations incrementally to form appropriate coalitions of sensor and processing resources.

Since the system is cooperative, we have developed a range of protocols that allow the agents to share meta-level information before they allocate resources. On one extreme the protocols are based on local utility computations, where each agent negotiates based on its local perspective. From there, a continuum of additional protocols exists in which agents base decisions on marginal social utility, the combination of an agent's marginal utility and that of others. We present a formal framework that allows us to quantify how social an agent can be in terms of the set of agents that are considered and how the choice of a certain level affects the decisions made by the agents and the global utility of the organization.

Our results show that by implementing social agents, we obtain an organization with a high global utility both when agents negotiate over complex contracts and when they negotiate over simple ones. The main difference between the two cases is mainly the rate of convergence. Our algorithm is incremental, and therefore the organization that evolves can adapt and stabilize as agents enter and leave the system.

*Effort sponsored by the Defense Advanced Research Projects Agency (DARPA) and Air Force Research Laboratory Air Force Materiel Command, USAF, under agreement number #F30602-99-2-0525, by the National Science Foundation under Grant number DMI0122173, and in part by the Air Force Office of Scientific Research under Grant No. F49620-03-1-0090. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright annotation thereon. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the Defense Advanced Research Projects Agency (DARPA), Air Force Research Laboratory, NSF, Air Force Office of Scientific Research, or the U.S. Government.

AAMAS'03, July 14–18, 2003, Melbourne, Australia.

Categories and Subject Descriptors

I.2.11 [**Artificial Intelligence**]: Distributed Artificial Intelligence—*Multiagent Systems*

General Terms

Design, Experimentation, Performance, Reliability

Keywords

self-organizing systems, emergent organization, multi-agent systems, negotiation, coalition formation

1. INTRODUCTION

The process of self-organization in a large-scale, open system is of key importance to the performance of the system as a whole. An appropriate organization can limit control and communication costs, significantly improving performance. We have observed useful system performance with an organization of as few as sixteen agents [3]. A static organization, however, may not be able to handle a dynamic environment. Re-organization, therefore, is necessary during operation as agents and resources are removed or added, or when their characteristics change. In this paper we present a distributed, incremental approach to self-organization through bottom-up coalition formation that we have applied to the distributed sensor network (DSN) of the EW Challenge Problem [3]. The process uses negotiation iteratively to enable managers of coalitions to refine the set of coalitions in the system to achieve efficient allocations of sensors and adapt dynamically to environmental changes.

Horling et al. [3] describe the EW Challenge Problem domain in detail. It consists of homogeneous sensor agents distributed throughout a region. The agents are fixed and communicate using an eight-channel RF system in which each can use only one channel at a time. An organization in such a domain helps facilitate the efficient assignment of tracking tasks to particular agents and limit contention on communication channels. We employ a one-level hierarchy in which agents are distributively divided into sectors, each of which has a manager. The manager monitors what is currently being tracked by its sector and, as new data arises, determines whether it needs to assign a new tracking task to an agent in its sector. To do this the manager must model what is currently being tracked and the internal states of the agents in its sector. Furthermore, when it assigns tracking tasks, the manager attempts to minimize contention on any one channel. Therefore, the division of the agents into sectors helps to minimize not only the computational load on

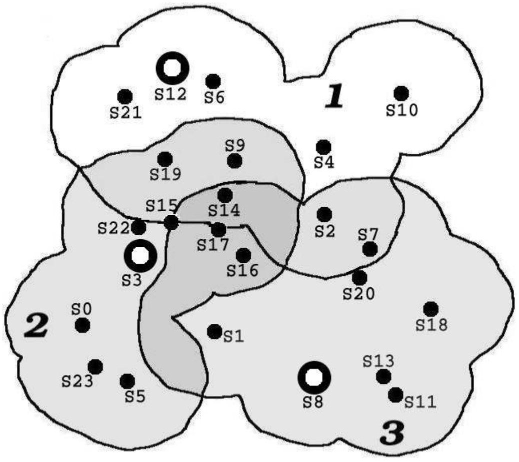

Figure 1: EWChallenge Domain

the managers but also the number of messages sent on any one channel. For our self-organization techniques, we assume all sector managers communicate with each other over channel zero and that the sector managers assign channels other than zero to agents as they enter the sector.

Figure 1 illustrates the domain. The empty circles are sector managers; the filled circles are sensor agents that are not managers. Although in actuality each agent has three separate sensor heads, for the purposes of this paper we assume agents have viewable areas of 360°. The sector areas represented by the "clouds" in the picture are defined by the intersection of the viewable areas of the sensors in the sectors. The areas of overlap show where the region covered by one sector intersects the region covered by another. Although sector boundaries overlap, each agent belongs to a single sector. To track a vehicle best, at least three sensor agents are required to triangulate the position of a vehicle moving through the region. If there are not three such sensors within the sector responsible for tracking, agents within that sector may request sensor data from other sectors.

We also assume that there is an overhead associated with passing a tracking task from one sector to another and that accessing sensor data of agents in different sectors may incur communication delays due to multiple hops or channel contention. It is desirable, therefore, for a sector to track well for as long as possible to minimize how often a tracking task is passed off to another sector, how often a tracking agent must access sensor data from different sectors, and how often tracking agents must negotiate over sensor allocation.

