The 2009 IEEE/RSJ International Conference on
Intelligent Robots and Systems
October 11-15, 2009 St. Louis, USA

Dynamic Coalition Formation Under Uncertainty

Daylond J. Hooper, Gilbert L. Peterson, and Brett J. Borghetti
Department of Electrical and Computer Engineering
Air Force Institute of Technology
Wright-Patterson AFB, OH 45433
Email: {daylond.hooper, gilbert.peterson, brett.borghetti}@afit.edu

Abstract— **Coalition formation algorithms are generally not applicable to real-world robotic collectives since they lack mechanisms to handle uncertainty. Those mechanisms that do address uncertainty either deflect it by soliciting information from others or apply reinforcement learning to select an agent type from within a set. This paper presents a coalition formation mechanism that directly addresses uncertainty while allowing the agent types to fall outside of a known set. The agent types are captured through a novel agent modeling technique that handles uncertainty through a belief-based evaluation mechanism. This technique allows for uncertainty in environmental data, agent type, coalition value, and agent cost. An investigation of both the effects of adding agents on processing time and of model quality on the convergence rate of initial agent models (and thereby coalition quality) is provided. This approach handles uncertainty on a larger scale than previous work and provides a mechanism readily applied to a dynamic collective of real-world robots.**

I. INTRODUCTION

Task allocation mechanisms in cooperative robotic systems tend to make one of two assumptions: 1) tasks require only a single agent [1], or 2) the number of agents required for a task are known *a priori* [2]. The application of dynamic coalition formation to task allocation permits removal of these assumptions and allows for greater flexibility in the system's execution and assignment of tasks. Additionally, it enables the system to better respond to changes in its composition (i.e. the number and types of robots in the system). Many artificial intelligence and game theory researchers have performed work related to coalition formation [3], [4]. Their research has created stable allocation schemes and mechanisms for forming coalitions. Unfortunately, one largely unaddressed area in coalition formation research is that of uncertainty [5]. This limits the applicability of coalition formation to real-world robotic systems, since many aspects of the environment are poorly defined. One way to bridge this representational gap is to use stochastic representations for the coalition formation related components. The robotic systems that contribute to uncertainty are the robot (or agent) type, individual quality (or inversely, agent cost), coalition value, and sensor noise. This paper improves upon the uncertainty captured in previous coalition formation mechanisms by:

This research was sponsored by AFRL Lab Task 06SN02COR. The views expressed in this article are those of the authors and do not reflect the official policy or position of the United States Air Force, Department of Defense or the U.S. Government.

- Incorporating both game theory and distributed artificial intelligence work to provide a coalition formation mechanism with stability.
- Explicitly capturing the uncertainty in agent type, agent cost, and coalition value through a unique agent model and evaluation mechanism.
- Providing an algorithm that solves for profitable and stable potential coalitions for a given task in a decentralized manner. The agents then form the most profitable stable coalition.

This paper also provides the results of applying the mechanism to a simulated robot collective with variations in the agents' model accuracy and the collective size.

II. UNCERTAINTY IN COALITION FORMATION

Coalition formation processes are traditionally performed with assumed knowledge of agent types, agent costs, coalition value, and the coalition action (or task). The usefulness of coalition formation is therefore constrained to well-defined environments. In real-world robotic systems, the environment is often poorly defined, preventing application of coalition formation to the system. Capturing the uncertainty, however, improves the applicability of coalition formation to real world systems. Three types of uncertainty exist: agent type, agent cost, and coalition value.

Agent type describes an agent's hardware construction, configuration, and general capabilities. The agent type is constant for each agent, though other agents might not know the agent type for a specific agent. Furthermore, the agent type may fall outside of the agent types that another agent can represent accurately. This creates difficulty in evaluating the quality of a potential coalition, since the agent types affect the agent costs.

