INVITED PAPER

Building Multirobot Coalitions Through Automated Task Solution Synthesis

A group of robots can move to, or push boxes to, specified locations by sharing information when individual robots cannot perform the tasks separately.

By Lynne E. Parker, *Senior Member IEEE*, and Fang Tang, *Student Member IEEE*

ABSTRACT | This paper presents a reasoning system that enables a group of heterogeneous robots to form coalitions to accomplish a multirobot task using tightly coupled sensor sharing. Our approach, which we call ASyMTRe, maps environmental sensors and perceptual and motor control schemas to the required flow of information through the multirobot system, automatically reconfiguring the connections of schemas within and across robots to synthesize valid and efficient multirobot behaviors for accomplishing a multirobot task. We present the centralized anytime ASyMTRe configuration algorithm, proving that the algorithm is correct, and formally addressing issues of completeness and optimality. We then present a distributed version of ASyMTRe, called ASyMTRe-D, which uses communication to enable distributed coalition formation. We validate the centralized approach by applying the ASyMTRe methodology to two application scenarios: multirobot transportation and multirobot box pushing. We then validate the ASyMTRe-D implementation in the multirobot transportation task, illustrating its fault-tolerance capabilities. The advantages of this new approach are that it: 1) enables robots to synthesize new task solutions using fundamentally different combinations of sensors and effectors for different coalition compositions and 2) provides a general mechanism for sharing sensory information across networked robots.

KEYWORDS | Coalition formation; information invariants; multirobot teams; schema theory; sensor sharing; task allocation

Manuscript received June 1, 2005; revised June 1, 2006.
The authors are with the Distributed Intelligence Laboratory, Department of Computer Science, University of Tennessee, Knoxville, TN 37996 USA (e-mail: parker@cs.utk.edu; ftang@cs.utk.edu).
Digital Object Identifier: 10.1109/JPROC.2006.876933

I. INTRODUCTION

Researchers generally agree that multirobot systems have several advantages over single-robot systems [1], [5]. The most common motivations for developing multirobot system solutions are that: 1) the task complexity is too high for a single robot to accomplish; 2) the task is inherently distributed; 3) building several resource-bounded robots is much easier than having a single powerful robot; 4) multiple robots can solve problems faster using parallelism; and 5) the introduction of multiple robots increases robustness through redundancy. The issues that must be addressed in developing multirobot solutions are dependent upon the task requirements and the sensory and effector capabilities of the available robots. The earliest research in multirobot systems focused on swarm intelligence approaches using homogeneous robot teams, inspired by insect societies (e.g., [3], [28]). In these approaches, individual robots typically perform the same type of subtask in the same environment, resulting in global group behaviors that emerge from the local interaction of individual robots. The fundamental research challenge in these systems is designing the local control laws so as to generate the desired global team behavior.

Other types of robot systems involve heterogeneous robots, which have differing sensor and effector capabilities. In these teams, the mapping of tasks to robots is much more important to the efficiency of the system, since robots vary in the quality of their solutions to tasks. Traditionally, this problem has been called the multirobot task allocation (MRTA) problem. Gerkey [14] has developed a taxonomy for describing these problems, distinguishing robots as either *single-task* (ST) or *multitask* (MT), tasks as either *single-robot* (SR) or *multirobot* (MR), and assignment types as either *instantaneous* (IA) or *time-extended* (TA). The vast majority of prior work on MRTA (e.g., [4],

[8], [16], [23], [31], [45], [47], [48]) has addressed single-task robots executing single-robot tasks using either instantaneous assignment (denoted ST-SR-IA) or time-extended assignment (denoted ST-SR-TA).

In this paper, we address a different problem in the MRTA taxonomy—namely, single-task robots performing multirobot tasks using instantaneous assignment (ST-MR-IA). In other words, we are addressing the development of heterogeneous robot coalitions that solve a single multirobot task. While this problem has been addressed extensively in the multiagent community (e.g., [24], [36], [37]), it has been noted by Vig [44] that most of the multiagent approaches to coalition formation cannot be directly transferred to multirobot applications, since robot capabilities and sensors are situated directly on the robots and are not transferable between robots. Our approach is aimed at enabling sensor-sharing across robots for the purpose of forming coalitions to solve single-multirobot tasks.

More generally, multirobot *coalition formation* deals with the issue of how to organize multiple robots into subgroups to accomplish a task. *Coalitions* are typically considered to be temporary organizations of entities that bring together diverse capabilities for solving a particular task that cannot be handled by single robots. Coalitions are similar to the idea of *teams*, except that they typically have a shorter duration and can change frequently over time. We are particularly interested in automated techniques for coalition formation, especially when the specific task solution is highly dependent upon the available capabilities of the heterogeneous robots, and thus cannot be specified in advance. This is especially challenging in heterogeneous robot systems, in which sensory and computational resources are distributed across different robots. For such a group to accomplish the task as a whole, it 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 approach called ASyMTRe (which stands for "Automated Synthesis of Multirobot Task solutions through software Reconfiguration," pronounced like the word "asymmetry"), which we first introduced in [41], [42]. This approach is aimed at increasing the autonomous task solution capabilities of heterogeneous multirobot systems by changing the fundamental abstraction that is used to represent robot competences from the typical "task" abstraction to a biologically inspired "schema" [2], [27] abstraction, and providing a mechanism for the automatic reconfiguration of these schemas to address the multirobot task at hand.[1] In doing this, we are able to simultaneously obtain a number of significant new benefits in multirobot coalition formation that have previously been difficult to achieve. These benefits include: 1) enabling robots to automatically generate task solutions based on sensor-sharing across robot coalition members, in configurations not previously explicitly defined by the human designer; 2) providing a way for robots to develop coalitions to address multirobot tasks; and 3) enabling flexible software code reuse from one multirobot application to another through the task-independent schema abstraction that is viewed as a generator of semantic information content which can be combined in many ways by various diverse tasks. 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 generating a coalition to solve single-multirobot tasks. The coalitions would then compete (with other coalitions or single robots) for task assignments using the higher level, more traditional task planning/allocation strategies.

The basic ASyMTRe approach is an anytime centralized reasoner, generating multirobot coalitions using complete information, with solution quality increasing as more time is available for the reasoning process. In order to allow for increased robustness, we also present a distributed version of ASyMTRe, called ASyMTRe-D (which we first introduced in [43]), which uses communication to enable distributed formation of coalitions. This distributed version offers a tradeoff of increased robustness versus solution quality compared to the centralized version. Our ultimate objective in this research is to eventually 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.

The rest of this paper is organized as follows. Section II describes the centralized ASyMTRe solution approach. Section III analyzes the theoretical soundness, completeness, and optimality of this approach. In Section IV, we describe the distributed version of ASyMTRe, called ASyMTRe-D. Section V describes the experimental results that validate this approach. We then present a review of related work in Section VI and conclude in Section VII.

II. THE ASYMTRE APPROACH

A. Schema Theory and Information Types

The basic building block of our approach is a collection of *schemas*, inspired by the work of Lyons and Arbib [27] and Arkin [2], which first applied schema theory to cognitive and agent systems. In [27], a formal model of computation is constructed, called robot schemas. In this earlier work, the schema includes a list of input and output ports, a local variable list, and a behavior, which defines how the input is processed to generate the output. A network of schemas can be built by manually connecting

[1]Our approach does not necessarily require a schema implementation, and could alternatively be implemented using traditional "behaviors." We selected the schema approach because schemas tend to be more fine-grained and less sensor- and task-specific than behaviors. Additionally, schemas have a more formal, commonly accepted interface definition, as defined by [27], which facilitates the formal ASyMTRe specification.

the outputs of one schema to the inputs of another schema. At the higher level, a nested network is established to represent the collaboration among robots. Arkin [2] further develops these ideas by presenting schema-based control for mobile robots. In his approach, perceptual schema interpret sensory stimuli, feeding the results to one or more motor schemas. Computations from multiple motor schemas are summed and normalized to generate the overall robot behavior.

Based upon this prior work, the fundamental building blocks of our approach are collections of environmental sensors (ES), perceptual schemas (PS), and motor schemas (MS). Our ASyMTRe approach extends this prior work by introducing a new component, called *communication schemas* (CS). Perceptual schemas process input from environmental sensors to provide information to motor schemas, which then generate output control vectors representing the way the robot should move in response to the perceived stimuli. Communication schemas transfer information between various schemas distributed across multiple robots. All of these schemas are assumed to be preprogrammed into the robots at design time, and represent fundamental low-level capabilities of individual robots.

Our ASyMTRe approach further extends this prior work on schema theory by autonomously and dynamically connecting the schemas at run time instead of using predefined (manual) connections. In our approach, schemas are situated in each robot, but are not connected to each other at the beginning of a task. Instead, they are configured using our automated ASyMTRe approach. Our automation is based upon recognizing and defining the fundamental information that is needed to accomplish the task. The information needed to automatically activate a certain schema remains the same regardless of the way that the robot may obtain or generate it. Thus, we label inputs and outputs of all schemas with a set of information types that are unique to the task. Note that we use the term *information types* as distinct from *data types*. This distinguishes our approach from Lyon [27]. *Information types* have semantic meaning and define the specific sensing or computational data of a schema or a sensor, such as the *global position* of a robot, rather than just the format of the data. Semantics of the information is built into these information types, and does not just refer to a data type (such as boolean or integer). These information labels provide us a method for automating the interconnections of schemas, enabling robots to share sensory and perceptual information as needed to accomplish the multirobot task.

