Real-Time Satisficing Multiagent Coalition Formation

Leen-Kiat Soh
Computer Science & Engineering Department
University of Nebraska
115 Ferguson Hall, Lincoln, NE 68588-0115
Tel: 402-472-6738
lksoh@cse.unl.edu

Costas Tsatsoulis
Dept. of Electrical Engineering & Computer Science
Information & Telecommunication Technology Center
University of Kansas
2335 Irving Hill Road, Lawrence, KS 66045
Tel: 785-864-7749
tsatsoul@ittc.ku.edu

Abstract

In a multiagent system where each agent has only an incomplete view of the world, optimal coalition formation is difficult. Coupling that with real-time and resource constraints often makes the rationalization process infeasible or costly. We propose a coalition formation approach that identifies and builds sub-optimal yet satisficing coalitions among agents to solve a problem detected in the environment. All agents are peers and autonomous—each controlling a set of resources, monitoring a part of the world, and able to change the world through its actions. Each is motivated to conserve its own resources while cooperating with other agents to achieve a global task or resource allocation goal. The (initiating) agent—that detects a problem—*hastily* forms an initial coalition by selecting neighboring agents that it considers to have high potential utilities, based on the capability of each neighbor and its respective inter-agent relationships. The initiating agent next finalizes the coalition via multiple concurrent 1-to-1 negotiations with only neighbors of high potential utility, during which constraints and commitments are exchanged in an argumentation setting. Finally, the initiating agent acknowledges the status of a coalition, a responsible act that seals the validity of a planned coalition.

Introduction

In this paper, we present a coalition formation strategy that facilitates sub-optimal yet satisficing collaborations among agents. The need for such collaborations arises due to the *reactiveness* requirement of agents in a time-critical and resource-constrained environment. In a non-hierarchical multiagent system, agents are peers and autonomous. Each controls a set of resources, monitors its part of the world, and reacts to events that it detects. For events that require collaborations, an (initiating) agent forms a coalition out of its neighbor agents. However, each agent only has an incomplete (and sometimes outdated and noisy) view of the world and its neighbors—making rational optimality in its coalition formation process impossible. So, in a situation where an optimal solution does not exist, where an optimal solution cannot be computed, or where an optimal solution cannot be derived on time, we have to look to sub-optimal solutions that are satisficing in that they meet minimum requirements rather than achieving maximum performance (Moon and Stirling 2001).

Our proposed coalition formation approach breaks the process into three stages: initial coalition formation, coalition finalization, and coalition acknowledgment. The objective of the initial coalition formation is to *hastily* identify potential candidates and rank them accordingly to their potential utilities. However, since the agent maintains only a partial view of the world, it needs to determine with certainty whether any of the candidates is willing to help. Instead of approaching all neighbors to compute an optimal coalition, the agent only selectively negotiates with top-ranked candidates to finalize the coalition so as to conserve both computational resources and communication usage. Some candidates may refuse to participate, some may agree. When a coalition is formed, all remaining negotiations are terminated. When a coalition no longer can be formed (due to some negotiation failures), all remaining negotiations are terminated as well. This strategy is thus opportunistic and high-risk since the formation of a coalition cannot be guaranteed. To help alleviate this weakness, our agents are equipped with learning mechanisms (case-based learning and reinforcement learning) to learn to form better coalitions faster, and a constant monitoring module for risk mitigation. Note that with this modular approach, we refine the validity of our coalition gradually as time permits. After the initial coalition formation step, an agent already has a potentially working coalition. As the agent gains more information about its neighbors (through negotiation), it is able to eliminate or confirm candidates.

In this paper, we first discuss some research work in coalition formation in rational and reactive agents and systems. Then, we present our coalition formation model, including the utility-theoretic initial coalition formation and the case-based argumentative negotiation for the coalition finalization step. In this section, we also discuss the learning mechanisms briefly and the coalition mitigation behavior in our agents. Next we present and describe our experiments and results. Based on our results, we discuss some possible future work, and finally we conclude.

Background

A definition from the rational coalition outlined in (Kahan and Rapoport 1984) states that a coalition game is based on the total utility that the member of the coalition can achieve by coordinating and acting together, assuming that information is complete. Our problem domain is not superadditive in which a merged coalition of any pair of sub-coalitions is better than any pair of sub-coalitions operating individually as we have to consider coalition formation costs such as communication and computation costs. Furthermore, subadditivity does not apply to our model either.

Sandholm and Lesser (1995) introduce a bounded rationality in which agents are guided by performance profiles and computation costs in their coalition formation process. In traditional coalition formation, a rational agent can solve the combinatorial problem optimally without paying a penalty for deliberation.

Zlotkin and Rosenschein (1994) describe a coalition driven by task-oriented utilities. In a task-oriented domain (TOD), a coalition can coordinate by redistributing their tasks among themselves.

