*Proc. of IEEE International Conference*
*on Intelligent Robots and Systems*, Edmonton, Canada, 2005.

# Distributed Multi-Robot Coalitions through ASyMTRe-D

Fang Tang and Lynne E. Parker
*Distributed Intelligence Laboratory, Department of Computer Science*
*University of Tennessee, Knoxville, TN 37996-3450*
Email: {*ftang|parker*}*@cs.utk.edu*

***Abstract*— This paper presents a distributed reasoning system, called ASyMTRe-D, which enables a team of robots to form coalitions to accomplish a multi-robot task through tightly-coupled sensor sharing. The theoretical foundation of the negotiation protocol is ASyMTRe, an approach we developed previously to synthesize task solutions according to the task requirements and the team composition. The goal of the ASyMTRe approach is to increase the task solution capabilities of heterogeneous multi-robot teams by changing the fundamental abstraction from the typical "task" abstraction to a "schema" abstraction and automatically reconfigure the schemas to address the task at hand. The decision-making in this prior work was fully centralized; the current paper presents a distributed version of this approach based on the Contract Net Protocol, which can achieve higher levels of robustness than the centralized version. The purpose here is not to improve the original protocol, but to apply it to our problem so that the autonomous task solution capabilities of robots can be achieved in a distributed manner. Simulation results are provided to validate the protocol with performance analysis. Finally, we compare ASyMTRe-D with the centralized ASyMTRe. Our future objective is to enable the human designer to specify the desired balance between solution quality and robustness, enabling the reasoning approach to invoke the appropriate level of information-sharing among robots to reach the specified solution characteristics.**

## I. Introduction

Multi-robot *coalition formation* deals with the issue of how to organize multiple robots into subgroups to accomplish tasks collectively. The motivation behind coalition formation is to enable the team members to work together as a group to accomplish tasks that cannot be handled by individual robots. A challenging class of problems in coalition formation is determining when it is appropriate to form a coalition and how the robots should cooperate within a coalition. We are particularly interested in automated techniques for determining *how* to solve a multi-robot task, when the specific task solution is highly dependent upon the available capabilities of the multi-robot team, and thus cannot be specified in advance. This is especially challenging in heterogeneous robot teams, in which sensory and computational resources are distributed across different robots. In such teams, some robots may be more resource-bounded than others (e.g., have more limited sensing capabilities) and thus may not be able to accomplish certain tasks. For example, a robot with only an acoustic sensor cannot navigate autonomously in an environment, but is able to navigate successfully with direct assistance from another robot. For such a team to accomplish the task as a whole, the team must determine how to couple the appropriate sensory and computational capabilities from each robot, resulting in automatically formed coalitions that serve specific purposes.

To address this challenge, we present our ASyMTRe approach.[1] This approach is aimed at increasing the autonomous task solution capabilities of heterogeneous multi-robot teams by changing the fundamental abstraction that is used to represent robot competences from the typical "task" abstraction to a biologically-inspired "schema" ([1], [8]) abstraction, and providing a mechanism for the automatic reconfiguration of these schemas to address the teaming task at hand. In doing this, we are able to simultaneously obtain a number of significant new benefits in coalescent multi-robot teaming that have previously been difficult to achieve. These benefits include: (1) enabling robots to generate solutions to new tasks that were not explicitly programmed by the human designer, but instead consist of new, automated combinations of low-level building blocks, or schemas; (2) enabling robot team members to automatically generate task solutions based on sensor-sharing across team members, in configurations not previously explicitly defined by the human designer; (3) providing a way for robots to develop coalitions to address multi-robot tasks; and (4) enabling flexible software code reuse from one robot teaming application to another through the task-independent schema abstraction that is viewed as a generator of semantic information content which can be used in many ways by various diverse tasks. In the formal multi-robot task allocation (MRTA) framework of [7], our approach addresses the class of problems encompassing "Single-Task" robots (ST), "Multi-Robot" tasks (MR), and "Instantaneous Assignment" (IA). Specifically, we are addressing the development of heterogeneous robot coalitions that solve a single multi-robot task. Eventually, we expect that the ASyMTRe approach can be layered with prior task planning/allocation approaches, with ASyMTRe serving as a lower-level solution generator for *how* to solve tasks, with the higher-level, more traditional task planning/allocation strategies using these lower-level solutions for implementing the task solutions.

We have developed a centralized ASyMTRe configuration algorithm ([13], [14]) that generates single-task solutions for robot teams. In this prior work [14], we have proven that the centralized approach is sound, complete, and optimal (given enough processing time). It is well-known that centralized approaches suffer from lack of robustness. One approach to

[1] "ASyMTRe" is an abbreviation for Automated Synthesis of Multi-robot Task solutions through software Reconfiguration, pronounced "Asymmetry", which is first introduced in [13].

achieving robustness is to duplicate the configuration process on every robot, but this approach requires that every robot be aware of the capabilities of all the other team members. Any change in the team would require an update of the knowledge base on all the robots. This type of system would not work efficiently when robot failure or sensor failure is common, or when robots join or leave the team dynamically. To increase the robustness of the system, we have developed a distributed, negotiation-based ASyMTRe, which we call ASyMTRe-D. This work shares the same theoretical foundation as the centralized ASyMTRe and is built upon the well-known Contract Net Protocol (CNP) [12]. The purpose here is not to improve the Contract Net Protocol, but to demonstrate that the autonomous solution generation process can also be distributed among robots to achieve the robustness of a distributed system over a centralized system. The distributed approach offers a more flexible and robust approach to coalition formation than the centralized approach. However, it trades off solution quality for robustness. Thus, for any particular application, the human designer will have to determine the appropriate balance along the quality-robustness spectrum. Our future objective is to allow the human designer to specify the desired balance between robustness and solution quality, and to have the reasoning system utilize the appropriate amount of information sharing among robots to achieve this balance. This paper provides the foundational groundwork for this future work.