Given the need for an organization such as that described above, the motivation for applying self-organization techniques is the need to move from predefined, hand-generated configurations of sensors and organizational relationships as in [3] to arbitrary configurations and dynamic construction of organizational structure. To achieve this, we use a bottom-up coalition formation technique to enable the agents in the system to construct the organization dynamically in a decentralized manner.

Through coalition formation, agents in a large system faced with a set of tasks partition themselves to maximize system performance. By this process, the system moves from being a set of single agents to a set of either disjoint or overlapping coalitions of agents. Our algorithm enables self-organization through coalition formation by having the agents discover their organizational relationships while partitioning themselves around the subtasks of a high-level task.

Our approach is similar to that of Shehory and Kraus [9, 10] in that it applies to a cooperative system of agents in a non-super-additive environment. We assume an overhead associated with each new member of a coalition and that a coalition reaches a point beyond which adding a member is no longer beneficial. Beyond these similarities our approach varies considerably from that of [9, 10]. In their work agents have a more global view of others in the system. The agents may not be aware of every other agent, but the assumption is that they know of a large number. Each agent then calculates a subset of the coalitions it may belong to, and the system engages in a greedy process of choosing coalitions based on their computed coalitional values. If the population changes, the coalition formation process must restart. In contrast, our approach is an incremental, local one in which agents need not know of that many other agents around them and the process of coalition formation can continue and adapt if the population changes. Another difference is that Shehory and Kraus [10] allow for overlapping coalitions. In our current work, we restrict our attention to disjoint coalitions although in future work, we plan to extend our techniques to overlapping coalitions where sensor agents may have membership in more than one sector.

Two other sets of related work are [2] and [4]. While the work of Horling et al. [2] does involve a local adaptation process, it uses evaluations of system performance to adapt an organization. Organizational adaptation in the work of Ishida et al. [4] is based on the tasks that enter the system and the system's current load. It is not an iterative search process designed to converge on a good organization. In both sets of work the adaptations may be revised as the situation changes, but the process of adapting is a single shot. The Contract Net Protocol [11] is also related to our work; we discuss it in Sections 3 and 4.

Finally, a common coalition formation problem related to the distribution of tasks asks: Given a fixed set of tasks and a set of cooperative agents, how can we pick groups of agents best suited to those tasks? Our problem is slightly different and resembles the work in Goldman and Rosenschein [1] aimed at partitioning information domains to facilitate future information retrieval requests. In our DSN, the system is given the high-level task of providing coverage for a region. This task encompasses the future tracking tasks that the system will perform but does not know *a priori*. The goal is to subdivide the region and assign portions of it to sectors (coalitions) of agents so that each sector is best able to perform the tracking tasks that it encounters.

Sections 2 and 3 present our model. Section 4 describes empirical results from testing different negotiation protocols that lead to different stable organizations. Section 4, also analyzes the performance of the organizations evolved in terms of their rate of convergence, fault tolerance, and message traffic while tracking. We conclude in Section 6 after presenting a formal framework in Section 5 that allows us to analyze the decisions that the agents make as a function of the value of the information they hold. We distinguish between local information agents and k-social agents who may be able to obtain information about k other transactions happening at the same time.

2. PROBLEM DESCRIPTION

In order to formalize our problem, we present the following assumptions and definitions.

2.1 Assumptions

In addition to the assumptions stated in the Introduction, we make the following assumptions:

- Although agents are arbitrarily distributed, there are a sufficient number of them and they are arranged such that every point in the region assigned to the system has at least one sensor that can see it.
- Although agents may enter or leave the system at any time, the agent population does not vary dramatically from one instance to the next. If the population were to fluctuate wildly, attempting to build organizational structure would be futile.
- An ideal sector has between eight and ten agents in it and at least three agents can see every point in the region it is responsible for. As the first assumption suggests, this is not always possible.

2.2 Definitions

At any time t we have a set of agents in the system, $A = \{A_1, A_2, ...A_n\}$, and each agent A_i has a vector of capabilities $B = \{b^i_1, b^i_2, ..., b^i_n\}$. For example, in the sensor domain presented, each agent controls the sensor associated with it and the capabilities are the regions each is able to cover.

Different organizations can result from a given set A of agents with their corresponding capabilities. These organizations are instantiations of different organizational templates. An organizational template is given by the organization's high-level goals, the roles that exist within the organization, the organization's control and communication hierarchies, and its evaluation function. Our implementation instantiates a simple template in which agent organizations are built using a single level hierarchy.

A sector S_i is a coalition of agents drawn from A that work together to accomplish a task. A sector manager SM is a representative of its sector that is responsible for handling negotiations with other sectors (as well as task and channel allocation within the sector). The manager may not remain constant throughout the life of the sector. Since the agents in A are homogeneous, any agent can serve as a sector manager.

