Winner Determination for Simultaneous Multi-Robot Task Allocation

Fang Tang and Spondon Saha
Computer Science Department
California State Polytechnic University, Pomona
Pomona, CA, 91768
Email: {ftang|ssaha}@csupomona.edu

Abstract

Multi-robot task allocation is an important problem for heterogeneous mobile robots. Simultaneous allocations with which multiple tasks are being allocated concurrently tend to lead to more efficient allocations than online or single task allocations. However, the simultaneous allocation also increases the complexity in the winner determination process, especially when robots are required to collaborate in order to accomplish certain tasks. This paper presents a winner determination algorithm for the simultaneous allocation of multi-robot tasks. The complete approach layers a low-level coalition formation algorithm for solving one multi-robot task with a high-level simultaneous task allocation approach. We implement a tree-based winner determination algorithm with an iterative deepening A* (IDA*) search and show that the algorithm is able to generate the optimal task-coalition mapping in the initial round and the IDA* performs efficiently based on time and space complexities.

Introduction

Multi-robot teams are widely used in today's robotic applications because they are expected to enhance the efficiency and reliability of a solution or they need to explicitly cooperate to accomplish a task. For heterogeneous robot teams, the multi-robot task allocation (MRTA) problem is of major research interest. It determines an efficient mapping between robots and tasks. We are particularly interested in domains where there are multiple tasks being offered for simultaneous allocation and each task may require multiple robots to work closely with one another. According to the taxonomy described in (Gerkey and Mataric 2004), our work can be classified as a single-task (ST) robot, multi-robot (MR) task and instantaneous assignment (IA) problem. In our definition, a multi-robot task is not trivially serializable, and cannot be decomposed further into subtasks that can be completed by individual robots operating independently. This type of task requires multiple robots to act in concert as in a *coalition* to achieve the task objective.

One motivating multi-robot task to consider is the site clearing application, a simplified version of the site preparation task (Parker et al. 2000), which has been identified by NASA as an important prerequisite for human missions to Mars. A heterogeneous robot team is assigned the task to clear obstacles by pushing them to the closest collection site. The area and locations of the robots, obstacles and collection sites are known. Obstacles have different weights and sizes, thus multiple robots need to work together to remove a single obstacle. Note that robots also have different capabilities. They may need to exchange sensing or computational information to work together and remove an obstacle. For instance, a robot with a range sensor can provide the relative angle of the obstacle to another robot with no range sensor, which then enables it to locate and push the obstacle. We further assume that a partial-order planner exists to determine the ordering constraints of removing the obstacles, in case certain obstacles need to be removed before other obstacles can be cleared. Since only some tasks have ordering constraints, the system can allocate a subset of the tasks to the robots for concurrent execution, with each task removing one obstacle. Thus, when making a task allocation decision, robots need to consider more than one task at a time. Additionally, when multiple coalitions are available, the system must determine which coalitions are the best fit to the current set of tasks.

In our prior work, Tang and Parker developed the ASyMTRe-D coalition formation algorithm for autonomous task solutions (Tang and Parker 2007). From our perspective, an individual task cannot technically be categorized in advance as a single-robot or multi-robot task. Instead, whether or not the task requires single or multiple robots depends on the capabilities of the heterogeneous team. The ASyMTRe-D approach is able to find combinations of robot capabilities that can accomplish a single task in either case, depending on the team composition. To handle multiple tasks, our approach layers ASyMTRe-D with a traditional auction-based task allocation approach to assign one task at a time to the robot coalition with the highest bid. In this work, we continue to use ASyMTRe-D to form coalitions at the low level, but extend the single task allocation to simultaneous allocation of multiple tasks.

