# Self-Organization through Bottom-up Coalition Formation[*]

Mark Sims, Glaudia V. Goldman, and Victor Lesser
Computer Science Department
University of Massachusetts
msims@cs.umass.edu

UMass Computer Science Technical Report 02-47

December 2, 2002

**Abstract**

This paper presents a distributed, incremental approach to self-organization through bottom-up coalition formation applied to a distributed sensor network. The agents engage in negotiations that lead to more efficient allocations of resources and better performance when tasks are assigned to the organization as a whole.

We take advantage of the cooperativeness of the system by allowing the agents to share information before they allocate resources. We have tested a range of protocols that agents can implement in order to make decisions that will affect the coalition formation. On one extreme these protocols are based on local utility computations, where each agent negotiates based on its own local perspective. From there, due to the cooperative character of the system, there is a continuum of additional protocols that can be studied. These protocols are based on marginal social utility, where each agent bases its decisions on the combination of its marginal utility and that of others. We present a formal framework that allows us to quantify how social an agent can be and how the choice of a certain level affects the decisions made by the agents and the global utility of the organization obtained.

[*]A version of this paper is under review for The Second International Joint Conference on Autonomous Agents and Multi Agent Systems (AAMAS2003 Melbourne, Australia).

Our results show that by implementing more social agents, we obtain an organization with a high global utility even though the agents do not negotiate over complex contracts. Our algorithm is incremental and therefore the organization that evolves can adapt and stabilize even if some of the agents become inactive during task execution.

## 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 thus significantly improving system performance. We have observed improved system performance with an organization consisting of as few as sixteen agents [3]. Re-organization is necessary during system operation when agents and resources are removed or added, or when their performance characteristics are changed. 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 of the EW Challenge Problem [3]. The process uses negotiation to enable managers of coalitions to refine the set of coalitions in the system to achieve efficient allocations of sensors iteratively and adapt dynamically to environmental changes.

Horling et al. [3] give a detailed description of the EW Challenge Problem domain. In brief, it consists of a distributed network of homogeneous sensor agents distributed throughout a region. The sensor agents are fixed and communicate using an eight-channel RF system in which an agent cannot use more than one channel simultaneously. An organization in such a domain is important to facilitate the efficient assignment of tracking tasks to particular agents and to limit contention on communication channels. We employ a one-level hierarchy in which sensor 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 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

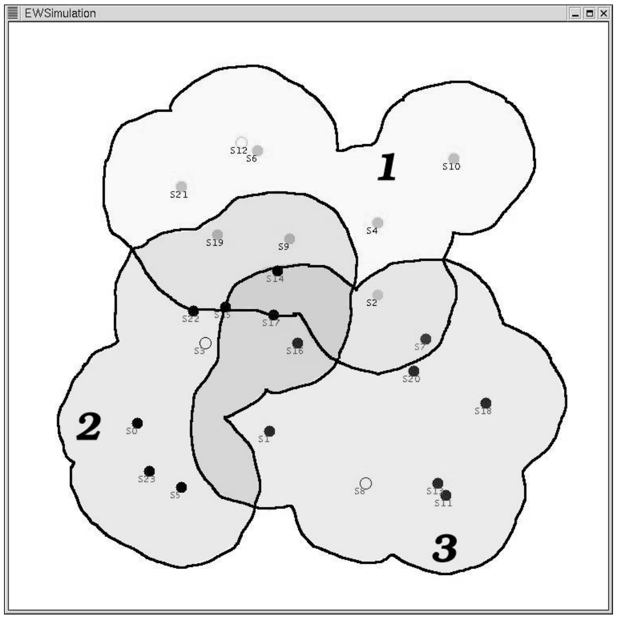

Figure 1: EWChallenge Domain

zero and that the sector managers assign channels other than zero to agents as they enter the sector.

