Hierarchical Distributed Task Allocation for Multi-Robot Exploration

John Hawley and Zack Butler

Abstract In order to more effectively explore a large unknown area, multiple robots may be employed to work cooperatively. When properly done, the group allocates specific portions of the overall exploration task to different robots such that the entire environment is explored with minimal excess effort. In this work, we present a new hierarchical market-based approach to this allocation problem. Our approach builds on standard auction approaches to provide agents with a mechanism to independently form coalitions and to divide a coalition into smaller coalitions in response to the progress of their cooperative exploration process. These coalitions allow a subset of the team to move together efficiently, especially in constrained environments when there are few avenues for exploration. We also present implementation and simulated experiments which show how this natural hierarchy forms and can lead to more efficient exploration than using a greedy allocation technique or without the use of coalitions.

1 Introduction

Exploration of unmapped terrain is a task well-studied in robotics, and is well suited to multi-robot systems. Teams of robots can fan out and visit locations in parallel to make the overall discovery process more efficient, and a variety of approaches have been proposed to coordinate this process. One common approach to coordinating multiple robots is through the use of market-based schemes for task allocation [5, 6, 13, 16]. When a new task is given to the team (or discovered by a team member, in the case of exploration), the robots bid on the right to take on that task. Bids are computed based on the difficulty of the robot to accomplish the task, creating an

Computer Science Dept.
Rochester Institute of Technology
jph1996@gmail.com, zjb@cs.rit.edu

essentially greedy assignment of tasks. In the case of exploration, tasks will generally take the form of a location or region to be visited.

Depending on the particular type of mission, there may be a surplus of tasks or a surplus of robots (or both at different times during the mission). Exploration of open terrain will generally have a surplus of tasks, but in indoor environments there may be few tasks, such as one for each hallway currently being explored. Most market-based systems for task allocation are designed for the former case (a surplus of tasks), and simply try to assign each task or set of tasks to the best available robot. When there is a surplus of robots, those without a task to accomplish simply remain idle. In the context of exploration, this may not be the best choice, as new tasks will be generated on the frontier of explored area. For example, when one robot is exploring a hallway and discovers a four-way intersection, it will generate three new tasks and we would like to have three robots close at hand if possible to make the process more efficient.

To address these issues, we propose a method through which robots can autonomously form coalitions during the exploration process via a market-based mechanism. That is, each robot decides for itself whether it is more profitable to take on a task for itself or join up with a group that already has one or more tasks. The coalition formation and dissolution is performed in addition to a standard market-based task allocation technique to handle the assignment of the exploration tasks.

1.1 Related Work

As mentioned, there have been many different approaches to multi-robot exploration. In general, they consist of determining locations to visit and assigning those locations to robots. For the locations, a common approach is to use frontiers [1, 7, 11, 15], identified as contiguous groups of map cells that represent explored open space adjacent to unexplored space¹. A goal point is created for each such group, and is located at the arithmetic mean of all points in the group. Other approaches to goal identification that have been implemented include random point selection, greedy exploration, map segmentation [14] and quad-tree subdivision [16]. Once determined, a variety of approaches exist to assign these to robots. Some rudimentary but successful approaches simply direct an agent towards the nearest goal point [15]. In the case where multiple agents are participating in the exploration, however, this task becomes far more complex. In such cases, more advanced techniques are often employed, including greedy mechanisms [1, 11], optimal centralized approaches [14], genetic algorithms [7], Voronoi-based approaches that can implicitly keep robots in different areas [3] and market-based mechanisms [6, 10, 16].

Among these techniques, market-based allocation strategies are quite popular. In these strategies, agents negotiate with each other and treat goal points as a commodity that they exchange. Such exchanges are determined by auction mechanisms,

¹ Our implementation uses frontiers to generate goal points, but the method is intended to function equivalently for other methods of goal point generation.

though the specific auction mechanism used varies between implementations. In some implementations, single-round single-item sealed-bid auctions that closely resemble greedy allocation strategies are preferred [16]. In other more distinctly market-based implementations, multi-round auctions may be used so that bids can account for the effects of previous allocations [4, 6], particularly in the calculation of goal point utility. Other issues addressed include handling constraints of communication across a dispersed robot team within the context of bidding [8, 10], which is important from a practical standpoint but is not considered in this work. Occasionally, combinatorial bids are used, in which agents bid on multiple goals in batches rather than individually [5], as it can be advantageous for an agent to pursue groups of nearby goals rather than treat each goal independently. Our work is similar in that an auction mechanism is used to assign goals to robots. However, we use a second auction process in parallel that assigns robots to coalitions. In this way, the coalitions will form naturally and in a purely decentralized way depending on whether the robot finds it more advantageous to pursue its own goal or join a nearby team. The work in [2] also involves groups of robots in a market-based allocation, but in that case leader robots can reassign tasks among ad-hoc groups for a more optimal assignment, whereas we are considering longer-term coalitions with a common goal.

