*Proc. of IEEE International Conference on Robotics and Automation*, St. Paul, MN, 2012.

# Task Allocation with Executable Coalitions in Multirobot Tasks

Yu Zhang and Lynne E. Parker

***Abstract*— In our prior work, we proposed the IQ-ASyMTRe architecture with a measure of information quality to reason about forming coalitions in multirobot tasks. The formed coalitions are guaranteed to be executable, given the current configurations of the robots and environment. A cost and a quality measure are associated with each coalition to further determine its utility for the task. In this paper, we show that IQ-ASyMTRe-like architectures can be utilized to significantly reduce the overall complexity of task allocation by considering only executable coalitions. For implementation, we apply a layering technique such that most existing methods for task allocation can be easily incorporated. Furthermore, we introduce a general process to address situations in which no executable coalitions are available for certain tasks, and integrate it with IQ-ASyMTRe to achieve more autonomy. Such an approach is able to autonomously decompose unsatisfied preconditions of the required task behaviors into satisfiable components, in order to generate partial order plans for them accordingly. We show how this process can be implemented using a market-based approach. Simulation results are provided to demonstrate these techniques.**

## I. INTRODUCTION

The task allocation problem addresses the issue of assigning available resources (i.e., robots) to tasks. While task allocation with single-robot (SR) tasks [6] can be solved optimally and efficiently, the problem with multirobot (MR) tasks is known to be NP-hard. Task allocation with multirobot tasks (also known as the *coalition formation* problem) involves the problem of determining possible coalitions (forming coalitions) and the problem of efficiently allocating tasks to a subset of these coalitions. This problem is extremely difficult to solve. For one, the number of possible coalitions grows exponentially with the number of robots. Furthermore, given a set of possible coalitions ($C$), one needs to check every possible set of assignments of coalitions to tasks ($T$) in order to determine the optimal solution; the number of such sets is $O(|T|^{|C|})$. As the number of coalitions increases, the problem quickly becomes intractable.

Although the problem is NP-hard, efficient heuristics can be designed to address it. Heuristics with worst case guarantees are valuable, as well as those that have been proven empirically to perform well in certain domains. It is desirable to be able to incorporate these methods easily in architectures for forming coalitions. To compute the solution, these methods require a cost measure for each coalition, which should be evaluated to approximate the overall costs incurred for accomplishing the task by the coalition. Costs of the allocated capabilities (e.g., sensors), communication, and coordination between robots should all be considered. Note that architectures for forming coalitions can define capabilities differently, as long as they specify cost measures for these capabilities in order to compute the coalition cost. A reward is associated with each task and there are precedence orders between different tasks that must be satisfied.

However, to enable the task allocation methods to run more efficiently, it is desirable to reduce the number of possible coalitions. Given a task, the most obvious way to reduce the number of coalitions is to consider the ones that satisfy the capability requirement of the task (referred to as *feasible coalitions* henceforth). However, the number of feasible coalitions can still be large. The problem is alleviated in non-super-additive environments [13] by restricting the maximum number of robots that can be allocated to a coalition. The assumption is that the coordination and communication costs increase significantly as the size of the coalitions grows, such that it is not beneficial to consider coalitions larger than a fixed size. However, such an approach does not actually solve the problem. For example, in a system with $15$ robots, even when we restrict the maximum size to be $3$, the number of possible coalitions is $575$, which can include hundreds of feasible coalitions among them. For environments that are not guaranteed to be non-super-additive, the situation becomes even worse.

One observation is that it is often unnecessary to consider all feasible coalitions in real applications. This is due to the fact that a large portion of feasible coalitions may not be in an executable state (i.e., certain preconditions of the required behaviors are not satisfied), such that the robots do not know how to execute them. These coalitions can be ignored for task allocation until ways to satisfy their preconditions are found and evaluated. Meanwhile, if the related tasks can be accomplished by executable (alternative) coalitions with reasonable costs, the satisfaction of the preconditions to enable these coalitions is likely to be unnecessary.

In this paper, we first show how task allocation can be achieved with executable coalitions and discuss how such an approach can significantly reduce the number of coalitions. We build this approach based on our previous work [17] (IQ-ASyMTRe) for forming executable coalitions given the current configurations of the robots and environment. Meanwhile, for tasks with no executable coalitions, instead of directly planning on the unsatisfied preconditions[1], we utilize the reasoning process of IQ-ASyMTRe to decompose them

This material is based upon work supported by the National Science Foundation under Grant No. 0812117.

Yu Zhang and Lynne Parker are with the Distributed Intelligence Laboratory in the Department of Electrical Engineering and Computer Science, University of Tennessee, Knoxville, TN 37996, USA, {yzhang51,parker}@eecs.utk.edu.

[1]Directly planning on these unsatisfied preconditions is only helpful if specific behaviors are implemented to satisfy these preconditions.

into satisfiable components and create *partial order plans* [10] accordingly.

To the best of our knowledge, this is the first work that addresses the task allocation problem with executable coalitions in order to reduce the problem complexity with multirobot tasks. Furthermore, we introduce an approach that can search for different ways to satisfy preconditions of the required task behaviors based on the current situations and autonomously create partial order plans. The implementation of this approach utilizes the reasoning process of our previous work (based on information invariant theory [4]) and is integrated with it in a natural way. This paper is organized as follows. A brief discussion of the related work is provided in Section II. In Section III, we first give a brief introduction of the IQ-ASyMTRe architecture. Afterwards, we discuss how to layer task allocation with IQ-ASyMTRe and then introduce the process to address tasks with no executable coalitions. Simulation results are presented in Section IV with discussions of conclusions and future work in Section V.