The remainder of this paper is organized as follows. Section II surveys our prior work on ASyMTRe, which provides a foundation of the distributed ASyMTRe-D. Section III then presents in detail the distributed negotiation process and the corresponding protocol. Experimental results are discussed in Section IV. We briefly outline related work in Section V, and provide concluding remarks in Section VI.

## II. Prior Work on ASyMTRe

We now briefly review the foundation of the ASyMTRe approach, which is based on one key idea – changing the fundamental abstraction of how we view robot capabilities from a "task" abstraction to a "schema" abstraction. Typically, the research community has defined heterogeneous robot capabilities in terms of *tasks* or *roles*. Under this view, a robot is provided with a set of software methods (usually referred to as control algorithms or "behaviors") enabling the robot to accomplish a set of pre-defined, application dependent tasks or roles (such as "push the box", "track the target", "go home", "forage", "defend home goal", etc.). The prevalence of this task-centric abstraction is evidenced by the significant focus of the heterogeneous robot teaming research on the topic of task allocation (e.g., [9], [2], [15], [16], [6]).

However, the development of software libraries of robot control code at the task level can severely restrict the reuse of this software in other applications. A fundamental problem is that software developed at the task level is often highly sensor, effector, and application-dependent. This dependency predefines how the robot team members will solve the given task. However, if the sensor capabilities of robot team members were to change (or, similarly, if new members were added that have very different sensor capabilities), their approach to solving the same task would be significantly different.

We address this problem in ASyMTRe by changing the common "task" abstraction to a "schema" abstraction, and providing a method enabling the robot team members to autonomously reconfigure the connections between the schemas to solve the task at hand. ASyMTRe is based upon a distributed extension to schema theory ([1], [8]) and the information theoretical work of Donald, et al. [4]. The following subsections outline this approach, which is the foundation upon which ASyMTRe-D is built.

### A. Schema Theory and Information Types

The basic building blocks of ASyMTRe are collections of environmental sensors (ES), perceptual schemas (PS), motor schemas (MS), and communication schemas (CS). A PS processes input from ES(s) to provide information to an MS, which then generates an output control vector corresponding to the way the robot should move in response to the perceived stimuli. A CS transfers information between various schemas distributed across robots, which is introduced to distinguish the connections within a robot from the connections across robots. All schemas are assumed to be pre-programmed into the robots at design time, and represent the fundamental individual capabilities of the robots. The connections between schemas are not fixed, but can be configured at run time.

ASyMTRe allows robots to reason about how to solve a multi-robot task based upon the fundamental information needed to accomplish the task. The information needed to activate a certain schema remains the same regardless of the way that the robot may obtain or generate it. Thus, we can label inputs and outputs of all schemas with a set of information types $F = \{F_1, F_2, ...\}$. $I^{S_i}$ and $O^{S_i} \subset F$ represent the input and output of schema $S_i$, respectively. Note that we use the term *information types* as distinct from *data types*. Semantics of the information is built into these information types, and does not just refer to a data type (such as boolean or integer). For example, the input information types of a *go_to_goal* schema could be {*current_position*, *goal_position*}, and its output types could be the specific motor commands. We assume that each schema has multiple inputs and outputs. An output of a schema can be connected to an input of another schema if and only if their information labels match. Using the mapping from schemas to information types, solution strategies can be configured at the schema level, rather than the sensor level, to determine *how* to solve tasks in a much more general manner. The benefit of the schema approach is that we can build up libraries of task-independent schemas, which also helps with code reuse.

### B. Knowledge Base

The knowledge base of information is represented as $(T, R_i)$, where $T = \{MS_1, MS_2, \cdots\}$ is the set of motor schemas that define the team-level task to be achieved, along

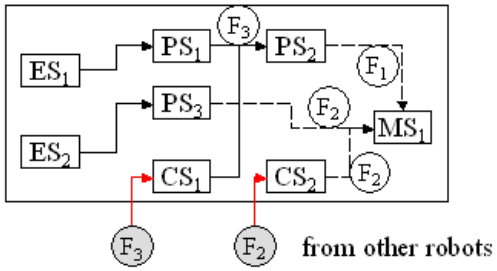


Fig. 1. An example of how the schemas are connected to accomplish a task. The solid-line arrows going into a schema represent an "OR" condition, meaning that it is sufficient for the schema to only have one of the specified inputs. The dashed-line arrows represent an "AND" condition, where all the indicated inputs are needed to produce a result. For example, $MS_1$ can calculate output only if it receives both $F_1$ and $F_2$. However, $PS_2$ can produce output based on either the output of $PS_1$ or $CS_1$.