Shehory et al. (1997) relax some of the restrictive assumptions of theoretical coalition formation algorithms for a real-world system. In their model, each agent has a vector of real non-negative capabilities. Each capability is a property of an agent that quantifies its ability to perform a specific type of action and is associated with an evaluation function. The authors' model assumes that all agents know about all of the tasks and the other agents. In our model, an initiating agent knows only the agents in its neighborhood and knows partially about the updated status of a selective subset of its neighbors after negotiation. The details of intra-coalitional activity are not necessary for agents outside of the coalition in the Shehory et al (1997)'s model. On the contrary, in our model, an agent performs a task contributing to that coalition and the execution of this is reflected in the agent's commitments, constraints, and perceptions.

Shehory and Kraus (1998) further extend their work to incorporating negotiations, computational and communication costs. This model is similar to ours. However, our model allows an agent to conduct multiple concurrent negotiations, and adjusts its negotiation strategies to redesign its coalition.

Tohme and Sandholm (1999) studies coalition formation among self-interested agents that cannot make sidepayments—reward each other with payments for agreement to join some coalition, making the evaluation of a coalition solely on its utility.

Sen and Dutta (2000) propose an order-based genetic algorithm as a stochastic search process to identify the optimal coalition structure. A significant difference between the authors' work and our model is the scope of coalition formation. The authors' algorithm searches for an optimal coalition structure, which consists of all the agents in the environment grouped into one or more coalitions. Our model, however, focuses on the formation of a single coalition for a particular event while allowing multiple coalitions to be formed concurrently.

Other work in coalition formation include (Ketchpel 1994, Klusch and Shehory 1996, Sandholm 1999, Moon and Stirling 2001)

Coalition Formation

In our approach, an initiator starts the coalition formation process first by selecting members in the agent's *neighborhood* that are qualified to be part of an initial coalition. A neighborhood of an agent consists of all other agents that the agent knows. Second, it evaluates these members to rank them in terms of their respective potential utility values to the coalition. Third, it initiates negotiation requests to the top-ranked candidates, trying to convince them to join the coalition. In the end, the coalition may fail to form because of the candidates' refusal to cooperate, or may form successfully when enough members reach a deal with the initiating agent. Finally, the agent sends an acknowledgement message to the coalition members to announce the success or failure of the proposed coalition. If it is a success, then all coalition members that have agreed to join will carry out their respective tasks at planned time steps.

This approach is opportunistic as the goal is to obtain a satisficing coalition and the success of the formation is not guaranteed. This is the risk that our agent is willing to take: the utility of responding timely to a problem is dominating the utility gained from the quality of the solution, since the domain is time-critical and dynamic. In a way, our agent has no choice but to attempt and accept failures in coalition formation several times as long as the failures are quickly reached. Here we discuss the three components of coalition formation in details: initial coalition formation, coalition finalization, and coalition acknowledgment. Also we briefly discuss the learning capability of our coalition formation design. Then, to alleviate the consequences of failed coalitions, we also describe the agent behavior that mitigates risks.

Initial Coalition Formation

The goal of the initial coalition formation process is to find a group of neighbors that can be of help to the initiating agent. In a neighborhood of an agent a_i, the agent a_i can communicate with each of its neighbors; but not all pairs of neighbors can communicate with each other. A neighborhood is formed out of the utility for cooperation. In our multisensor target tracking domain, the neighborhood of a_i is the group of agents with sensors covering overlapping areas with a_i's sensor. In our CPU re-allocation domain, the neighborhood of a_i is the group of agents residing on the same CPU platform as a_i. In addition, these *candidates* are ranked according to their potential utility values to accomplish the task at hand. Thus, the ranking criteria include the capability of a candidate to provide useful resources and the past and current negotiation relationships between the initiating agent and that candidate. As a result, even if a candidate is the most capable among all candidates, its potential utility is reduced if it does not have a good relationship with the initiating agent.

First, upon detecting a resource or task allocation problem, the initiator describes the problem by observing its environment and its own activity. The parametric description helps guide the identification of the coalition candidates. Second, to establish who can provide useful resources or perform certain tasks, the initiator identifies them from its knowledge profile of its neighborhood. This knowledge profile consists of the following for each neighbor: (a) the name and communication id, (b) the resources it has, and (c) the tasks that it can perform. By matching the neighbor profile with the profile of the problem, the initiator selects useful neighbors and forms the initial coalition.

Since computational resources are limited, and negotiating consumes CPU and bandwidth, the initiator does not start negotiation with all members of the coalition, but first ranks them and then initiates negotiation selectively with the top-ranked ones. Ranking of the coalition members is done using a multi-criterion utility-theoretic evaluation technique.