Each sector S_i has an area defined by the viewable areas of the sensor agents that it is responsible for as shown by the "clouds" in Figure 1. We denote this as $Area_{S_i}$. Each sector has a utility value U_{S_i} that is a function of the number of agents in the sector and how well the sector can provide coverage of the sub-region it is responsible for. More specifically, let AN_i be the average number of sensors in S_i that can see each point in the region covered by S_i. Let NU_i be a function ranging between 0 and 1 dependent on the number of agents in S_i. Space limitations preclude a complete description of NU_i, but it is an empirically defined factor that equals 1 if S_i has eight sensors, falls off slowly at first so that between 6 and 10 agents still gets a fairly high rating, and then falls off quickly for sectors of other sizes. A sector with only one agent, for instance, has $NU_i = 0.001$. Finally, $U_{S_i} = AN_i \times NU_i$.

The coalition formation process results in a set of sectors called a coalition structure [6] $CS = \{S_1, S_2, ..., S_m\}$ where S_i is the i^{th} sector in CS. A coalition structure's global utility is the sum of the utilities of the individual sectors in it:

$$U_{CS} = \sum_{i \epsilon CS} U_{S_i}$$

The coalition formation process decomposes a high-level task T assigned to the system into subtasks $\{t_1, t_2, ..., t_m\}$ which may overlap and are assigned to the different sectors. In the sensor network, two coverage subtasks overlap if $Area_{S_i} \cap Area_{S_j} \neq \emptyset$. Figure 1 shows that all three sectors illustrated have areas of overlap.

Each agent is able to perform a portion of the subtask assigned to its sector based on its capabilities. In the sensor network example, an agent is able to provide partial coverage of the region its sector is responsible for.

With the assumptions and definitions above, we can formulate the self-organization problem as follows: Given a high-level task T[1] and a set of agents A, subdivide T into m subtasks $\{t_1, t_2, ..., t_m\}$ and A into a coalition structure $CS = \{S_1, S_2, ..., S_m\}$ of m sectors such that each of the subtasks is assigned to one sector, where $\bigcup S_i = A$, $\forall i \neq j$ $S_i \cap S_j = \emptyset$, and U_{CS} is maximal.

2.3 Market Analogy Definitions

Because our approach involves an iterative negotiation process, comparing the system to a marketplace is useful. A **buyer** is a sector manager whose sector does not have the necessary sensors to perform its subtask adequately. In other words, the sensor agents that comprise the sector do not provide sufficient coverage of the area for which the sector is responsible. A **seller** is a sector manager whose sector has sensor agents able to provide coverage of a region the buyer would like to cover. Sector managers can be buyers and sellers simultaneously. The only agents involved in the negotiations of this marketplace are sector managers.

The **product** in the sensor network is the ability to provide coverage for a certain region and is transfered from one sector manager to another through the exchange of sensor agents between sectors. The product is the resource the buyer needs to improve its performance of its subtask. Finally, the **value** of a product to a buyer or seller is a function of the buyer's and the seller's marginal utility gains from the transaction and depends on the negotiation strategy they are using. When determining with whom to transact, buyers and sellers may consider either their own local marginal utility gains or the social marginal utility. The **local marginal utility** is the difference between a sector's utility before a transaction and the utility after the transaction. The **social marginal utility** is the sum of the local marginal utilities of both the buyer and the seller. In the local case, the buyer and seller value products differently. In the social case, they value products the same.[2]

With this analogy, the problem of bottom-up coalition formation translates into deciding which sellers the buyers

[1] In our case the high-level task is to provide coverage for the entire region.

[2] Although the analogy to a marketplace is useful, it is worth noting that our system is indeed cooperative and, therefore, agents may be willing to negotiate at the social level. This is not reasonable in a competitive market.

should attempt to buy from and which buyers the sellers should sell their products to such that U_{CS} is maximized.

3. NEGOTIATION STRATEGIES

A well-known strategy for assigning resources that has also been used to organize a DSN is the Contract Net Protocol (CNET) [11]. CNET provides a general framework to describe negotiation processes between agents. In its original version it involved agents' making decisions based on each agent's own perspective. For the DSN domain, an example of how CNET enables agents to build an organization is as follows: A task manager with a task to be fulfilled (such as finding a sensor agent to provide signal data) broadcasts a task announcement with a deadline for receiving bids. Just before the deadline, agents capable of performing the task send their bids to the manager who then evaluates the bids and awards contracts appropriately. Once an agent receives a contract, that agent is committed to it.

In our example, many agents may be able to provide coverage for the same area, but assigning the task to different agents may lead to different global utilities. In CNET each task that is assigned by a task manager (at its highest abstraction level)[3] was assumed to be independent of other tasks, so that the order of processing tasks by different task managers did not affect the global utility of the system.

We are interested in evaluating the performance of the whole organization in terms of the agents' decisions and the structure that results from these decisions. We assume that all agents are interested in maximizing the global utility of the system and, therefore, require a negotiation protocol to enable this. CNET in its original formulation is not sufficient for this purpose. For example, assume agents A_1 and A_2 are both able to cover a region that sector manager SM_1 needs covered, but only A_2 is able to cover a region SM_2 needs covered. If SM_1 awards a contract to A_2, A_2 may no longer be available to SM_2.