When multiple tasks are considered, combinatorial auctions ((Sandholm 2002), (Berhault et al. 2003)) are often used for robots to express their desirability on accomplishing combination of tasks. However, these tasks are often single-robot tasks and each robot can win more than one task at a time. A typical example is the multi-robot routing prob-

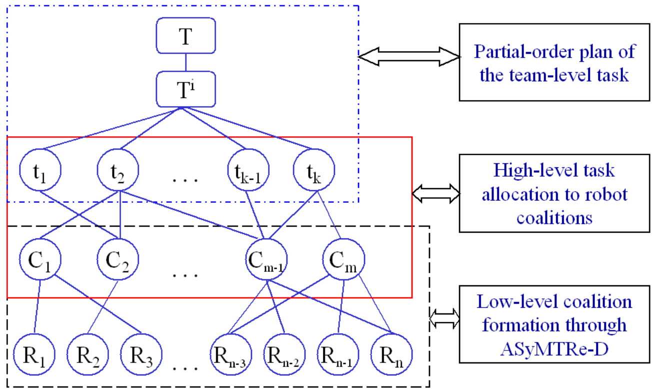

Figure 1: The relationships between tasks, coalitions and robots.

lem, where robots need to visit a set of locations in the environment. It is naturally efficient to group nearer locations together for allocation since a robot can visit multiple locations on a single trip. Our problem domain is fundamentally different from the above problem because a robot coalition can only remove one obstacle at a time since the coalition needs to work together to push an obstacle to a collection site. In other words, our problem is equivalent to a combinatorial auction problem where only task bundles with size one are considered (Dias et al. 2006). The winner determination process however, is more challenging since each winner is a coalition (subset) of robots instead of a single robot. As a result, our approach needs to guarantee that the winning coalitions do not share the same team members. In this paper, we introduce our winner determination algorithm using an iterative deepening A* (IDA*) search that is adapted from the multi-agent system. The contributions of this work are twofold. First, it successfully adapts the winner determination algorithm used in combinatorial auction to the MRTA problem. Second, it can be layered with the ASyMTRe-D approach and produces a complete approach to autonomous task solution generation for multiple tasks.

The remainder of the paper is organized as follows. We first give an overview of our complete approach to autonomous solution generation, with a focus on the implementation of the winner determination algorithm using IDA*. We then evaluate the performance of the winner determination algorithm and compare it with a baseline algorithm. We then give a background of the related work and provide concluding remarks in the conclusion and future work.

The Approach: Layering Coalition Formation with Task Allocation

Overview

The main idea of our approach to task allocation is illustrated in the algorithm listed in Table 1 and Figure 1. We assume that there exists a partial order planner that finds a set of subtasks T^i from the team-level task T according to the partial order plan. Here, T^i represents the set of subtasks

Table 1: Simultaneous Multi-Robot Task Allocation

Input: (T, R)

1. Find the set of tasks T^i , such that both the ordering constraints and the preconditions of tasks are satisfied.
2. Configure solutions for each task t_j in T^i by forming a set of coalitions C^i, based on t_j's objective and the current team capabilities.
3. Allocate tasks in T^i to coalitions in C^i, such that:
   - The task contribution value is maximized for T^i.
   - A coalition can win at most one task at a time. Assuming $C' \subseteq C^i$ is the set of winning coalitions selected to perform tasks in T^i, then the following condition must be satisfied: $\forall_{C'_i, C'_j \in C', i \neq j}, C'_i \cap C'_j = \emptyset$.
4. Monitor the execution of tasks. If the entire task is not complete, start the allocation process (go to step 1) when robots are within Δt time to complete their current tasks. Otherwise, exit.

$\{t_1, t_2, ..., t_k\}$ that satisfies the ordering constraints and any precondition requirement, and thus can be allocated concurrently. At the low level, coalitions $\{C_1, ..., C_m\}$ from the team of robots $\{R_1, ..., R_n\}$ are formed by ASyMTRe-D to address the given set of subtasks T^i. Each coalition C_i may include a varying number of robots. These coalitions are not mutually exclusive and may share the same team members, or could even be identical coalitions. The coalitions then compete for the assignment of subtasks using an auction-based task allocation approach with our proposed algorithms for winner determination.

Low-Level Coalition Formation

To better understand the integrated system, we now describe our previous work on coalition formation, called ASyMTRe-D (Parker and Tang 2006). The ASyMTRe-D approach has been developed for addressing the formation of heterogeneous robot coalitions that solve a single single-robot or multi-robot task. We share the same motivation behind coalition formation as mentioned in (Shehory 1998); that is, robots in a coalition should work together to share resources and cooperate on task execution due to their decision that they would benefit more from working together as a coalition than they would from working individually.