TABLE I
CONNECTION CONSTRAINTS FOR SCHEMAS

| Sensor/Schema | Input Sources | Output Feeds into: |
|---|---|---|
| ES | Sensor Signals | PS |
| PS | ES, PS or CS | PS, CS or MS |
| CS | PS, or CS | PS, CS, or MS |
| MS | PS, CS, or ES | Actuators |

with application-specific parameters as needed.[2] A robot, $R_i$, is represented by $R_i = (ES^i, S^i)$. $ES^i$ is a set of environmental sensors that are installed on $R_i$, and $S^i$ is the set of schemas that are pre-programmed into $R_i$ at design time. Each schema is represented by $(S^i_j, I^{S^i_j}, O^{S^i_j})$. A schema can be activated if and only if its input can be obtained from the output of schemas or sensors on the local robot or can be directly transferred from other robots. Additionally, a set of *Connection Constraints* are used to specify the restrictions on correct connections between various schemas (see Table I).

### C. Potential Solutions

A *potential solution* is one way to connect schemas on an individual robot for it to fulfill its part of the task (i.e,. for all $MS_j \in T$, the inputs of $MS_j$ are satisfied, along with all the inputs from the schemas that feed into $MS_j$). We represent a potential solution by

$$PoS^i_j = (S^i_1, S^i_2, \cdots, S^i_k, F^i_1, F^i_2, \cdots, F^i_h). \quad (1)$$

where $PoS^i_j$ is the $j$th potential solution for $R_i$, $S^i_x$ ($1 \leq x \leq k$) is the $x$th schema of $R_i$ that needs to be activated, and $F^i_y$ ($1 \leq y \leq h$) is the $y$th information type that needs to be transferred to $R_i$. For example, in Fig. 1, if we assume that $T = \{MS_1\}$, one potential solution is to activate $\{PS_1, PS_2, PS_3, MS_1\}$, provided that the robot has both $ES_1$ and $ES_2$. Another potential solution is to activate $\{PS_2, CS_1, CS_2, MS_1\}$ when $F_3$ and $F_2$ can be transferred from other robots.

[2]In future work, we will develop a more general task specification, similar to the formal specification of tasks in [5].

### D. Solution Quality

With multiple potential solutions available, we introduce *utility* to measure their qualities. We define a *sensori-computational system* (SCS) [4], which is a module that computes a function of its sensory inputs and produces outputs. It is represented by $SCS^i_j = (S^i_j, ES^i_j, O^{S^i_j})$, where $S^i_j$ is the $j$th $PS/CS$ on $R_i$, $ES^i_j$ is the sensory input, and $O^{S^i_j}$ is the output. Each $SCS^i_j$ is assigned a cost $C^i_j$ and a success probability $P^i_j$, where $C^i_j$ represents the sensing cost of using $ES^i_j$ and $P^i_j$ represents the success rate of $S^i_j$ to generate a satisfactory result. We calculate the utility[3] of activating $SCS^i_j$ or producing $O^{S^i_j}$ by $U^i_j$:

$$U^i_j = \max(0, w \cdot P^i_j - (1-w) \cdot (C^i_j / \max_j(C^i_j))). \quad (2)$$

Here, $w$ ($0 \leq w \leq 1$) is a weight factor that balances the relative importance of the success probability and the cost. According to (1), we can measure the quality of a potential solution $PoS^i_j$ by summing the utilities of all the $SCS^i_j$ that need to be activated on the local robot and the utilities of the information types that are obtained from other robots. The goal is to maximize the utility of the selected potential solution.

### E. Finding Solutions

In centralized ASyMTRe, solutions are found by searching through the potential solution space for viable solutions of high utility. Since the optimization problem is NP-hard, finding optimal solutions quickly is not possible. However, we have identified robot ordering heuristics that enable centralized ASyMTRe to find good solutions quickly. If more time is available, our anytime algorithm will continue to search for better solutions and is guaranteed to find the optimal solution, given sufficient time [14]. However, as previously noted, a purely centralized approach may suffer from a lack of robustness. Thus, we now show how this approach can be distributed in order to improve system robustness. The solution generation process is achieved through a distributed negotiation process that is inspired by CNP. The purpose here is not to improve the original protocol, but to apply it to our problem so that the autonomous task solution capabilities of robots can be achieved in a distributed manner.

## III. THE DISTRIBUTED ASYMTRE-D APPROACH

### A. Negotiation Process

In ASyMTRe-D, robots are viewed as a set of information sources, where some robots do not have sufficient information to solve the task by themselves. To accomplish the task for the team as a whole, more capable robots can provide useful information to less capable robots. As previously noted, the sharing of information, and thus the cooperation among robots, are achieved through a distributed negotiation process, based on the Contract Net Protocol [12]. Each robot decides what

[3]In fact, the utility of a solution should also consider other aspects, such as the quality of information, frequency of the output, the computational complexity, etc. We will extend our utility definition to include these aspects in future work.