We define the inputs and outputs of the schemas to belong to the set of information types $F = \{F_1, F_2, \ldots\}$. For schema S_i, I^{S_i} and $O^{S_i} \subset F$, represent the input and output sets of S_i, respectively. As in [27], we assume that each schema has multiple inputs and outputs. There are two types of inputs to a schema (see Fig. 1). The solid-line arrows entering a schema represent an "OR" condition,

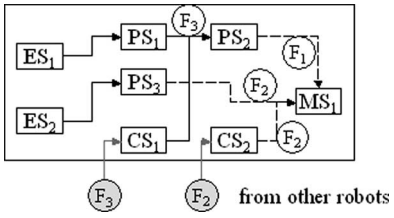


Fig. 1. ***An example of how the schemas are connected to accomplish a task. (For clarity, we have eliminated the superscripts in this figure.)***

Table 1 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

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, in Fig. 1, 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. An output of a schema can be connected to an input of another schema if and only if their information types match. Using the mapping from schemas to information types, robots can collaborate to define different task strategies in terms of the required flow of information in the system. Once the interconnections between schema are established, the robots have executable code to accomplish their task.

Given a set of n robots and a task T, the solution configuration problem can be represented as (R, T, U), where $R = \{R_1, R_2, \cdots, R_n\}$ is the set of n robots, $T = \{MS_1, MS_2, \cdots\}$ is the set of motor schemas that define the group-level task to be achieved, along with application-specific parameters as needed,[2] and U provides utility information to be defined later. 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, where $O^{ES^i_j} \subset F$ is the output of ES^i_j (that is, the jth ES on robot R_i). S^i is the set of schemas that are preprogrammed 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 another schema or sensor. A set of *Connection Constraints* regulate the connections between schemas. As shown in Table 1, these constraints specify the restrictions on correct connections between various schemas.

[2]More complex task definitions, including task sequences, can be defined in a manner similar to the formal, schema-compatible, specification of tasks in [13].

Given the above information, the problem (R, T, U) has a solution if and only if for each $R_i \in R$ and for all $\mathrm{MS}_j \in T$, the inputs of MS_j are satisfied, along with all the inputs from the schemas that feed into MS_j. A solution is a combination of schemas that can satisfy the above requirements.

B. 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 $\mathrm{MS}_j \in T$, 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

$$\mathrm{PoS}_j^i = \left(S_1^i, S_2^i, \cdots, S_k^i, F_1^i, F_2^i, \cdots, F_h^i\right) \tag{1}$$

where PoS_j^i is the jth potential solution for R_i, S_x^i $(1 \le x \le k)$ is the xth schema of R_i that needs to be activated, and F_y^i $(1 \le y \le h)$ is the yth information type that needs to be transferred to R_i. For example, in Fig. 1, if we assume that $T = \{\mathrm{MS}_1\}$, one potential solution is to activate $\{\mathrm{PS}_1, \mathrm{PS}_2, \mathrm{PS}_3, \mathrm{MS}_1\}$, provided that the robot has both ES_1 and ES_2. Another potential solution is to activate $\{\mathrm{PS}_2, \mathrm{CS}_1, \mathrm{CS}_2, \mathrm{MS}_1\}$ when F_3 and F_2 can be transferred from other robots.

C. Solution Quality

With multiple potential solutions available, we introduce *utility* to measure their qualities. We define a *sensori-computational system* (SCS) [10] as a module that computes a function of its sensory inputs and produces outputs. An SCS_j^i is formally represented by $(S_j^i, \mathrm{ES}_j^i, O^{S_j^i})$, where S_j^i is the jth PS/CS on R_i, ES_j^i is the sensory input, and $O^{S_j^i}$ is the output. Each SCS_j^i is assigned a cost C_j^i and a success probability P_j^i, where C_j^i represents the sensing cost of using ES_j^i and P_j^i represents the success rate of S_j^i to generate a satisfactory result. The success probabilities would typically come from a learning process that performs task monitoring during execution, similar to the ideas described in [33]. We calculate the utility[3] of activating SCS_j^i or producing $O^{S_j^i}$ by U_j^i

$$U_j^i = \max\left(0, w \cdot P_j^i - (1-w) \cdot \left(C_j^i / \max_j\left(C_j^i\right)\right)\right). \tag{2}$$

Here, w $(0 \le w \le 1)$ is a weight factor that balances the relative importance of the success probability and the relative cost. We measure the quality of a potential solution PoS_j^i by summing the utilities of all the SCS_j^i 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.

D. Potential Configuration Space (PCS)

When a group of robots is brought together to accomplish a task, each with its unique sensors and corresponding schemas, the brute force way to solve the problem is to perform an exhaustive search of all the possible combinations of schemas within or across robots. We refer to this complete space as the original configuration space (OCS). However, the number of possible connections in the OCS is exponentially large in the number of robots. In the general case, the robots will be developed separately, and it is highly possible that the schemas with the same functionality are represented differently on different robots. By definition, schemas S_i and S_j are of the same functionality, $\mathrm{func}(\mathrm{S}_i) = \mathrm{func}(\mathrm{S}_j)$, and are thus in the same equivalence class, if and only if $I^{S_i} = I^{S_j}$ and $O^{S_i} = O^{S_j}$, and they have the same success probability and sensing cost. To reduce the size of the search space, we generate a reduced configuration space, called the potential configuration space (PCS), by including only one entry for each equivalence class of schema. The extent of the size reduction that we achieve depends upon the specific robot group composition and the degree of overlap in their capabilities. Step I in Fig. 2 shows an example of reducing the configuration space. Assume, without loss of generality, that the robot group is composed of n robots, each of which has h schemas, and each schema requires k inputs. The OCS is of size is $O((nhk)^{nh})$. After the reduction, the PCS is of size $O((h'k)^{h'})$, where h' is the number of equivalence classes, and $h' \le nh$. This

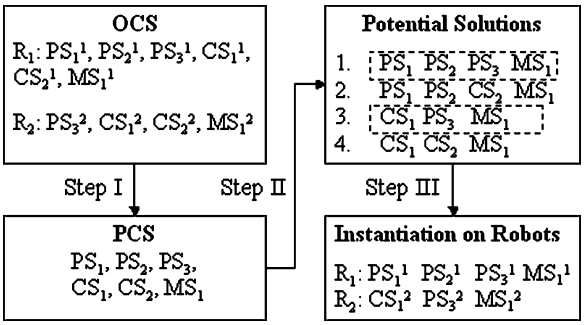


Fig. 2. *The ASyMTRe solution process. Step I: Reduce the OCS to the PCS by creating one entry per schema equivalence class. Step II: Generate the list of potential solutions based on the schemas in the PCS. Step III: Instantiate the selected potential solutions on specific robots. Here, we assume* $T = \{\mathrm{MS}_1^1, \mathrm{MS}_1^2\}$, $\mathrm{func}(\mathrm{PS}_k^i) = \mathrm{func}(\mathrm{PS}_k^j)$, $\mathrm{func}(\mathrm{CS}_k^i) = \mathrm{func}(\mathrm{CS}_k^j)$, *and* $\mathrm{func}(\mathrm{MS}_k^i) = \mathrm{func}(\mathrm{MS}_k^j)$.

[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.

analysis assumes that every output of any schema can be a potential input of any schema. In practice, the size of the OCS and PCS are even smaller because of the connection constraints shown in Table 1. The conversion from the OCS to the PCS not only reduces the size of the entire search space, but also discards the duplicated solutions. As an example, consider a case of n homogeneous robots that have the same p schemas. Then, the OCS would have np schemas, but the PCS would only have p schemas, since the duplicates would be removed from the PCS. Thus, in practice, the reduction can have a significant impact on the search space size, although theoretically, the search space could still be exponential.

Theorem 1: If a solution exists in the OCS, it must exist in the PCS.

Proof: Assume to the contrary that one of the solutions present in the OCS does not exist in the PCS. This could only occur if at least one schema S_i in the solution does not exist in the PCS, which means there exists an S_i in the OCS, but there does not exist an S_j in the PCS such that $\text{func}(S_i) = \text{func}(S_j)$. According to the definition of the PCS, there will be one entry for each equivalence class of schema. Thus, there cannot be a solution in the OCS that does not exist in the PCS. ■

E. The ASyMTRe Configuration Algorithm

After the PCS is generated, the ASyMTRe configuration algorithm searches through the PCS to generate a list of *potential solutions* for each robot, which are the exhaustive combinations of schemas (within or across the robots) that could be connected for a single robot to accomplish its part of the task. Note that the list of potential solutions are not for the entire task, but are ways to connect schemas in the robot coalition in order for some individual robot to contribute to the task. During the configuration process, the algorithm searches through the list of potential solutions to assign the best solution for each robot. As shown in Theorem 1, the algorithm will not miss any solutions by searching in the PCS. Once solutions are found in the PCS, we must map them back to the OCS and instantiate them on robots by translating the schemas in the solutions into robot-specific schemas. Steps II and III in Fig. 2 demonstrate an example of the instantiation process.

