Proceedings of the 2009 IEEE International Conference on Systems, Man, and Cybernetics
San Antonio, TX, USA - October 2009

Sequential Auctions for Heterogeneous Task Allocation in Multiagent Routing Domains

George Thomas
Department of Computer Science
University of Iowa
Iowa City, IA 52242
george-thomas@uiowa.edu

Andrew B. Williams
Department of Computer and Information Sciences
Spelman College
Atlanta, GA 30314
williams@spelman.edu

***Abstract*—Many realistic problem domains are composed of heterogeneous tasks distributed in a physical environment. A team of mobile agents has to autonomously allocate these tasks, navigate to them and finally execute them. Recently auctions have been used for task allocation among homogeneous agents. Less studied is the case of allocation where both the tasks and the agents are heterogeneous in nature. In this paper, we investigate the market-based allocation of heterogeneous tasks to heterogeneous agents in domains where the distribution of the task heterogeneity is known *a priori*. We present a model of task heterogeneity, and define a metric that allows us to assess the fitness of a team for a particular task domain. We then present a sequential, round-based, auction setup for allocating heterogeneous tasks to heterogeneous teams and empirically investigate the performance of three different allocation strategies.**

***Index Terms*—Coordination, Task Allocation, Multiagent Systems, Robotics**

I. Introduction

Many interesting problems can be solved better, faster or more reliably by using a team of agents rather than just a single agent. Today, there are many application domains that envisage the use of teams of agents functioning autonomously to coordinate, cooperate and efficiently perform various tasks in those domains. In robotics, for example, it is envisaged that in a few years, teams of autonomous robots will be used in domains where it is difficult, hazardous or monotonous for humans to work. Examples of such applications include search and rescue [1], facility surveillance [2]and exploration of unknown, distant terrain [3]. One of the major research questions that arise in such applications is how to allocate the individual tasks in the domain to individual agents such that the tasks are accomplished optimally, or as efficiently as is practical.

By nature, the various tasks in any realistic problem domain are never all identical; rather, each individual task may require a particular subset of skills for its successful execution. Tasks may share particular skill requirements among each other but they may also differ in some skill requirements too. Our focus is on such heterogeneous task domains. Current allocation techniques and methods have focused on teams composed of homogeneous agents where every agent is endowed with all the skills necessary to perform any task. Little work has been done in exploring how the allocation is affected when the team of agents is no longer homogeneous but rather heterogeneous, as the tasks themselves. This is a significant consideration because, in most applications, fully homogeneous teams are impractical due to cost constraints. To equip every agent in a team with the skills necessary to handle every kind of task it may encounter is far more expensive than specializing specific agents within that team to handle specific tasks, which brings about a division of labor and lower cost in constructing that team of specialized agents.

In this paper, we define a model of task heterogeneity that captures how heterogeneity is reflected in real robotic systems, when there is some knowledge of the distribution of the heterogeneous tasks. We then define a metric that measures how effective a heterogeneous team will be on a such task domains and use it to direct our selection of randomly generated heterogeneous teams. We then present a sequential, single item auction for heterogeneous task allocation and introduce three different strategies to guide the task allocation. We extensively test our methods on a representative spectrum of task sets.

Our heterogeneous task models are situated within the context of multiagent routing task domains. These are domains where a team of mobile agents have to navigate to various locations within an environment and execute tasks at those locations. Team performance is assessed by measuring the total time taken to complete all the tasks in that domain, or the *maximum makespan* of the set of tasks. For simplicity, we have assumed that the tasks are known in advance, independent, of small duration, and uniform randomly distributed within the environment.

We organize this paper as follows: we present a brief review of literature in this field, and we formally introduce our allocation problem. We then describe our heterogeneous task model and its various properties, and our team effectiveness metric. We then extend an existing auction based allocation mechanism originally designed to handle homogeneous agents to handle heterogeneous agents, and present our allocation strategies. We introduce a metric to measure how "good" a heterogeneous team is and then describe the heterogeneous teams we use. We end with extensive empirical analysis of our methods and present some key findings.

II. Background

General research on heterogeneity in agents has focused on diversity in behavior, exploring coevolution of robot controllers based on differing levels of difficulty [4], and using social entropy [5] or evolutionary fitness [6] to measure diversity in robot teams. Our work differs from these in that we examine the effect of heterogeneity in physical abilities on teams of agents trying to perform heterogeneous tasks.

Using market systems to allocate tasks in multi-robot routing domains has recently become a popular method for task allocation [7]–[9]. The theoretical nature of this problem is similar to the $kTRP$ problem [10], with heterogeneous tasks. Previous work on agent heterogeneity has focused on robots that can do particular tasks. We focus on a deeper level of granularity where agents possess specific abilities, which we have discretized for ease of representation. Tasks require certain skills to be completed effectively, and these skills are covered by specific abilities. This model more accurately simulates how realistic robotic agents are designed.