TABLE II

MESSAGES USED IN THE ASYMTRE-D PROTOCOL

| Type | Format |
|---|---|
| Simple Request | (‘$F1$’, *from*, *numinfo*, $F_1, \cdots, F_{numinfo}$) |
| Complex Request | (‘$F2$’, *from*, *numinfo*, $F_1, \cdots, F_{numinfo}$) |
| Simple Reply | (‘$H1$’, *from*, *to*) |
| Complex Reply | (‘$H2$’, *from*, *to*, *utility*) |
| Confirmation | (‘$C$’, *from*, *to*) |
| Cancellation | (‘$A$’, *from*, *to*) |

TABLE III

HANDLING POSSIBLE COMMUNICATION FAILURES

| Message Loss | Countermeasures | Time | Result |
|---|---|---|---|
| Reply | finite waiting time ($t$) | 0.75s | repeat requests |
| Request | repetitive requests | 10s | report failure |
| Confirmation | finite waiting time | 4s | cancellation |

information it needs and then requests it from others. The solution is evaluated based upon each robot's local information, and the final decision is determined by mutual selection. The negotiation process is totally distributed, with no centralized control or centralized data storage.

Such a distributed system offers a reliable, extensible, and flexible mechanism to make ASyMTRe suitable for applications where robot or sensor failures are common, or the robot team composition is dynamic (robots may join or leave frequently). The negotiation process is triggered at the beginning of each task to generate initial solution strategies, and is called to re-plan solutions to accommodate changes in robot teams or tasks. It is important to note, however, that the distributed approach trades off solution quality for team robustness. We note again that the intent of this approach is not to develop a new negotiation protocol, but instead to develop a method for the robot team to vary their reasoning between fully centralized and fully distributed decision-making, according to the desired balance between solution quality and robustness.

### *B. Distributed ASyMTRe-D Negotiation Protocol*

The distributed negotiation protocol involves the following major steps with the message types listed in Table II:

- **Make request**. Depending on the requirements of each potential solution, a robot broadcasts requests for the information types it needs to obtain from other robots. Simple requests are sent out at the beginning to estimate the potential number of robots ($pn$) that can provide the required information. It has been shown in [14] that the ordering in which robots are been considered in the configuration process is an essential factor to solution completeness and solution quality. Each robot will wait for a period of time that is proportional to its $pn$ value before sending out the complex requests. Thus, the robots with fewer potential helpers have higher priorities to make requests, since they will likely have fewer chances for success. However, sending simple requests increases the communication and computation cost. For tasks that are time critical, this step can be ignored and robots can directly send out complex requests instead.
- **Serve request and submit help**. After evaluating the required information, each robot replies based on a first-come-first-serve (FCFS) order. Simple replies are sent out without the estimation of utilities to enable the requesting robot to collect information about its $pn$. Otherwise, the robots will estimate the utility of providing the required information by (2). Since a requesting robot selects the potential solution with the highest utility, some capable robots are more likely to be chosen than others. Similar to [11], we assume robots work in a non-super-additive environment. Thus, the larger a coalition is, the higher the communication and computation costs are. Thus, we impose a *max-to-help* ($k$) constraint on each robot, which limits the number of robots that one can provide information. This constraint can reduce the complexity of the robots executing the solution due to motion constraints and balances the burden among capable robots.
- **Rank and confirm help**. Solutions are ranked by decreasing utilities. Each robot then selects the solution with the highest utility and sends a confirmation message. When there are multiple solutions with the same utility, the selection also follows the FCFS rule. If no robot responds to the request after the timeout, the robot will repeat the negotiation process until it reports “failure” after a period of time. The confirmation message will be broadcast to all robots, so that the other robots that are also willing to help can be released from their commitment and serve more requests.

The distributed ASyMTRe-D negotiation protocol acts as a greedy planner, since each robot selects the locally best solution to accomplish the task. However, it may not yield a global best solution, suffering from the usual problems of greedy algorithms. In [14], we have given an example of this problem and presented the centralized approach that takes into account all the orderings of robots (if given enough time), therefore generating the best solution for the team as a whole. Clearly, this represents the tradeoff between the robustness of a distributed solution and the solution quality of a centralized solution, which will be discussed further in experiments.

To ensure a general and robust negotiation process, some traditional mechanisms are built into the distributed protocol [3]. First, our protocol employs timeouts during the negotiation process (see Table III). The current settings of timeout values are based on experiments and estimation, which can be tuned as parameters to the program. A robot will wait for a finite time ($t$) for any replies, and if there is no reply, it will send out requests again. This process will continue for a period of time before the robot reports “failure”, which is either due to no robots being available to help, or to the requests or replies getting lost. A helping robot will also wait for a finite time for the confirmation. In this way, the robot can be released to help other robots if the confirmation gets lost or it is not selected to help. Similar to [6], our protocol also uses

broadcast messaging, rather than point-to-point, because it is efficient in transferring data and does not require the system to know specific destination information.