The potential utility of a candidate is a weighted sum of (1) its ability to help towards the problem at hand, (2) its past relationship with the initiator, and (3) its current relationship with the initiator. Formally,

$$PU_{\alpha_k,a_i} = W_{\Lambda_{ini}(a_i,e_j)} \bullet$$

$$\left[rel_{past,a_i}(\alpha_k,t) \quad rel_{now,a_i}(\alpha_k,t) \quad ability_{a_i}(\alpha_k,e_j,t)\right]$$

where $W_{\Lambda_{ini}(a_i,e_j)} = \begin{bmatrix} w_{past,a_i,e_j} \\ w_{now,a_i,e_j} \\ w_{ability,a_i,e_j} \end{bmatrix}$ and

$w_{past,a_i,e_j} + w_{now,a_i,e_j} + w_{ability,a_i,e_j} = 1$ are weights used by the agent a_i to factor the three groups of attributes: past relationship, current relationship, and the ability of the candidate to perform the requested task. Note that ultimately these weights may be dynamically dependent on the current status of a_i and the task e_j. PU_{α_k,a_i} is the potential utility of the candidate α_k, as seen by the agent a_i. $rel_{past,a_i}(\alpha_k,t)$ is the past relationship value between the candidate α_k and the agent a_i, $rel_{now,a_i}(\alpha_k,t)$ is the current relationship value between the two, and $ability_{a_i}(\alpha_k,e_j,t)$ is the ability value of the candidate α_k computed by the agent a_i.

Each agent keeps a profile of its neighborhood, and current and past relationships with its neighbors, and the selection of the potential members of an initial coalition is based on this profile. The current relationship is based on the negotiation strains and leverage between two agents at the time when the coalition is about to be formed. The past relationship, however, is collected over time and enables an agent to adapt to form coalitions more effectively.

Ability. The ability value is based on a set of general heuristics and domain-specific criteria. The general heuristics are: (1) the uniqueness of the resource or functionality that a candidate can provide related to the problem at hand, (2) the quality of the resource and functionality that a candidate can offer—including volume, duration, efficiency, and so on, (3) how many resources that are useful in solving the problem does the candidate have, and (4) how many different functions can the candidate perform towards solving the problem.

In addition to these general heuristics, the evaluation also involves domain-specific criteria. Here, we basically examine the quality of the resource and functionality of the candidate driven by their applicability to the problem at hand as determined by domain-specific requirements. For example, if the initiator needs a neighbor to compute the first 100 prime numbers and neighbor A does not know how to, then the ability of neighbor A for solving the problem is zero. If neighbor A can only compute the first 25 prime numbers, then the initiator realizes that this neighbor can only fulfill partial requirements and if the initiator cannot find another neighbor to compute the next 75 prime numbers, then neighbor A's ability drops to zero as well. However, if a domain-specific rule states that one should compute for prime numbers as soon as possible to prevent computational stress at a later time, then neighbor A would be still useful and would have a non-zero ability.

Current Relationship. The current relationship between an agent and a candidate is defined as follows: Suppose (a) the number of concurrent negotiations that an agent can conduct is m, (b) the number of tasks that the agent is currently executing as requested by the candidate is k, (c) the number of all tasks that the agent is currently executing is K, and (d) the number of ongoing negotiations initiated by the agent to its candidate is n. Then, the current relationship value is a weighted sum of the following attributes:

(a) negotiation strain between the agent and the candidate: n/m,
(b) negotiation leverage between the agent and the candidate: k/m, and
(c) degree of strain imposed on the agent by the candidate: k/K.

The first attribute is inversely proportional and the other two are proportional to the current relationship value. The first attribute approximates how demanding the agent is of a particular neighbor. The more negotiations an agent is initiating to a neighbor, the more demanding the agent is and this strains the relationship between the two and the negotiations suffer. The last two attributes are used as leverage that the agent can use against a neighbor that it is negotiating with.

The current relationship can be computed readily from the status profile of the agent of its tasks and negotiation processes.

Past Relationship. Here we define the past relationship between an agent and a candidate. Suppose that

(a) the number of all previous negotiations initiated by the agent to the candidate is pT,
(b) the number of all previous successful negotiations initiated from the agent to the candidate is s,
(c) the number of negotiation requests from the candidate to the agent that the agent agrees to entertain is r,
(d) the total number of all negotiation requests initiated by the agent to all its neighbors is qT,
(e) the total number of all successful negotiations initiated by the agent to all its neighbors is q,
(f) the number of all previous negotiations initiated by the candidate to the agent is rT,
(g) the number of all previous successful negotiations initiated by the candidate to the candidate is c, and
(h) the number of all previous negotiations initiated by the candidate to all its neighbors is uT.

In our model, the past relationship value is a weighted sum of the following attributes:

(a) the helpfulness of the candidate to the agent: s/p ,
(b) the importance of the candidate to the agent: p/qT ,
(c) the reliance of the agent on the candidate: s/q ,
(d) the friendliness of the agent to the candidate: r/rT ,
(e) the helpfulness of the agent to the candidate: c/r ,
(f) the relative importance of the agent to the candidate: rT/p , and
(g) the reliance of the candidate on the agent: rT/uT .