The configuration process involves greedily searching the list of potential solutions and selecting the solution with the maximum local utility from each robot's perspective (U_i). Our goal is to maximize the total utility $\sum_i U_i$. Similar to [37], we assume robots work in a nonsuperadditive environment. Thus, the larger a coalition is, the higher the communication and computation costs are. Therefore, in practice, we impose a *maximum cooperation size* constraint on the algorithm, to reduce the complexity of the robots executing the resulting solution. In addition, we impose heuristics on the search process in the form of robot orderings, which tend to guide the search towards good solutions quickly. The first ordering sorts robots according to increasing robot sensing capabilities. Less capable robots (whose motor schemas must be activated in the solution, as defined by the task definition) are considered first because finding a solution is relatively easy when the sensing resources are abundant. The second ordering sorts robots by the number of other robots with which they can cooperate, as calculated during the first search process. This number represents the number of robots that can provide the information that is needed by the current robot. Thus, robots with fewer other robots to cooperate with will have the chance to find their collaborators earlier in the search process.

We formulate our approach as an *anytime algorithm* [46] that provides a satisfactory answer within a short time, with its solution quality increasing over time. In our domain, the algorithm first generates an ordering of robots with which the algorithm can configure solutions. If more time is available, another ordering of robots is selected sequentially and the above procedure is repeated until the deadline is reached. The algorithm will report the solution with the highest utility it has found so far or report no solution if a solution is not found. Since we have a finite number of robots and a finite solution space, the algorithm will ultimately find the optimal solution if there exists one, given sufficient search time. All of the previously described aspects of the ASyMTRe approach are combined to yield the centralized ASyMTRe configuration algorithm shown in Table 2.[4]

III. ASyMTRe ALGORITHM ANALYSIS

We now analyze the soundness, completeness, and optimality of the ASyMTRe configuration algorithm.[5] Here, *soundness* is the proof of the correctness of the generated solutions with respect to the environmental setting. *Completeness* is the guarantee that a solution will be found if it exists. *Optimality* is ensuring that the system will select an optimal solution.

A. Soundness

Theorem 2: The ASyMTRe configuration algorithm is correct.

Proof: Assume to the contrary that the algorithm is not correct. This could occur in two ways: 1) at least one of the connections made is not a valid connection and 2) not

[4]Note that other search techniques to find a solution, such as evolutionary approaches, could also be used here. Exploring the tradeoffs with other search approaches is a subject of our future research.

[5]Due to the similarity between our configuration algorithm and the coalition formation algorithm presented in [37], we plan to analyze the bounds on our solution quality in future work. It has been proved in [37] that similar algorithms are of low logarithmic ratio bounds to the optimal solution.

Table 2 The ASyMTRe Configuration Algorithm

The ASyMTRe Configuration Algorithm ((R, T, U), Deadline)
(R, T, U): the robot group, task and utility
n: the number of robots in the group
m: the number of configurations to accomplish the task
k: a constant, which specifies the number of iterations

1) Generate all sequential orderings of the robots.
2) Generate the PCS based on equivalence classes.
3) Generate a list of potential solutions (size m) by connecting schemas in the PCS to satisfy task-requirements.
4) Sort the robots according to increasing sensing capabilities.
5) Configure solutions on the robots:
   - For each robot R_i in current ordering:
     - For each potential solution j:
       - If R_i can accomplish the task by itself, assign solution j to R_i.
       - Else check the other $n-1$ robots to see if one can provide the needed information.
       - If the estimated utility U_i of R_i using solution j is greater than the utility of its previous solution, and the constraints on cooperation size are satisfied, update the solution strategy on R_i.
   - Continue the above process until:
     - All robots can perform the task.
     - Or, after k number of trials.
   - Calculate the coalition utility U. If U is greater than the utility of previous solutions, update the coalition.
   - If more time is given: If the special ordering (increasing number of robots that one can cooperate with) has not been checked, use this ordering; otherwise, select an ordering sequentially; repeat the above process.
6) If a solution exists, report the current best solution. Otherwise, report "Failure".

all needed connections are made to compose a solution. We argue that these two cases will not occur.

First, an output of a schema S_i can be connected to an input of a schema S_j if and only if $O^{S_i} = I_j^{S_k}$, which means the output of S_i has the same information type as one of the inputs required by S_j. Since we know all the input and output information types for every schema, and we maintain the connection constraints that specify the types of schema connections that the system allows, it is not possible to generate an invalid connection.

Second, a potential solution is a combination of schemas such that the inputs for each $MS_i \in T$ are satisfied, along with all the inputs from the schemas that feed into MS_i. Since the algorithm complies with this standard, all needed connections will be made to become a potential solution. Thus, a solution generated by ASyMTRe will be a correct solution. ■

B. Completeness and Optimality

Theorem 3: The ASyMTRe configuration algorithm is complete and optimal given enough time.

Proof: We prove completeness and optimality by showing that the algorithm can sequentially search the entire solution space given enough time. Step 5 (in Table 2) shows the major configuration process in which, given an ordering of robots, the algorithm searches through the list of potential solutions to find solutions for every coalition member. Note that the ordering is sequentially selected rather than randomly generated. If the algorithm is given enough time, it will ultimately test all possible orderings of robots, which is $O(n!)$, and reports the solution with the highest utility. Since the solution space is eventually explored in its entirety by the algorithm, it is complete and optimal, given enough time. ■

Despite this proof, we note that in reality, practical constraints require a fast response, especially when dealing with failures that require quick reconfiguration. Since the coalition formation problem can be cast as an instance of the Set Partitioning Problem [14], we know that the coalition formation problem is NP-hard, with worst case time complexity of $O(n!)$. Therefore, it is not possible to generate an optimal solution in practice when n is large. However, in our experiments, the algorithm consistently returns a good coalition solution within a few seconds, as will be presented in Section V-D.

IV. THE DISTRIBUTED ASyMTRe-D APPROACH

In ASyMTRe-D, the sharing of information, and thus the cooperation among robots, is achieved through a distributed negotiation process, based on the Contract Net Protocol [39]. Each robot decides what information it needs and then requests this information 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 group composition is dynamic. The negotiation process is triggered at the beginning of each task to generate initial solution strategies, and is called to replan solutions to accommodate changes in the robot group or task. It is important to note, however, that the distributed approach trades off solution quality for robustness. The intent of this approach is not to develop a new negotiation protocol, but instead to develop a method for the robot group to vary their reasoning between fully centralized and fully distributed decision-making, according to the desired balance between solution quality and robustness. As we show in our empirical studies in Section V-D, the time requirement for distributed solutions scales better than the centralized approach as the robot team size increases.

A. Distributed ASyMTRe-D Negotiation Protocol

The distributed negotiation protocol involves the following major steps.

- **Make request**. Depending on the requirements of each potential solution, a robot broadcasts requests for the information types it needs from other robots. These requests are either *simple* requests or *complex* requests. Simple requests ask other robots if they can provide particular information types. These requests are sent out at the beginning to estimate the potential number of robots (pn) that can provide the required information. Each robot will wait for a period that is proportional to its pn value before sending out the complex requests, which ask for utility information for supplying the requested information types. Thus, the robots with fewer potential helpers have higher priorities to make requests, since they will likely have fewer chances for success. 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 in a first-come-first-served (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. Because we assume robots work in a nonsuper-additive environment, 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 resulting 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. 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.

To ensure a general and robust negotiation process, some traditional mechanisms are built into the distributed protocol [9]. First, our protocol employs timeouts during the negotiation process. The 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 for any replies, and if there is no reply, it will send out requests again. This process will continue for a period 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 [16], 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.

V. EXPERIMENTAL RESULTS

We have validated the ASyMTRe approach using a series of experiments both in simulation and on physical robots to demonstrate the solution generation process of both the centralized and the distributed approaches and the performance comparison between them. We used Player/Stage [15] as the simulation interface and ActivMedia Pioneer 3 mobile robots in the physical experiments. In this section, we present our experimental approaches and analyze the results.

A. Multirobot Transportation Through Centralized ASyMTRe

1) Task Description: In this application (which we first reported in [6] and [35]), a group of robots must navigate from their starting positions to a set of goal positions (one per robot) defined in a global coordinate reference frame. Assume that all the robots are programmed with the motor schema *go-to-goal*, which moves the robot from its current position to a goal position, defined in a global coordinate reference frame. To successfully use this motor schema, a robot must know its own current position relative to its goal. If every robot can localize itself, obviously the solution is to have every robot navigate independently. However, in some groups, there might be robots that do not have the sensing, effector, and algorithmic capabilities to localize (e.g., see [34]); they need help from more capable robots to provide the information needed to fulfill the task. In the following paragraphs, we detail the application by introducing the environmental sensors and various schema used in this application.