## IV. Experiments and Discussion

### A. Task Description

The distributed ASyMTRe-D negotiation protocol has been implemented and tested in simulation and physical robot teams [10] on a multi-robot transportation task. Since only the performance of the negotiation protocol is analyzed here, data described in this section are all from simulation results. In a transportation task, a team of robots is required to navigate from a set of starting positions to a set of goal positions (one per robot) in a global coordinate reference frame. This requires that each robot be able to perceive its current position relative to the goal position in global coordinates. For robots that cannot localize, more capable robots can serve as navigation assistants to guide them [10]. The environmental sensors used for the robots in these experiments are the laser scanner (`laser`) and an omnidirectional camera (`camera`). Additionally, robots are assumed to possess communication devices (`comm`). We have implemented the corresponding schemas on the robots (both physical and simulated): $PS_1$, which estimates its *own global position* ($F_1$) using a laser and an environmental map; $PS_2$, which gives the *goal position* ($F_4$, hard-coded); $PS_3$, which estimates *relative position of another robot* ($F_3$) using camera and visual fiducial; $PS_4$, which estimates the current robot's own global position according to another robot's global position and that other robot's position relative to the current robot $PS_5$, which estimates *global position of another robot* ($F_2$) according to its own global position and the estimated relative position of that robot; $CS_1$, which transfers information between robots; and $MS_1$, which calculates *motor commands* ($F_5$) that lead the robot toward the commanded direction.

We also have $T = \{MS_1\}$ and assume that all robots are pre-programmed with the motor schema $MS_1$, as well as the perceptual schemas corresponding to the physical sensors they have available to them. The input and output information used in this task is shown in Table IV. Here, we only provide fuzzy estimates for costs and probabilities; in actual applications, these estimates would be specific numeric values. Also, each robot has the same scale of cost and probability estimation.

### B. Simple Case

To illustrate the results, we now present a simple experiment in which five robots are brought together to form a team, three of which are of type I, III, and IV, respectively, and two of which are of type II, as shown in Fig. 2. Type I robot has a laser and can localize using $PS_1$. Type II robot only has communication capability, thus it needs to obtain its $F_1$ from other robots to navigate. Type III robot has a camera and can estimate $F_3$ using $PS_3$, however to accomplish the task, it still needs to obtain $F_2$. Type IV robot has both laser and camera, thus it can localize and produce $F_2$ and $F_3$ at the same time. Additionally, we set the max-to-help parameter ($k$) to be either 1 or 2, the weight $w$ to be 0.5, and various timeouts to be the values in Table III.

TABLE IV

Input and output information types for various schemas and their corresponding sensing costs and success probabilities

| $S_i$ | ES | $I^{S_i}$ | $O^{S_i}$ | $C_i$ | $P_i$ |
|---|---|---|---|---|---|
| $PS_1$ | Laser | Laser signal | $F_1$ | High | High |
| $PS_2$ | none | Hardcoded | $F_4$ | none | none |
| $PS_3$ | Camera | Camera signal | $F_3$ | Medium | Medium |
| $PS_4$ | Camera | $F_2$ and $F_3$ | $F_4$ | Medium | Medium |
| $PS_5$ | Camera | $F_1$ and $F_3$ | $F_2$ | Medium | Medium |
| $CS_1$ | Comm | $F_1$<br>$F_2$ | $F_2$<br>$F_1$ | Low | High |
| $MS_1$ | none | $F_1$ and $F_4$ | $F_5$ | none | High |

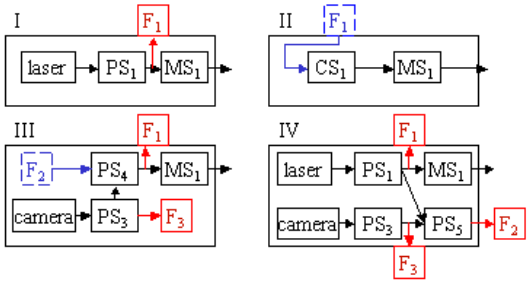


Fig. 2. Four types of robots (I, II, III, and IV) defined by various connections of schemas to accomplish the task. The dashed-line box represents the information required by the robot. The solid-line box represents the information produced by the robot. We assume that all the robots have communication capabilities and $CS_1$ is omitted except for the type II robot.

We performed 120 trials (5!) with different orderings of robots in both cases ($k$ = 1, 2) and observed that the group of robots successfully coalesced to accomplish the task 100% of the time when $k$ is 2, and 50% of the time when $k$ is 1. The unsuccessful cases are because all the capable robots were engaged in activities helping others. However, those robots that report "failure" can wait until other robots are finished with their current activity, and try again to find solutions. In this case, capable robots help the resource-bounded robots in a sequential manner. The order in which each robot starts the negotiation process is also a factor because it assigns a priority to the robot. A robot with a higher priority has a better chance of obtaining the required information. Fig. 3 shows two successful coalitions of the five robots. On the left, a type IV robot helps both type II robots by providing $F_1$ to

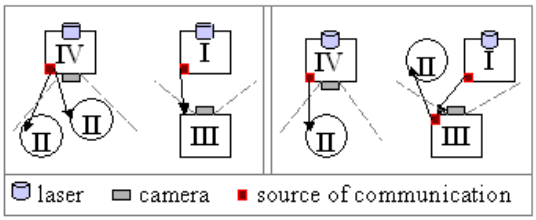


Fig. 3. Two successful simulation results. The arrows represent the flow of information among robots.