Figure 1 illustrates the domain. The "clouds" represent sectors. The empty circles, such as S12, are sector managers and the filled-in circles are sensor agents that are not managers. The areas of overlap show where the regions covered by one sector intersect with those covered by another. Note that the agents located in an overlap belong only to one sector although they can request sensor data from an agent in another sector. In the figure, S7 and S16 both belong to Sector 3. In order best to track a vehicle, at least three sensor agents are required to triangulate the position of a vehicle moving through the region. In addition, although each agent has three sensor heads and can sense with only one head at a time, for the purposes of this paper we assume agents have a viewable area of 360°.

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 outside of the sector responsible for tracking a vehicle 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 agents in 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 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.

Coalition formation is the process whereby agents in a large system faced with a set of tasks partition themselves such that system performance is greatest. In effect, through coalition formation, 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 [8, 9] in that it applies to a cooperative system of agents in an environment that is not super-additive. We assume that there is an overhead associated with each new member of a coalition and that a coalition reaches a point beyond which it is no longer beneficial to add a member. Beyond these common assumptions our approach varies considerably from that of [8, 9]. In their work agents have a more global view of other agents in the system. The agents may not be aware of every other agent, but the assumption is that the agents know of a large number of other agents. Given this knowledge, each agent calculates a subset of the possible coalitions it may belong to. The system then 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 [9] allow for overlapping coalitions. In our current work, we restrict our attention to disjoint coalitions although for future work, we plan to extend our techniques to overlapping coalitions where sensor agents that can see into multiple sectors 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, their process 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 current load on the system. Unlike our approach, neither is an iterative search process designed to converge on a good organization. The adaptations may be revised as the situation changes, but the process of adapting is a single shot.

Finally, the classic coalition formation problem can be formulated as follows: Given a fixed set of tasks and a set of cooperative agents, how can we best pick from those agents the groups most well suited to handle those tasks? Our problem is slightly different and resembles the work in Goldman and Rosenschein [1] aimed at partitioning information domains in order to facilitate future information retrieval requests. In our distributed sensor network, 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 need to 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. In Section 4, we also analyze the performance of the organizations evolved in terms of the number of nodes used to track and the message traffic that sectors incur when actual tracking tasks are assigned to the sensors. 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 ksocial 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.

- A good 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.

## 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 $B = \{b_1^i, b_2^i, ..., b_n^i\}$. 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 organization templates as defined below:

**Definition 1 (An Organization Template)** — *An organization template is given by 1 ) the topology of the organization, 2) the communication model, 3) the cost model and 4) a vector of capabilities that are required to be a sector manager.*

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 is chosen by the other agents and 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 has an area defined by the viewable areas of the sensor agents that it is responsible for as shown in the cloud 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. For instance, a sector with low utility may have eight agents, but they are so spread out that it is impossible to triangulate the position of any vehicles traveling though it.

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 $i^{\text{th}}$ sector in $CS$. A coalition structure's global utility is the sum of the utilities of the individual sectors in it:

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

When a high-level task $T$ is assigned to the system, the coalition formation process decomposes $T$ 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 the 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 $\bigcup S_i = A$ and $\forall i \neq j, S_i \cap S_j = \emptyset$ and $U_{CS}$ is maximal.

## 2.3 Market Analogy Definitions

Because our approach involves an iterative negotiation process, comparing the system of agents to a marketplace is useful. A **buyer** is a sector manager whose sector does not have the necessary sensor resources 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. Coalition managers can be buyers and sellers simultaneously. Also, note that the only agents involved in the negotiations of this marketplace are the sector managers, not every agent.

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

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

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 gain 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 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 distributed sensor network is the Contract Net Protocol (CNET) [10]. The CNET provides a general framework to describe negotiation processes between agents. The CNET in its original version involved agents' making decisions based on each agent's own perspective. For the distributed sensor network domain, an example of how the 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. Note that this protocol does not take into account the effect of the *quality* of the contract on the global utility of the system.

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

[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.

global utilities. In the 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. In fact, no utility measure was considered in the CNET, only whether a task could be performed by some deadline.

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. The 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. 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 in the 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 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).