In order to correct for the above problem, we have developed two general classes of negotiation protocol for self-organizing through coalition formation in the marketplace of Section 2.3: local marginal utility based and social marginal utility based.[4] In our case, because agents negotiate, even if A_2 initially joins SM_1, SM_1 and SM_2 may be able to adjust the allocation of sensors to coalitions such that A_2 moves to SM_2 and A_1 joins SM_1.

For an illustration of the dynamics of the protocols we have developed, refer to Figure 2. In the local marginal utility based protocols, a round of negotiation proceeds as follows: A buyer broadcasts a message (1) requesting coverage of a region. Each manager within range, who has an agent that can cover that region and whose local marginal utility of giving up the agent is positive, responds with a message (2) stating that it could provide coverage to the buyer. Unlike CNET, the seller is not bound to honor this offer. The seller is free to make offers to as many buyers that send requests as it likes.

The buyer waits for a period of time, collecting responses from sellers. When the period is over, the buyer selects

[3]This task may be sub-contracted and its sub-parts are indeed dependent.

[4]The idea of negotiating over marginal utility is similar to the TRACONET [8] extension of CNET in which bidding and awarding decisions are based on marginal cost calculations.

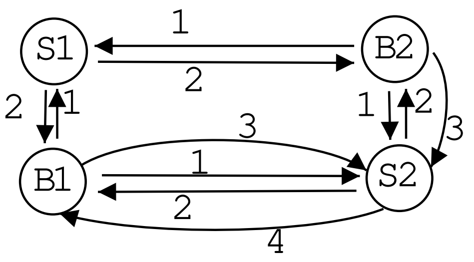

Figure 2: Negotiation Message Types

1. Buyers request. 2. Seller's potential utility change. 3. Buyer chooses seller. 4. Seller chooses buyer.

the seller whose product would provide the buyer with the greatest local marginal utility gain and sends a message (3) to that seller requesting the coverage offered.

Finally, given the multiple responses from buyers that the seller receives, the seller chooses to give its product to the buyer that maximizes the *seller's* local utility (4).

Negotiation in the social case is slightly different. As in the local case, the buyer sends its product request (1). This time, however, the seller responds with an offer even if its change in local utility would be negative and reports to the buyer what its local utility change would be (2). The buyer collects responses from sellers and chooses to request the product it needs from the seller that maximizes the sum of the buyer's local marginal utility and the seller's local marginal utility assuming the sum is positive. The buyer reports the sum (3) to the seller (i.e., the buyer requests the coverage offered).

Given the product requests the seller receives, it chooses to give the product to the buyer that reported the highest social marginal utility to it (4). Although the social marginal utility gain will be positive, the seller's or even the buyer's (but not both) local change in utility may be negative. In Figure 2 both Buyer 1 and Buyer 2 accept the offers Seller 2 made to them. Seller 2 then chooses (as seen by message (4) in the Figure) to give the product to Buyer 1 because the social marginal utility reported by Buyer 1 was higher than that reported by Buyer 2.

In addition to negotiation, we assume that a discovery process occurs when an agent enters the system; it must learn of the other agents near it and they must learn of it. To make this happen quickly, an entering agent joins the nearest sector to it, by listening for beacons on channel zero. If there is no sector within range, the agent elects itself manager of a new sector and begins attracting entering agents to it by broadcasting a periodic beacon on channel zero. It also starts negotiating with other managers over the resources it needs to perform its subtask.

We also assume that a maintenance process takes place throughout the life of the system. Sector managers must make sure that the members of their sectors still exist, and members must make sure that their managers still exist. In our approach each member of a sector periodically sends a brief message to its manager on the manager's channel. If the manager does not receive a message from a member, the manager assumes the agent is no longer a member of the coalition and adjusts its evaluation of the coalition accordingly. Likewise, the manager periodically sends a message to each of its members on the channel the member uses. If the member does not receive a message from its manager, that member assumes the manager is no longer active as a manager and joins the nearest coalition to it (as if it were entering the system for the first time).

4. RESULTS

To examine the above classes of protocols on large numbers of agents, we built an asynchronous simulation testbed for the EW Challenge [3] DSN. One limitation of the simulation is that it does not model delays due to computation time. In order to deal with this, it has the ability to add random delays to messages that are sent. In future work we plan to explore how increased delays affect overall system performance.

Note that while we would like to compare our results to optimal, the sizes of the configurations tested in this work are too large to generate optimal values.

4.1 Organization Results

To compare the performance of local and social utility based negotiation mechanisms, we varied factors such as when agents can initiate and respond to requests, whether sellers can initiate negotiations by advertising coverage, and how many agents a seller can transfer to a buyer during a negotiation. For each variation, we compared our results to those generated by CNET. In the CNET adapted for our domain, a buyer broadcasts a request. The seller collects requests and responds with an offer that the seller is obliged to fulfill if the buyer accepts it. This differs from our protocols. Since we are dealing with cooperative agents whose priority is the welfare of the system, a seller need not honor an offer. In other words, a CNET agent will respond only to a single request, while an agent that uses our protocols may respond to several requests simultaneously.

In total we tested fourteen protocols. We ran 100 experiments each on 40, 70, and 90 node configurations in 100 x 100 foot regions populated by agents with viewable sensor regions with 20 feet radii. For a given number of nodes, we generated an arbitrary configuration and then ran each of the 14 protocols on that configuration. By far the best performing protocols were those that were socially based.