Note that the above attributes are based on data readily collected whenever the agent initiates a negotiation request to one of the candidates or whenever it receives a request from one of its neighbors. The higher the value of each of the above attributes, the higher the potential utility the candidate may contribute to the coalition. The first three attributes tell the agent how helpful and important a particular neighbor has been. The more helpful and important that neighbor is, the better it is to include that neighbor in the coalition. These attributes imply that the arguments that the agent sends over to the candidate will garner strong evidence support. The agent expects the particular candidate to be *grateful* and more willing to agree to a request based on the agent's friendliness, helpfulness and relative importance to that candidate (in this way our method uses a form of *reciprocity* in agent interactions). On the other hand, the second set of attributes tells the agent the estimated chance of having a successful negotiation with the candidate based on how the candidate has behaved when interacting with the agent.

Finally, the initiator makes use of the potential utilities to carry out task allocations and assignments. Based on the overall potential utility of the initial coalition, the initiator may want to lower its demands to improve the chance of forming a coalition. By the same token, if the potential utility of a candidate is high, then the initiator may want to increase its demand with that candidate. We are currently investigating various algorithms such as greedy, lazy, worried, and weary. An initiating agent becomes greedy during a negotiation when (1) it tries to minimize its own rationalization and computing process, (2) it selects the candidate with the higher overall utility values to approach hoping for a successful negotiation, (3) it cares mostly about high-priority tasks, (4) it tries to maximize its chance of getting a particular task done—by including sub-utilities in the focused utility evaluation, and (5) it hopes to shift its responsibility (partially) to the candidates via successful negotiations—expecting the candidates to spawn their own coalitions to help respond to the problem at hand. In a lazy algorithm, the initiator prefers to concentrate its effort on a few candidates, as it does not want to spend resources or time on too many. In a worried algorithm, the agent asks for more than it needs to ensure, that if some of the candidates refuse to help, it will get what it needs. This translates to insurance policies. Finally, in a weary algorithm, the initiator prefers not to upset a candidate—especially one that has a high uniqueness—by being over-demanding. This leads to demand caps that make sure that an additional demand does not hurt the negotiation.

When a candidate is approached, it examines the requested the task against its current and planned activities. If the candidate realized that the requested task is do-able, then it agrees to negotiate. Otherwise, it refuses. So, a candidate does not compute the potential utility reciprocally when acting as a responding agent. In other words, the potential utility measures are computed only to help the initiating agent rank the neighbors so as to choose the best candidates for coalition formation.

Coalition Finalization

After obtaining the initial coalition and the coalition candidates ranked according to their respective potential utility values, the initiator invokes the coalition finalization step. This step consists of negotiations. To conserve computational resources and communication resources, the initiator only approaches the top-ranked candidates. When a candidate agrees to negotiate, the initiator proceeds with a one-to-one negotiation guided by a negotiation strategy. However, since the initiator conducts multiple, concurrent 1-to-1 negotiations, each negotiation process has access to the coalition status. This awareness allows the agent to modify its instructions for each negotiation process. In the following, we briefly talk about the argumentative negotiation model (Soh and Tsatsoulis 2001) and the coalition-aware, concurrent negotiations.

Argumentative Negotiation. Our agents use a variation of the argumentative negotiation model (Jennings et al. 1998) in which it is not necessary for them to exchange their inference model with their negotiation partners. Note that after the initial coalition formation, the initiator knows who can help. The goal of negotiations is to find out who is willing to help. To do so, first the initiator contacts a coalition candidate to start a negotiating session. When the candidate or responder agrees to negotiate, it computes a persuasion threshold that indicates the degree to which it needs to be convinced in order to free or share a resource or perform a task (alternatively, one can view the persuasion threshold as the degree to which an agent tries to hold on to a resource). Subsequently, the initiator attempts to convince the responder by sharing parts of its local information. The responder, in turn, uses a set of domain-specific rules to establish whether the information provided by the initiator pushes it above a resource's persuasion threshold, in which case it frees the resource. If the responder is not convinced by the evidential support provided by the initiator, it requests more information that is then provided by the initiator. The negotiation continues based on the established strategy and eventually either the agents reach an agreement, in which case a resource or a percentage of a resource is freed, or the negotiation fails. Note that, motivated to cooperate, the responder also counter-offers when it realizes that the initiator has exhausted its arguments or when time is running out for the particular negotiation. How to negotiate successfully is dictated by a negotiation strategy, which each agent derives using case-based reasoning (CBR). CBR greatly limits the time needed to decide on a negotiation strategy, which is necessary in our real-time domain since the agent does not have to compute its negotiation strategy from scratch. Please see (Soh and Tsatsoulis 2001) for details.

As *negotiation strategy* we define the set of guidelines (or protocol) that govern the behavior of an agent during a particular negotiation. In contrast to other work in negotiation where the negotiating parties followed a predefined, static protocol, our agents dynamically establish a new strategy depending on their current state and the state of the world. The goal is to situate a negotiation and to improve the chances of its success by taking into account the dynamically changing world state. This is accomplished by using CBR to select, adapt, and eventually learn negotiation strategies.