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

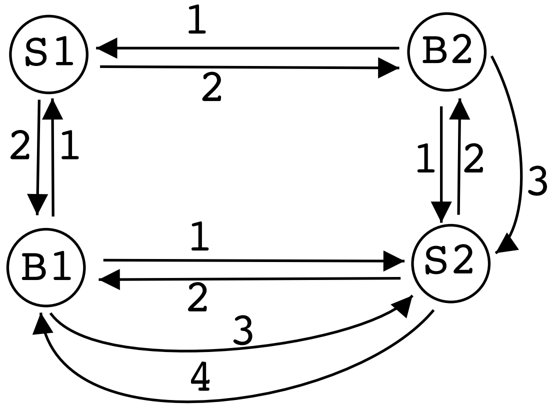

Figure 2: Social Negotiation Message Types

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

Negotiation in the social case is similar. 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 occurs 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 classes of protocols described above on large numbers of agents, we built an asynchronous simulator that enables us to use the EW Challenge sensor network domain as a testbed. The sizes of the configurations tested in this work are too large to generate optimal values. The simulator employs simple agents instantiated as objects in a single process. The agents become active at randomly different times, and what stage of a negotiation they are in is dependent only on when they become active, not on a schedule imposed by the simulator. One limitation of the simulation is that it does not model delays due to computation time. In order to deal with this, the simulator 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.

### 4.1 Organization Results

We compared the performance of local utility based negotiation mechanisms and socially based ones. In both classes of protocol, we varied factors such as when agents can initiate and respond to negotiation requests, whether or not sellers can initiate negotiations by advertising available coverage, and how many agents a seller can transfer to a buyer during any given negotiation cycle. For each variation, we also compared our results to those generated by the CNET. In the CNET adapted for our domain, a buyer sends out a message requesting a product. The seller collects requests for a time and responds with an offer that the seller is obliged to fulfill if the buyer accepts it. This is different from the protocols we have developed and are comparing to the CNET. Since we are dealing with cooperative agents whose main priority is the welfare of the system, in our protocols we do not include the restriction that a seller must honor an offer. Although agents using both locally based protocols and CNET make decisions only on local information, the agents following CNET will only respond to a single request, while agents that use our locally based protocols may respond to several requests at the same time.

In total we tested fourteen protocols. We ran 100 experiments each on 40, 70, and 90 node configurations in 100 x 100 foot regions where agents were arbitrarily distributed and had viewable sensor regions with radii of 20 feet. 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 the following 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-Node 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.

- **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.

Out of the fourteen protocols Single-Node Social Protocol and Multiple-Node Social Protocol performed best. It makes sense that SNSoc and MNSoc would perform better than the locally based protocols because of their increased social context. It is difficult to understand, however, why MNSoc, in which complex transfers of agents are allowed, does not perform better than SNSoc. We were surprised by this result 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 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.

Tables 1, 2, and 3 summarize the results for the protocols above for 40, 70, and 90 node configurations. They show the average change in global utility from the initial state to a stable state and the approximate number of negotiation cycles required to reach the stable state. From the experimental data obtained, the fewer nodes in a configuration there are, the greater the difference is between the socially based utility and the locally based utility. 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.

|  | *Single* |  | *Multiple* |  | *CNet* |  |
|---|---|---|---|---|---|---|
|  | local | social | local | social | local | social |
| $\Delta U_{CS}$ | 6.5% | 61.2% | 6.3% | 60.6% | 2.1 | 15.6% |
| Cycles | 3 | 18 | 2 | 13 | 1 | 3 |

Table 1: 40 Node Configuration

|  | *Single* |  | *Multiple* |  | *CNet* |  |
|---|---|---|---|---|---|---|
|  | local | social | local | social | local | social |
| $\Delta U_{CS}$ | 24.6% | 50.0% | 23.6% | 47.8% | 14.7 | 14.7% |
| Cycles | 7 | 22 | 5 | 18 | 2 | 2 |

Table 2: 70 Node Configuration

## 4.2 Message Traffic and Fault Tolerance

So far we have discussed the system's organizational performance in terms of global utility only. In an environment such as the sensor network domain, the real test is how well the system is able to track the vehicles that enter its region. Furthermore, it is very important that the system be able to reorganize itself and continue to track well after several of its nodes go down. Also, as we mentioned in the Introduction, we want to keep to a minimum the number of messages used for tracking that are sent between sectors.