Coalition formation has been addressed in different contexts as well. Often, these works consider tasks which can or must be completed by a team of agents instead of a single agent. A foundational work in this area is that of Shehory and Kraus [9], which includes distributed algorithms for coalition formation with provable bounds on task completion efficiency. In a more closely related context, the AsyMTRe-D algorithm [12] allows robots to create small coalitions based on their capabilities to solve complex tasks. As such, it makes decisions on a discrete basis to form necessary groupings rather than the real-valued numeric bidding used here to form opportunistic groups.

2 Hierarchical Exploration

Our exploration technique includes both goal assignment and formation as well as maintenance and dissolution of coalitions through different auction mechanisms. The first type of auction is a *Goal Auction*, in which agents offer and bid on goals, similar to existing mechanisms for task allocation. The second type is an *Agent Auction*, in which an agent auctions its services in the event that it does not have its own goals to pursue, potentially forming a coalition with other agent(s). These two auction mechanisms take place asynchronously, but care must be taken so that an agent does not transfer a goal in a *Goal Auction* that it has used to make a bid in an *Agent Auction*.

2.1 Goal Auctions

In order for an exploration strategy to be truly distributed, there must be a sharing of responsibility for goals among agents. The agent initially responsible for a goal is trivially the agent that discovered the open space to which the frontier is adjacent. During the course of exploration, however, the agent that generated a goal point may become no longer the most optimal agent for exploring that goal point. In this work, we use a market architecture as a simple and effective way for agents to transfer responsibility for goal points. As agents traverse the environment, they build a local map and periodically share that information with the other agents in the team. When an agent generates new goal points, usually by reaching a current goal point, it can hold an auction so that goal points may be transferred to more optimal agents. Depending on the structure of the environment, a frontier may be discovered or enlarged on the way to another goal, and will be put up for auction if so. An agent may also periodically hold auctions even when no new goals are discovered so that goals can still be transferred between agents during long travels through explored regions. While many complex auction strategies exist, here we use simple single-round highest-bidder closed auctions.

In order for agents to appropriately bid on goals, measures of *cost* and *utility* of goals are required, as in much previous work in market-based allocation. Here, cost is calculated as the distance an agent must travel to reach a goal point. Since the environment is only partially known, the cost is optimistically calculated by treating unexplored space as open space within a standard A* search. Utility is defined in a way dependent on the type of goal used, but generally describes the expected increase in explored area from visiting that goal. The value of a goal is then calculated as $value = utility - \beta \cdot cost$ where β represents a coefficient representing the relative values of *cost* and *utility*. To compute utility when frontiers are used as goal locations, we estimate how much unexplored space would be revealed by that agent (based on its sensing radius) were it to be at that particular goal point, resulting in the *expected information gain* [11] of that goal point. We note however that the general form of the hierarchical task allocation that we present does not rely on any particular definition of utility.

2.2 Coalitions

In order to hierarchically distribute goals to agents, agents can form coalitions. Here, we define a *coalition* as a set of agents simultaneously and intentionally moving to explore the same goal point. A coalition is comprised of exactly one *supervisor* and zero or more *workers*. While an agent is deciding what to do next, it is in a third state, *retasking*. An agent is always in exactly one of these three states. See Fig. 1 for a representation of the transitions between these states.

The *supervisor* of a coalition is the agent responsible for that coalition's goal. When an agent does not have any goals that it is responsible for, it can obtain a

Fig. 1 State diagram describing the transitions between *supervisor*, *worker*, and *retasking* states.

goal from another agent. In some cases, this entails joining another agent in a coalition. Coalitions necessarily form when there are more agents than available goals, which is often the case in highly structured environments. When a coalition has been formed to pursue a goal and the resulting exploration of that goal reveals two or more new goals, that coalition will divide into smaller coalitions so that the newly generated goals can be effectively explored. In this way, coalitions hierarchically divide and allocate tasks accordingly.