III. Formalization

A. Problem Description

Let $T = \{t_1, \ldots, t_n\}$ be a set of n tasks. Each task t has associated with it a skill vector $S_t = (s_1, s_2, \ldots, s_{\mathcal{K}})$ which describes which members of the superset of all possible skills, of cardinality $\mathcal{K}$, is required by t. The elements of S_t are binary values and are interpreted as $s_j = 1$ indicating skill j is required for the task t, and $s_j = 0$ indicating skill j is not required for t, for $j = 1$ to $\mathcal{K}$. An agent team, $R = \{r_1, \ldots, r_m\}$, of m agents has associated with each agent an ability vector, also of size $\mathcal{K}$, denoted by $A_r = (a_1, a_2, \ldots, a_{\mathcal{K}})$ composed of binary values indicating which abilities the agent possesses. As before, $a_j = 1$ indicates agent r possesses the ability j, and $a_j = 0$ indicates it does not, for $j = 1$ to $\mathcal{K}$.

An agent possessing a particular ability has the matching skill that a task might require. Therefore, an agent r is *qualified* for task t if

$$A_r - S_t \geq 0 \tag{1}$$

Our heterogeneous task allocation objective is to form a partition, $\mathcal{T}$, of the set of tasks T, where $\mathcal{T} = \{\mathcal{T}_1, \ldots \mathcal{T}_m\}$, such that assigning the set of tasks $\mathcal{T}_r$ to agent r, where r is qualified to perform all the tasks in $\mathcal{T}_r$ results in minimizing the time taken to complete all the tasks, i.e minimizing the maximum makespan across all agents. More precisely, if $PC(r, \mathcal{T}_r)$ denotes the minimum path cost of agent r over the tasks in $\mathcal{T}_r$, then our objective is

$$\min_{\mathcal{T}} \max_{r \in R} PC(r, T_r) \tag{2}$$

Note that $PC(r, T_r)$ is the time taken for agent r to move from its starting location, navigate to each of the tasks in T_r and finish executing them.

B. Heterogeneous Task Model

We now describe our heterogeneous task model more formally. A *task class* is a set of heterogeneous tasks that all have identical skill vectors. The skill vector, thus, uniquely identifies the task class. A set of n tasks, T, can be partitioned into a set $\mathcal{C} = \{c_1, \ldots, c_N\}$ of N disjoint subsets, with $N \leq n$. Each element of $\mathcal{C}$ is thus a task class based on its skill vector and $\mathcal{C}$ is called the *task classification* of T. The cardinality, $\mathcal{K}$, of the universe of all possible skills for this task set T, is called the *dimension* of T.

We restrict our investigation to task sets where the distribution of tasks among the task classes $\mathcal{C}$ is known. So we associate with each task set a probability table P where p_i is a rational representing the probability that a task belonging to task class i appears in that set. If the exact number of tasks belonging to each task class is known, then P is computed as $p_i = |c_i| / (\sum_{j=1}^{N} |c_j|)$ with $\sum_{i=1}^{N} p_i = 1$. Within a task set, we assume that every skill is required by at least one task class; otherwise that skill should not be part of the universe of skills for that task set. Similarly, we also assume that the every task class has at least one actual task belonging to it occurring in the task set.

IV. Heterogeneous Teams

When considering the allocation of heterogeneous tasks to heterogeneous agent teams, the heterogeneity of the team plays an important role. This heterogeneity is limited by the cost of the individual abilities and the overall budget available to construct that team. There are many metrics that could be used to decide how well suited a team is for a task set. Teams can be measured based on how good the resulting allocation is in terms of absolute performance; or how robust a team is in terms of the redundancy and failure tolerance it provides. We chose the raw performance metric, biasing our team selection method in favor of teams that complete the tasks in minimum maximum makespan.

1) Heterogeneous Agent Construction Cost: Let vector $Q = (q_1, \ldots, q_{\mathcal{K}})$ of real values be the construction costs associated with including the corresponding ability in any agent; then the cost of constructing agent r, with ability vector $A_r = (a_1, \ldots, a_{\mathcal{K}})$, is given by $\mathcal{C}cost(r) = \sum_{i=1}^{\mathcal{K}} a_i \cdot q_i$. The *Team Construction Cost* for constructing a team R is then given by $\mathcal{C}cost(R) = \sum_{r \in R} \mathcal{C}cost(r)$. If we had unlimited budget, we would make every agent possess all possible skills. This is never the case so the team construction cost has to be within the available budget B; therefore, $\mathcal{C}cost(R) \leq B$.