Because of space limitations, we include results from six protocols with the following characteristics:

- **Single-Node Social Protocol (SNSoc):** Only single nodes are transferred per negotiation cycle. Sector managers are simultaneously buyers and sellers. Sellers advertise regions of coverage they are willing to give up. Value is based on social marginal utility.

- **Multiple-Node Social Protocol (MNSoc):** Same as above, but either one or two nodes may be transferred per negotiation cycles.

- **CNET Single Social Protocol (CNETSoc):** Socially based CNET with single node transfer.

- **Single-Node Local Protocol (SNLoc):** Same as SNSoc except that value is based on local marginal utility.

	Single		*Multiple*		*CNet*	
	local	social	local	social	local	social
40 nodes						
$\%\Delta U_{CS}$	6.5	61.2	6.3	60.6	2.1	15.6
Cycles	3	18	2	13	1	3
70 nodes						
$\%\Delta U_{CS}$	24.6	50.0	23.6	47.8	14.7	14.7
Cycles	7	22	5	18	2	2
90 nodes						
$\%\Delta U_{CS}$	44.2	70.9	42.7	67.6	36.7	39.7
Cycles	7	24	5	15	3	4.3

Table 1: 40, 70 and 90 Node Configurations

- **Multiple-Node Local Protocol (MNLoc):** Same as MNSoc except that value is based on local marginal utility.

- **CNET Single-Node Local Protocol (CNETLocal):** Same as CNETSoc, but locally based.

Table 1 summarizes the results for the protocols above for 40, 70, and 90 node configurations. They show the average percent change in global utility $\%\Delta U_{CS}$ from the initial state to a stable state and the approximate number of negotiation cycles required to reach the stable state. In this context the initial state is the set of rough sectors immediately after the discovery phase. The stable state occurs when agents are no longer able to engage in successful negotiations.

Of all the protocols SNSoc and MNSoc performed best. It makes sense that these would perform better than the locally based protocols because of their increased social context. We were surprised to find, however, that MNSoc achieved slightly lower global utility than SNSoc did since Sandholm [5] suggests that a contract over multiple objects can help avoid local maxima that result from single object contracts. One possible explanation is that transferring more than one agent in a single transfer causes the system to become stable more quickly. As a result, the system falls into local maxima more often than it does when only transfers of single agents are allowed. For example, if a sector manager gives up two nodes to another at time t, then the set of possible actions that the same sector manager can take at time $t + 1$ is reduced, and it may not be able to make a socially beneficial transfer that was unknown at time t. This conclusion is supported by the fact that the average number of negotiation cycles required to reach a stable state when MNSoc is used is less than the number when SNSoc is used. The conclusions in [5] consider a non-cooperative multi-agent system. In our cooperative organizations, the need for larger contracts is lessened by a more informative, social utility function.

While SNSoc does ultimately achieve higher global utility, it is interesting to note that early on in the self-organization process, MNSoc actually achieves higher utility. Figure 3 compares the average utility profiles of MNSoc and SNSoc for 90 node configurations. It gives the percentage of the final maximum utility achieved by SNSoc versus the average number of negotiation cycles. It shows that before the first 14 negotiation cycles, MNSoc does better. Only after this point does SNSoc pull ahead. The profile suggests a way of choosing a negotiation protocol and limiting how long the negotiation phase lasts. For instance, if the organization must form quickly due to high communication costs or other

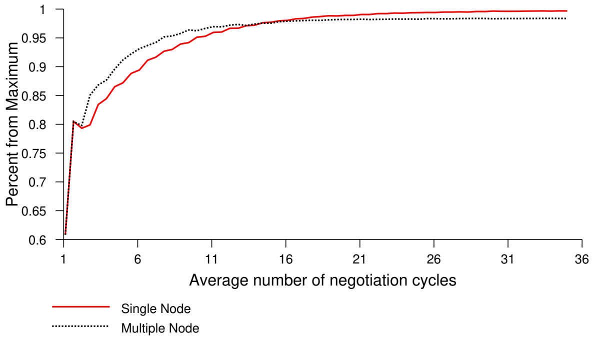

Figure 3: Utility profiles for SNSoc and MNSoc 90 node configurations.

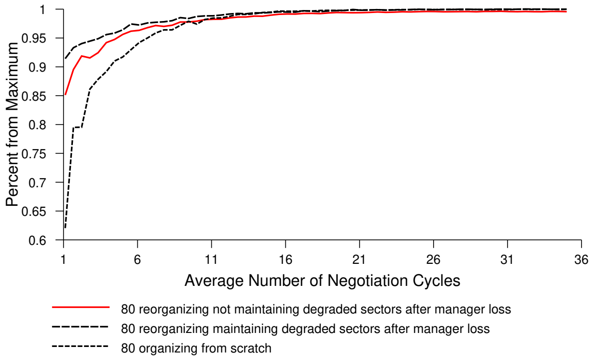

Figure 4: MNSoc utility profiles for 80 nodes reorganizing two ways plus 80 nodes organizing from scratch