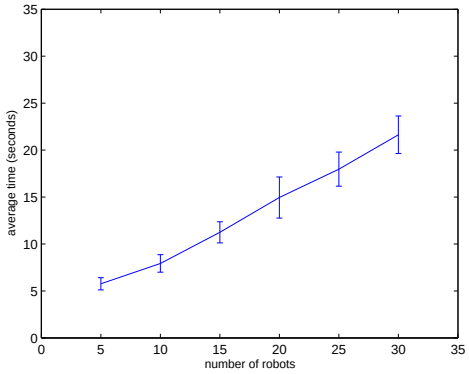


Fig. 4. The size of the robot team (*n*) vs. the average time to achieve solutions using parameter settings in Table III.

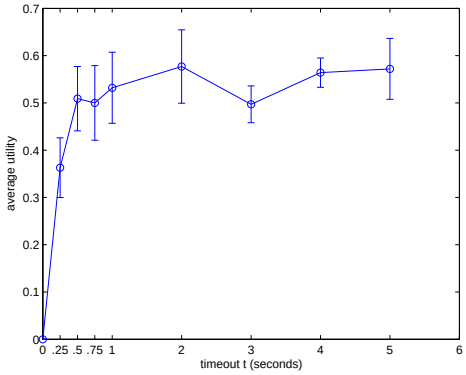


Fig. 6. Timeout value (*t*) vs. the average utility of the generated solutions ($n$ = 10).

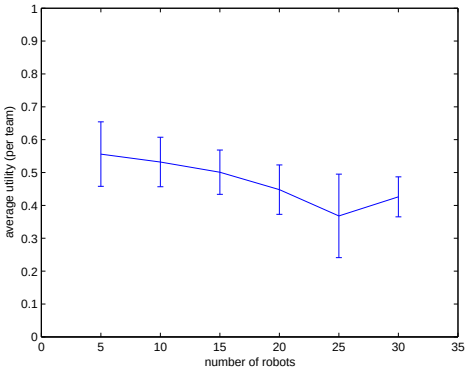


Fig. 5. The size of the robot team (*n*) vs. the average solution quality.

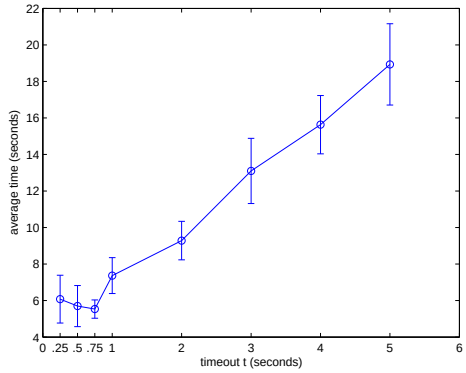


Fig. 7. Timeout value (*t*) vs. the average time to achieve solutions ($n$ = 10).

them respectively, and a type I robot helps a type III robot by providing $F_2$. On the right, a type III robot helps one of the type II robots by providing it $F_1$ after the type III robot has been helped by a type I robot.

### C. Team Size and Timeouts

The above simple case presents the example operational results of the negotiation protocol. The following experiments explore the performance of this protocol by varying the size of the robot team and the values of the timeout *t*. First, we enlarge the team by duplicating the robots with the same set of capabilities as in the simple case. Fig. 4 shows that the average time to find a solution increases linearly as we increase the team size, because more messages need to be processed when there are more robots. The computational complexity is a function of the number of robots ($n$) and the average number of potential solutions ($m$). As suggested in [7], we consider the calculation and comparison of the utility as the major operation in the protocol. Thus, the computational complexity of the protocol is $O(mn)$, since there are at most $mn$ requests/replies for a robot to process. Fig. 5 shows the variation of the solution quality with increasing team sizes. The solution quality degrades as the team size increases because each robot greedily searches for the best solution for itself without considering benefiting the whole team. Thus, the more robots that are involved, the higher chance that the total quality will be reduced.

In the second experiment, we try different lengths of the timeout for the finite waiting time (*t*) to determine the optimal setting.[4] Given a specific robot team of size *n*, each robot receives a certain number of messages (*e*) before timeout to select the robot helper with the highest utility. Assuming that the average message size (*s*) and the maximum data transfer rate ($r$ = 11 Mb/s for wireless communication) are known, we can use the following formula to calculate the minimum timeout value (*t*) required for the *rank and confirm help* step of the negotiation process:

$$t_{min} = n \cdot e \cdot s/r + v \tag{3}$$

Here, we introduce *v* to account for the variations caused by communication delays and communication load of the system in actual experiments and set its value to be 0.5 to ensure a robot receives an adequate number of messages. In our experiment, a robot receives an average about 8 messages

[4]The other two timeout values in Table III are only from estimation. In our future work of analyzing the robustness of the protocol, these two values will be fine-tuned to obtain the optimal setting.

during the *rank and confirm help* step and the average size of a message is 11 bytes. Thus, the minimum timeout *t* is approximately 0.5 + 0.00064 (= 0.50064) seconds. If the timeout *t* is less than the minimum timeout, a robot will likely receive fewer messages and the solution quality will degrade. If the timeout *t* is 0, a robot will receive no messages and will remain idle until it reports failure. Fig. 6 shows that given a timeout *t*, the solution quality fluctuates within a small range, and as long as *t* is above some threshold (in our case, 0.5 seconds), the qualities remain similar. Fig. 7 shows that with a team size of 10, a smaller *t* is preferred, since smaller *t* values give quicker solutions. Combining the results from Fig. 6 and Fig. 7 gives us a near-optimal setting of approximately 0.75 seconds for *t* in our example, because the solution quality is high and the average time is low at that point.