The environmental sensors are: laser scanner with an environmental map (`laser`), omnidirectional camera (`camera`), and `DGPS`, providing up to 2^3 different combinations of robot capabilities, as shown in Table 3. All robots also have a communication sensor, (`comm`). We assume: 1) a robot with a laser or DGPS can estimate its current global position in the environment; 2) a robot with a camera or laser can estimate the relative position of another robot in the environment, as long as the other

Table 3 Eight Types of Robot With Different Sensing Capabilities

Type	Available Sensor(s)
R_1	comm
R_2	DGPS, comm
R_3	laser, comm
R_4	camera, comm
R_5	DGPS, laser, comm
R_6	DGPS, camera, comm
R_7	laser, camera, comm
R_8	DGPS, laser, camera, comm

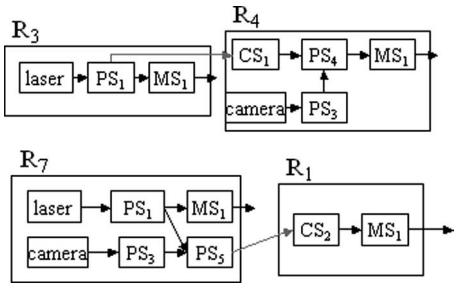


Fig. 3. *Two ways to connect the schemas in the transportation task.*

robot is within its sensing range; and 3) a robot has the computational ability to convert a relative position to a global position.

We have implemented the following schemas on the robots: PS_1, which estimates global position using laser (with an environmental map) or DGPS; PS_2, which gives the goal position; PS_3, which estimates the relative position of another robot using camera or laser, and fiducial marker; PS_4, which estimates self global position according to another robot's global position and relative position; PS_5, which estimates the global position of another robot according to its own global position and the estimated relative position of the other robot; CS_i, which transfers information between robots; and MS_1, which calculates motor commands that lead the robot toward the goal. We define the task, $T = \{MS_1\}$, meaning that MS_1 should be active on all coalition members.

In Table 4, we define the set of information types F and label the input and output information for each schema used in this application. According to the flow of information, the centralized configuration algorithm generates all the possible connections that can connect the available schemas and lead the robot to achieve its goal. Two specific connections are shown in Fig. 3. The first solution involves R_3 using the laser to localize and then communicating this information to R_4. R_4 combines the received information with the detected relative position of R_3 (using its camera) to calculate its own global position. The second solution involves R_7 using its laser to globally localize itself and using the camera to calculate the relative position of R_1. With this information, R_7 can calculate the global position of R_1 and communicate this information to R_1.

With multiple solutions available, a robot needs to determine which solution it should use. This is decided by each robot's sensing cost and the estimated success rate of the particular solution it chooses. Typically, the sensing cost is determined by the sensory and computational requirements of the solution. Perceptual processes with a significant amount of sensor processing, such as laser scan matching or image processing, are given higher sensing costs. Perceptual processes with a relatively low processing requirement, such as DGPS, are assigned lower sensing costs. Success probability is an estimated value based upon learning and experience. Perceptual processes that are easily influenced by environmental factors, such as image processing under different lighting conditions, are given lower success probabilities. Here, we only provide fuzzy estimates for costs and probabilities (see Table 4); in actual applications, these estimates will likely be specific numeric values.

Table 4 Input and Output Information Types for Various Schemas and Their Corresponding Sensing Costs and Success Probabilities

Type	Description
F_1	Self_Global_Position
F_2	Other_Global_Position
F_3	Other_Relative_Position
F_4	Goal_Position
F_5	Motor_Commands

S_i	I^{S_i}	O^{S_i}	C_i	P_i
PS_1	Laser or DGPS	F_1	High Low	High High
PS_2	Hardcoded	F_4	none	none
PS_3	Camera or Laser	F_3	Medium High	Medium Medium
PS_4	F_2 and F_3	F_1	Medium	Medium
PS_5	F_1 and F_3	F_2	Medium	Medium
CS_1	F_1	F_2	Low	High
CS_2	F_2	F_1	Low	High
MS_1	F_1 and F_4	F_5	none	High

2) Physical Robot Implementation: Several of the schemas described above were implemented on two Pioneer robots equipped with a SICK laser range scanner and a Cannon pan-tilt-zoom camera. Both robots possess a wireless ad hoc networking capability, enabling them to communicate with each other. Experiments were conducted in a known indoor environment using a map generated using an autonomous laser range mapping algorithm. Laser-based localization used a standard Monte Carlo localization technique. The implementation of PS_3 makes use of prior work in [34] for performing vision-based sensing of the relative position of another robot. This approach makes use of a cylindrical marker designed to provide a unique robot ID, as well as relative position and orientation information suitable for a vision-based analysis. Using these two robots, three sets of experiments based on sensor availability were tested to illustrate the ability of these building blocks to

Fig. 4. *Physical robot implementation of the transportation task. The initial setup of the two robots is on the left. The result when the two robots reach their goal points (indicated by the squares on the floor) is shown on the right.*

generate fundamentally different cooperative behaviors of the same task through sensor sharing.

Experiment set 1 is a baseline case in which both robots have full use of their laser scanner and camera. Each robot localizes itself using its laser scanner and reaches its own goal independently. Experiment set 2 involves a fully capable robot R_3 with laser, as well as a robot R_4 with only a camera. They automatically connect their schemas according to the first solution shown in Fig. 3 to accomplish the task. Experiment set 3 involves a sensorless robot R_1 with communication capabilities only and a fully capable robot R_7. They automatically connect their schemas according to the second solution shown in Fig. 3. A snapshot of these experiments is shown in Fig. 4.

In extensive experimentation, data on the success rate was collected with an average of ten trials of each set. Robots in experiment set 1 were 100% successful in reaching goal positions. Experiment set 2 had four failures and set 3 had one failure. The failures were caused either by: 1) variable lighting conditions that led to a false calculation of the relative robot positions using the vision-based robot marker detection or 2) the robot not being in the field of view of the observer. However, even with these failures, these overall results are better than what would be possible without sensor sharing. In experimental sets 2 and 3, if the robots did not share their sensory resources, one of the robots would never reach its goal position, since it would not have enough information to determine its current position. Thus, our sensor sharing mechanism extends the ability of the robot group to accomplish tasks that otherwise could not have been achieved. To increase the robustness of the application, once a failure is detected, the ASyMTRe reasoning process can be called to reconfigure solutions for the available robots. We discuss this further in Section V-C2 with ASyMTRe-D.

B. Cooperative Box Pushing Through Centralized ASyMTRe

1) Task Description: Cooperative box pushing was studied by Donald *et al.* [11] in the development of information invariants for analyzing the complexity of alternative cooperation algorithms. To illustrate the connection of our approach to the theory of information invariants, we have defined our box pushing experimentation in a similar manner to [11]. The goal is to organize the group into coalitions, such that each coalition is able to push a box with exactly two robots,[6] which we call pusher robots. Assume that all the robots are programmed with the motor schema *Push* (MS_1), which pushes the box along a straight line. To use this motor schema, a robot must be able to perceive the box's vector relative to itself. Here, the relative vector includes the applied force, relative displacement, the angle between the actual pushing direction and the line of pushing. For example, the two pusher robots can record the relative displacements while they are pushing the box; by comparing these two values, they can decide which robot should push harder. We define five environmental sensors for this task, as follows: `laser`, `camera`, `gripper`, `bumper`, and `odometry`. These sensors define 2^5 possible robot capabilities. Three methods were presented in [11] to push a box with two robots. In addition, we generate a fourth method in which the pusher robots do not have the capabilities to perceive the relative vector of the box. The alternative cooperative box pushing methods are as follows.

1) Using bumper, two robots compute their applied forces (PS_1) and share this information with each other, allowing them to decide which robot should push harder.
2) Using odometry, two robots compute the relative displacements (PS_2) along the line of pushing; they exchange the location information and take actions to reduce the difference between the relative displacements.
3) Using grippers, two robots compute the angles (PS_3) between the line of pushing and its actual moving direction; they take actions to reduce its angle on the next step—no explicit communication is needed.
4) Using laser or camera, a helper robot can help the pushers calculate the relative vector of the box (PS_4), enabling the pusher robots to take actions. To do this, the helper robot needs to move along with the box (MS_2) and keep the pushers in its field of view.

The information set in the box pushing application is $F = \{\text{force,displacement,angle,box\_relative\_vector}\}$. The schemas and their input and output information are defined in Table 5. Fig. 5 shows one of the connections that enables the robots to push a box. To calculate the utility, we assume that the sensing costs and success probabilities for {`gripper`, `bumper`, `odometry`} are the same. The

[6]This can be generalized to an arbitrary number of robots.