Since initiating a negotiation and responding to one are fundamentally different tasks, although still governed by the same methodology, each agent has two different case bases: one with strategies for initiating negotiations and one with strategies for responding to negotiation requests. Cases of both initiating and responding negotiation strategies have the same description, but different strategies. In the following we discuss the joint situation description of the two case types and then discuss the two types of strategies separately.

Each case also contains the negotiation strategy that was used in the past together with the outcome of the negotiation, such as: "offer accepted," "offer rejected," "ran out of time," or "ran out of resources." The strategy tells the agent how to conduct the negotiation. For the initiator the negotiation strategy includes:

1. a ranking of the classes of information it should use as arguments: during a negotiation each agent attempts to minimize the number and length of messages sent, since with fewer messages the agents can avoid message loss due to communication failures, and reduce traffic among the agents. The agents want to send short messages as well since the transfer is faster and the bandwidth is constrained. Thus, it is important for an initiating agent to decide which information pieces are more important to send to the responding agent;
2. the time constraint: how long (in real time) the agent should be negotiating, since the target may leave the area;
3. the number of negotiation steps: a "step" is a complete negotiation communication act where the initiator sends arguments and the responder makes a counter-offer or requests more convincing arguments. Clearly the more steps that are allowed the higher the chance of reaching an agreement, but also the more time and resources are spent;
4. the CPU usage: more CPU resources for a negotiation mean faster negotiation, but also less CPU available for other tasks.

The responder has a slightly different negotiation strategy. It shares some elements of the initiator's protocol, specifically the time constraint, the number of negotiation steps, and the maximum CPU usage, but it also introduces two more parameters:

1. the power usage: this defines how much power the responder is willing to use to turn on its radar;
2. persuasion thresholds for resources: as already mentioned, each resource has a persuasion threshold associated with it which determines how difficult it will be to convince the responder to free the resource.

Coalition-Aware Negotiations. Each agent has n negotiation threads. This means that an agent can conduct multiple, concurrent negotiations. Since a negotiation is not guaranteed to be successful, an initiating agent usually approaches more than the number of agents needed for a task.

This coalition awareness has several benefits. First, it allows an agent to free up its negotiation threads, communication channels, and communication bandwidth for other negotiation tasks. Second, it allows an agent to immediately abandon failing coalition, re-assess its environments, and start another coalition formation. Third, by terminating useless negotiations, an agent is able to base its reasoning on updated, more correct status profile.

This coalition awareness allows a negotiation thread to (1) determine whether to proceed with the current negotiation, and (2) to evaluate the acceptability of a counter-offer.

To improve the reflectiveness of each negotiation process, each negotiation thread is aware of the status of the coalition that it belongs to. For example, if a coalition requires 30 units of a resource, and the coalition already successfully recruits two candidates that total 35 units, then other negotiation processes are terminated. Similarly, if a coalition is no longer viable, then the initiator aborts its negotiation. For example, suppose that a coalition needs a carpenter and a woodcutter to build a table. Suppose the initiator approaches two carpenters and two woodcutters for help. The negotiations with the two woodcutters eventually fail. The agent immediately realizes that it no longer can form a successful coalition; and thus it aborts the two negotiations with the carpenters.

Here we show an example for counter-offer study. Suppose that the initiator is in need of 40 units of a resource. It has asked for help from 4 candidates: 25 units from candidate A, 15 units from candidate B, 10 units from candidate C, and 5 units from candidate D. In this case, the agent is worried and thus asks for more resources: 55 units instead of 40 units. Suppose that after a while, candidates B and C both agree to help and thus the initiator has formed a partial candidate where it now gets 25 units. Suppose that now candidate A counter-offers with 15 units of resource, the negotiation thread in charge of the negotiation with candidate A must check its awareness link, an updated snapshot of the health of the coalition. By comparing the need of the coalition, the negotiation thread realizes that the counter-offer is acceptable and thus responds accordingly.

The implementation of the coalition awareness is straightforward. Each negotiation thread of an agent shares the same coalition repository. When a negotiation thread reports its completion to the core (behavior) thread of the agent, the core thread updates the thread's information with the coalition repository, and subsequently determines the viability of the coalition. This information is automatically and immediately available at each negotiation thread.

Currently, once a coalition is agreed upon, agents who are involved cannot bail out. But, during negotiations, any of the agents involved can opt out: an initiating agent may terminate some of its negotiation activities once it realizes that it no longer needs those negotiations to be successful; and a responding agent may decide to reject any further negotiation when it realizes that it has started performing another task.

Coalition Acknowledgment

After a coalition has been decided—meaning all negotiation threads have terminated, the agent has to acknowledge the coalition. If the coalition is a failure, then the agent has to send out a *discard* message to each coalition member that has agreed to a deal to cancel the deal. If the coalition is a success, then the agent must send out a *confirm* message to the same coalition members. This acknowledgment is necessary for effective task planning of our agents.