The fundamental idea of ASyMTRe-D is to change the abstraction that is used to represent robot competences from the typical "task" abstraction to a biologically-inspired "schema" abstraction and provide a mechanism for the automatic reconfiguration of these schemas to address the multi-robot task at hand. To achieve this, we view robot capabilities as a set of environmental sensors that are available for the robot to use, as well as a set of perceptual schemas, motor schemas, and communication schemas that are preprogrammed into the robot at design time. The ASyMTRe-D approach autonomously connects schemas at run time instead of using pre-defined connections. According to the information invariants theory (Donald 1995), the information needed to activate a certain schema or to accomplish a task remains the same regardless of the way that the robot may obtain or generate it. We can therefore label inputs and outputs of all schemas with a set of information types, such as *self global positioning data*. Two schemas can be connected if their input and output information labels match. Thus, schemas can be autonomously connected within or across robots based upon the flow of information required to accomplish a task. With the run time connection capabilities, task solutions can be configured in many ways to solve the same task or can be reconfigured to solve a new task. We have implemented the ASyMTRe-D approach using a distributed negotiation protocol. We have further extended ASyMTRe-D such that it generates multiple eligible coalitions for one task. However, the size of a coalition is limited to a predefined maximum coalition size assuming robots work in a non-super-additive environment (Shehory 1998).

To evaluate the performance of robot coalitions, we calculate a task contribution value for each coalition that represents the contribution that the coalition can bring to the overall task completion. This value is determined by multiple factors such as the robot-inherent cost and the task-specific cost. The robot-inherent cost is calculated by the sum of the sensing and/or computational cost of activating the required schemas in a coalition. To guarantee the solution quality, the cost is also combined with the success probability of the coalition through a weighted linear function. In ASyMTRe-D, each schema or sensor has a predefined cost and success probability associated with it. For example, using stereo vision to calculate the depth information might be less accurate than using laser range data, while the latter approach consumes more power and thus costs more. The task-specific cost is determined by the task completion time and other task-related factors when necessary. In our site clearing task, the completion time is determined by the locations of the box and the robot coalition. We convert the above two costs into a numeric value through another weighted linear function that is domain-specific and determined by the user. In an idea situation, a higher contribution value means that a task with a shorter completion time is assigned to a coalition with a lower cost and a higher success rate.

The high-level task allocation approach is implemented with an auction process in order to handle multiple tasks, which includes typical steps such as task announcement, coalition formation through ASyMTRe-D, bid submission, winner determination and award acceptance. Since our main contribution in this paper is the winner determination process (step 3 in the allocation algorithm), we focus on it in the following discussion. More information about the ASyMTRe-D approach for coalition formation and layering ASyMTRe-D with a high level auction approach can be found at (Tang and Parker 2007).

Winner Determination for Simultaneous Task Assignments

Since robots have different capabilities, the coalitions they form also produce different contribution values given the same task. If we allow each coalition to win multiple tasks, a potential problem arises because it is possible that a more capable coalition (thus with a higher contribution value) will receive multiple tasks while some less capable coalitions will remain idle. In this way, the system does not fully utilize the capabilities of the whole robot team. Since robots cannot multi-task in our site clearing problem domain, our approach needs to ensure that each coalition can win at most one task at a time and no two coalitions share the same team member(s). With this approach, the set of tasks can be distributed among different coalitions, instead of being assigned to the single best coalition. Eventually, the set of tasks can be accomplished in a more efficient manner. To sum up, the goal of the winner determination step is to guarantee that: (1) the overall task contribution value for T^i is maximized and (2) each coalition can win at most one subtask with no winning coalitions sharing the same team member(s).

Table 2: An example of the list of bids submitted.