Table 5 Perceptual Schemas, Communication Schemas, and Motor Schemas in Multirobot Box Pushing

Schema	Input	Output
PS_1	Bumper	force
PS_2	Odometry	displacement
PS_3	Gripper	angle
PS_4	Laser or Camera	box_relative_vector
CS_1	F_i	F_i
MS_1	force, displacement, angle, or box_relative_vector	Motor commands
MS_2	box_relative_vector	Motor commands

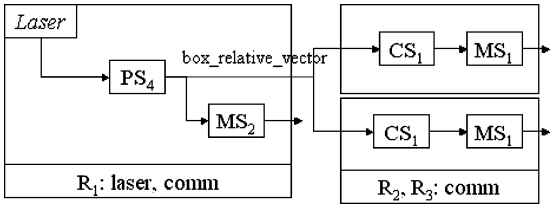


Fig. 5. *One way of connecting the schemas to accomplish the "Push" goal. This solution involves R_1 using its laser to perceive the relative vector of the box, and then communicating this information to the pusher robots.*

Table 6 Box Pushing: Robot Coalition Composition

Robot	Environmental Sensor(s)
R_1	comm
R_2	bumper, comm
R_3	odometry, comm
R_4	gripper, comm
R_5	laser, camera, comm
R_6, R_7	bumper, gripper, odometry, comm

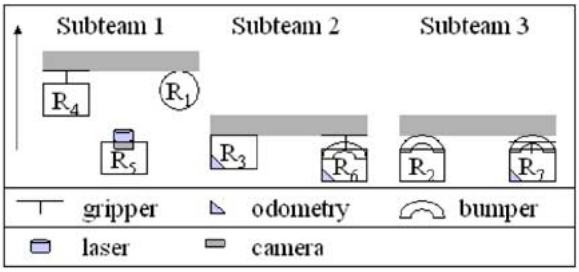


Fig. 6. *Results of applying ASyMTRe to an instance of the box pushing task, in which the robot group is composed of seven robots from six different types. The group is autonomously divided into three coalitions; each heterogeneous coalition can successfully push a box.*

`laser` and `camera` values are the same as the multirobot transportation application.

2) Results of Box Pushing: To demonstrate how the box pushing task can be achieved, we selected a group of seven robots from six different types, as shown in Table 6. Fig. 6 presents the results, in which the group is divided into three coalitions. In coalition 1, robots R_1 and R_4 push the box together. Since R_1 cannot perceive the box's relative vector by itself, they are helped by R_5 using either laser or camera (with the specific choice depending on the utility). In coalition 2, robots R_3 and R_6 push the box together using method 2, since they both have odometry. In coalition 3, robots R_2 and R_7 push the box together using method 1, since they both have bumper.

C. Multirobot Transportation Through ASyMTRe-D

The previous experiments on the centralized ASyMTRe approach have validated the correctness of the ASyMTRe approach for automatically generating schema connections within or across robots. Now, we evaluate the ASyMTRe-D approach through more complex experiments, in order to demonstrate two aspects of the ASyMTRe-D approach: the continuous reasoning capabilities of the robot group over a period while performing a more complex task, and the robustness of the approach.

1) Simulation Setup and Results: To illustrate how ASyMTRe-D can be used in more complex tasks requiring new coalitions over time, we implemented the transportation task in simulation, using a larger number of robots. In this task, robots are randomly assigned a series of goal positions to visit.[7] In some cases, robots are assigned the same goal position. Once a subgroup of robots has been assigned their goal positions, the robots form coalitions as needed using ASyMTRe-D. When all the robots accomplish their current tasks (i.e., visit their assigned goal positions), new positions are assigned and new coalitions are formed as needed. In the experiment reported in Fig. 7, only two robots out of the group of seven can navigate independently to goal positions. The remaining robots need navigation assistance, as described previously in the earlier version of the transportation task. The task priority for robots in this application is: 1) help robots in the group that share the same goal position; 2) help robots in any other group; and 3) navigate to robot's own goal position.

Fig. 7 is one of the typical runs of the transportation task in simulation. Here, we generated two random goals

[7]Note again that we are not addressing the higher level assignment of single-robot tasks (i.e., goal positions) to multiple robots. This issue is addressed by other task allocation approaches fitting the ST-SR-IA and ST-SR-TA taxonomic category. Here, we assume that other task allocation approaches would determine the assignments of goal positions to robots. We focus instead on how robots can repeatedly form different coalitions to enable individual robots to reach their assigned positions.

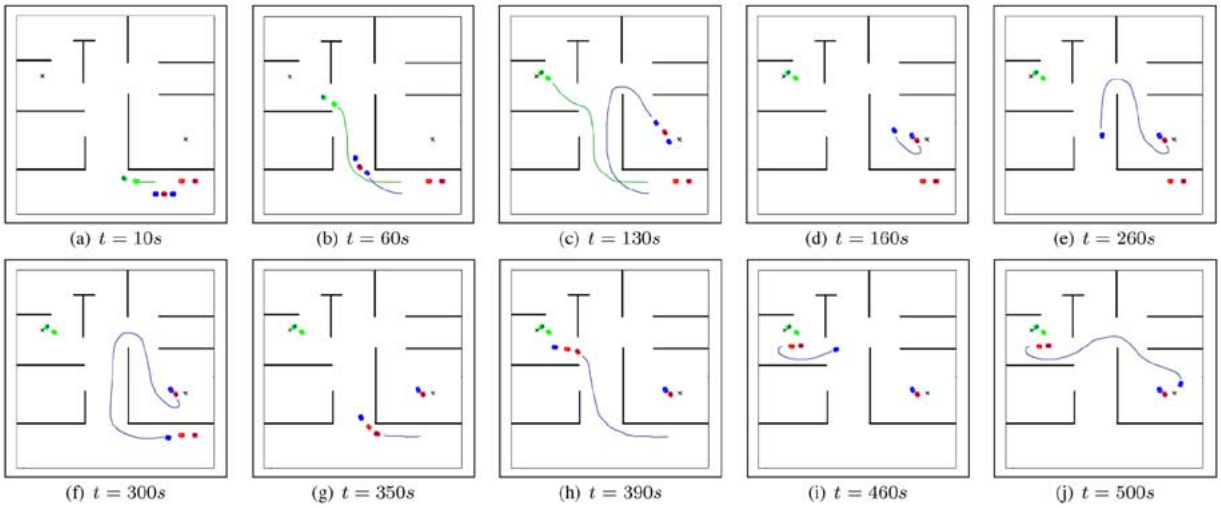


Fig. 7. *The simulation results of a more complex multirobot transportation mission, where robots are given successive goal positions. (a)–(c): the first group (four robots) configures the solutions, two robots start out to go to the position on the left (represented by the cross), while two robots wait at the starting positions for help; the other group configures solutions and starts out moving to the position on the right. (d)–(f): after the third round of configuration that involves the two capable robots and the rest of the robots that still need help, the leader robot in the second group (right) goes back to pick up the two robots. (g) and (h): the other two robots follow the leader to their goal position. (i) and (j): the leader robot goes back to its original goal. (a)* $t = 10$ *s. (b)* $t = 60$ *s. (c)* $t = 130$ *s. (d)* $t = 160$ *s. (e)* $t = 260$ *s. (f)* $t = 300$ *s. (g)* $t = 350$ *s. (h)* $t = 390$ *s. (i)* $t = 460$ *s. (j)* $t = 500$ *s.*

for the robot group, assigning the first goal position to a group of three robots, and the other goal position to a group of four robots. As shown in this figure, each of the more capable robots leads less capable robots to their goal positions. One of the more capable robots then returns to form a second coalition of robots for reaching their goal positions. Any introduction of a new position or the failure of a robot to find help will trigger the ASyMTRe-D negotiation process to configure new coalitions. This process continues repeatedly as new position assignments are made.

2) Physical Experimental Setup and Results: In order to evaluate the robustness of the ASyMTRe-D approach, we designed three different sets of experiments on a group of physical robots. According to the different categories of failures in a typical experiment defined by [9], we illustrate how the ASyMTRe-D approach will help the coalition recover from partial/sensor failure, or complete robot death. The robot group is composed of the following types of robots: R_1 with a laser-scanner and an environmental map for the robot to localize, and a camera mounted backward to calculate the global position of another robot (within its field of view) based on marker recognition; R_2 with a camera mounted in the front to track the marker within its field of view; R_3 with no sensors. We also assume that all robots have communication capabilities. To accomplish a task, robots of the coalition must navigate from a starting position to a goal position.

We conducted three sets of experiments to illustrate the fault-tolerant capabilities of ASyMTRe-D. In experiment set 1, R_1 and R_2 form a coalition such that R_2 follows R_1 by tracking the marker mounted on R_1. There are no failures introduced in this set. In experiment set 2, R_1 and R_2 still form a coalition with R_2 tracking R_1. During the execution, the camera on R_2 is covered such that it cannot detect R_1 anymore. This sensor failure triggers the reasoning process to generate new solutions for the two to accomplish the goal. The new solution is for the leading robot R_1 to calculate R_2's global position and communicate this information to R_2. An example of this experiment is shown in Fig. 8. In experiment set 3, R_1 and R_2 form a coalition at the beginning, then during execution, a simulated robot death is introduced on R_2, which triggers the whole group to reconfigure solutions. The new solution is that R_1 goes back to pick up R_3, and they navigate together to the goal position. An example of this experiment is shown in Fig. 9.

We performed ten successful trials for each experiment set, and collected data on the completion time (reported in Figs. 8 and 9) and the number of successful trials. In total, we have performed 32 trials, with one failure in set 2 and 3, respectively. Both failures happened because of false marker detection data when the leader robot tried to guide the follower robot. More fault detection mechanisms can be built into the application to increase its robustness, such as monitoring group members to detect their faults [16]. These experiments show the software reconfigurability of ASyMTRe-D upon failures.