Agent cost is the expense incurred by an agent while executing a task. The agent cost directly affects an agent's quality within a coalition. Agent cost is not constant for a given task type, and may be affected by distance, energy investment, or the types of the other agents. Agent cost also dictates the division of payoff upon completion of a task. Inaccurate state representations and poorly represented agent types create uncertainty in agent cost, which makes evaluating the quality of the coalition difficult.

Coalition value is the third source of uncertainty. The coalition value describes a coalition's suitability for a task. If the agent types in a coalition are unequal, individual

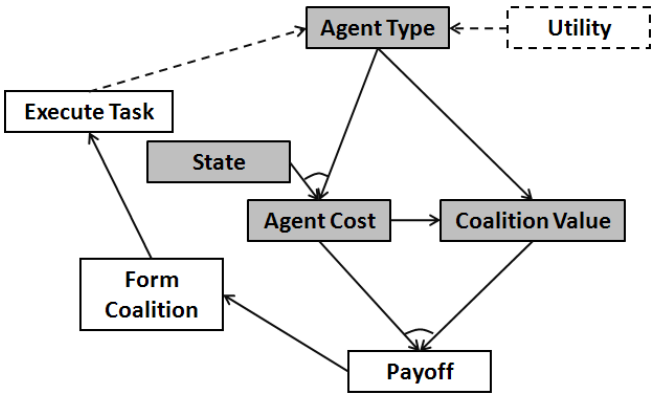

Fig. 1. AND/OR graph showing dependencies in coalition formation processes. The darker boxes are sources of uncertainty. The arrows indicate the direction of the dependency. Dashed lines indicate temporal values, fed in at the beginning or end of the formation cycle. The quality value input and execution result feedback are unique to the procedure presented in this paper.

quality dictates the coalition value. Coalition value is directly affected by the agent costs within the coalition. The agent costs are used to determine coalition value, which in turn is used to determine payoff.

Both agent cost and coalition value uncertainty are largely unaddressed, since most work assumes transferable utility (TU) is present [6]–[8]. TU exists where there is a common currency used to make side payments within a coalition [9]. If an agent is not contributing according to expectations, other agents transfer a payment to the poor agent. This increases the agent's quality, improving the coalition and improving stability. This is impractical for many situations since the currency may not be transferable, giving nontransferable utility (NTU). A real-world mobile robotic collective would not have TU. In fact, the assumption of TU has been called "exceedingly restrictive–for many purposes it renders n-person theory next to useless" [10]. An NTU coalition is often less stable than one with TU. In NTU coalitions, the characterization of each individual agent's quality is of crucial importance, since poor characterization causes a poor estimate of contribution.

If the quality of agent type estimates are poor, they inhibit the quality of the agent cost and coalition value estimates. Agent type estimates and state representations are applied to determine agent cost. The agent costs (and perhaps state, depending on approach) are applied to determine coalition value. The coalition value, together with agent cost, is used to determine individual payoff estimates. The individual payoff estimates reflect an expected quality of an agent, relative to other agents, within a coalition. This individual payoff is used to determine individual profit, which an agent uses to evaluate a coalition. These dependencies are shown in Figure 1. With uncertainty in agent type, agent cost and coalition value are affected. Even if the agent types are fully known, the state representation (which is by nature uncertain in robotic collectives) creates uncertainty in agent cost and coalition value.

III. RELATED WORK

There are three primary stability evaluation methods in game theory: the kernel [11], the core [12], and the Shapley value [13]. Each of these have different knowledge that they require in order to evaluate the coalition stability. The kernel is only concerned with coalition value, and the core and Shapley value focus on agent quality. If full knowledge of the information required by each stability concept is present, the uncertainties in the other sources are irrelevant. Conversely, full knowledge of the other two sources of uncertainty provides only minimal insight into the information they require. An aspect of the three sources of uncertainty is that knowledge of one does not necessarily imply knowledge of the others.