## II. RELATED WORK

Although many approaches are available for forming coalitions to enable multirobot cooperation, most of them address either single-robot (SR) tasks [1], [5] or loosely-coupled multirobot tasks [2], [3], [18]. The ability to also address the tightly-coupled multirobot tasks is desirable, since robots often do not have all the capabilities and need to share them to accomplish a task. Architectures [9], [14], [17] that utilize schema theory [8] and define inputs and outputs based on the notion of *information type*[2] enable capability sharing; results have been provided to demonstrate the flexibility of such an ability. The work of IQ-ASyMTRe [17] improves over [9], [14] and guarantees forming executable coalitions. Hence, we build our approach based on [17].

For more theoretic analysis of the task allocation problem with multirobot tasks, researchers often choose a numerical representation of capabilities [11], [12], [13]. In [11], Sandholm discusses the requirements that approximation algorithms must satisfy in order to have worst-case guarantees. In [13], Shehory provides an efficient approximation algorithm based on a greedy approach for the set covering problem and implements it on multirobot systems with reasonable solution bounds. In [12], Service discusses two different formulations of the problem (i.e., service and resource models) and efficient approximation algorithms are provided to address them. Although the representations of capabilities in these algorithms are different from that in IQ-ASyMTRe, it is desirable to be able to interface with them easily to improve the overall task allocation performance.

To provide such a solution, a layering technique that allows different task allocation methods to be easily incorporated is adopted. A similar approach is presented in [15] for the ASyMTRe architecture [9]. However, as discussed in [17], ASyMTRe suffers from several issues which are addressed by IQ-ASyMTRe to enable coalition execution. While the issue with a large number of coalitions is addressed by assuming non-super-additive environments in [15], to provide a new aspect, we utilize IQ-ASyMTRe to address the task allocation problem with executable coalitions.

Finally, the issue needs to be addressed when no executable coalitions exist for tasks. One approach is to use AI planning techniques separately as in [7]. However, this approach is only applicable for sequencing executable behaviors, with the previous behaviors satisfying preconditions of the following ones. The method we propose is able to find ways to satisfy these preconditions even no behaviors are provided to directly satisfy them.

## III. TASK ALLOCATION WITH IQ-ASYMTRE

In this section, after a brief review of IQ-ASyMTRe [17], we discuss the layering technique as well as addressing tasks with no executable coalitions using a market-based approach. Finally, specific algorithms are provided.

### A. The IQ-ASyMTRe Architecture

The IQ-ASyMTRe architecture defines basic building blocks of robot capabilities to be collections of environmental sensors (ESs), perceptual schemas (PSs), motor schemas (MSs), and communication schemas (CSs). Each schema can be activated when its inputs are satisfied and may produce certain outputs. Inputs and outputs are labeled using information instances (see definition below). Then, according to a set of rules, connections can be created among the schemas on the robots to allow information to flow through the system to activate the required MSs (i.e., the task behaviors).

*1) Information Type, Instance and Conversion:* IQ-ASyMTRe uses both *information type* and *information instance* for a complete reference of information. In IQ-ASyMTRe, an information type is specified as $(\mathcal{F}_i,\ N_i)$, where $\mathcal{F}_i$ is consistent with our previous discussion of information type (Section II) and that defined in [9], [14]. $N_i$ is the number of referents associated with $\mathcal{F}_i$. An information instance also captures information about the related entities. An information instance of type $\mathcal{F}_i$ can be represented as $F_i(\mathit{Ref}_{1:N_i})$, where $\mathit{Ref}_j$ is used to refer to the $j$th referent for the information instance.

Each referent, $\mathit{Ref}_j$, can be instantiated to a particular entity or remain uninstantiated, waiting for future instantiations. Fully instantiated information instances represent actual information that can be used, while partially instantiated ones represent a class of information. For example, $F_G(X)$ can be the global position information of any entity that $X$ is instantiated to, while $F_G(r)$ represents the global position information of the robot $r$.

Inspired by information invariant theory [4], IQ-ASyMTRe introduces *information conversion* as a special PS (*Reduction Perceptual Schema*, denoted as RPS) to express the conversions between different information instances. Table I shows the RPSs used in the discussions in this paper, in which $+$ represents the *AND* condition.

[2]Information types differ from data types in that they have semantic meanings, e.g., the global or relative position information.