### D. Discussion

*1) Solution quality:* As shown in the above experiments, the solution quality drops with increasing team sizes because of the greedy search process. Additionally, the preference of a smaller coalition size also plays an important role. A larger coalition size would involve a penalty in the quality calculation, since we assume that it would require higher communication and computation costs. In centralized ASyMTRe, a *maximum cooperation size* is specified [11], resulting in small-sized robot coalitions (see Fig. 8, left). However, there is no such limit to the coalition size in ASyMTRe-D (remember that the *max-to-help* only limits the number of robots that one can help with), possibly resulting in a long chain, or hierarchical coalition structure (see Fig. 8, right), which introduces more complex teaming solutions than are necessary. Ongoing work includes applying the maximum coalition constraint to ASyMTRe-D.

*2) Robustness:* The robustness of ASyMTRe-D over centralized ASyMTRe is enhanced through the characteristics of the negotiation protocol and the introduction of timeouts and broadcasting. It remains as future work to characterize the robustness of the system in detail. However, several prior works in multi-robot cooperation illustrate that similar techniques generate robust multi-robot coordination (e.g., [9], [3]), and thus we expect future experiments to show the same for ASyMTRe-D. As identified in [3], three categories of malfunctions (i.e., communication failures, partial robot malfunctions, and robot death) will be considered in our future work to validate the robustness. Communication failures are handled in the traditional ways, such as acknowledgements, timeouts, etc. Partial robot malfunctions or robot death will be handled through the activation of the negotiation or reasoning process among a coalition or the whole team to generate a new solution based on current team capabilities.

*3) Comparison:* When comparing ASyMTRe-D with centralized ASyMTRe (see Table V), we observe that ASyMTRe-D is robust and flexible with little maintenance of the knowledge base since any change in the robot team only needs to be updated locally. However, the solution quality is only optimized locally because of the greedy search process. If we run centralized ASyMTRe on a single robot (method 2 in Table V), the best solution can be found given enough time [14]. However, except for the major concern of single point failure, this method requires a complete sharing of robots' capabilities at the beginning and sending the solutions back to all robots at the end. The centralized knowledge base also needs to be updated when there is any change in the team composition. To increase the robustness, we could run centralized ASyMTRe on every robot (method 3 in Table V). However, robots still need to share capability information with each other at the beginning or whenever the team composition changes. This method requires more work to maintain the knowledge base than the centralized approach on a single base station, since the knowledge base updates must be duplicated on all robots. Our ongoing work also includes a formal comparison between the solution qualities of the two approaches.

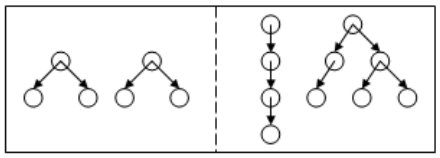

Fig. 8. On the left, robot are grouped into small size coalitions. On the right, coalition size is not limited.

*4) Automating Balance of Solution Quality and Robustness:* As previously noted, an ultimate objective of this research is to develop an automated decision maker that can choose the appropriate balance of solution quality (as obtainable with more complete information) with robustness (as obtainable with a distributed decision-making process). This process is easily automated for small problems, since method 3 in Table V is clearly the best solution in this case. However, for larger problems, it is unclear whether it is best to employ the fully distributed solution, or to use the centralized solution (either at a base station or on all robots), or perhaps to use the distributed solution combined with more initial information sharing to enable robots to reason with more complete information. Our future work approach is to analytically formulate the desired degree of solution quality and robustness and its relationship to the team size, mix of capabilities, time available for the solution, communications data rate, human user priorities, and so forth, to determine the proper amount of information exchange and computation that should be carried out by each robot team member. This should enable an automated system for determining the proper tradeoffs between the higher solution quality possible with the centralized process and the robustness that is possible with the distributed process. The distributed decision-making process reported in this paper is a necessary foundation for achieving this ultimate objective.

## V. Related Work

Coalition formation is not a new concept in the multi-agent community. Many approaches have been proposed to coordinate agent behaviors, in which agents are organized into coalitions to achieve a higher-level goal [11] that requires

TABLE V

COMPARISON BETWEEN CENTRALIZED ASYMTRE AND ASYMTRE-D

| **Method** | **Computational Complexity** | **Solution Quality** | **Robustness** | **Maintenance of Knowledge Base** |
|---|---|---|---|---|
| 1. ASyMTRe-D | $O(mn)$ | Locally optimal | High | Low |
| 2. Centralized ASyMTRe on one robot (base station) | Optimal solution: $O(mn!)$ First solution: $O(mn^2)$ | Globally optimal given enough time | Single point failure | Medium |
| 3. Centralized ASyMTRe on every robot | Optimal solution: $O(mn!)$ First solution: $O(mn^2)$ | Globally optimal given enough time | Yes, with redundancy | High |

complex planning and calculation. Our work goes beyond the mapping of capabilities to tasks by abstracting the problem at the schema level, rather than the task level, and by automatically instantiating solutions into executable code.