Workers are agents who have joined the supervisor because they do not have any goals of their own to pursue. Each other agent is therefore trivially the supervisor of a coalition of size one – the coalition containing only that agent. Agents that are workers do not have any goals for which they are responsible and therefore do not hold auctions to transfer goals to other agents and cannot be supervisors. In addition, an agent will belong to exactly one coalition at any time. The supervisor of a coalition is responsible for notifying the workers of that coalition of any changes to the current goal of the coalition.

2.3 Coalition Formation

The mechanism used to form coalitions is similar to the market used to allocate goals to agents. However, instead of holding a goal auction, an agent holds an agent auction, which allows it to discover the most profitable goal for it to pursue. The

agent auction is initiated by broadcasting a *Request-for-work* message. See Fig. 2 for more details regarding the interaction between an auctioneer and bidders that takes place in response to a *Request-for-work* message.

Fig. 2 *Request-for-work* mechanism, initiated by an agent that does not have any of its own goals to pursue.

In order to compare joining a coalition to alternative courses of action, an expected profit must be calculated for a potential coalition. The profit of a coalition can be calculated by

$$Profit = \frac{\max_{i \in A}(Utility(i, g)) - \beta \cdot \sum_{i \in A} Cost(i, g)}{|A|}$$

where A is the set of agents belonging to the coalition and g is the coalition's goal. In the case where this results in a negative value for *Profit*, the formula

$$Profit = \left(\max_{i \in A}(Utility(i, g)) - \beta \cdot \sum_{i \in A} Cost(i, g) \right) \cdot |A|$$

must be used so that larger coalition sizes are penalized (i.e. given more negative profit values) rather than rewarded. Since there is no way to know how many avenues of exploration a goal will produce, it is assumed that smaller coalitions provide a more even distribution of workers among available goal points and are therefore desirable. This is relevant as an agent may find it best to join some coalition if it has no goal of its own, and so it is possible the smallest negative profit will be chosen.

2.4 Coalition Maintenance

In many cases, particularly in highly structured environments such as hallways, exploration of a goal point produces only a single new goal point. In these cases, it is sensible for a coalition to continue on to the new goal. This is accomplished by the supervisor sending a retask message to coalition workers informing them of the new goal to pursue. A worker may leave a coalition at any time, but this retasking provides a particularly opportune time for workers to consider alternative courses of action based on the utility of the new goal.

The case in which a coalition explores a goal that results in multiple new goals requires particular attention. If a worker discovers a new goal point, that worker will quit the coalition and become its own supervisor. It will then respond to future *Request-for-work* broadcasts to obtain its own workers. If the supervisor of a coalition discovers multiple goal points, it is the supervisor's responsibility to decide which workers will pursue which goals. This can be done in different ways, but in our implementation, we have chosen a greedy approach, as follows. First, agents are ordered by topological distance to the supervisor, including the supervisor, which trivially has a distance of zero. Each agent is then assigned to the most profitable goal for that agent, beginning with the supervisor. The first agent assigned to a goal will become a supervisor responsible for that goal, and any further agents assigned to the same goal will be transferred to the new supervisor as workers. Once an agent has become its own supervisor, it is no longer affiliated with the agent that was previously its supervisor.

In order to accommodate all these interactions, three types of retask messages are required. A *Retask-simple* message simply instructs a worker to calculate a path to and pursue a new goal point. A *Retask-become-supervisor* message instructs a worker to become a supervisor that is responsible for the included goal point. Implicitly, the new supervisor is to calculate a path to and pursue the new goal. A *Retask-change-supervisor* message instructs an agent to join a new supervisor's coalition. Upon doing so, the worker will be given a new goal point to pursue.

Fig. 3 Three maps used for experiments: sparse, dense, structured.

2.5 Coalition Dissolution

Coalitions may be dissolved for a number of reasons. A worker may choose to quit its current coalition and reevaluate a new task at any time. This is particularly useful when the worker is far away from the coalition’s goal, since the state of the exploration changes over time, and it is possible, if not likely, that a better alternative will arise for the worker. A worker may also receive its own goal, either by revealing newly explored open territory, or by bidding in goal auctions (workers do not hold goal auctions because they have no goals of their own to auction, but they still bid in goal auctions held by other agents).