Task	Coalitions submitted
t_1	$\{R_1, R_2, 3\}$, $\{R_2, R_3, 4\}$ and $\{R_3, 2\}$
t_2	$\{R_1, R_2, 4\}$, $\{R_2, 5\}$ and $\{R_3, 3\}$
t_3	$\{R_1, R_3, 4\}$, $\{R_2, R_3, 4\}$ and $\{R_3, 8\}$

The winner determination process is similar to a combinatorial optimization problem called the Set Partitioning Problem (SPP). Here, we need to partition the set of robots such that each partition is a coalition of robots that can accomplish one subtask and the overall contribution value of the partitions is maximized. The SPP problem has been examined thoroughly in the multi-agent society for agent coalition formation (Shehory 1998) and combinatorial auction (Sandholm 2002). In combinatorial auctions, multiple tasks are put up for auction and each bidder can bid on subsets of tasks and win more than one subset. In (Sandholm 2002), an optimal winner determination algorithm is developed for determining the winners for a set of single-robot tasks, where each bidder can win multiple tasks. Inspired by the above algorithm, we also use tree search to achieve our goal of assigning coalitions to tasks to maximize the overall contribution value. The input to our algorithm is a list of bids, with each bid containing the information such as the robot coalition members, its contribution value, and the subtask it can accomplish. Unlike (Sandholm 2002), we assign multi-robot tasks to coalitions of robots. Additionally, we do not exclude lower bids that are submitted for the same task since we do not allow a coalition to win more than one task.

The bids are preprocessed such that each task t_i maintains a set of coalitions C^{t_i} that bid for t_i, as shown in Table 2. The last element in the coalition set represents its contribution value. A dummy bid with an empty coalition and a zero bid value is inserted to each C^{t_i}, corresponding to the case where no coalition can accomplish t_i. These bids are then used to build our search tree such that each path from the root to a leaf node represents exactly one possible partition of the robots. The contribution value for each partition is the sum of the bid values of that path. Starting with a dummy root node, the following two rules ensure that the above property is maintained. First, each path is composed

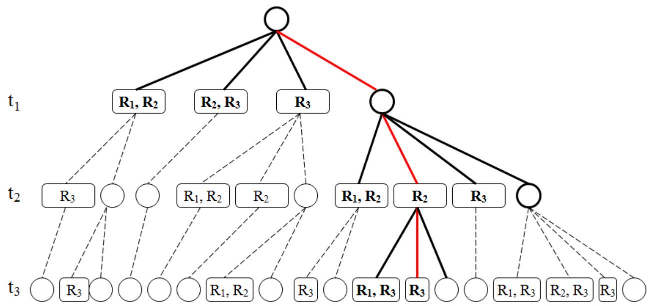

Figure 2: The bid tree corresponding to the bids submitted in Table 2. The dotted lines represent the branches that can be pruned with the IDA* search. The red lines represent the optimal allocation with a total contribution of 13.

of nodes with disjoint coalitions, that is, bids that do not share team members with each other. When inserting a bid, we need to make sure that the coalition members in this bid do not appear anywhere on the current path from the root to its parent node. If no bid satisfies this condition, we can still insert a dummy bid. Second, each node (coalition) on a given path attacks a different task on that path. To achieve this, we make sure nodes with the same depth are composed of bids submitted for the same task t_i. The order of the tasks which we use to build the search tree does not affect the final result. Figure 2 shows a complete search tree corresponds to the bids submitted in Table 2.

Similar to Proposition 3.3 in (Sandholm 2002), the worst case of the tree size (number of nodes) is analyzed below. The maximum depth of the tree is the number of tasks (k) being allocated. Let the total number of bids (including the dummy bids) be m and $n_i = |C^{t_i}|$. The upper bound on the tree size is given by $n_1 \times n_2 \times ... \times n_k$ with $n_1 + n_2 + ... + n_k = m$. The maximum possible number of nodes is bounded by $n_1 = n_2 = ... = n_k = m/k$. Therefore, the number of nodes in the tree is no greater than $(m/k)^k$. The bound shows that worst case time/space complexity of the algorithm is polynomial in the number of bids if the number of tasks being considered in the current round is fixed.