TABLE I

EXAMPLES OF RPSs

| RPS | Description |
|---|---|
| $F_G(X) + F_R(Y, X) \Rightarrow F_G(Y)$ | global + relative $\Rightarrow$ global |
| $F_R(Y, X) \Rightarrow F_R(X, Y)$ | relative $\Rightarrow$ relative |

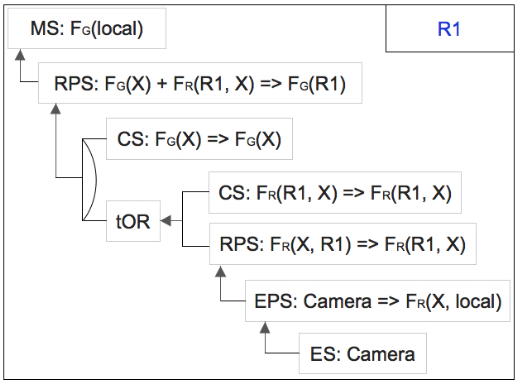



Fig. 1. A solution space for a robot to obtain $F_G(local)$ with only a camera sensor. The referent *local* refers to the robot itself. The solution space encodes two solutions. One solution is to have another robot send over its global position (CS: $F_G(X) \Rightarrow F_G(X)$) and use the camera to sense the relative position (EPS: Camera $\Rightarrow F_R(X, local)$), in which EPS represents a special type of PS for sensors. A RPS (RPS: $F_R(Y, X) \Rightarrow F_R(X, Y)$) is used to convert $F_R(X, R_1)$ to $F_R(R_1, X)$. The other solution (*tOR*) is to have both information instances (CS: $F_G(X) \Rightarrow F_G(X)$ and CS: $F_R(R_1, X) \Rightarrow F_R(R_1, X)$) sent over by another robot.

*2) Solution Space and Potential Solution:* A solution space encodes all potential solutions. To create it, the IQ-ASyMTRe reasoning algorithm checks all schemas that can provide the inputs of the required MS. The algorithm then checks recursively for the inputs of those schemas. Figure 1 (from [17]) shows a solution space as an *and-or* tree for retrieving $F_G(local)$. The *tOR* node is introduced to manage multiple options of connection. After the solution space is created, potential solutions can be extracted from the solution space by making decisions on which schema node to use at each *tOR* node (the rest of the nodes are trimmed), as the extraction proceeds from the root to the leaves.

*3) Incorporation of Information Quality:* To facilitate coalition execution, IQ-ASyMTRe uses a measure of information quality introduced in [16], which assesses the utility of the required information for enabling the coalitions. Such information often influences the coalition execution in the form of sensor constraints. For example, in a cooperative robot navigation task, a sensor constraint is for the follower to keep the leader in its field of view (FOV). Here, the information quality measure specifies how well the follower is tracking the leader, considering their relative configuration and environmental influence.

By assuming independence between information instances and defining the information quality measures to be within [0, 1], IQ-ASyMTRe computes the coalition quality measure by simply multiplying the related information quality measures. The multiplications naturally reflect the fact that the more dependencies there are, the less reliability there is.

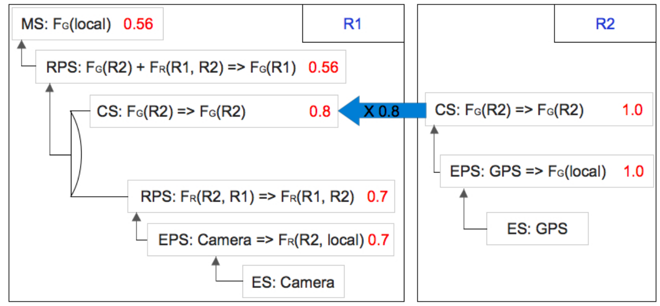



Fig. 2. An instantiated potential solution extracted from the solution space in Figure 1 with the information quality measures set arbitrarily. Information quality measures reflect the utility (e.g., retrievability) of using the information in the current situation. The measure for $F_G$ with a GPS sensor is set to be 1, given that the sensor is usually reliable. The measure for $F_R$ with a camera sensor reflects the retrievability of the information based on sensor models and the relative configuration. In IQ-ASyMTRe, these measures are dynamically determined using the approach in [16].

The process for computing the coalition quality measure is shown in Figure 2 for one of the potential solutions in Figure 1 after instantiations. The coalition quality is $0.56$. Note that the unreliability of communication (i.e., using CS) is accounted for by multiplying by a pre-defined or dynamically determined scalar (arbitrarily picked as $0.8$ in the figure).

### *B. Layering IQ-ASyMTRe with Task Allocation*

Tasks can be represented as the required behaviors (i.e., MSs) to be activated. IQ-ASyMTRe creates coalitions for tasks when the required information for the behaviors can be retrieved. Since the reference of information is complete, IQ-ASyMTRe guarantees forming executable coalitions. Task allocation can be implemented similarly as in [15] based on a marker-based approach. An auctioneer (i.e., the central task allocation process) announces tasks to the robots, while the robots (running IQ-ASyMTRe algorithms) reason about possible coalitions and submit bids for tasks. The auctioneer (also running a task allocation algorithm) then determines the winner coalitions for the tasks and assigns them to these coalitions. A coalition (and a robot) can only win one bid at a time. Robustness can be achieved by setting timers for certain events (e.g., when no bids for a task are received after a period of time, the task can be re-announced).

*1) Interface with Task Allocation:* To interface with the task allocation methods, information about the robots in the coalitions, the costs of coalitions, rewards for the tasks, and precedence orders between tasks must be included in the bids. One assumption we make is that capabilities are not shared between different coalitions. This is almost always true in multirobot systems (unlike in multi-agent systems), since capabilities are not transferable. As a result, information regarding the capabilities allocated by the coalitions does not need to be provided. This reduces the complexity of the central task allocation process.