Chalkiadikis and Boutilier [5] incorporate agent type uncertainty to determine the Bayesian core. Their approach applies reinforcement learning to select the appropriate agent type from a set of candidates, allowing them to then determine agent cost and coalition cost to develop a coalition. Unfortunately, the agent type and agent cost are not as closely related in real-world applications as their approach implies. Failing to know the cost of even one agent limits knowledge of the coalition value, in turn making the computation more difficult since that agent's influence is unknown [14]. Furthermore, their application of reinforcement learning to select the agent type from a known set limits the types of agents that can be in a collective. If the true agent type falls outside the known set, the characterization may be erroneous and the mechanism may form poor coalitions.

Shehory and Krauss [6] provide a mechanism that does not explicitly require *a priori* information about the agent types, but solicits this information from the other agents during a coalition formation process. Their work assumes transferable utility, infeasible for many real-world robotic systems. Shehory and Krauss apply an algorithm to solve for all possible coalitions and perform a greedy selection of the preferred coalitions in order to determine the best coalition structuring. The resulting coalition structure is not guaranteed optimal due to the greedy selection strategy. Once the agent type information is solicited, there is no further uncertainty in agent type so the uncertainty is not incorporated into the calculation directly. Their approach generates poor coalitions if an agent cannot represent another agent's type appropriately.

Some work handles uncertainty by deflecting it. Soh and Tsatsoulis [7] provide a representation that allows for agent type and cost uncertainty with some agents, but requires adequate *a priori* knowledge of the agents in a neighborhood, reducing the complexity of the calculation to the well-understood neighborhood. There is not, however, uncertainty associated with the agents in the neighborhood, thus the coalition formation process is free from uncertainty. The full knowledge of the other agents in a neighborhood limits the applicability of this mechanism to a more uncertain collective. Vig and Adams [8] use the algorithm from Shehory and Krauss [6] in a multi-robot setting to form coalitions.

The uncertainty in their approach is limited to sensor noise and communication failure, which creates uncertainty in the robot's internal state estimate. However, the types and capabilities are assumed known *a priori*. This may be a valid assumption for static robotic collectives, but does not apply in collectives that allow for the addition of unknown robot types. Unfortunately, none of these approaches directly handle all three sources. The closest to date has been Chalkiadakis and Boutilier [5], which directly addressed uncertainty in agent type yet assumed knowledge of agent cost and coalition value. We extend Chalkiadakis and Boutilier's [5] work by generating agent types for the other agents through evaluation of an internal model.

A quality-based approach to allocating tasks to multiple robots involves the use of auction methods. Much work has been performed in auction methods to address efficiency and optimality [1], [2]. Unfortunately, auction methods make a single critical assumption: the number of robots required for a task are known *a priori*. Thus, the application of auction methods directly to coalition formation is limited to static coalition sizes.

IV. THE COALITION FORMATION PROCEDURE

On the notification of a new task, each agent begins the procedure with the determination and broadcast of its quality value for the issued task. An agent's overall quality value defines its individual capability for execution of a task. This quality value is calculated by evaluating the quality of each ability across the set of its abilities, an ability set. The quality value is broadcast to all members of the collective so they can individually process the task allocation. This coalition formation mechanism takes advantage of the quality values to gain insight into the broadcasting robot's abilities and form improved coalitions. However, since the quality for agent i, y_i, is based on a cost estimate and is limited by the abilities a receiving robot can represent, which may not be the same as the broadcaster, the quality values over the abilities are captured with uncertainty.

Let $N = \{1, 2, 3..., n\}$ be the set of agents in a collective. The ith coalition formed in this collective is $S_i \subseteq N$. The coalition structure is the set of all coalitions formed $CS = \{S_1, S_2, ..., S_g\}$, where g is the total number of coalitions formed. Intuitively, $S_1 \cup S_2 \cup ... \cup S_g = N$. An agent can be a member of multiple coalitions at once, so $S_a \cap S_b \subseteq S_a \cup \emptyset$. The tasks assigned the collective are $T = \{T_1, T_2, ..., T_p\}$. Each coalition $S_i \in CS$ has an associated task $T_j \in T$. Note that $i = j$ or $i \neq j$ is irrelevant, since the sequencing of the coalitions in S and the tasks in T are arbitrary. Each task T_j may have a sequence of subtasks (which do not decompose), defined as $\langle t_{1,j}, t_{2,j}, ..., t_{r,j} \rangle$. Each task also has an associated payoff U_j. The task cost C_j is equal to the coalition cost $x_{T_j}^{S_i}$, which is derived from the agent cost in this approach. A task T_j is a tuple, $\langle U_j, t_{1,j}, t_{2,j}, ..., t_{r,j} \rangle$. The cost for the task is not included in the tuple since the cost is dependent upon the coalition assigned to the task.