After Smith [12] first introduced the Contract Net Protocol, several approaches addressing multi-robot cooperation through negotiation were developed, such as M+ [2], Traderbots [16], and MURDOCH [6]. M+ [2] is a cooperative scheme for task allocation and has an interface to higher level task planner. The Traderbots approach [16] presents a market-based method that is distributed, but incorporates opportunistic centralized planning to improve the solution quality. MURDOCH [6] employs a *publish/subscribe* communication model to carry out auctions, which has the advantage of anonymous communication. All works assume that a task is divided into subtasks or task trees by a general planner; the robots then bid and negotiate to carry out the subtasks. The goal is to maximize the utility for executing a certain task. Our approach is also based upon the economy model, which utilizes broadcasting and timeouts to ensure a robust communication model. However, our approach is different in that we do not assume that a task can be subdivided into independent subtasks/roles. Instead, our approach addresses multi-robot tasks, which require more than one robot working together to solve a single task. By abstracting the problem at the schema level rather than at the task level, we believe that we can generate more flexible solution strategies that determine *how* to solve multi-robot tasks, and are not dependent upon predefined solution strategies in the form of task decompositions or roles that are specified in advance by the human designer.

## VI. CONCLUSIONS AND FUTURE WORK

This paper has presented the distributed ASyMTRe-D negotiation protocol that coalesces multiple robots into teams by synthesizing single-task solutions based upon the team composition. The distributed negotiation process enables each robot to find the best solution locally by maximizing the utility for executing the task. Compared with our prior centralized ASyMTRe, distributed ASyMTRe-D provides a more robust and flexible way to form coalitions. However, it also presents a tradeoff between solution quality and robustness.

Our ongoing work includes developing the mechanisms to allow ASyMTRe-D to automatically determine the appropriate level of information sharing to achieve the desired balance between solution quality and robustness. Additionally, we are working towards verifying this approach with more physical experiments, discovering motion constraints in the physical applications, and incorporating the constraints into the solution generation process.

## REFERENCES


[1] R. C. Arkin. Motor schema based navigation for a mobile robot: an approach to programming by behavior. In *Proceedings of IEEE Conference on Robotics and Automation*, pages 264–271, 1987.

[2] S. Botelho and R. Alami. M+: A schema for multi-robot cooperation through negotiated task allocation and achievement. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 1234–1239, 1999.

[3] M. B. Dias, M. Zinck, R. Zlot, and A. Stentz. Robust multirobot coordination in dynamic environments. In *Proceedings of IEEE International Conference on Robotics and Automation*, 2004.

[4] B. R. Donald, J. Jennings, and D. Rus. Towards a theory of information invariants for cooperating autonomous mobile robots. In *Proceedings of the International Symposium of Robotics Research*, Hidden Valley, PA, October 1993.

[5] C. H. Fua, S. S. Ge, and K. W. Lim. Boas: Backoff adaptive scheme for task allocation with fault tolerance and uncertainty management. In *Proceedings of the IEEE Intl. Symposium on Intelligent Control*, Taipei, Taiwan, September 2004.

[6] B. Gerkey and M. J. Mataric. Sold!: Auction methods for multi-robot coordination. *IEEE Transactions on Robotics and Automation*, 16(5):758–768, 2002.

[7] B. Gerkey and M. J. Mataric. A formal analysis and taxonomy of task allocation in multi-robot systems. *Intl. Journal of Robotics Research*, 23(9):939–954, September 2004.

[8] D. M. Lyons and M. A. Arbib. A formal model of computation for sensory-based robotics. *IEEE Transactions on Robotics and Automation*, 5(3):280–293, 1989.

[9] L. E. Parker. ALLIANCE: An architecture for fault tolerant, cooperative control of heterogeneous mobile robots. In *Proc. of the 1994 IEEE/RSJ/GI Int'l Conf. on Intelligent Robots and Systems (IROS '94)*, pages 776–783, Munich, Germany, Sept. 1994.

[10] L. E. Parker, M. Chandra, and F. Tang. Enabling autonomous sensor-sharing for tightly-coupled cooperative tasks. In L. E. Parker, A. Schultz, and F. Schneider, editors, *Multi-Robot Systems Volume III: From Swarms to Intelligent Automata*. Kluwer, 2005.

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

[12] R. G. Smith. The Contract Net Protocol: high-level communication and control in a distributed problem solver. *IEEE Transactions on Computers*, C-29(12), December 1980.

[13] F. Tang and L. E. Parker. ASyMTRe: Automated synthesis of multi-robot task solutions through software reconfiguration. In *Proceedings of IEEE International Conference on Robotics and Automation*, 2005.

[14] F. Tang and L. E. Parker. Coalescent multi-robot teaming through ASyMTRe: a formal analysis. In *Proceedings of the 12th International Conference on Advanced Robotics*, 2005.

[15] B. B. Werger and M. J. Mataric. Broadcast of local eligibility for multi-target observation. *Distributed Autonomous Robotic Systems 4*, pages 347–356, 2000.

[16] R. Zlot, A. Stentz, M. B. Dias, and S. Thayer. Multi-robot exploration controlled by a market economy. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 3016–3023, 2002.