It is also possible for the supervisor to dissolve a coalition. If the exploration of the coalition goal results in no new goals, the supervisor will make use of the *Request-for-work* mechanism (see Fig. 2) to obtain a task. If this results in the supervisor joining another coalition, it will become a worker itself and will transfer the workers of the old coalition to its new supervisor by sending them *Retask-change-supervisor* messages. It is also necessary to notify the new supervisor of the addition and the workers of the change in supervisor, by sending it a *Transfer-workers* message.

3 Experiments

To evaluate the utility of the coalition-based algorithm, experiments were performed on a variety of maps with different numbers of participating agents. A simulator was written in Java that communicates with clients over TCP/IP. The simulator notifies clients of newly explored area and provides messaging between clients in both point-to-point and broadcast manners. Robots are assumed to have accurate localization. Communication between clients and the server is asynchronous. Clients are not provided with any means of contacting other clients directly.

In order to make comparisons in a proof-of-concept sense, we tested our algorithm against two other basic techniques. One of these techniques is simply using our algorithm without the agent auctions; this will mimic traditional auction-based task allocation. In this case, any agent without a goal assigned will simply be idle

and remain at its present location. The other point of comparison is a greedy algorithm in which each agent is responsible for the goals to which it is closer than any other agent. An agent pursues whatever goal it has that is the closest topologically. If an agent is not responsible for any goals, it will broadcast a request to other agents and will pursue the goal it receives that is the closest topologically. Note that this mechanism does not transfer responsibility for the goal, it merely provides the agent with an interim goal to pursue until it is responsible for its own goal rather than remaining stationary. This algorithm should eliminate some inefficiency due to idling but without using explicit coalitions.

Experiments were performed on a set of four maps, the three shown in Fig. 3 as well as an open map with no obstacles. Each map is represented as an 800 by 600 pixel bitmap, and agents have a radius of vision of 40 pixels. The first map is devoid of obstacles, except for a boundary preventing agents from reaching the edge of the map. The second and third maps contain increasingly many variously sized, shaped, and positioned obstacles. The fourth map is a highly structured office-building-like map specifically designed to test the performance of exploration methods in an inherently hierarchical environment. Team sizes of 2, 4, 8, 16 and 32 homogeneous agents were tested, and in all cases, all agents started in the center of the map for all tests, simulating a standard group deployment.

3.1 Results/Discussion

As expected, results varied between map types. In general, the greatest benefit of coalitions was seen on the structured map, while hierarchical allocation methods consistently outperformed the no-coalition algorithm and did not perform noticeably worse than the greedy control algorithm in any of the tests.

Perhaps the most intuitive measure of the performance of an exploration algorithm is the amount of area explored versus time. Since the simulator calculates its state in time increments, henceforth referred to as ticks, the units of time used in the graph are arbitrary and correspond to simulator ticks rather than any wall time unit. Area explored is simply measured in pixels, since the exploration environment maps are loaded as bitmaps, providing pixels as a convenient measure of area. In the simpler, more regular environments, there was little difference between the different algorithms, whereas in the structured map, the hierarchical approach was able to explore more quickly than either of the comparison approaches. Plots for the 16 robot case are shown in Fig. 4.

We can also look at the relative progress of the different algorithms with respect to time across different map types. In this case, area explored and exploration time must be expressed as percentages, since the maps vary in amount of free space and thus time required for exploration. In particular, we compute the ratio A_h/A_{nc} where A_h and A_{nc} are the area explored by the hierarchical and no-coalition methods respectively at a given time. The value of this ratio over time is shown in Fig. 5a for both the open and structured environments. From this graph, it can be seen that the

Fig. 4 Area Explored Versus Time for 16 agents exploring the (a) open, (b) sparse, (c) dense and (d) structured environments.