An agent type for agent i is represented by $A_i = \langle a_1^i, a_2^i, a_3^i, ..., a_p^i \rangle$, where each a_k^i is an ability. This is the true agent type, represented internally by agent i. Another agent j that can characterize n abilities represents agent i as $A_i^j = \langle a_1^{ij}, a_2^{ij}, a_3^{ij}, ..., a_n^{ij} \rangle$. This characterization allows each agent to model the other agent types through ability sets. This representation allows the agents to represent the abilities they understand. The set of abilities agent j uses to define agent i might not accurately represent agent i. In other words, A_i and A_i^j might not be subsets of each other. Define each $a_p^{ij},\ p \in \{1, 2, ..., n\}$ as a Gaussian distribution. When a task is provided, the agents determine an ability set $\{a_{t1}^{ij}, a_{t2}^{ij}, a_{t3}^{ij}, ..., a_{tp}^{ij}\},\ p \leq n$ that represents the abilities that j expects i to use to execute task T. This representation includes itself, so reasoning about the potential coalitions is more straightforward. Each ability a_m^{ij} is composed of a quality $q_m^{ij} \in (0, 1]$ and the relationship between these abilities are assumed known, so that the covariance matrix for them can be defined. As an agent completes tasks in coalitions, updates on the quality and covariance values are performed through a Kalman Filter update. The observation model H is defined as a $1 \times n$ matrix with the positions of the unused abilities set to 0 and the used abilities evenly dividing the weight, i.e., if three abilities are used, each has a value in the corresponding position of the H matrix of $\frac{1}{3}$. The propagation of the Kalman Filter is merely a carryover from the previous update.

The agent cost for agent i is $c_i = f(T_k, A_i)$. The cost is determined by calculating the costs associated with each subtask $\langle t_{1,k}, t_{2,k}, ..., t_{r,k} \rangle$. This is based on an agent's subjective understanding of the environment and estimates of its resource expense related to a task. This gives an estimated cost for each subtask, $\langle c(t_{1,k}), c(t_{2,k}), ..., c(t_{r,k}) \rangle$. Since a plan with abilities needed for each subtask is assumed available through a planning mechanism within the control architecture, the cost c_i is determined using

$$c_i = \sum_{t_{b,j} \in T_k} \frac{c(t_{b,k})}{q_1^{ij}} + \frac{c(t_{b,k})}{q_2^{ij}} + ... + \frac{c(t_{b,k})}{q_p^{ij}}$$

This shows that each ability scales the estimated cost for the task, generating task cost. In other words, the quality of the abilities contributing to a task affect the cost of executing the task. A very low-quality ability drives the cost up significantly. The ability-centric costs $\frac{c(t_{b,k})}{q_1^{ij}}$, etc. are added to each other for each subtask, creating a subtask cost, which is sent along with an estimated quality. The subtask costs are summed to create a total cost, c_i. More accurately, this total cost is c_i^j, or the total cost that agent j believes agent i will incur. The costs can be learned over time if the domain is not well known, or generated ahead of time if it is. Since the agents are to be real-world robots, it is assumed that the domain is not well known and the costs are learned over time.