In IQ-ASyMTRe, coalition members include robots that provide the necessary information and the robots that execute the required behaviors. For example, in Figure 2, while $R_1$ is the robot that executes the behavior (i.e., the MS), $R_2$

provides the required information to $R_1$. Hence, both $R_1$ and $R_2$ are in the coalition. Coalitions with a single robot can be created when individual robots can execute the desired behaviors without help. Rewards of tasks and precedence orders between tasks are often specified a priori. Next, we discuss how the costs of coalitions are computed.

*2) Coalition Cost:* The cost of a coalition should be computed to approximate the actual cost for the coalition to accomplish the task. In IQ-ASyMTRe, the costs of the (sensory and computational) capabilities can be computed as the summation of the costs of the MSs, PSs and ESs activated; similarly, communication and coordination costs can be computed based on the costs of the CSs used. When execution times can be estimated, they can be incorporated by defining the costs of schemas to be unit-time costs. Furthermore, to consider the influence of information quality, we associate the coalition quality measure with the success ratio ($\theta$) of the coalition using a task-specific function, $F : [0,1] \times T \rightarrow [0,1]$, in which $[0,1]$ is the space of all possible values of coalition quality and $T$ is the space of all possible types of tasks.

Note that coalition quality is computed to reflect the utility for using the information required by the coalition. While certain tasks are not influenced much by the utility of the information, others can be significantly affected. For example, in the robot navigation task, the success ratio is not influenced much by whether the follower robot is in a desirable configuration relative to the leader (e.g., close to the leader), since the robots can communicate to coordinate their actions. On the other hand, in a robot tracking task, whether the robot can successfully track the target is highly dependent on their relative configuration, since the target can move out of sight easily when the configuration is undesirable. Given that the coalition is committed to accomplish the task once assigned, the expected cost can be computed for a coalition $c$ and task $t$ as $cost(c,t) = E(\widehat{cost}(c,t)) = \widehat{cost}(c,t)/F(Q_c, Y_t)$, in which $\widehat{cost}(c,t)$ represents the summation of the costs of all activated schemas in the coalition for the task, $Q_c$ and $Y_t$ represent the coalition quality of $c$ and the task type of $t$, respectively. When the expected costs of all executable coalitions for a task are greater than the reward of the task (i.e., all coalitions are very likely to fail given the current situation, so that executing the task would not be beneficial), the task would be handled as if no executable coalitions exist according to the process in the following section.

### *C. Tasks with No Executable Coalitions*

One obvious advantage with forming executable coalitions is that robots actually know how to execute them to accomplish the tasks. Furthermore, since the number of executable coalitions often are much smaller than the number of feasible coalitions in the current situation, the complexity for task allocation can be significantly reduced. Consequently, however, IQ-ASyMTRe cannot address tasks for which no executable coalitions exist.

*1) Extending MS:* First of all, in IQ-ASyMTRe, we notice that the preconditions of behaviors are input information

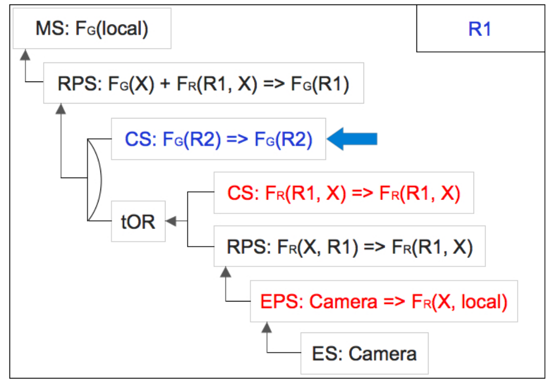



Fig. 3. An illustration example for using IMS in the robot navigation task.

instances of the required MSs for the tasks (e.g., $F_G(local)$ in Figure 1). A task is not executable if the required information cannot be retrieved by any coalitions in the environment. In order to introduce executable coalitions, the robots must have the capability to obtain information that is initially not retrievable. To incorporate this capability into the general framework of schema theory, we extend the definition of MS so that it can not only output commands for actuators, but also output information. In such a way, the IQ-ASyMTRe algorithms can remain almost unchanged. Although the introduction of this new kind of MS does not influence how MSs are used, it is referred to as IMS (Information MS) when we need to distinguish it.

To utilize this capability, all that needs to be done is to create and assign new tasks for IMSs. However, it is unlikely that IMSs always exist to directly retrieve the required information. To address this issue, we can also utilize the capability of IQ-ASyMTRe to reason about alternative ways to retrieve the information. An illustration example is presented in Figure 3 for the robot navigation task. When no potential leader is in the FOV of the follower ($R_1$), no executable coalition exists. Suppose that an IMS (i.e., *find-entity*) is implemented on both $R_2$ (a potential leader) and $R_1$ to search for entities within the communication range in the environment. Given that $R_1$ knows that $R_2$ can localize (from CS: $F_G(R_2) \Rightarrow F_G(R_2)$ in blue), $R_1$ either needs to find a way to retrieve $F_R(R_1, R_2)$ or $F_R(R_2, R_1)$ (in red). While the first can be more conveniently retrieved by $R_2$ executing *find-entity*, the latter is more convenient for $R_1$.