IDA*

At each iteration of the IDA*, we perform a depth-first search, cutting off a branch when its estimated maximum possible contribution value is less than a given threshold, which starts at the estimate of the contribution value of the root, and decreases for each iteration of the algorithm. At each iteration, the threshold used for the next iteration is the maximum contribution of all values that are less than the current threshold. To guarantee the optimality, we apply an admissible heuristic. Assume n is a node, the heuristic $h(n)$ is defined as the sum of the maximum contribution for each *unassigned* robot on the current path of n that performs the rest of the *unallocated* tasks on the same path. Formally speaking, we have:

$$h(n) = \sum_{R_i \notin n.bid} Contribute(R_i) \qquad (1)$$

$$Contribute(R_i) = \max_{R_i \in t_j.bid_k.robots} \frac{t_j.bid_k.val}{t_j.bid_k.size} \qquad (2)$$

Where $n.bid$ is the set of robots that appear on the current path of n and *Contribute*(R_i) represents the maximum potential contribution of a robot R_i among all unallocated tasks. To calculate *Contribute*(R_i), we first find the set of unallocated tasks ($\{\forall_j | t_j$ is unallocated on the current path of $n.\}$) that robot R_i participates as a coalition member, and then compute a set of potential contribution values by dividing the bid value ($t_j.bid_k.val$) by the number of robots in that coalition ($t_j.bid_k.size$). The highest value in the set represents the maximum potential contribution for a robot R_i. This heuristic will never underestimate the legitimate contribution a robot could make on the current partition and thus it is an admissible heuristic for this search problem. The result of the search tree based on IDA* is shown in Figure 2 corresponding to the bids submitted in Table 2. Here, the dotted lines represent the branches been cut. In this example, the initial contribution in the first iteration is 15 and it is reduced to 13 for the next iteration and the optimal allocation is found afterwards.

Experiments

We have implemented the IDA* for winner determination in Python. We have also implemented a baseline breadth-first search (BFS) algorithm for comparison. To validate the correctness and evaluate the performance of the IDA*, we generate bids with different distribution as described below. From the theoretical analysis, we know that the size of the bid tree is largely influenced by two factors: (1) the number of bids submitted and (2) the number of tasks been allocated. Thus, in the experiment, we used a fixed small-size robot team with 10 robots, with a varying number of tasks from 10, 15, to 20 and a varying number of bids from 10 to 160. The following two bid distributions are considered:

- *Random*: For each bid, we pick the number of robots (coalition size) randomly from 1, 2, ..., 10. The contribution value is also random.
- *Uniform*: This distribution accepts bids of a fixed coalition size, in our case, 3 robots. We select this number because this size was frequently used in our previous coalition formation simulation and physical experiments. We want to keep the coalition size relatively small, since a large coalition would generate more interferences among robots. The contribution value is random.

Each data point in our experiment is averaged over 10 runs. Each run only considers one round of allocation such that leftover tasks from the first round are reinserted into the task queue for the future allocation. The experiments we have done are threefold. We first measure the correctness of both algorithms. We then measure and compare the performances of both the BFS and IDA* algorithms with regard to their time and space complexities. For space complexity, we record the number of nodes generated in the search process. The objective is to see how IDA* behave with an increasing number of bids and tasks, without affecting the solution quality.

Results and Discussion

Table 3: Space Complexity (Number of Nodes in the Bid Tree) Comparison of Randomly Distributed Bids

Tasks	Bids	10	20	30	40	50
10 tasks	IDA*	24	75	105	136	345
	BFS	106	979	4681	9205	29945
15 tasks	IDA*	26	93	146	189	420
	BFS	119	1154	6136	24402	57530
20 tasks	IDA*	25	78	198	254	484
	BFS	163	2042	9788	36604	130201

Table 4: Time Complexity (in seconds) Comparison of Randomly Distributed Bids

Tasks	Bids	10	20	30	40	50
10 tasks	IDA*	0.02	0.12	0.31	0.43	1.53
	BFS	0.01	0.11	0.59	1.32	4.60
15 tasks	IDA*	0.02	0.14	0.33	0.56	1.69
	BFS	0.01	0.13	0.70	3.42	10.17
20 tasks	IDA*	0.03	0.15	0.44	0.73	1.72
	BFS	0.02	0.23	0.96	4.91	21.70

Table 5: Space Complexity (Number of Nodes in the Bid Tree) Comparison of Uniformly Distributed Bids

Tasks	Bids	40	70	100	130	160
10 tasks	IDA*	61	114	171	228	271
	BFS	2364	2364	5240	9643	17089
15 tasks	IDA*	95	159	258	317	443
	BFS	974	3419	8051	15113	20586
20 tasks	IDA*	110	227	348	455	585
	BFS	1284	4531	10365	20586	34796

In our experiments, the BFS will be given enough time to perform a complete search and the best solution will be returned. Since our bid tree has a finite branch factor and depth, the solution returned by BFS is the optimal solution. We then compare the solution from IDA* with the solution found through BFS and found that the total contribution value are always the same for both algorithms. However, the task-coalition mappings are different in many cases because when multiple optimal allocations are available, the BFS only keeps track of the first best solution.