Fig. 5 Ratio of Area Explored by 16 Agents in the open and structured environments — (a) Hierarchical algorithm vs no-coalition algorithm; (b) Hierarchical algorithm vs greedy algorithm.

hierarchical allocation method performed better in the very early stages of exploration in both environments, but that it performed much better throughout in the structured environment. This is as expected, because the hierarchical method allocates agents more effectively when there are more agents than goals. Even the greedy algorithm, which allocates the extra agents by assigning them to their re-

	8 robots			16 robots		
	No coalition	Greedy	Coalitions	No coalition	Greedy	Coalitions
Open environment	1393	1442	1478	972	943	934
Structured environment	4006	3929	3887	3753	3370	2827

Table 1 Time (in simulator ticks) required to explore 80% of the free space in different environments with different team sizes and algorithms.

spective nearest goals instead of idling them, suffers in comparison during this initial phase in both environments, though it does catch up effectively by the end. This comparison is shown in Fig. 5b.

Other team sizes showed similar trends with the benefit of coalitions generally larger as the team size increases, as expected. However, these environments do suffer from diminishing returns. Table 1 shows the time required to explore 80% of the free space of the environment for 8 and 16 robots using the three different algorithms.² The more open environment showed greater improvement when going to the larger team, while the structured environment showed less improvement (presumably since there are fewer avenues of exploration) but more effect of coalitions, especially when the larger team is employed.

Fig. 6 Total distance traveled by the robot team over time. (Left) 8 robots in the open environment (Right) 16 robots in the structured environment.

In addition to the time required for exploration, we also considered the total distance traveled by the team. In general, since both the greedy approach and the coalition-based approach do not allow robots to idle, we expect these methods to produce more total travel even when the exploration time is less. As can be seen in Fig. 6, this is borne out in the experiments. Distance traveled for these techniques goes up almost perfectly linearly with time. When using coalitions, especially in large teams, some robots do pause while determining their next course of action, but this does not have a major effect on total distance traveled. Without coalitions,

² We use the time to explore 80% instead of 100% since the last portion of exploration, though very important, is highly dependent on the locations of the robots near the end of exploration and is not as directly comparable across algorithms.

Fig. 7 Percentage of Area Explored and Coalitions Formed Versus Time for 16 agents exploring the dense environment.

the team initially has lower distance traveled since several robots will be idle at the outset. During the bulk of the exploration, some robots may remain idle in the larger teams, but not in the smaller teams.

Qualitative Observations: Qualitative observations do not provide concrete support for the validity of hierarchical task allocation, but they can help explain the quantitative observations made and provide some insight into future ideas worth pursuing. In particular, we can identify different stages of exploration under which the hierarchical allocation method performs more or less effectively.

As mentioned earlier, our motivation for using coalitions is largely to handle the situation when agents outnumber goals. In many exploration processes, there will eventually come a point at which there are more goals to be explored than there are agents. Once agents are no longer forming coalitions, the hierarchical allocation method essentially becomes a standard market-based allocation algorithm. In some environments, however, this may never occur. For example, in the structured environment and with 16 agents exploring, there was always at least one coalition of more than one agent. This can be seen in Fig. 7. In this figure, the area explored and number of coalitions formed are represented as percentages. Coalitions Formed is calculated as the number of coalitions that exist at any given time divided by the number of agents, since the number of agents is the maximum number of coalitions that may form. This figure also demonstrates the decrease in rate of area being explored as the number of coalitions drops significantly around 70% of the way through the exploration. Even in those instances when there are more goals than agents, there will eventually be fewer goals than agents again near the end of the exploration. The hierarchical allocation method does not appear to perform particularly well once there are more goals than agents, even after the number of goals decreases back below the number of agents. It is not immediately clear why this

is, but observations of the experiments indicate that building the coalitions from physically dispersed robots seems to be unhelpful when there is little area left to explore.

4 Conclusions / Future Work

Overall, these results indicate that hierarchical coalition-forming task allocation techniques for robotic exploration can perform better than greedy or coalition-free approaches. This is particularly the case in very dense or structured environments, but even in open space, hierarchical task allocation performed no worse than simple traditional auctioning or greedy allocation. The improvements apply to teams of various sizes, depending somewhat on the environment. This improvement in time does come at the cost of greater energy expenditure in terms of total distance traveled by the team.

We have also considered several potential enhancements to the basic distributed coalition formation technique. For example, existing allocation methods for robot teams often use combinatorial auctions in which several nearby goals can be bid on and assigned as a group, exploiting their collocation. With coalitions, a group of nearby goals may be bid on by a coalition, such that the size of the goal set is equal (or close) to the size of the coalition. More generally, the utility of a coalition to achieve a goal (or set of goals) may be dependent on the capabilities of the members of the coalition, especially if the system is heterogeneous. This could allow for such systems to effectively use their varied abilities in similar fashion to other task allocation strategies in addition to the advantages of the coalitions.