The coalition cost C_S for coalition S_i is the sum of the agent costs in the coalition, which is based upon the agent types:

$$C_S = \sum_{k \in S_i} c_k = \sum_{k \in S_i} f(T_j, A_k).$$

This cost is applied to the coalition formation mechanism for generation of a profit vector $P_S = \{p_1, p_2, ..., p_m\}$ of a given coalition where $p_j = P_S \cdot \frac{1}{c_j^i \sum_{r \in S} \frac{1}{c_r^i}} = U_k - C_S \cdot \frac{1}{c_j^i \sum_{r \in S} \frac{1}{c_r^i}}$. The sum of the profit vector in relationship to the payoff of the task is the coalition value. The coalition value is not calculated explicitly, as it is implied through this relationship.

The coalition formation decision procedure is broken into two roles: proposer and responder. These roles are concurrently executed in each agent. The proposer generates coalitions based upon the agent's subjective view and suggests them to other agents, also making them available to the responder. The responder receives a suggested coalition and determines the quality of the proposal given the agent's subjective view of the potential coalition. Upon receipt of a proposed coalition, the agent no longer needs to propose that coalition to the other agents, so the responder role reduces the workload of the proposer by removing the coalition from the generated set. Because agent type is uncertain, the agents negotiate coalitions since full information needed for the generation of coalitions is not available. The proposer's procedure is shown in Algorithm 1.

Algorithm 1 Proposer(Task T)

1: Determine and broadcast quality and subtask costs.
2: Receive quality $y_j \forall j$ from other robots. Update beliefs.
3: **for** each $S \in N$ where $i \in S$ **do**
4: generate required abilities A_j for each agent j.
5: determine $C_S = \sum_{j \in S} c_j^i$.
6: calculate $p_j = U_T - C_S \cdot \frac{1}{c_j^i \sum_{r \in S} \frac{1}{c_r^i}}$ $\forall j \in S \Rightarrow P_S$.
7: **if** p_i *current* $< p_i$ *singleton* **then**
8: break.
9: **end if**
10: **for** each $j \neq i \in S$ **do**
11: send S, P_S to j.
12: **if** response = negative **then**
13: break.
14: **end if**
15: **end for**
16: save S in PS (the set of acceptable coalitions).
17: **end for**
18: **while** a coalition is not formed and $PS \neq \emptyset$ **do**
19: select from PS the S_m maximizing p_i.
20: issue S_m to its members of S_m for final approval.
21: **if** approved **then**
22: announce S to collective and exit.
23: **else**
24: remove S from PS.
25: **end if**
26: **end while**

The proposer procedure is three-staged: the first stage calculates the anticipated benefit of each coalition for this agent. If the coalition is better than the coalition of just itself (p_i *singleton*) then in the second stage, all of the agents in the potential coalition are proposed the coalition with associated profit vector (lines 10-17). Once all mutually acceptable coalitions are produced, in the third stage (lines 21-26), the best coalition generated is sent to the other agents for final approval. If approved, the agent begins execution on the task with the formed coalition.

Algorithm 2 Responder(message)

1: **if** message type is final approval with S, P_S **then**
2: **if** $i \in V$ where $p_i^S < p_i^V$ **then**
3: respond with negative
4: **else**
5: respond with positive, notify proposer of V
6: **end if**
7: **else**
8: receive S, P_S
9: Generate required abilities set $\forall j \in S$
10: Generate $C_S = \sum_{j \in S} c_j^i$.
11: Determine $p_j^i = U_T - C_S \cdot \frac{1}{c_j^i \sum_{r \in S} \frac{1}{c_r^i}} \Rightarrow P_S^i$.
12: Evaluate $p_i \in P_S$ and $p_i^i \in P_S^i$: $y = |1 - \frac{p_i^i}{p_i}|$.
13: **if** $y \leq q$ **then**
14: respond with positive
15: **else**
16: respond with negative
17: **end if**
18: **end if**

The responder handles messages from the proposer. It evaluates the message type, then proceeds according to the type. If the message type is for final approval, the proposer evaluates the request in terms of any currently approved coalitions (lines 2-6). If it is a proposal, it generates its own expected profit vector P_S^i (lines 8-11) then compares it to the offered profit (line 13) to determine whether the proposal is acceptable. The parameter q in line 13 is user defined, and dictates the degree to which the models must agree before accepting a coalition. A reasonable value is 0.3, which requires that a coalition proposal must contain a profit for agent i which reflects i's expected profit within 30%. Smaller values for q require greater agreement in the models.