To examine the system's ability to track and message traffic while tracking, we built a mathematical model of the system that for a given organization of sectors generates tracks through the environment and calculates

|  | *Single* |  | *Multiple* |  | *CNet* |  |
|---|---|---|---|---|---|---|
|  | local | social | local | social | local | social |
| $\Delta U_{CS}$ | 44.2% | 70.9% | 42.7% | 67.6% | 36.7 | 39.7% |
| Cycles | 7 | 24 | 5 | 15 | 3 | 4.3 |

Table 3: 90 Node Configuration

|  | 70 Nodes | 90 Nodes |
|---|---|---|
| Avg. $\Delta U_{CS}$ after deactivation | -58.4% | -58.04% |
| Avg. $\Delta U_{CS}$ after reorganization | 69.0% | 67.9% |

Table 4: Percent utility changes due to 25% node deactivation

the number of messages within a sector used for tracking and the number of messages between sectors used for tracking. Because the quality of tracking increases as the number of nodes used to track increases, the model also calculates the average number of nodes used to track vehicles.

For Single-Node Social Protocol described in Section 4.1 we examined the system's ability to reorganize itself after 25% of the nodes were deactivated. We ran 100 experiments each for 70 and 90 node configurations, letting the system reach a stable state and then deactivating the nodes.

Table 4 summarizes the changes in utility. In each case the system utility decreased by approximately 58% after deactivation. The system then reorganized itself to a new stable state with a system utility within approximately 30% of the original. Put another way, killing off nodes caused a sharp decrease in utility. The system then reorganized itself increasing the global utility from the diminished value by approximately 68%.

Tables 5 and 6 summarize how deactivating nodes affected the messages used for tracking. For the 70-node configuration before deactivation, approximately 186 messages within sectors were used versus 31 between sectors. After deactivation 141 messages were within sectors and 25 were between. Then, after reorganization, 170 messages were within sectors and 31 were between. For the 90-node configuration, the results were similar. These results show that the incremental self organization algorithm successfully divides the agents into coalitions such that most of the messages occur inside a coalition and fewer messages are needed across coalitions.

Table 7 shows how many nodes were used to track before, after deactivation and after achieving again a stable organization.

As a final test, we compared the organization achieved by a 70-node configuration to that achieved when a stable 90-node configuration has 20 of its nodes deactivated and then reorganizes. We ran 100 experiments on 90-node configurations, letting the system reach a stable state each time and then deactivating 20 nodes in order to have the system reorganize itself. We

| | Within Sector | Between Sectors |
|---|---|---|
| Before deactivation | 186 mssgs | 31 mssgs |
| After deactivation | 141 mssgs | 25 mssgs |
| After reorganization | 170 mssgs | 31 mssgs |

Table 5: Message traffic before and after deactivation for 70 node configuration

| | Within Sector | Between Sectors |
|---|---|---|
| Before deactivation | 193 mssgs | 30 mssgs |
| After deactivation | 156 mssgs | 29 mssgs |
| After reorganization | 179 mssgs | 31 mssgs |

Table 6: Message traffic before and after deactivation for 90 node configuration

then used each set of 70 nodes as a starting configuration and let the system organize itself from scratch.

The utilities achieved in the two cases differed by only 0.06% on average. The number of nodes used to track and the number of messages between sectors and within sectors were also practically the same. The amount of time it took for the systems to organize themselves, however, did differ considerably. After the 20 nodes were deactivated, the system needed an average of 18 negotiation cycles to reorganize itself while the system that used the 70 nodes as the initial configuration required 22 cycles.

| | 70 nodes | 90 nodes |
|---|---|---|
| Before deactivation | 48 | 50 |
| After deactivation | 33 | 39 |
| After reorganization | 42 | 45 |

Table 7: Number of nodes used to track before and after deactivation.