Tables 3 and 5 compare the BFS and the IDA* algorithms with regard to their space complexities. We can see that the size of the bid tree increases when we increase the number of bids and the number of tasks, which matches our theoretical analysis. IDA* outperforms BFS in terms of the number of nodes expended. We also notice that the bid tree generated

Table 6: Time Complexity (in seconds) Comparison of Uniformly Distributed Bids

Tasks	Bids	40	70	100	130	160
10 tasks	IDA*	0.09	0.34	0.78	1.47	2.40
	BFS	0.05	0.25	0.75	1.74	3.54
15 tasks	IDA*	0.13	0.54	1.02	1.78	3.13
	BFS	0.07	0.41	0.96	2.24	4.35
20 tasks	IDA*	0.14	0.55	1.22	2.27	3.71
	BFS	0.08	0.38	1.10	2.58	5.13

by the randomly distributed bids is much larger than the tree generated by the uniform distribution. This is because the bids in a uniform distribution have a higher chance of sharing the same team member(s) and thus result in a sparser bid tree. To see the tendency of the growth clearly, we have included more bids (from 40 to 160 bids) in our uniform distribution. Tables 4 and 6 compare the time complexities of both algorithms. For the random distribution, the time complexities for both algorithms well represent their space complexities: (1) the more tasks and bids that are considered, the longer it takes to find the optimal solution; and (2) IDA* takes much less time than BFS when the difference in the bid tree is large. We should note that IDA* has the overhead of heuristic evaluation for every node. We conclude that both the BFS and IDA* are able to generate the optimal task-coalition mappings while the IDA* performs more efficiently in its space and time requirements.

Related Work

In past work, many task allocation approaches deal with the single-robot (SR) and instantaneous assignment (IA) ((Parker 1998) and (Gerkey and Mataric 2002)) problem. Typically, a high- level task is decomposed into independent subtasks or roles either by a general autonomous planner or by a human designer, which can be achieved concurrently by individual robots. The IA mechanism enables subtasks to be allocated iteratively until all subtasks are assigned. A greedy algorithm or online assignment is usually applied such that only one task is under consideration at a time to find the best robot-task pair. Our task allocation strategy is different in that we are allocating MR-tasks. Assume there are more tasks than eligible coalitions, our algorithm optimally solves the initial assignment by maximizing the total contribution value and then uses the Greedy algorithm to assign the remaining tasks as the robots become available.

Some recent approaches are beginning to allocate multi-robot tasks. The work of (Zlot and Stentz 2006) addresses complex multi-robot tasks by trading task trees, in which subtasks are achieved in order according to their precedence constraints. However, their tasks can be decomposed into multiple loosely-coupled single-robot tasks, which is different from our problem domain. The work of (Kalra, Ferguson, and Stentz 2005) addresses tight coordination in a security sweep domain. Their coordination is achieved by trading joint plans that generate more revenue than individual plans. The work of (Jones et al. 2006) also addresses tight coordination in a treasure-hunt problem through allocating roles that are predefined through play scripts. The articles mentioned above are all built upon an interactive task allocation system using a market-based economy (Dias 2004). The fundamental difference of our work lies in the low-level ASyMTRe-D for coalition formation that autonomously generates the team solution instead of using a predefined solution. The winner determination algorithm adapted from the multi-agent system completes our coalition formation approach by allowing multiple tasks being assigned simultaneously. The work of (Vig and Adams 2006) also adopts the coalition formation technique from multi-agent systems. In their work, tasks are assigned one at a time in an iterative manner until all tasks are cleared. In our allocation algorithms, we make multiple task-coalition assignments simultaneously with a centralized winner determination algorithm.