Fig. 8. *Partial robot failure. In this experiment, the leader robot R_1 uses its laser to navigate and the follower robot R_2 uses its camera to follow the marker on the leader. During task execution, the camera on R_2 is covered (as indicated by the arrow), and the coalition reconfigures to continue the task. The new solution is that R_1 uses its camera to guide R_2 to the goal. Over ten trials, the average time for these experiments was 112.6 s, with standard deviation of 2.63 s, compared to the no-failure experiment time average of 96.1 s, with standard deviation of 1.45 s.*

D. Comparison Between Centralized/ Distributed ASyMTRe

The above experiments present the example operational results of applying ASyMTRe to various multirobot applications and the robustness of the ASyMTRe-D approach. The following experiments explore the performance of both approaches and compare their scalabilities and solution qualities by varying the size of the robot group (n). First, we define a simple case where the group is composed of five robots with various capabilities (see Table 3) to accomplish the transportation task. We then increased the group size by duplicating the robots with the same set of capabilities in the simple case. We ran these instances on both centralized and distributed ASyMTRe and collected data on the reasoning time and the overall coalition utility. In centralized ASyMTRe, the total time for generating a solution includes: the time to generate all the orderings of robots, which increases exponentially ($O(n!)$), and the actual reasoning time ($O(mn^2)$). In ASyMTRe-D, the time is the average reasoning time ($O(mn)$) for the group to generate a solution. Here, m is the solution size. For these experiments, the negotiation timeout values are set as follows: wait for reply: 0.75 s; try repetitive requests: 10 s; and wait for confirmation: 4 s. As shown in Fig. 10, the average time to generate a solution increases as the robot group size increases, linearly for ASyMTRe-D, but exponentially for centralized ASyMTRe, but the coalition utility does not vary significantly with varying group sizes. Additionally, in Fig. 11, the coalition utility is plotted for four different coalition sizes. In centralized ASyMTRe, the coalition utility increases as more time is given because of the anytime aspect of the centralized reasoning algorithm. Additionally, the centralized results always have a higher utility than that of the ASyMTRe-D, since the centralized approach operates with complete information.

When comparing centralized ASyMTRe with ASyMTRe-D (see Table 7), we observe that ASyMTRe-D is robust and flexible with little maintenance of the knowledge base since any change in the team capabilities only needs to be updated locally. However, ASyMTRe-D trades off its solution quality because of the local greedy

Fig. 9. *Simulated robot failure. In this experiment, the leader robot R_1 uses its laser to navigate and the follower robot R_2 uses its camera to follow the marker on the leader. Then, there is a simulated failure of R_2 and the coalition reconfigures the solution. The new solution is that R_1 goes back to pick up another follower R_3 and guide it to the goal. Over ten trials, the average time for these experiments was 187.1 s, with standard deviation of 5.45 s.*

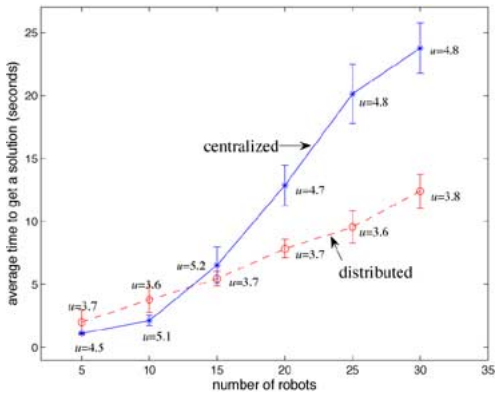


Fig. 10. *The average time increases linearly for ASyMTRe-D, but exponentially for centralized ASyMTRe with increasing team size. The coalition utility is also shown on each data point given the specific group size and reasoning time. The computation of the time and utility is averaged over ten samples. The standard deviation of the utility is 0.3 with an average of 3.7 for ASyMTRe-D, and is 0.2 with an average of 4.8 for centralized ASyMTRe.*

search process. If we run centralized ASyMTRe on a single robot (method 2 in Table 7), the best solution can be found given enough time. However, except for the 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 the team capabilities change. To increase the robustness, we could run centralized ASyMTRe on every robot (method 3 in Table 7). However, robots still need to share capability information with each other at the beginning or whenever the team capabilities change. 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.

VI. RELATED WORK

The multirobot coalition formation approach incorporated into the ASyMTRe system is closely related to several bodies of research, including *coalition formation in multiagent research*, *teamwork theories*, and *task allocation*. The following subsections describe this prior work in more detail.

A. Coalition Formation in Multiagent Research

Coalition formation is not a new concept in the multiagent community (e.g., [24], [36], [37]). In these systems, agents are organized into coalescent teams to achieve a higher level goal [19]. In particular, the work of Shehory [37] inspired some aspects of our work. Shehory's work describes a method of allocating a set of interdependent tasks to a group of agents by forming coalitions. Tasks are assumed to have a precedence order, and agents use coalition formation to achieve efficient task allocation. This problem is similar to the set-partitioning problem and is well known to be NP-hard. However, by applying limitations on the permitted coalitions (e.g., the coalition size k), their greedy distributed set-partitioning algorithm has a low ratio bound $O(n^k)$. These algorithms also have the anytime property, returning better solutions over time. Shehory found this property to be especially important for operating in dynamic environments. In future work, we plan to use these results to try to prove similar approximation bounds on the ASyMTRe approach.

In other research, Sandholm *et al.* [36] present an approach to find coalitions via a partial search, with the generated results being guaranteed to be within a bound from the optimum. Although their algorithm reduces the search space dramatically, it is still exponential in the number of agents, and thus is not applicable for large groups. When there are a large number of agents, self-organization helps to improve the performance and to generate coalitions dynamically based upon interactions and communication among local agents [26]. However, as noted in [44], many of the multiagent approaches to coalition formation cannot be directly transferred to multirobot applications, since robot capabilities and sensors are situated directly on the robots and are not transferable between robots. Our ASyMTRe work differs from the prior work in multiagent coalition formation, in that we abstract the problem at the *schema* level, rather than the task level, permitting more flexibility in the solution approach, and allowing the system to take into account the situated nature of sensors and robot capabilities. Additionally, our approach enables sensor-sharing across multiple robot team members.

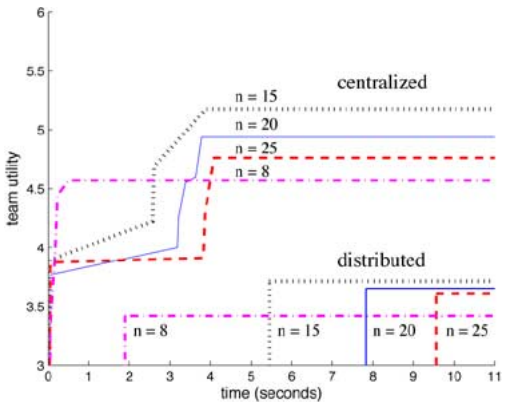


Fig. 11. *Given a specific group size (n = 8, 15, 20, 25), the solution quality of the centralized approach increases over time, which is also relatively better than the distributed approach. In these applications, we were able to run the centralized algorithm to its completion only when n is 8, which generates an optimal solution.*

Table 7 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

B. Teamwork

Similar to coalitions, agents form teams to work together to accomplish a common goal [40]. Several teamwork models have been developed to provide mechanisms for agents to negotiate with each other to agree upon a plan to achieve the team level goal (e.g., [7], [18], [20], [25], [40]). A common technique is the use of joint intentions (e.g., [7]) to develop shared plans to achieve the team level goal. In this approach, a belief–goal–commitment model is presented with formal definitions of events, belief, goal, and mutual belief. Other models reason about the proper teamwork, such as joint responsibility [20] and SharedPlan [17]. Based on some of the above theoretic work [7], [17], Tambe built STEAM (an abbreviation for Shell for Team) [40], a general teamwork model that takes into account the flexibility in a dynamic environment and reusability for different task domains. Jennings' work on joint responsibility [20] is also an extension of the joint intention model, involving satisfaction of defined preconditions to start cooperation and generation of plans for agents to behave during cooperation and in faulty cases. These approaches provide powerful high-level models for problem solving and role assignment taking into account the team capabilities. However, they do not address the issue of how agents can autonomously determine their proper contributions to the solution based upon their sensing, effector, and computational capabilities. These lower level issues are those that are addressed in our ASyMTRe approach.

We also note that, like many autonomous planning techniques, our representation of the robot capabilities is similar to that used in STRIPS planning [12]. These ideas have been extended to behavior-based robotic systems in the work of [29], which provides a mechanism for encoding complex, sequential, and hierarchical tasks within a behavior-based framework, suitable for plan generation. STRIPS-type planning is usually based on preconditions that describe the state of the world, rather than the capabilities of the robot team members. However, we abstract the problem of robot capabilities problem in a different manner. By representing robot schema inputs in terms of information requirements instead of resource requirements and environmental state conditions, we allow for more flexibility in how the robots share sensor and perceptual information to accomplish a multirobot task.