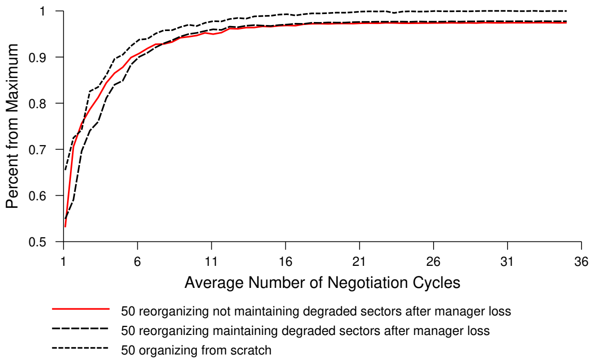

Figure 5: MNSoc utility profiles for 50 nodes reorganizing two ways plus 50 nodes organizing from scratch

constraints and achieving only 95% of the maximum utility is acceptable, then it is better to use MNSoc since it will reach this level of utility more quickly.

Also, evident in the graph is that in both cases, most of the utility increase occurs early on in the self-organization phase. This is corroborated by other results not shown here that show that the number of transfers of agents from one sector to another is high early on and then falls off rapidly.

Also of interest is that the fewer nodes in a configuration there are, the greater the difference is between the socially based protocols and the locally based ones (see Table 1). One explanation is that when there are many nodes in a fixed space, it is easier for these nodes to partition themselves to cover a given region. Thus, individual negotiation decisions in a dense region do not have as great of an effect on the ultimate social utility of the configuration as they do in less dense regions. While the more informed decisions possible through the socially-based utility functions certainly produce large improvements in utility in dense regions, their greatest impact is seen in less dense regions.

4.2 Fault Tolerance and Message Traffic

A DSN must be able to reorganize itself after several of its nodes go down. An additional concern of ours was that the maintenance process as described in Section 3 would hinder reorganization since if a sector manager fails, the nodes in its sector simply join other sectors near them rather than try to maintain the degraded sector. Therefore, we implemented a second maintenance scheme whereby if a manager fails, another node in its sector takes over its role.

To examine how well both mechanisms respond to node-failure, we performed the following experiment for SNSoc and MNSoc. For 100 different configurations we let 90 nodes organize until they reached a stable organization. Then starting from that stable state, we removed 10 nodes at random and let the system reorganize once using the original maintenance mechanism, once using the new one. Finally, using the remaining 80 nodes as a starting configuration, we let those nodes organize from scratch. We repeated this process three more times, each time starting from the stable 90-node organization, removing the same set of nodes as in the previous experiment plus an additional 10.

In all cases MNSoc organized more quickly than SNSoc with only minor differences in final utility. Figure 4 shows the utility profiles for MNSoc when 10 nodes fail. The graph for SNSoc is similar. When only 10 nodes go down, maintaining degraded sectors enables the system to reorganize most efficiently. In other words, since losing only a small number of nodes does not perturb the structure of the stable organization much, it is best to work within that structure and make only minor adjustments after node failure.

The above does not hold as more nodes go down. When 20 nodes fail, the two mechanisms have almost identical profiles. When 30 and 40 nodes go down, SNSoc reorganizes most quickly with the original maintenance mechanism, followed closely by organizing from scratch. With MNSoc organizing from scratch actually does better than reorganizing with either maintenance mechanism; in fact it achieves better utility than the other two options (see Figure 5). When large numbers fail, trying to maintain a previous structure hinders reorganizing and prevents the system from finding a globally beneficial solution.

The final check of our self-organization procedure was to verify that it keeps inter-sector communication low. We built a simple message traffic model of the domain and tested it on the stable 70-node organizations. The model showed that on average approximately 85% of messages occur within sectors, indicating that our algorithm successfully divides the agents such that most messages indeed occur within sectors.

5. THEORETICAL MODEL

In addition to experimentation, we developed a theoretical model of the negotiation process that builds on the idea that increased social context can improve system performance.

Because our system is cooperative, we assume that agents can share information, although the process of obtaining this information may be costly. In a DSN, decisions about which agent covers which area are affected by the interdependencies that exist between the agents.

DEFINITION 1 (INTERDEPENDENCY). *Given that sector managers SM_i and SM_j are responsible for tracking tasks in sectors S_i and S_j, we define an interdependency between these two sector managers if* $Area_{S_i} \bigcap Area_{S_j} \neq \emptyset$

A chain of interdependencies *is given by an ordered list of sector managers* $SM_{i_1}, SM_{i_2}, \ldots, SM_{i_{n-1}}, SM_{i_n}$ *such that* $Area_{S_{i_1}} \bigcap Area_{S_{i_2}} \neq \emptyset \ \ldots \ Area_{S_{i_{n-1}}} \bigcap Area_{S_{i_n}} \neq \emptyset$. *We denote by n the maximal length of a possible chain of interdependencies in a given system.*[5]

We distinguish agents based on their information horizon, the amount of information they can gather given by the length of an interdependencies' chain. We define an agent that knows the information in a chain of k interdependencies as follows:

DEFINITION 2 (K-SOCIAL, $k \leq n$). *Agent A is* **k-social** *if its decision about the action it will perform is based on information known by each agent in a chain of interdependencies of length k whose first element is A.*[6]

To explain the process in which an agent (a buyer or a seller) must decide which offer to accept or to whom it should sell a resource, we refer the reader to Figure 6.