*2) Information Task Request:* Task allocation is performed in two phases. In the first phase, called the *Easy Auction* phase, robots search for executable coalitions for the broadcasted tasks and submit bids to the auctioneer. The auctioneer assigns tasks based on all submitted bids. For tasks that no bids are submitted, an additional auctioning step is initiated for IMS tasks. In this second phase (called the *IMS Auction* phase), the robots reason about the alternative ways to retrieve the input information instances as demonstrated in Figure 3. The robots then submit *information task requests* for requesting IMS tasks to the auctioneer. The auctioneer then considers these new tasks as preconditions[3] for the initiating tasks in the partial order plan. These new tasks are then auctioned using the *Easy Auction* phase. Note that these new tasks are handled in the same way and the two-phase process may apply recursively.

In this way, partial order plans can be autonomously generated. Since these IMSs are created to provide input information for the required MSs, once the new tasks are accomplished, the input information of the required MSs would be satisfied and the initiating tasks can be executed. However, caution must be taken to avoid IMS tasks removing the already satisfied input information. Detail discussion is out of the scope of this paper and the issue will be addressed in our future work.

Note that the creation of partial order plans unavoidably introduces scheduling issues, which greatly increase the complexity of the task allocation problem. In our current approach, these scheduling issues are ignored. For each task allocation process, we consider only tasks for which all preconditions in the partial order plan are satisfied.

### D. Algorithms for Task Allocation

The algorithms for the auctioneer and robot processes are provided in Algorithms 1 and 2, respectively. The auctioneer maintains a list of new tasks and a list of announced tasks. In each task allocation process, the auctioneer announces tasks for which all preconditions in the partial order plan are satisfied. It then receives bids and allocates tasks to winner coalitions. Tasks for which no bids are submitted or no bids are beneficial are moved to the list of announced tasks. Tasks in the list of announced tasks are announced in the *IMS Auction* phase. Whenever the auctioneer receives information task requests (IMS tasks), the auctioneer updates the preconditions of the initiating tasks in the list of announced tasks and moves them to the list of new tasks. Note that new tasks are not announced until all preconditions in the partial order plan are satisfied. For the robots, when there is a winning bid, they set up coalitions to accomplish the task; otherwise, they reason about the solutions for the announced tasks.

## IV. SIMULATION RESULTS

We demonstrate the different aspects of our approach in simulation for the cooperative robot navigation task. In all simulations, blue robots are leader robots that have a localization capability, while red robots are follower robots. Robots in the simulations are running the same program with different configurations (e.g., communication ports) as separate processes. All data is collected based on a 2.4GHz Core 2 Duo laptop with 2GB memory and all robot processes are running on the same machine. Every robot has a laser fiducial sensor to detect other teammates. The range of these sensors is restricted to 4 meters, and the angle is restricted to 180 degrees in front of the robot.

[3]Here, preconditions refer to execution orders specified for partial order plans, which differ from our previous mentioning of preconditions as input information for behaviors. References of preconditions in the remaining discussions should be unambiguous given the context.

**Algorithm 1** Auctioneer Process

```
Create empty new_task and announced_task lists.
while true do
  Receive new tasks and put them on the new_task list.
  for all tasks in announced_list that are initiating tasks
  for the new IMS tasks received do
    Update the task's preconditions.
    Move the task from announced_list to new_list.
  end for
  IMS Auction: announce tasks in announced_list.
  Easy Auction: announce tasks in new_task list for which
  preconditions are satisfied.
  Move the announced tasks to announced_list.
  Wait a while for bids.
  Collect bids from robots.
  Invoke task allocation algorithms to determine the task
  assignments.
  Remove tasks that are assigned from new_task list.
  Move tasks for which no bids are submitted or no bids
  are beneficial to announced_list.
end while
```

**Algorithm 2** Robot Process

```
while true do
  if the robot has a winning bid then
    Set up the coalition and execute the task.
  end if
  Receive new task announcements.
  for all received tasks do
    if task announced for Easy Auction then
      Invoke IQ-ASyMTRe to search for executable
      coalitions and submit bids.
    else if task announced for IMS Auction then
      Invoke IQ-ASyMTRe to submit information task
      requests.
    end if
  end for
end while
```

### A. IQ-ASyMTRe with Coalition Quality

First of all, we provide different scenarios in which the difficulty for finding executable coalitions gradually increases. As shown in Figure 4, the goal is for the last follower robot to achieve a localization capability. Since there is only one leader, given the configurations of the other followers, the only possible coalition is for the last follower to set up a grand coalition (in which all robots are included). Since the robots initially have only local information, the last follower must request information from others until it discovers the situation and the only coalition solution.

Figure 5 shows the time requirements and coalition quality measures when we gradually increase the number of followers from 1 to 9. The blue line shows the times (in seconds) that the last follower uses to find the only coalition from initially receiving the task. The coalition quality (see

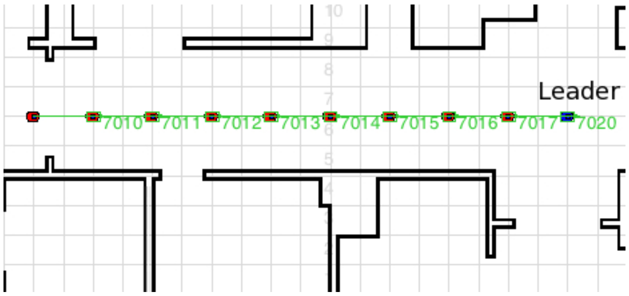



Fig. 4. A configuration of a line of followers with one leader at the front (the right). Each robot blocks the view of the robot immediately behind it.