Our objectives in the automated synthesis of multirobot coalitions are similar to the work of Jones and Mataric [22], which addresses the automated synthesis of communication-based multirobot controllers. Similar to [22], our work also enables robots to share task-relevant information using communication. However, our work differs in that the composition of the coalition and the fundamental actions of robots (i.e., the schemas that are activated), are dependent upon the particular sensor and effector capabilities of robot team members. Thus, our work focuses on coalitions performing single-multirobot tasks, rather than teams of robots performing independent subtasks.

C. Task Allocation

Research specific to heterogeneous robots often focuses on the issue of *task allocation*, which is the problem of determining a suitable mapping between robots and tasks. Several approaches to robot team task allocation have been developed (e.g., [16], [31], [45], [48]). Since it has been shown that developing the optimal mapping of tasks to robots is NP-hard [30], existing mechanisms for MRTA typically use some type of heuristic greedy strategy to achieve the mapping, leading to the generation of suboptimal solutions. A formal analysis comparing the computation, communication requirements and solution qualities of several well-known approaches is presented in [14].

Typically, a task is decomposed into independent subtasks [31], hierarchical task trees [47], or roles [21], [38] either by a general autonomous planner or by the human designer. Independent subtasks or roles can be achieved concurrently, while subtasks in task trees are achieved according to their interdependence. Examples of behavior-based approaches to MRTA include ALLIANCE [32], which uses *motivational behaviors* to achieve fault-tolerant, adaptive action selection, enabling a team of robots to select appropriate actions contributing to a mission based on the mission requirements, the activities of other teammates, and the robot's internal states. BLE [45] is another behavior-based approach to multirobot coordination, which allows robots to execute tasks by continuously broadcasting locally computed eligibilities and only selecting the robot with the best eligibility to perform the task. In this case, task allocation is achieved through behavior inhibition.

After Smith [39] first introduced the Contract Net Protocol (CNP), many market-based approaches addressing multirobot cooperation through negotiation were developed, including M+ [4], MURDOCH [16], TraderBots [8], [47], [48], and Hoplites [23]. In these approaches, a task is divided into subtasks or hierarchical subtask trees (in the case of [47]) for the robots to bid and negotiate to carry out the subtasks. Each robot can estimate the utility of executing a subtask (or hierarchical tree of subtasks), which measures the quality and cost factors of the resulting actions. The goal is to greedily assign subtasks or task trees to the robot that can perform the task with the highest utility.

Our ASyMTRe-D approach is also based upon an economy model, which utilizes broadcasting and timeouts to ensure a robust communication model. Of the prior market-based approaches, Hoplites [23] is most closely related to ours, in that both approaches address tightly coupled multirobot tasks. However, Hoplites (and other market-based approaches) primarily address the coordination of interacting robot tasks (i.e., multiple single-robot tasks operating in parallel, ST-SR-IA). In contrast, our work addresses robot coalitions for performing single-multirobot tasks (ST-MR-IA). Robots are no longer working on independent subtasks or hierarchical task trees; instead, they are sharing sensor and effector capabilities to solve a single task that requires multiple robots working together simultaneously.

As in ASyMTRe, Fua and Ge [13] also address the problem of multirobot tasks (ST-MR-IA) using the COBOS cooperative backoff adaptive scheme. In the COBOS approach, task specifications include the capabilities a robot must possess to be eligible for a task. These capabilities can be represented using schemas (as in ASyMTRe) or resources (as in MURDOCH, [16]). However, COBOS assumes that the list of required capabilities for a given task is fixed; ASyMTRe, on the other hand, discovers alternative combinations of capabilities (i.e., schemas that generate required information) that can accomplish the task, and does not assume a fixed list of capabilities (or even information types) that are needed to accomplish a task. By abstracting the task using schemas and information requirements, rather than specific solutions based on specific sensors, we believe ASyMTRe generates more flexible solution strategies for multirobot coalition formation that are not dependent upon a fixed list of capabilities or resources required for each task.

VII. CONCLUSION AND FUTURE WORK

This paper has presented ASyMTRe—a mechanism for the automatic generation of multirobot coalitions in groups of heterogeneous robots. Built upon schema and information invariants theories, our approach enables the robot coalition to dynamically connect schemas within and across robots to accomplish a single-robot task using coalitions of multiple robots. We have shown that the centralized ASyMTRe configuration algorithm is sound, complete and optimal given enough time. In addition, we have also presented the ASyMTRe-D negotiation protocol that forms multirobot coalitions through a negotiation process. The distributed negotiation process enables each robot to find the best solution locally by maximizing the utility for executing the task. Compared with the centralized ASyMTRe, distributed ASyMTRe-D provides a more robust and flexible method for forming coalitions. However, it also presents a tradeoff between solution quality and robustness. We presented physical and simulated robot implementations and analytical studies of both of these approaches.

In broader application, we believe that the ASyMTRe approach can interface with existing approaches for task allocation and task planning that have traditionally dealt with assigning single-robot tasks (or hierarchical task trees) to the appropriate robot (i.e., the ST-SR-IA problem). In combining these approaches, we envision ASyMTRe being used at the lower level to form a set of possible coalitions that can address multirobot tasks. These coalitions would then compete (with other coalitions or with single robots) for the assignment of tasks at a higher level. Note that in this combined approach, it would be misleading to independently categorize a task as either a "single-robot" task or a "multirobot" task (i.e., SR or MR, using the taxonomy of [14]), without taking into account the robot team capabilities. In reality, the number of robots required to perform a task is dependent on the mix of capabilities of the available robots, even in the same application. So, by combining ASyMTRe for coalition formation at the lower level with "traditional" task allocation and task planning approaches at the higher level, both single robots and coalitions can be assigned tasks for the benefit of the entire team of robots within the same large-scale application.

Our ongoing work on ASyMTRe includes incorporating the learning of new semantic information labels, hierarchically and autonomously "chunking" schemas into higher level capabilities, addressing issues of information quality, developing formal methods for incorporating motion constraints in the physical applications, exploring alternative configuration search techniques, interfacing ASyMTRe with more expressive goal specifications, and extending ASyMTRe to enable human–robot coalitions. ■

Acknowledgment

The authors would like to thank M. Chandra for her work in implementing some of the physical robot experiments. The authors would also like to thank the anonymous reviewers for their helpful suggestions for improving an earlier version of this paper.

REFERENCES

[1] T. Arai, E. Pagello, and L. E. Parker, "Editorial: Advances in multi-robot systems," *IEEE Trans. Robot. Autom.*, vol. 18, no. 5, pp. 655–661, Oct. 2002.

[2] R. C. Arkin, "Motor schema based navigation for a mobile robot: An approach to programming by behavior," in *Proc. IEEE Conf. Robotics and Automation*, 1987, pp. 264–271.

[3] G. Beni and J. Wang, "Swarm intelligence in cellular robotics systems," in *Proc. NATO Advanced Workshop Robots and Biological System*, 1989, pp. 703–712.

[4] S. Botelho and R. Alami, "M+: A scheme for multi-robot cooperation through negotiated task allocation and achievement," in *Proc. IEEE Int. Conf. Robotics and Automation*, 1999, pp. 1234–1239.

[5] Y. Cao, A. Fukunaga, and A. Kahng, "Cooperative mobile robotics: antecedents and directions," *Auton. Robots*, vol. 4, pp. 1–23, 1997.

[6] M. Chandra, "Software Reconfigurability for heterogeneous robot cooperation," Master's thesis, Dept. Comput. Sci., Univ. Tennessee, Knoxville, 2004.

[7] P. R. Cohen and H. Levesque, "Teamwork," *Nous*, vol. 25, no. 4, pp. 487–512, 1991.

[8] M. B. Dias and A. Stentz, "A free market architecture for distributed control of a multirobot system," in *Proc. Int. Conf. Intelligent Autonomous System*, 2000, pp. 115–122.

[9] M. B. Dias, M. Zinck, R. Zlot, and A. Stentz, "Robust multirobot coordination in dynamic environments," in *Proc. IEEE Int. Conf. Robotics and Automation*, 2004, pp. 3435–3442.

[10] B. R. Donald, J. Jennings, and D. Rus, "Towards a theory of information invariants for cooperating autonomous mobile robots," presented at the *Int. Symp. Robotics Research*, Hidden Valley, PA, 1993.

[11] ——, "Information invariants for distributed manipulation," *Int. J. Robot. Res.*, vol. 16, no. 5, pp. 673–702, 1997.

[12] R. E. Fikes and N. J. Nilsson, "STRIPS: A new approach to the application of theorem proving to problem solving," *Artif. Intell.*, vol. 2, no. 3/4, pp. 189–208, 1971.

[13] C.-H. Fua and S. S. Ge, "COBOS: Cooperative backoff adaptive scheme for multirobot task allocation," *IEEE Trans. Robot.*, vol. 21, no. 6, pp. 1168–1178, 2005.

[14] B. Gerkey and M. J. Mataric, "A formal analysis and taxonomy of task allocation in multi-robot systems," *Int. J. Robot. Res.*, vol. 23, no. 9, pp. 939–954, 2004.