A recent work (Jones, Dias, and Stentz 2009) addresses problems in the multi-robot and time-extended allocation domain. In their application, fire-fighting trucks need to visit different locations to extinguish fires and they may need the help of cleaning robots to clear the routes with debris. A tiered-auction approach is implemented with the first tier auctioning a cluster of locations based on distance and each cluster is assigned to the truck with the highest bid, and the second tier handling collaboration between the truck and the cleaning robot once the truck wins a task cluster. One potential problem of the first tier auction is that it is possible that a truck will be assigned too many tasks because of its higher bids. If the tasks are more evenly distributed among the trucks, then the overall mission can be completed in a more timely manner.

Conclusion and Future Work

Simultaneous multi-robot task allocations, where multiple tasks are being considered at the same time, can lead to more efficient allocations than traditional iterative allocations, where tasks are considered sequentially. We have described our approach for layering the low-level ASyMTRe-D for generating multi-robot task solutions through coalition formation, with an auction-based simultaneous task allocation approach for assigning tasks to coalitions. The winner determination algorithm considers all possible mapping of tasks to coalitions that are proposed by the robots and generates an optimal mapping between tasks and coalitions. Our ongoing work includes the demonstration of the integrated work on the site clearing task with a comparison of simultaneous allocation with single task allocation.

References

Berhault, M.; Huang, H.; Keskinocak, P.; Koenig, S.; Elmaghraby, W.; Griffin, P.; and Kleywegt, A. 2003. Robot exploration with combinatorial auction. In *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems*.

Dias, M. B.; Zlot, R.; Kalra, N.; and Stentz, A. 2006. Market-based multirobot coordination: a survey and analysis. *Proceedings of IEEE*. Special Issue on Multi-Robot Systems.

Dias, M. B. 2004. *TraderBots: A New Paradigm for Robust and Efficient Multirobot Coordination in Dynamic Environments*. Ph.D. Dissertation, Robotics Institute, Carnegie Mellon University.

Donald, B. R. 1995. Information invariants in robotics. *Artificial Intelligence* 72:217–304.

Gerkey, B. P., and Mataric, M. J. 2002. Sold! auction methods for multi-robot coordination. *IEEE Transactions on Robotics and Automation* 18(5):758–768.

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

Jones, E.; Browning, B.; Dias, M. B.; Argall, B.; Veloso, M.; and Stentz, A. 2006. Dynamically formed heterogeneous robot teams performing tightly-coordinated tasks. In *Proceedings of IEEE International Conference on Robotics and Automation*.

Jones, E.; Dias, M. B.; and Stentz, A. T. 2009. Time-extended multi-robot coordination for domains with intra-path constraints. In *Robotics: Science and Systems (RSS)*.

Kalra, N.; Ferguson, D.; and Stentz, A. 2005. Hoplites: a market-based framework for complex tight coordination in multi-robot teams. In *Proceedings of IEEE International Conference on Robotics and Automation*.

Parker, L. E., and Tang, F. 2006. Building multi-robot coalitions through automated task solution synthesis. *Proceedings of IEEE*. Special Issue on Multi-Robot Systems.

Parker, L. E.; Jung, D.; Huntsberger, T.; and Pirjanian, P. 2000. Opportunistic adaptation in space-based robot colonies: Application to site preparation. In *Proceedings of World Automation Congress*.

Parker, L. E. 1998. ALLIANCE: An architecture for fault tolerant multi-robot cooperation. *IEEE Transactions on Robotics and Automation* 14(2).

Sandholm, T. 2002. Algorithm for optimal winner determination in combinatorial auctions. *Artificial Intelligence* 135:1–54.

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

Tang, F., and Parker, L. E. 2007. A complete methodology for generating multi-robot task solutions using ASyMTRe-D and market-based task allocation. In *Proceedings of IEEE International Conference on Robotics and Automation*.

Vig, L., and Adams, J. A. 2006. Multi-robot coalition formation. *IEEE Transactions on Robotics* 22(4).

Zlot, R., and Stentz, A. 2006. Market-based multirobot coordination for complex tasks. *International Journal of Robotics Research* 25(1). Special Issue on the 4th International Conference on Field and Service Robotics.