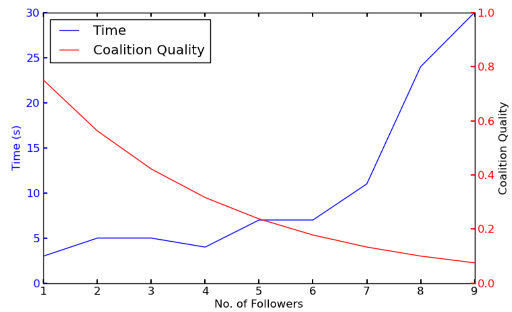



Fig. 5. Time and coalition quality measures for the scenarios in Figure 4.

Section III-A) computed by the follower is also shown in red. To make the results easier to interpret, we do not scale down the quality of information transferred in this simulation and maintain a fixed distance for all adjacent robots. We also assume that the information quality of the localization information retrieved by the leader is 1. When the last follower is the only follower (i.e., corresponding to the point with only 1 follower in Figure 5), the coalition quality is only influenced by the information quality of the relative position information retrieved using the fiducial sensor, which has a measure of 0.75 in this setting using the sensor models in [16]. When there are two followers, the coalition quality is influenced by the quality measures of both relative positions (of the same value). One can then easily induce the coalition quality with $N$ followers.

We can clearly see from Figure 5 that the time required to find the coalition increases as the number of followers increase. In real applications, however, the coalitions with many robots involving in such tight coordination would most likely be ignored, since the expected cost of such coalitions can easily get higher than the reward of the task (see the computation of the expected cost in Section III-B). We can see from this simulation the flexibility of the IQ-ASyMTRe for finding executable coalitions and how the measure of coalition quality can help in making coalition decisions.

### B. Executable Vs. Feasible Coalitions

In this simulation, we demonstrate the advantage of task allocation with executable coalitions with (naturally) limited sensing capabilities (i.e., the range and angle restrictions of fiducial sensors). One observation is that robots in the multirobot systems may often be divided into local groups that are spatially separated (e.g., see Figure 6). Although robots in different groups may still be able to communicate, spatial separation (and other kinds of spatial restrictions) can make a large portion of the feasible coalitions not executable for the robots in many situations.

It is assumed in this simulation that there are 4 followers and 8 leaders in the environment. The task is to achieve a localization capability for all of the followers. From the given information, it is not difficult to conclude that the number of possible feasible coalitions is 3824, since any coalition that includes any leader and any follower is feasible. Furthermore, when we assume a non-super-additive environment, the number drops to 192. This number is still not small for task allocation algorithms.

As we discussed, although the number of feasible coalitions can be large, the number of executable coalitions may be limited. It is desirable to use IQ-ASyMTRe to search for executable coalitions based on the current configurations of the robots and environment. To show this, we generate 10 random configurations of the robots and run IQ-ASyMTRe to find the executable coalitions. Figure 7 shows 2 random configurations out of the 10 and Table II shows the results. The table shows the number of followers that can find a coalition to help it localize (i.e., **Foll. Enab.**), as well as the number of executable (**Exec.**) and feasible (**Feas.**) coalitions for any environments and non-super-additive (**n.s.a**) environments with maximum coalition size of 3.

It is obvious to see the reduction of the number of coalitions for all random configurations. We can see from this simulation that the limitation of sensing capabilities can restrict the number of coalitions that need to be considered. By utilizing such a 'disadvantage', one can make task allocation much more efficient. One important note is that to find the executable coalitions, IQ-ASyMTRe checks only from the feasible ones. The feasibility of coalitions is automatically guaranteed by the reasoning process (by requiring necessary information to be retrieved). Furthermore, the search process for checking all feasible coalitions is naturally distributed. By trading off computation that is linear in the number of feasible coalitions, the magnitude of the possible exponential growth is reduced.

### C. Tasks with No Executable Coalitions

However, one obvious issue of task allocation with IQ-ASyMTRe is that tasks may not have executable coalitions. We have proposed an approach that enables the robots to autonomously decompose unsatisfied preconditions of the required task behaviors into satisfiable components to create partial order plans. In this experiment, we provide an example that illustrates how such an approach works.

In this simulation (see Figure 8(a)), there are three tasks to be allocated and each one is for one follower to achieve a localization capability and navigate to the goal. The challenge is that one of the followers (the follower at the bottom) does not have a leader in its sight. Figure 8 shows the snapshots