The procedures presented above do not appear to address total quality directly. The quality is addressed during the generation of the abilities in line 4 of the proposer's procedure. The value of each ability affects the quality of each agent. This affects the profit of the coalition, so the quality is captured by the procedure. The procedure favors smaller coalitions since the payoff is static for the task and the profit is divided among the performing coalition. This may often lead to singleton coalitions unless the ability sets are designed appropriately. A system designer may choose to build a more complicated model where an agent's quality is affected by both the task type and the number of agents in a proposal. This requires an expansion of the quality value generation and evaluation, but leads to a much more sophisticated system which could favor coalitions up to a given size or of at least a given size.

V. RESULTS

A simulation was developed with multiple agents and a simulated task to move an item. The agents are broken into three types. Each agent type has ability sets representing solo execution and group execution of a task, though only the group execution quality is sent. The quality is determined from the values for each ability type within an agent, i.e. two agents of identical type have identical quality for group execution in this simulation. Type 1 is an agent which is good at the task and improves little if in a group, i.e. the difference between ability qualities in the solo and group abilities is low. Type 2 is an agent which is somewhat good at the task by itself, but improves significantly if in a group. Type 3 is an agent which is poor at the task and improves significantly if in a group. Each agent type possesses at least two abilities that the other agents do not. Agents of type 2 and 3 tend to propose coalitions, whereas agents of type 1 tend to decline coalitions since their profit is not increased by joining a group. The effects of different quantities of each agent type and the quality of each agent's internal model of the other agents are investigated. The factors of interest are the computation time and the number of communication messages sent. The simulation is designed such that communication does occasionally fail (i.e., communication messages may be blocked or lost) and multiple agents may issue proposals to each other at the same time.

The testing is broken into two parts: perfect models and imperfect models. The effects of varying the quantity and type of agents on the computation time are tracked in the perfect model testing. The imperfect model testing evaluates convergence rates: the quantity of tasks needed for the models to converge sufficiently to form coalitions.

A. Perfect Model Testing

With perfect models, the agents can evaluate and propose potential coalitions to any other agent, confident that the other agents will accept if the other agents cannot receive a better payoff alone. The effects of a specific type of agent on the message quantity and processing time may be significant, depending upon the types of the other agents in the collective. For example, a 12 member collective composed solely of agents of type 1 on a standard task completes rapidly since no agents issue proposals. However, changing this setup to a 12 member collective containing 4 agents of type 1 takes an average of 4,821 seconds to complete. The configurations selected for testing contain a split between the agent types in order to examine the worst case scenario for processing time. The quantity of each agent type for each run is shown in Table I. Each agent has two values for each ability: solo and group. The solo value gives the quality if an agent chooses to work alone, and the group value if the agent chooses to work in a group. This approach favors two-member coalitions, though it does not prevent larger coalitions in the case of poor models or communication failures.

Fig. 2 indicates the time taken for each run. The dots are realized run times and the line indicates the trend. Note that the plot is semi-logarithmic, reflecting the 2^n effects on the growth rate when an agent is added. The runtime effects are not significantly affected by the agent types, though this is due to the mixture of types in the collective.

TABLE I

THE RUN NUMBERS WITH THE QUANTITY OF EACH AGENT TYPE. EACH RUN WAS EXECUTED 10 TIMES.