In our multiagent system, each agent maintains a dynamic job queue. When a responding agent agrees to a request to perform a set of tasks, it registers the tasks with the job queue with the agreed start time and execution duration. However, since the negotiations conducted by the initiating agent are concurrent, that means that a task already registered may have to be discarded before it is executed at the planned start time.

This coalition acknowledgment demonstrates two useful agent characteristics towards responsible coalition formation. First, an initiating agent is responsible—it informs coalition members of the success or failure of the coalition and releases the coalition members from their agreements in case of a coalition failure caused by other negotiations. Second, a responding agent may unilaterally release itself from its agreement if it does not receive a confirmation of an agreed task.

This coalition acknowledgment step is a feature that allows the initiator to conduct concurrent negotiations asynchronously.

Learning Better Coalition Formation

Since our coalition formation strategy is opportunistic in that it tries to form satisficing solutions with no guarantees. Indeed, a coalition formation cannot be guaranteed due to the uncertain communication facility—messages may be lost, corrupted, or jammed. To improve the chance of successfully forming coalitions, we install learning mechanisms in our agent design aimed at learning to form better coalitions faster. We briefly discuss them in this section.

First, when a coalition is initially formed, an agent is motivated to go back to the same neighbor (for a particular task) that the agent has had good past relationship in their interactions. This reinforcement learning is evident in the computation of the potential utility described previously.

Second, when an initiating agent sends over different information classes to argue with the responding agent, one of the information classes includes a profile of the neighbors including the past relationship attributes. As a result, the responding agent is reinforced to agree to a negotiation by its previous interactions with the initiating agent: the more successes the initiating has had with a particular neighbor, the more effectively it can argue with that neighbor due to its reinforcement learning.

Third, we use CBR to retrieve and adapt negotiation strategies for negotiations. When a negotiation is completed, the agent also determines whether to learn the new case—storing it in the agent's case base. The case-based learning is performed in two modes: incremental and refinement. During the incremental learning the agent matches the new case to all cases in the case base and if it is significantly different from all other stored cases, then it stores the new case. When we want to keep the size of the case base under control, we use refinement learning—replacing the most similar old case with the new case if the replacement increases the diversity of the case base. This case-based learning allows an agent to learn how to achieve a successful negotiation (a coalition finalization) better and faster.

We are currently looking into learning to form coalitions better by adjusting the weights in the potential utility, driven by the failure and success rates of coalition formation. For example, if a coalition fails to form or forms with low resultant utility (after the tasks have been carried out or resources have been re-allocated), then the initiator may pinpoint the weak coalition members and identify the reweighting scheme that would have reduced their ranking in the previously carried out initial coalition formation step. Similarly, the initiator may learn from a successful coalition formation by computing the set of new weights that would distance the good coalition members from those discarded.

Coalition Mitigation

To alleviate the impact of a coalition failure, our agent behavior design adopts a natural mitigation approach. Our agent constantly monitors its environments. When a coalition fails, the agent continues to monitor and if it finds the same problem still present, it can start another round of coalition formation. Since our agents are reflective and situation-aware, this new coalition will have a different problem profile, neighborhood profile, and agent profile. This in effect allows the agent to look at the problem from a slightly different viewpoint, which may eventually lead to a successful coalition being formed. The rate of such a recovery (the number of coalition failures before success) is critically dependent on the dynamism of the problem.

This mitigation approach belies the principles of our coalition formation approach: the agents are willing to fail many times before getting it right because of the dynamism of the system that does not allow the agents to rationalize optimally an does not allow the agents to guarantee the successful formation of an optimal coalition.

Results

We have built a fully-integrated multiagent system with agents performing end-to-end behavior. We have conducted experiments using both a software simulation and a physical hardware setup. In the following, we report on one particular experiment performed using a simulation software called Radsim.

Application

The driving application for our system is multisensor target tracking, a distributed resource allocation and constraint satisfaction problem (Soh and Tsatsoulis 2001). The objective is to track as many targets as possible and as accurately as possible using a network of fixed sensors under real-time constraints. To track accurately, the agents controlling the sensor must be able to react to the incoming targets in a timely manner. For example, to track accurately a target moving at half a foot per second requires one measurement each from at least three different sensors within a time interval of less than 2 seconds. Moreover, the environment is noisy and subject to uncertainty and errors such as message loss and jammed communication channels. This further complicates the collaborations among the agents.

There are eight agents, each controlling one sensor, and two targets. The sensors are fixed and each target moves in a non-overlapping rectangular route as shown in Figure 1. In this experiment, Radsim was first started followed by the simultaneous invocations of the eight agents. Each agent has one behavior thread, one communication thread, one execution thread, and two negotiation threads. A tracking coalition requires at least three coalition members. Each agent interacts autonomously with the software simulation via software sockets: monitors the world, reasons and responds to events in the world, and actuates its radar and changes the world.