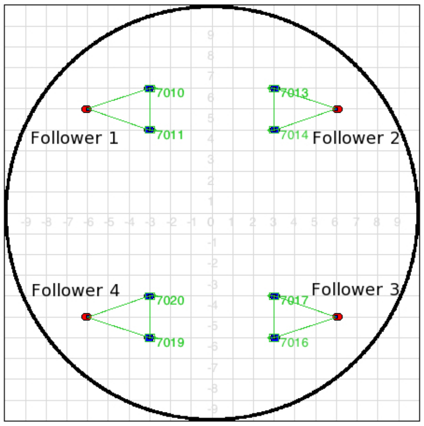
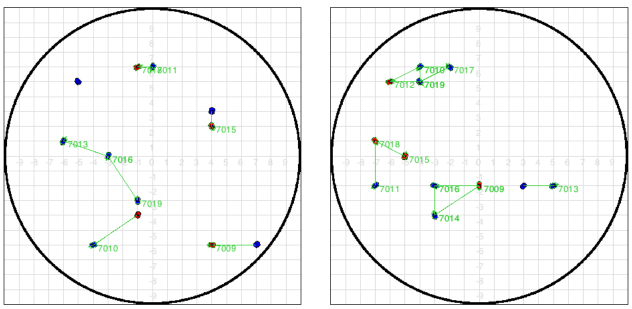



Fig. 6. A configuration with four groups of robots spatially separated. Each group has one follower and two leaders. The range and angle restrictions of the fiducial sensors separate each one from the others.



(a) (b)

Fig. 7. (a) A random configuration corresponding to the entry 8 in Table II. (b) A random configuration corresponding to the entry 10 in Table II.

from an execution of task allocation and execution. The process can be summarized as follows:

*1) Figure 8(a):* Tasks are announced in the Easy Auction phase and the followers invoke the IQ-ASyMTRe algorithms to search for executable coalitions and submit bids for tasks. The bottom follower robot cannot find an executable coalition and hence submits no bids.

TABLE II

EXECUTABLE VS. FEASIBLE COALITIONS

| Conf. | Foll. Enab. | Exec. | Feas. | Exec. n.s.a | Feas. n.s.a |
|---|---|---|---|---|---|
| Fig. 6 | 4/4 | 12 | 3824 | 12 | 192 |
| 1 | 4/4 | 33 | 3824 | 17 | 192 |
| 2 | 3/4 | 13 | 3824 | 9 | 192 |
| 3 | 2/4 | 3 | 3824 | 3 | 192 |
| 4 | 2/4 | 5 | 3824 | 3 | 192 |
| 5 | 2/4 | 6 | 3824 | 5 | 192 |
| 6 | 1/4 | 3 | 3824 | 3 | 192 |
| 7 | 2/4 | 15 | 3824 | 9 | 192 |
| 8 | 4/4 | 4 | 3824 | 4 | 192 |
| 9 | 4/4 | 11 | 3824 | 9 | 192 |
| 10 | 4/4 | 12 | 3824 | 11 | 192 |

*2) Figure 8(b):* Two task assignments are made and two followers start navigating with the leaders. The auctioneer notices that no bids are submitted for a task, so it initiates the IMS Auction phase for the task. Since other followers are executing tasks, they ignore the new auction. The bottom follower receives (again) the task in the IMS Auction phase and submits information task requests to the auctioneer (i.e., *find-entity*).

*3) Figure 8(c):* The auctioneer receives the requests and announces the new task and both the bottom follower and leader submit bids. The auctioneer assigns the task to the bottom follower and it starts executing *find-entity*. (For simplicity, we configure the potential leader to be easily found.) Once a potential leader is found, the follower notifies the auctioneer and the auctioneer re-announces the initiating task in the Easy Auction phase, which is put on hold due to its unsatisfied precondition in the partial order plan.

*4) Figure 8(d):* The bottom follower submits a bid for the task since it now has a leader in its sight, followed by the auctioneer assigning the task to it. Finally, the bottom follower starts navigating.

The novelty of this approach is that robots can autonomously decompose unsatisfied preconditions (i.e., input information) of the required behaviors into satisfiable components to create partial order plans, depending on the current situations. Such an approach utilizes the capability of IQ-ASyMTRe to reason about alternative ways to satisfy these preconditions even when no IMSs are implemented to directly satisfy them.

## V. CONCLUSIONS

In this paper, we show how to layer the IQ-ASyMTRe architecture with task allocation. Furthermore, we show the advantage of task allocation with executable coalitions. The reduction of the number of coalitions is the result of the limited sensing capabilities. IQ-ASyMTRe takes advantage of this 'disadvantage' and searches for executable coalitions on which task allocation is based. Finally, for tasks with no executable coalitions, we introduce a new type of MS and provide a process that can autonomously create partial order plans to satisfy the preconditions of the required behaviors. Simulations are provided to demonstrate these techniques.

For future work, we plan to analyze the impact of our approach on the overall performance of task allocation. Also, more complicated tasks are to be implemented in simulation and with physical robots to further validate the approach.