## 5 Theoretical Model

In addition to experimental work to assess the protocols we have developed, we have built a theoretical model of the negotiation process that builds on the idea that increased social context can improve system performance.

Due to the cooperative nature of our system, we may assume that agents can share information, although the process of obtaining this information may be costly or time consuming when this information is distributed (e.g., geographically). In a distributed sensor system, decisions about which agent covers which area are affected by the interdependencies that exist between the agents.

**Definition 2 (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 their corresponding sectors follow*

$$Area_{S_{i_1}} \bigcap Area_{S_{i_2}} \neq \emptyset \ \ldots \ Area_{S_{i_{n-1}}} \bigcap Area_{S_{i_n}} \neq \emptyset$$

*We denote by $n$ the maximal length of a possible chain of interdependencies in a given system.*[4]

Then, agents can be distinguished based on the information horizon they can *see*, that is the information agents can gather up to some horizon that is 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 3 (An Agent is ksocial, $k \leq n$)** *An agent $A$ is* **ksocial** *if its decision about the action it will perform is based on the information known by all the agents in the chain of interdependencies of length $k$ and whose first element is the agent $A$.*

To explain the decision process in which an agent (a buyer or a seller) is involved when it is facing a decision as to which offer to accept, or to whom it should sell a resource, we refer the reader to Figure 3.

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

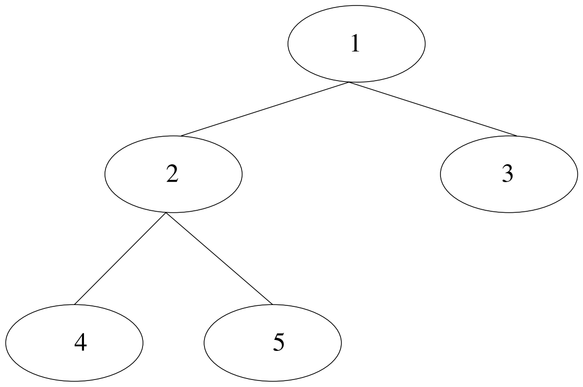

Figure 3: A Decision Tree example for $n = 3$.

We first build a tree describing the chain of interdependencies[5] assuming that each node in this tree represents a sector manager, and each edge represents an interdependency. In the simple example shown in the figure assume that the numbers at each node in the tree stand for the names of the sector managers. The depth of the tree is the length of the interdependencies chain. Assume in this simple example that $n = 3$. The root is at level $k = 0$. The agent at the root (i.e., agent 1) is the agent that is going to make a decision.

In Figure 3, the root 1 needs either to decide to whom to sell a resource that 1 currently owns, or it has to decide from whom it will buy a resource it needs. In order for 1 to make this decision, we denote by $\Delta_{ij}$ the change in $i$'s utility caused by agent $j$'s selling to or buying from agent $i$. If 1 is local ($k = 0$) it does not care at all about the others and it compares $\Delta_{12}$ and $\Delta_{13}$ and chooses to interact with either 2 or 3 based on the larger change in its own utility caused by each one of these interactions. This is analogous to the protocols described above in which sector managers value transactions based on their own local marginal utility.

If 1 is $k = 1$*social* it compares $\Delta_{12} + \Delta_{21}$ to $\Delta_{13} + \Delta_{31}$. In this case, 2 and 3 may be interacting *at the same time* with other agents, but 1 sees only up to horizon 1 so it cannot know what other decisions will be taken beyond the horizon of 1. So 1's decision may be wrong because if it accepts

[5]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 i that already appeared in the tree needs to appear again this node is set to be a leaf.

the transaction with 2 based on its comparison of $\Delta_{12} + \Delta_{21}$ to $\Delta_{13} + \Delta_{31}$, it may lose if 2 does not accept 1 and chooses 4, for example. 1 could not have known about this and therefore chooses mistakenly 2 instead of 3 who would not have rejected it. We have not implemented a protocol in which the agents are $k = 1 social$. If we had, it would proceed as follows: 1) A buyer broadcasts a request for coverage. 2) A seller responds with an offer (if it has an agent that can provide the coverage). 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 buyer's utility. The seller chooses the buyer that gives the highest sum.