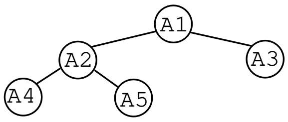

Figure 6: A Decision Tree example for $n = 3$.

We first build a tree describing the chain of interdependencies[7]. Each node represents a sector manager, and each edge represents an interdependency. In the figure, the numbers of each node represent the sector managers' names. The depth of the tree is the length of the interdependencies chain. Assume that $n = 3$. The root is at level $k = 0$, and the agent at the root is the one making a decision.

In Figure 6, agent $A1$ needs either to decide to whom to sell a resource that $A1$ currently owns, or it must decide from whom to buy a resource it needs. We denote by Δ_{ij} the change in i's utility caused by agent j's selling to or buying from agent i. If $A1$ is local ($k = 0$) it does not consider the other agents, and by comparing Δ_{12} to Δ_{13} chooses to interact with either $A2$ or $A3$ based which would produce the greater change in its own utility. This is analogous to the locally based protocols described in Sections 3 and 4.

If $A1$ is $k = 1social$, it compares $\Delta_{12} + \Delta_{21}$ to $\Delta_{13} + \Delta_{31}$. In this case, $A2$ and $A3$ may be interacting simultaneously with other agents, but since $A1$ sees only up to horizon 1, it is unaware of any pending decisions further down the interdependency chain. So $A1$'s decision may be wrong. For example, if $A1$ accepts the transaction with $A2$ based on the above comparison, it may lose, if $A2$ chooses $A4$ instead of $A1$. We have not implemented a protocol in which the agents are $k = 1social$. If we had, it would proceed as follows: 1) A buyer broadcasts a request for coverage. 2) A seller responds with an offer. 3) The buyer does not wait to collect responses from sellers; it simply responds to the seller, telling it what the buyer's local change in utility would be if it were to receive the seller's offer. 4) The seller ranks all responses it receives based on the sum of its local marginal utility and the reported buyers' utilities. The seller chooses the buyer that gives the highest sum.

If $A1$ is $k = 2social$, it knows whether $A2$ is negotiating with $A4$ and $A5$ while $A2$ is negotiating with $A1$. Here $A1$ decides by computing $max\{\Delta_{21} + \Delta_{12}, max\{\Delta_{24} + \Delta_{42}, \Delta_{25} + \Delta_{52}\}\}$.

If the "winner" of this maximum is $\Delta_{21} + \Delta_{12}$, sector manager $A2$ will **not** transact with $A4$ or $A5$. Therefore, sector manager $A1$ to make his decision compares $\Delta_{21} + \Delta_{12}$ to $\Delta_{13} + \Delta_{31}$ and chooses $A2$ or $A3$ accordingly. If the "winner" of the above maximum is $\Delta_{24} + \Delta_{42}$ or $\Delta_{25} + \Delta_{52}$, then the value for $max\{\Delta_{21} + \Delta_{12}, max\{\Delta_{24} + \Delta_{42}, \Delta_{25} + \Delta_{52}\}\}$ is set to $-\infty$, so that $A1$ does not take $A2$ as an option in its decision because from its perspective $A2$ will accept the transaction with either $A4$ or $A5$ and not with $A1$; in this case $A1$ chooses $A3$.

The $k = 2social$ case is analogous to the social marginal utility protocols we have developed. In those protocols, the seller at the root does not actually do all of the calculations described above. Rather, parts of the calculation are done further down the tree and propagated up. For example, if $A1$ in Figure 6 is a seller, $A2$ is a buyer and calculates $max\{\Delta_{21} + \Delta_{12}, max\{\Delta_{24} + \Delta_{42}, \Delta_{25} + \Delta_{52}\}\}$. If the "winner" is $\Delta_{21} + \Delta_{12}$, $A2$ propagates this value up to $A1$. Otherwise, it does not, and $A1$ knows only to consider negotiating with $A3$.

Notice that Sandholm and Lesser [7] assume self-interested agents, which are necessarily $k = 0social$[8]. In their case, agents must transact on complex deals in order to approximate the maximal utility. We take advantage of the cooperativeness of the system by allowing the agents to be social and, thus, obtain better deals in terms of the complete system without the need to transact over more complex deals.

Results obtained from our simulations show that social agents attain higher utilities than local agents and that in configurations with few agents, the difference between the organizational utility obtained by social agents and that ob-

[5] In general, an interdependency exists if the set of resources needed by two agents making a decision intersect.

[6] In the setting analyzed so far, the agent referred to in this definition corresponds to the sector manager.

[7] The interdependencies may actually be represented by a graph, because there may be cycles of interdependencies among the resources. So this tree is a mapping from this graph to a tree, i.e., whenever a node that has already appeared in the tree needs to appear again it is set to a leaf.