Finally, even though the environment is assumed to be unknown, the robots could use their experiences from the initial exploration to inform future decisions. For example, it seems from our experiments that the number of coalitions follows a pattern of increase and decrease throughout exploration. Being able to detect this on the fly may allow us to idle robots toward the end of the exploration process to conserve energy with minimal loss of exploration efficiency. Also, while it is impossible to know for sure which goals will branch into multiple new goals to explore, it may be possible to employ pattern matching techniques to predict a likelihood that a goal branches. The ability to predict branching with any accuracy could be conveniently incorporated into a coalition forming exploration strategy. The coalition profit calculation could be easily modified to account for the optimal coalition size for a goal, based on estimated branching. In heterogeneous systems, even the membership of coalitions could be informed by the expectation of the needs of the exploration process. Together, we hope to show in the future that these improvements can lead to a cooperative exploration system that is even more efficient and effective.

References

1. Burgard, W., Moors, M., Fox, D., Simmons, R., Thrun, S.: Collaborative multi-robot exploration. In: Proceedings of ICRA, pp. 476–481 (2000)
2. Dias, M.B., Stentz, A.: Opportunistic optimization for market-based multirobot control. In: Proceedings of IROS, pp. 2714–9 (2002)
3. Fu, J.G.M., Bandyopadhyay, T., Ang Jr, M.: Local voronoi decomposition for multi-agent task allocation. In: Proceedings of ICRA, pp. 1935–40 (2009)
4. Lagoudakis, M., Berhault, M., Koenig, S., Keskinocak, P., Kleywegt, A.: Simple auctions with performance guarantees for multi-robot task allocation. In: Proceedings of IROS, pp. 698–705 (2004)
5. Lin, L., Zheng, Z.: Combinatorial bids based multi-robot task allocation method. In: Proceedings of ICRA, pp. 1145–1150 (18-22 April 2005)
6. Ma, X., Meng, F., Li, Y., Chen, W., Xi, Y.: Multi-agent-based auctions for multi-robot exploration. In: The Sixth World Congress on Intelligent Control and Automation, pp. 9262–9266 (2006)
7. Ma, X., Zhang, Q., Li, Y.: Genetic algorithm-based multi-robot cooperative exploration. In: International Conference on Control and Automation, pp. 1018–1023 (2007)
8. Pei, Y., Mutka, M., Xi, N.: Coordinated multi-robot real-time exploration with connectivity and bandwidth awareness. In: Proceedings of ICRA, pp. 5460–5 (2010)
9. Shehory, O., Kraus, S.: Methods for task allocation via agent coalition formation. *Artificial Intelligence* **101**(1–2), 165–200 (1998)
10. Sheng, W., Yang, Q., Ci, S., Xi, N.: Multi-robot area exploration with limited-range communications. In: Proceedings of IROS, pp. 1414–19 (2004)
11. Simmons, R., Apfelbaum, D., Burgard, W., Fox, D., Thrun, S., Younes, H.: Coordination for multi-robot exploration and mapping. In: Proceedings of the National Conference on Artificial Intelligence (AAAI, 2000)
12. Tang, F., Parker, L.: Distributed multi-robot coalitions through ASyMTRe-D. In: Proceedings of IROS, pp. 2606–13 (2005)
13. Walsh, W., Wellman, M.: A market protocol for decentralized task allocation. In: Proceedings of International Conference on Multi Agent Systems, pp. 325–332 (3-7 Jul 1998)
14. Wurm, K.M., Stachniss, C., Burgard, W.: Coordinated multi-robot exploration using a segmentation of the environment. In: Proceedings of IROS, pp. 1160–5 (2008)
15. Yamauchi, B.: A frontier-based approach for autonomous exploration. In: Proceedings of Computational Intelligence in Robotics and Automation (CIRA'97), pp. 146–151 (1997)
16. Zlot, R., Stentz, A., Dias, M., Thayer, S.: Multi-robot exploration controlled by a market economy. In: Proceedings of ICRA, pp. 3016–3023 (2002)