If 1 is $k = 2social$ then it also *knows* whether 2 is negotiating with 4 and 5 at the same time 2 is negotiating with 1. Now agent 1 decides based on the following reasoning: It computes

$$max\{\Delta_{21} + \Delta_{12}, max\{\Delta_{24} + \Delta_{42}, \Delta_{25} + \Delta_{52}\}\}$$

If the "winner" of this maximum is $\Delta_{21} + \Delta_{12}$, it means that sector manager 2 will **not** transact with 4 or 5. Therefore, sector manager 1 to make his decision compares $\Delta_{21} + \Delta_{12}$ and $\Delta_{13} + \Delta_{31}$ and chooses 2 or 3 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 1 does not take 2 as an option in its decision because from its perspective 2 will accept the transaction either with 4 or 5 but will not accept the transaction with 1, so 1 considers its other options; in this case it chooses 3. The change in the system's utility because of 1's decision will be, following this example, either $\Delta_{12} + \Delta_{21}$ or $\Delta_{13} + \Delta_{31}$ based on the winner in 1's decision.

The above $k = 2social$ case is analogous to the social marginal utility protocols we have developed, although in our experiments $n >> 3$. In those protocols, the seller at the root of the tree does not actually do all of the calculations described above, however. Rather, parts of the calculation are done further down in the tree and propagated up. For example, if the root 1 in Figure 3 is a seller, then 2 is a buyer and calculates $max\{\Delta_{21} + \Delta_{12}, max\{\Delta_{24} + \Delta_{42}, \Delta_{25} + \Delta_{52}\}\}$ and then if the "winner" is $\Delta_{21} + \Delta_{12}$, 2 propagates this value up to 1. Otherwise, it does not and 1 knows only to consider negotiating with 3.

Notice that Sandholm and Lesser [7] assume that the agents were self-interested, and therefore the agents are necessarily $k = 0social$. In their case, agents must transact on more complex deals in order to approximate the maximal utility (i.e., large-transfer contracts, multi-agent contracts and swaps). The information about the other agents in the negotiation processes (that is the information in the nodes at some depth $k > 0$) is not available to self-interested agents. We take advantage of the cooperativeness of the system by allowing the agents to be more social (i.e., we allow the sector managers to base their decisions on k values larger than 0) and thus obtain better deals in terms of the complete system without the need to transact over more complex deals.

## 6 Conclusions and Future Work

This paper presents an incremental approach to self-organization based on bottom-up coalition formation. Agents negotiate during the process of self-organization in order to maximize the global utility of the system. The algorithm proposed in this work approximates the optimal value. This is achieved by the negotiation process the agents are involved in, where in a distributed manner they try to maximize their local or social utility. The decisions made by the agents at each negotiation step affect the structure and the performance of the resulting organization.

Our approach is novel in the sense that it allows for different levels of social agents to be tested. That is, our protocols can represent a complete continuum of agents: from locally-oriented (also applicable for selfish agents) to fully-informed agents (when each agent knows exactly the decisions faced by each one of the other agents). Empirical results show that social agents do attain higher utilities than locally-based or CNET-based agents do. That is, although stable organizations are achieved in all the cases tested, negotiating with social awareness in an incremental fashion avoids many of the local maxima yielded in non-social utility based cases. We also show that the organizations obtained are robust to agent failure, i.e., the agents do indeed reorganize after some percentage of them are deactivated. Furthermore, the reorganized system is as good as the one that would have been achieved if the organizational process had restarted completely.

Future work will look at other types of organization templates. At the topology level, we will study more complex topologies, for example organizations based on hierarchies with multiple levels. These topologies may require various communication models for the exchange of information at the different levels. Other cost models that are worth studying include the computation of the global utility resulting from agents' negotiating based on combinations of local utilities 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. 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.

## 7 Acknowledgments

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 and by the National Science Foundation under Grant number DMI0122173. 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 or the U.S. Government.

## 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, 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] 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.

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

[10] 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.