[8] In this paper, we are assuming that communication between the cooperative agents is free. It is not reasonable to assume that self-interested agents will exchange this information for free, although they may benefit from it.

tained by local agents is greater than it is in configurations with many agents. We can understand these results from the theoretical model. We denote by $P(s_i)$ the probability of sector i's manager's making the "correct" decision. If sector i's manager chooses to transact with agent j and j also chooses to transact with agent i, then we say that agent i has made the right choice. Therefore, $P(s_i)$ is a conditional probability that the root of the interdependency tree has made the right decision. This probability is conditioned on the decisions of the other nodes in the tree. The more social the sector manager is, the more accurate $P(s_i)$ is. If the sector manager knew all the information in the complete tree then $P(s_i)$ would be 1 (in the decentralized version of the problem[9]). Hence, the result we obtained that being social is better than being local is supported by the theoretical model. The utility of the organization is higher as long as the value of $P(s_i)$ increases, and this value increases as long as the sector manager considers larger values of k (i.e, the agents are more social).

The result that being social in sparsely populated configurations has a greater effect than being social in a dense configuration is also supported by the model. In such a setting, because there are fewer interdependencies, $P(s_i)$ will be greater than in a dense configuration with many interdependencies. Therefore, the difference in utility will be greater as well.

6. CONCLUSIONS AND FUTURE WORK

This paper presents an incremental approach to self-organization based on bottom-up coalition formation. Agents negotiate to maximize the system's global utility by using a variety of protocols based on local or social marginal utility.

Our approach is novel in the sense that it allows for different levels of social agents to be tested. Our protocols can represent a continuum of agents from locally-oriented to fully-informed. Empirical results show that social agents do attain higher utilities than locally-based or CNET-based agents do. In other words, although the system achieves a stable organization in all the cases tested, negotiating with social awareness in an incremental fashion avoids many of the local maxima of non-social utility based negotiations. We also show that the organizations obtained are robust to agent failure; the agents do indeed reorganize after some number of them are deactivated, and as long as the number of nodes that fail is not so great as to obliterate the structure already in place, reorganizing to a stable state happens more quickly than simply organizing from scratch and achieves similar utility values.

Future work will look at other types of organization templates. We will study more complex topologies, such as organizations based on hierarchies with multiple levels which may require various communication models for the exchange of information at the different levels. Another cost model worth studying involves the computation of the global utility resulting from agents' negotiating based on combinations of local and social marginal utilities. Adding explicitly the cost of sending a message (e.g., given by delays) and analyzing the trade-off faced by agents between obtaining more accurate information and the time it may take to gather it deserves more research as well. In this work, we have assumed that all the agents are homogeneous. Further work will look at systems where the sector managers require certain computational capabilities that only some of the agents have.

[9]The optimal centralized organization may still be different from the optimal decentralized version.

7. REFERENCES

[1] C. V. Goldman and J. S. Rosenschein. Partitioned multiagent systems in information oriented domains. In O. Etzioni, J. P. Müller, and J. M. Bradshaw, editors, *Proceedings of the Third International Conference on Autonomous Agents (Agents'99)*, pages 32–39, Seattle, WA, USA, 1999. ACM Press.

[2] B. Horling, B. Benyo, and V. Lesser. Using self-diagnosis to adapt organizational structures. In J. P. Müller, E. Andre, S. Sen, and C. Frasson, editors, *Proceedings of the Fifth International Conference on Autonomous Agents*, pages 529–536, Montreal, Canada, 2001. ACM Press.

[3] B. Horling, R. Vincent, R. Mailler, J. Shen, R. Becker, K. Rawlins, and V. Lesser. Distributed sensor network for real time tracking. In *Proceedings of the 5th International Conference on Autonomous Agents*, pages 417–424, Montreal, June 2001. ACM Press.

[4] T. Ishida, M. Yokoo, and L. Gasser. An organizational approach to adaptive production systems. In *National Conference on Artificial Intelligence*, pages 52–58, 1990.

[5] T. Sandholm. Contract types for satisficing task allocation: I theoretical results. In *AAAI 1998 Spring Symposium: Satisficing Models, 1998.*, pages 68–75, Standford University, CA, March 1998.

[6] T. Sandholm, K. Larson, M. Andersson, O. Shehory, and F. Tohme. Coalition structure generation with worst case guarantees. *Artificial Intelligence*, 111(1-2):209–238, 1999.

[7] T. Sandholm and V. Lesser. Issues in automated negotiation and electronic commerce: Extending the contract net framework. In V. Lesser, editor, *Proceedings of the First International Conference on Multi-Agent Systems (ICMAS'95)*, pages 328–335, San Francisco, CA, USA, 1995. The MIT Press: Cambridge, MA, USA.

[8] T. W. Sandholm. An implementation of the contract net protocol based on marginal cost calculations. In *Proceedings of the 12th International Workshop on Distributed Artificial Intelligence*, pages 295–308, Hidden Valley, Pennsylvania, 1993.

[9] O. Shehory and S. Kraus. Task allocation via coalition formation among autonomous agents. In *Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence (IJCAI-95)*, pages 655–661, Montréal, Québec, Canada, 1995.

[10] O. Shehory and S. Kraus. Methods for task allocation via agent coalition formation. *Artificial Intelligence*, 101(1-2):165–200, 1998.

[11] R. G. Smith. The contract net protocol: High-level communication and control in a distributed problem solver. *IEEE Transaction on Computers*, C-29(12):1104–1113, 1980.