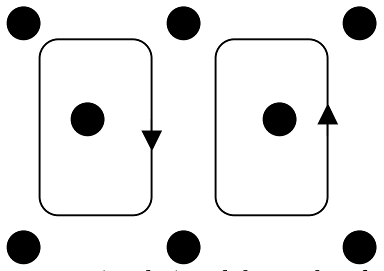

Figure 1. 8 sensors (circles) and the tracks of 2 targets in our experiments.

When a target is detected, the initiator measures the target to obtain the initial target location and velocity. This is the problem profile. Then, based on a geometric model of all its neighbors and the projected target trajectory, the initiator finds the radar coverage areas that the path crosses and identifies areas where at least three radars can track the target (remember that tracking requires almost simultaneous measurement from at least three sensors). Ranking of the coalition members is done using a multi-criterion utility-theoretic evaluation, guided by both general heuristics and domain-specific criteria. The domain-specific criteria are: (1) the target's projected time of arrival at the coverage area of a sensor: there has to be a balance between too short arrival times which do not allow enough time to negotiate and too long arrival times which do not allow adequate tracking; (2) the target's projected time of departure from the coverage area of a sensor: the target needs to be in the coverage area long enough to be illuminated by the radar; (3) the number of overlapping radar sectors: the more sectors that overlap the higher the chance that three agents will agree on measurements, thus achieving target triangulation; and (4) whether the initiator's coverage overlaps the coverage area of the coalition candidate.

Discussion of Results

Here we report on some preliminary experiment results for one typical run. In this run, the total number of attempts to form a coalition was 150. The total number of coalitions successfully formed (after coalition finalization) was 30, or 20%. The total number of coalitions confirmed by all three coalition members was 26, or 86.7% of all successfully formed coalitions. Finally, the total number of coalitions executed on time was 18, or 61.5% out of all successfully confirmed coalitions.

First, the percentage of successfully formed coalitions was only 20.0%. Out of the 120 failed attempts, 86 (71.7%) of them were caused by one of the coalition members outright refusing to negotiate, 17 (14.2%) were caused by the communication channels being jammed, and 17 (14.2%) were caused by busy negotiation threads. When an initiating agent initiates a negotiation request to a candidate and that candidate immediately refuses to entertain the negotiation, it can be due to (1) the responding agent does not have idle negotiation threads, or (2) the responding agent cannot project the requested task into its job queue. Thus, we expect this failure rate to decrease once we increase the number of negotiation threads allocated per agent. When an agent fails to send a message to another agent, or fails to receive an expected message, we label this as a communication "channel-jammed" problem. When an initiating agent fails to approach at least two candidates, it immediately aborts the other negotiation process that it has invoked for the same coalition. This causes the coalition to fail.

Second, the probability of a successfully formed coalition getting confirmed completely was 86.7%. For each coalition successfully formed, three confirmations were required. Out of 30 coalitions, 4 coalitions were confirmed only by two of the members. The causes were (1) the acknowledgment message sent out by the initiating agent was never received by the responding agent expecting a confirmation, and (2) the agreed task had been removed from the job queue before the confirmation arrived. The first cause happened since communication channels could be jammed. The second cause happened because of a contention for a slot in the job queue by two tasks. For example, suppose agent A receives a request from agent B to track a target starting at 8:00 a.m. Agent A responds to the request and starts a negotiation. Then later on, agent A receives a request from agent C to track a target also starting at 8:00 a.m., but using a different sensing sector (each sensor has three difference sensing sectors). Agent A checks its job queue and sees that it is free at that time and thus agrees to negotiate. Note that a task is inserted into the job queue only after the agent agrees to perform it. Now, suppose that both negotiations are successful. The negotiation between A and B ends first and then that between A and C. When the first negotiation ends, agent A adds the task requested by B to the job queue. Immediately after, when the second negotiation also ends successfully, agent A adds the second task, requested by C to the job queue, and this causes the second task to replace the first task. This is a problem with over-commitment.

Third, the probability of a confirmed coalition getting executed was 61.5%. Out of 26 coalitions confirmed, only 16 of them were executed completely. Of the 10 failures, there were two cases where none of the members executed its planned task; one case where only one of the members executed; and seven cases where only two members executed. The cause for the failure to execute was that the agreed task had been removed from the job queue before the execution took place.

Current and Future Work

Overall, our results showed that our agents are able to form coalitions quickly and in time to track a moving target in the environments. The agents were able to negotiate, plan synchronized tracking tasks and execute them accordingly. We are investigating solutions to address the following problems in our multiagent system:

(1) Channel Jammed – To prevent negotiation messages from getting lost and holding up negotiation threads, a better use of the available communication channels is needed. This will increase significantly the chance for response and acknowledgment messages to be received on time, and, in turn, the success rate of coalition formation.

(2) Task Contention and Over-Commitment – Currently, if an agent is approached by two other agents for two separate tasks around the same time slot, it entertains both requests and may run into task contention and over-commitment; we are studying ways of preventing this.