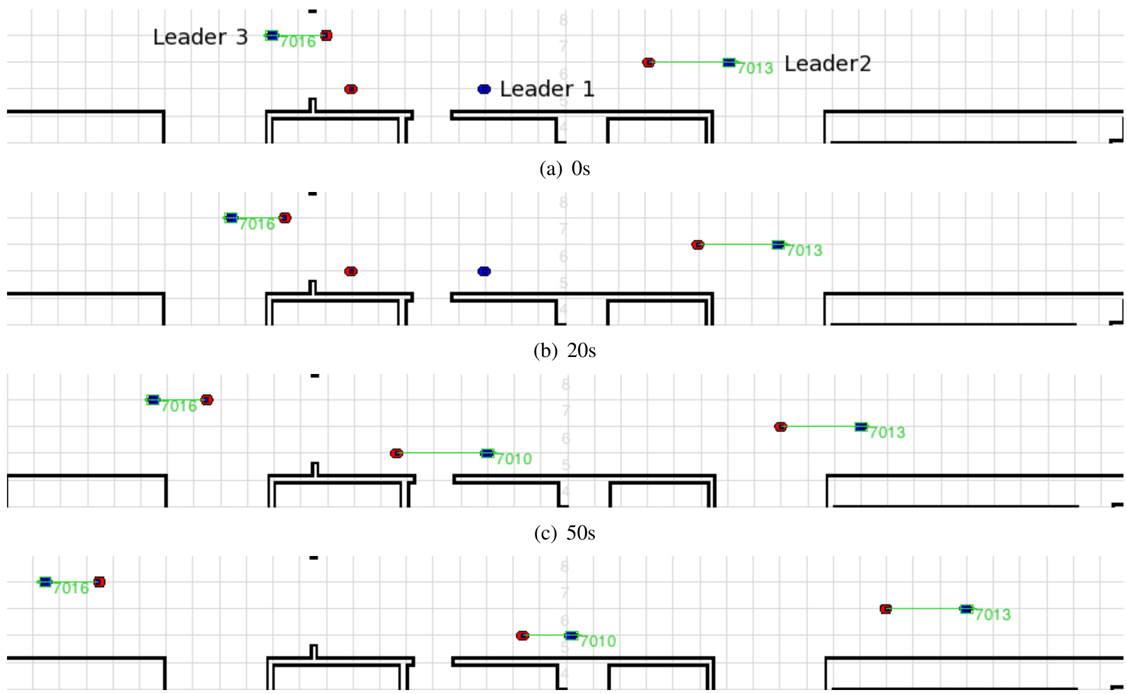
## REFERENCES

[1] S.C. Botelho and R. Alami. M+: a scheme for multi-robot cooperation through negotiated task allocation and achievement. *Proc. of the IEEE Int'l Conf. on Robotics and Automation*, 2:1234–1239, 1999.

[2] A.K. Das, R. Fierro, V. Kumar, J.P. Ostrowski, J. Spletzer, and C.J. Taylor. A vision-based formation control framework. *IEEE Transactions on Robotics and Automation*, 18:813–825, 2002.

[3] M.B. Dias and A Stentz. A free market architecture for distributed control of a multirobot system. In *Proc. of The 6th Int'l Conf. on Intelligent Autonomous Systems*, pages 115–122, 2000.

[4] B.R. Donald, J. Jennings, and D. Rus. Information invariants for distributed manipulation. In *Int'l Workshop on Algorithmic Foundations of Robotics (WAFR)*, pages 431–459, 1997.




(a) 0s

(b) 20s

(c) 50s

(d) 90s

Fig. 8. A scenario created for task allocation with 3 tasks (for the three followers to achieve a localization capability), out of which one task has no executable coalitions. The labels under each subfigure show the execution time in seconds.


[5] B.P. Gerkey and M.J. Mataric. Sold!: Auction methods for multi-robot coordination. *IEEE Transactions on Robotics and Automation, Special Issue on Multi-robot Systems*, 2001.

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

[7] R. Lundh, L. Karlsson, and A. Saffiotti. Plan-Based Configuration of an Ecology of Robots. In *Proc. of the IEEE Int'l Conf. on Robotics and Automation*, pages 64–70, Rome, Italy, 2007.

[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, Jun. 1989.

[9] L.E. Parker and F. Tang. Building multirobot coalitions through automated task solution synthesis. *Proc. of the IEEE*, 94(7):1289–1305, July 2006.

[10] S.J. Russell and P. Norvig. *Artificial Intelligence: A Modern Approach*. Pearson Education, 2003.

[11] T. Sandholm, K. Larson, M. Andersson, O. Shehory, and F. Tohme. Coalition structure generation with worst case guarantees. *Artificial Intelligence*, 111(1-2):209 – 238, 1999.

[12] T. Service and J. Adams. Coalition formation for task allocation: theory and algorithms. *Autonomous Agents and Multi-Agent Systems*, 22:225–248, 2011.

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

[14] P.M. Shiroma and M.F.M. Campos. CoMutaR: A framework for multi-robot coordination and task allocation. In *Proc. of The IEEE/RSJ Int'l Conf. on Intelligent Robots and Systems*, pages 4817 –4824, Oct. 2009.

[15] F. Tang and L.E. Parker. A complete methodology for generating multi-robot task solutions using ASyMTRe-D and market-based task allocation. In *IEEE Int'l Conf. on Robotics and Automation*, pages 3351–3358, Apr. 2007.

[16] Y. Zhang and L. E. Parker. A general information quality based approach for satisfying sensor constraints in multirobot tasks. In *IEEE Int'l Conf. on Robotics and Automation*, 2010.

[17] Y. Zhang and L.E. Parker. IQ-ASyMTRe: Synthesizing coalition formation and execution for tightly-coupled multirobot tasks. In *IEEE/RSJ Int'l Conf. on Intelligent Robots and Systems*, 2010.

[18] R.M. Zlot. *An auction-based approach to complex task allocation for multirobot teams*. PhD thesis, Carnegie Mellon University, Pittsburgh, PA, USA, 2006. AAI3250901.