[15] B. Gerkey, R. Vaughan, K. Stoy, and A. Howard, "Most valuable player: A robot device server for distributed control," in *Proc. 2001 IEEE/RSJ Int. Conf. Intelligent Robotics and Systems*, 2001, pp. 1226–1231.

[16] B. P. Gerkey and M. J. Mataric, "Sold! Auction methods for multi-robot coordination," *IEEE Trans. Robot. Autom.*, vol. 18, no. 5, pp. 758–768, 2002.

[17] B. J. Grosz, "Collaborative systems," *AI Mag.*, vol. 17, no. 2, pp. 67–85, 1996.

[18] H. Hexmoor and G. Beavers, "Towards teams of agents," presented at the Proc. Int. Conf. Artificial Intelligence, Las Vegas, NV, 2001.

[19] B. Horling and V. Lesser, *A survey of multi-agent organizational paradigms*, Univ. Massachusetts, Cambridge, Tech. Rep. 04-45, 2004.

[20] N. Jennings, "Controlling cooperative problem solving in industrial multi-agent systems," *Artif. Intell.*, vol. 75, no. 2, pp. 195–240, 1995.

[21] N. R. Jennings and C. Kirkwood-Watts, "Distributed mobile robotics by the method of dynamic teams," presented at the 4th Int. Symp. Distributed Autonomous Robotic Systems (DARS 1998), Karlsruhe, Germany.

[22] C. Jones and M. Mataric, "Automatic synthesis of communication-based coordinated multi-robot systems," in *Proc. IEEE/RSJ Int. Conf. Intelligent Robots and Systems (IROS)*, 2004, pp. 381–387.

[23] N. Kalra, D. Ferguson, and A. Stentz, "Hoplites: A market-based framework for planned tight coordination in multirobot teams," in *Proc. IEEE Int. Conf. Robotics and Automation*, 2005, pp. 1170–1177.

[24] M. Klusch and A. Gerber, "Dynamic coalition formation among rational agents," *IEEE Intell. Syst.*, vol. 17, no. 3, pp. 42–47, May 2002.

[25] H. Levesque, P. Cohen, and J. Nunes, "On acting together," in *Proc. 8th Nat. Conf. Artificial Intelligence*, 1990, pp. 94–99.

[26] K. H. Low, W. K. Leow, Jr., and M. A. Ang, "Task allocation via self-organization swarm coalitions in distributed mobile sensor network," in *Proc. 19th Nat. Conf. Artificial Intelligence*, 2004, pp. 28–33.

[27] D. M. Lyons and M. A. Arbib, "A formal model of computation for sensory-based robotics," *IEEE Trans. Robot. Autom.*, vol. 5, no. 3, pp. 280–293, Jun. 1989.

[28] M. J. Mataric, "Designing emergent behaviors: From local interactions to collective intelligence," Proc. 2nd Int. Conf. Simulation of Adaptive Behavior, J. Meyer, H. Roitblat, and S. Wilson, Eds., 1992, pp. 432–441.

[29] M. Nicolescu and M. Mataric, "A hierarchical architecture for behavior-based robots," in *Proc. 1st Int. Conf. Autonomous Agents and Multi-Agent Systems*, 2002, pp. 227–233.

[30] L. E. Parker, "L-ALLIANCE: Task-oriented multi-robot learning in behavior-based systems," *J. Adv. Robot.*, 1997.

[31] L. E. Parker, "ALLIANCE: An architecture for fault-tolerant multi-robot cooperation," *IEEE Trans. Robot. Autom.*, vol. 14, no. 2, pp. 220–240, Apr. 1998.

[32] L. E. Parker, "Toward the automated synthesis of cooperative mobile robot teams," in *Proc. SPIE: Sensor Fusion and Decentralized Control in Robotic Systems II*, 1998, pp. 82–93.

[33] L. E. Parker, "Lifelong adaptation in heterogeneous multi-robot teams: Response to continual variation in individual robot performance," *Auton. Robots*, vol. 8, no. 3, pp. 239–267, Jun. 2000.

[34] L. E. Parker, B. Kannan, F. Tang, and M. Bailey, "Tightly-coupled navigation assistance in heterogeneous multi-robot teams," in *Proc. IEEE Int. Conf. Intelligent Robots and Systems*, 2004, pp. 1016–1022.

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

[36] T. Sandholm, K. Larson, M. Andersson, O. Shehory, and F. Tohme, "Coalition structure generation with worst case guarantees," *Artif. Intell.*, vol. 111, no. 1–2, pp. 209–238, 1999.

[37] O. Shehory, "Methods for task allocation via agent coalition formation," *Artif. Intell.*, vol. 101, no. 1–2, pp. 165–200, 1998.

[38] R. Simmons, S. Singh, D. Hershberger, J. Ramos, and T. Smith, "First results in the coordination of heterogeneous robots for large-scale assembly," in *Proc. ISER '00 7th Int. Symp. Experimental Robotics*, pp. 323–332.

[39] R. G. Smith, "The contract net protocol: High-level communication and control in a distributed problem solver," *IEEE Trans. Comput.*, vol. C-29, no. 12, pp. 1104–1113, Dec. 1980.

[40] M. Tambe, "Towards flexible teamwork," *J. Artif. Intell. Res.*, vol. 7, pp. 83–124, 1997.

[41] F. Tang and L. E. Parker, "ASyMTRe: Automated synthesis of multi-robot task solutions through software reconfiguration," in *Proc. IEEE Int. Conf. Robotics and Automation*, 2005, pp. 1513–1520.

[42] F. Tang and L. E. Parker, "Coalescent multi-robot teaming through ASyMTRe: A formal analysis," in *Proc. 12th Int. Conf. Advanced Robotics*, 2005, pp. 817–824.

[43] F. Tang and L. E. Parker, "Distributed multi-robot coalitions through ASyMTRe-D," in *Proc. IEEE Int. Conf. Intelligent Robots and Systems*, 2005.

[44] L. Vig and J. A. Adams, "Issues in multi-robot coalition formation," in *Multi-Robot Systems Volume III: From Swarms to Intelligent Automata*, L. E. Parker, A. Schultz, and F. Schneider, Eds. Boston, MA: Kluwer, 2005.

[45] B. B. Werger and M. J. Mataric, "Broadcast of local eligibility for multi-target observation," in *Proc. 5th Int. Symp. Distributed Autonomous Robotic Systems (DARS)*, 2000, pp. 347–356.

[46] S. Zilberstein, "Using anytime algorithms in intelligent systems," *AI Mag.*, vol. 17, no. 3, pp. 73–83, 1996.

[47] R. Zlot and A. Stentz, "Complex task allocation for multiple robots," in *Proc. IEEE Int. Conf. Robotics and Automation*, 2005, pp. 1515–1522.

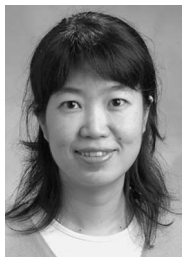
[48] R. Zlot, A. Stentz, M. B. Dias, and S. Thayer, "Multi-robot exploration controlled by a market economy," in *Proc. IEEE Int. Conf. Robotics and Automation*, 2002, pp. 3016–3023.

ABOUT THE AUTHORS

Lynne E. Parker (Senior Member, IEEE) received the Ph.D. degree in computer science in 1994 from the Massachusetts Institute of Technology (MIT), Cambridge, performing research on cooperative control algorithms for multirobot systems in MIT's Artificial Intelligence Laboratory.

She joined the faculty of the Department of Computer Science, University of Tennessee, Knoxville, as Associate Professor in August 2002, where she founded the Distributed Intelligence Laboratory. She also holds an appointment as Adjunct Distinguished Research and Development Staff Member in the Computer Science and Mathematics Division at Oak Ridge National Laboratory, Oak Ridge, TN, where she worked as a researcher for several years. She is the author of five edited volumes on the topic of multirobot systems. Her current research interests are in distributed robotics, artificial intelligence, sensor networks, embedded systems, and multiagent systems.

Dr. Parker is a member of the American Association for Artificial Intelligence (AAAI), the Association for Computing Machinery (ACM), and Sigma Xi. She was awarded the 2000 U. S. Presidential Early Career Award for Scientists and Engineers for her research in multirobot systems, as well as the 1999 U.S. Department of Energy Office of Science Early Career Scientist and Engineer award.



Fang Tang (Student Member, IEEE) received the B.S. degree in computer science from Sichuan University, Chengdu, China, in 2000 and the M.S. degree in computer science from the University of Tennessee, Knoxville (UTK), in 2003. She is currently working toward the Ph.D. degree at UTK, performing her research on multirobot cooperation in the Distributed Intelligence Laboratory.

She joined the Department of Computer Science at UTK as a Graduate Student in August 2001. She has been conducting research in the Distributed Intelligent Laboratory at UTK since 2003, and worked on the Software for Distributed Robotics project funded by DARPA. Her current research focuses on developing mechanisms to enable multiple robots to cooperate by autonomously forming coalitions. Her research interests also include multiagent systems, human-robot cooperation, and machine learning. She is a student member of the Association for Computing Machinery (ACM) and the American Association for Artificial Intelligence (AAAI).