Run	Type 1	Type 2	Type 3	Total
1	2	2	2	6
2	3	2	2	7
3	2	3	2	7
4	2	2	2	7
5	4	2	2	8
6	2	4	2	8
7	2	2	4	8
8	5	2	2	9
9	2	5	2	9
10	2	2	5	9
11	4	4	4	12

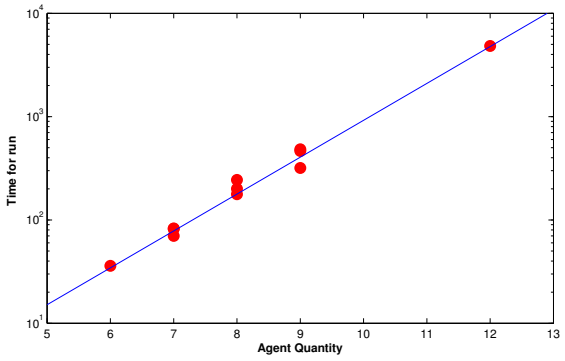

Fig. 2. Semi-logarithmic plot showing the effect on runtime of adding an agent. The realized values for each quantity of agents is shown relative to a line indicating the trend. Each agent has perfect models of all other agents.

B. Imperfect Model Testing

The convergence rate when the models are imperfect depends greatly upon the initial quality of an agent's models relative to the true configuration of the other agents. If the models are of poor quality, the agents may have to process a task multiple times prior to being able to form a coalition, since their models improve when the utilities are sent each time a task is presented. Each agent's unique abilities are not represented by the agents of any other type. Thus, this testing incorporates not only improvement of the models, but also the effects of not properly representing the other agents. The models were tested on four configurations: the first configuration has each agent assuming that all the other agents are identical to itself. The second configuration has each model consisting of skills identical to the modeling agent and ability values initially set to 0.5. The third configuration assumes all the agents except for the modeling agent are poor at the task, with each ability set to 0.01. The final configuration assumes all the agents except for the modeling agent are excellent at the task, with each ability quality set

to 1.0. A six-agent collective with two of each agent type is used for these tests. For each run, a task is re-evaluated five times. One run at each configuration is sufficient to evaluate the quality of the model convergence, since the convergence rate in the developed simulation is only based upon the broadcast quality and the model's initial quality, not on environmental effects or the coalition formed. In real-world scenarios, another opportunity for learning would be through performing a postmortem evaluation of the task, which would cause the convergence rate to become partially dependent upon the formed coalition.

Since agents only agree to form coalitions if their internal model agrees (within 30%) with the issued proposal, the first exposure to the task under these conditions does not form a coalition successfully, except when each agent assumes the others are identical to itself or are highly capable. The models converge to at least some coalitions agreed upon within three runs, regardless of the initial model quality. After five runs, the models are similar enough to each other for a given task that the proposing agents begin to issue proposals that better reflect the true cost and payoff. The selection mechanism within each agent is then better able to distinguish the quality of coalitions. Fig. 3 shows the number of iterations required to form a coalition. More accurate initial models converge faster, but even very poor initial models generate acceptable coalitions after eight iterations. More complex domains with widely varying task types may cause the convergence to take more iterations, but convergence occurs regardless of initial model quality.

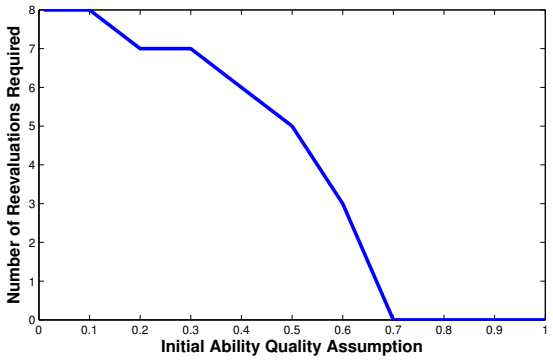

Fig. 3. The number of iterations required to form coalitions at varying initial model qualities.