(3) Time Modeling – Sensor-related tasks must be modeled more closely to help plan and schedule a compact and sensible job queue. We have performed time profiling on various calls and have found that sensor-related tasks have a high variance of execution duration. We need to determine the bounds such that a tracking task can be safely scheduled and expected to be executed.

There are also several areas that we plan to examine in the future:

(1) Inter-coalition and intra-coalition competitions – task distribution, priorities, 'health' of coalitions, etc.

(2) Coalition awareness and the effects of coalition monitoring on speed (how much should a negotiation process monitor about the coalition when negotiating, and how reflective we want the negotiations to be of the coalition)

(3) How can we incorporate agent behavior into coalition formation tendencies: how greedy should the initiating agent be, how lazy should it be? Will irresponsible agents work together and actually produce faster response to world events?

(4) Online learning of better coalition formation strategies through distributed cooperative case-based learning

Conclusions

We have described a coalition formation strategy that aims at obtaining satisficing solution for time-critical, noisy, and incomplete resource or task allocation problem. Because of the nature of the strategy, a coalition is not guaranteed to form successfully especially when message passing among agents is not reliable. In our approach, our coalition formation process is divided into three stages: initial coalition formation, coalition finalization, and coalition acknowledgment. Initially, coalition candidates are selected hastily from an agent's neighborhood and subsequently ranked according to their respective potential utilities. Next, during the finalization phase, the coalition is refined and verified through negotiations, where information is exchanged between two agents to clarify commitments and constraints. The agent is able to coordinate directly and indirectly through a coalition awareness link with its negotiation threads. Finally, the coalition acknowledgment step confirms or discards already-agreed requests. This releases an agent from uselessly honoring a lost-cause coalition commitment. We have incorporated utility theory, case-based reasoning, argumentative negotiation, and real-time profiling in the above methodology and design.

Finally, we have built a multiagent system complete with end-to-end agent behavior. Our preliminary results are promising in that an initiator was able to form satisficing coalitions quickly given its constraints. Our results also showed that we need to manage our communication channels better, handle task contention and over-commitment, and model domain-related time constraints better.

Acknowledgments

The work described in this paper is sponsored by the Defense Advanced Research Projects Agency (DARPA) and Air Force Research Laboratory, Air Force Materiel Command, USAF, under agreement number F30602-99-2-0502. 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), the Air Force Research Laboratory, or the U.S. Government.

References

Jennings, N. R., Parsons, S., Noriega, P., and Sierra, C. 1998. On Argumentation-Based Negotiation. In Proceedings of the International Workshop on Multi-Agent Systems, Boston, MA.

Kahan, J. P. and Rapoport, A. 1984. *Theories of Coalition Formation*, Lawrence Erlbaum.

Ketchpel, S. 1994. Forming Coalitions in the Face of Uncertain Rewards. In Proceedings of the AAAI'94, Seattle, WA, July, 414-419.

Klusch, M., and Shehory, O. Coalition Formation among Rational Information Agents. In Proceedings of the MAAMAW'96, Eindhoven, Netherlands, 22-25.

Moon, T. K. and Stirling, W. C. 2001. Satisficing Negotiation for Resource Allocation with Disputed Resources. In Working Notes of 2001 Fall Symposium Series on Negotiation Methods for Autonomous Cooperative Systems North Falmouth, MA, November, 106-115.

Sandholm, T. W., Larson, K., Andersson, M., Shehory, O., and Tohme, F. 1999. Coalition Structure Generation with Worst Case Guarantees. *Artificial Intelligence* 111(1-2):209-238.

Sandholm, T. W. and Lesser, V. R. 1995. Coalition Formation amongst Bounded Rational Agents. In Proceedings of the IJCAI'95, Montreal, Canada, 662-669.

Sen, S. and Dutta, P. S. 2000. Searching for Optimal Coalition Structures. In Proceedings of the ICMAS'00, Boston, MA, July, 286-292.

Shehory, O. and Kraus, S. Methods for task allocation via agent coalition formation. AI 101, 1-2, 165-200, 1998.

Shehory, O. M., K. Sycara, and Jha, S. 1997. Multi-Agent Coordination through Coalition Formation. In Proceedings of the ATAL'97, Providence, RI.

Soh, L.-K. and Tsatsoulis, C. 2001. Reflective Negotiating Agents for Real-Time Multisensor Target Tracking. In Proceedings of the IJCAI'01, Seattle, WA, July, 1121-1127.

Tohme, F. and Sandholm, T. 1999. Coalition Formation Process with Belief Revision among Bounded-Rational Self-Interested Agents. *Journal of Logic & Computation*, 9(6): 793-815.

Zlotkin, G. and Rosenschein, J. S. 1994. Coalition, Cryptography and Stability: Mechanisms for Coalition Formation in Task Oriented Domains. In Proceedings of the AAAI'94, Seattle, WA, July, 432-437.