VI. CONCLUSIONS AND FUTURE WORK

This paper presented a new coalition formation mechanism that directly allows for uncertainty within a collective. This approach is built upon a modeling and evaluation procedure that allows for model convergence over time and dynamic allocation of agents to a task. This procedure takes no longer than previous approaches [6], yet incorporates the environmental and agent type uncertainty when generating coalitions. This can be very useful in real-world autonomous mobile robotic systems, since it allows for introduction of new, previously unknown robot types into an existing system. This coalition formation mechanism has stability based upon the strong Bayesian core developed by Chalkiadakis and Boutilier. The algorithm solves for profitable and stable potential coalitions for a given task in a decentralized manner. The results of applying the mechanism to a simulated robot collective with variations in the agents' model accuracy and the collective size were provided. This paper showed that the models converge with repeated formation attempts, allowing for more accurate coalitions over time. The mechanism scales similarly to other coalition formation algorithms, yet enables the formation of coalitions when an agent type falls outside of a known set. This approach is well-suited to applications where turnover rates of robots are high yet lifespan is long. It also provides the ability of more general collectives to form coalitions in more diverse environments with agents or robots requiring minimal *a priori* knowledge about each other.

Future work includes the application of the algorithm to a real-world robot collective performing intelligence, surveillance, and reconnaissance, and the analysis of the effects of quality functions and parameters on the collective's performance.

REFERENCES

[1] M. B. Dias and A. T. Stentz, "A free market architecture for distributed control of a multirobot system," in *6th International Conference on Intelligent Autonomous Systems (IAS-6)*, July 2000, pp. 115–122.

[2] B. Gerkey and M. Mataric, "Multi-robot task allocation: analyzing the complexity and optimality of key architectures," in *IEEE International Conference on Robotics and Automation, 2003. ICRA '03.*, vol. 3, Sept. 2003, pp. 3862–3868 vol.3.

[3] G. Chalkiadakis and C. Boutilier, "Coalitional bargaining with agent type uncertainty," in *Proceedings of IJCAI-07*, Hyderabad, India, 2007, pp. 1227–1232.

[4] M. J. Osborne and A. Rubinstein, *A Course in Game Theory*. Cambridge, Massachusetts: The MIT Press, 1994.

[5] G. Chalkiadakis and C. Boutilier, "Bayesian reinforcement learning for coalition formation under uncertainty," in *AAMAS '04: Proceedings of the Third International Joint Conference on Autonomous Agents and Multiagent Systems*. Washington, DC, USA: IEEE Computer Society, 2004, pp. 1090–1097.

[6] O. Shehory and S. Kraus, "Methods for task allocation via agent coalition formation," *Artificial Intelligence*, vol. 101, no. 1-2, pp. 165–200, May 1998.

[7] L.-K. Soh and C. Tsatsoulis, "Utility-based multiagent coalition formation with incomplete information and time constraints," *IEEE International Conference on Systems, Man and Cybernetics, 2003*, vol. 2, pp. 1481–1486 vol.2, Oct. 2003.

[8] L. Vig and J. Adams, "Multi-robot coalition formation," *IEEE Transactions on Robotics*, vol. 22, no. 4, pp. 637–649, Aug. 2006.

[9] J. C. Gomez, "Generalizing the extended core to games with nontransferable utility," Tech. Rep., November 2002, university of Minnesota, Department of Economics.

[10] R. D. Luce and H. Raiffa, *Games and Decisions: Introduction and Critical Survey*. Dover, 1989, reprint. Originally published:Wiley, 1957.

[11] M. Davis and M. Maschler, "The kernel of a cooperative game," *Naval Research Logistics Quarterly*, vol. 12, no. 3, pp. 223–259, 1965.

[12] D. B. Gillies, *Contributions to the Theory of Games*. Princeton University Press, 1959, vol. IV, ch. Solutions to general non-zero-sum games, pp. 47–85.

[13] E. Winter, "Chapter 53: The shapley value," in *Handbook of Game Theory with Economic Applications*, R. Aumann and S. Hart, Eds. Elsevier, 2002, vol. Volume 3, pp. 2025–2054.

[14] M. Allen and S. Zilberstein, "Agent influence as a predictor of difficulty for decentralized problem-solving," in *AAAI 2007*, Vancouver, BC, Canada, July 22-26 2007, pp. 688–693.