It can be argued that the makeup of a heterogeneous team plays a bigger role in the performance of that team on a task set than any other factor. Allocating a task set where many tasks require a particular skill s to a team that has very few members possessing the ability that satisfies s will result in a poor overall minimum maximum makespan and bad performance. On the other hand, a team in which many members possess the abilities satisfying the most commonly required skills in a task set will probably perform better on that task set. This gives us an intuitive idea for a metric measuring the performance,

with respect to overall minimum maximum makespan, of a heterogeneous team on a heterogeneous task set. *For every task in a task set, maximize the number of agents on the team qualified to execute it.*

2) Task Coverage: Let R be a team of m agents that are to be allocated to a task set T. Let

$$priority(c_i) = \frac{p_i}{|\{R' : r \in R \text{ and } r \text{ is qualified for } c_i\}|}$$

where c_i and p_i are as previously defined. Then the *task coverage* for R over T is

$$coverage(R,T) = \frac{1}{m \cdot \sum_{i=1}^{N} priority(c_i)} \quad (3)$$

This is a normalized value with a maximum of 1 for a team of all homogeneous agents. Heterogeneous teams occupy the range of $[0,1]$ with the higher the coverage, the better the performance. In general, this is true but coverage can be dominated by other factors so if the difference between the coverages of two teams is minor, their performance differences are in the same range.

V. METHODS

A. Sequential, Single-Item Auctions for Heterogeneous Task Allocation

Lately, market-based mechanisms such as auctions have been used to provide efficient solutions to some of these problems. Auctions lend themselves to task allocation among cooperative agents because they provide a good balance between a fully centralized and a fully distributed solution. Sequential, single item auctions have been used extensively in multiagent routing domains [8], [9], [11] because of their tractability and good performance. The auction proceeds in multiple rounds and only one task is up for sale in a round. Agents bid for unsold tasks in every round and the task is awarded to the agent with the lowest bid in each round. Only after the sale is complete is the next task announced. Since agents sequentially build up the set of tasks they are awarded, some synergies can be taken into account; so while these auctions still lead to sub-optimal allocations, they proceed fast [12] and in some cases, the performance can be bounded with respect to the optimal [13]. None of the implemented systems have explicitly considered heterogeneity. We use a sequential, single item auction(SSI) and bidding rules similar to those in [8], [12], but we extend the auctions to explicitly handle heterogeneous tasks. There is no central auctioneer but every agent serves as auctioneer and bidder; allocations are computed in distributed fashion by all the agents.

Let $PC(r, T^{'})$ be the smallest cost for visiting and then executing all tasks in $T^{'}$ from the current location of agent r. Initially all tasks are unallocated. Every agent computes its path cost for every unallocated task for which it is qualified, and then bids for only one task - the task for which it has the smallest overall path cost PC. This bid is sent to all other agents. Every agent then finds the overall lowest bid, among all received bids, and allocates that task to the agent that made that lowest bid. The process continues until all tasks have been allocated. Finally, agents compute their minimum cost paths to visit and execute all their tasks and proceed to execute the tasks they have been assigned.

A key issue is how individual agents should compute their bids for tasks. Given auctions of this nature, the bidding rules that should be followed have been analyzed in [8], [14]. The basic idea behind the bidding rule used for maximum makespan is that agents bid on each task the marginal increase in makespan for the entire team between the current allocation of tasks to the team and the new allocation, if this task is won by the bidding agent. Every agent, which can generate its bids without knowing the location of the other agents, computes these bids for all unawarded tasks and then broadcasts only its overall lowest bid. Lagoudakis *et al.* [13] show that for homogenous agents, this bidding rule can perform no worse than twice the number of agents times the optimal cost solution, which is unfortunately not a constant factor guarantee. No guarantees are known for the heterogeneous case.

To compute the path cost, PC, during bidding and for final path construction, each agent has to solve a Traveling Salesman Problem(TSP), which is NP-hard. As a matter of design, we do not depend on any particular TSP solving techniques for our auction method to work; but any appropriate TSP solution method can be substituted, or plugged in. We chose to use a combination of well-known TSP heuristics [15]. To create a TSP tour, and thus compute PC, we incrementally add task locations and build up an individual tour. We add each location by inserting it through the cheapest-insertion heuristic and then running 2-Opt to optimize the resulting tour. While this method does not have any approximation factor guarantees, in practice, the tours generated are very good.

We investigated three different allocation strategies within SSI auctions:

1) **FirstFit(FF):** Our first method adopts a naive approach. We use the bidding rule mentioned previously but only qualified agents submit their first overall lowest bid for a task and tasks are allocated to the agent that submits the first, lowest bid ; hence it is a form of first-fit.
2) **BestFit(BF):** Here, each agent computes a 2-tuple bid for each qualified task. The first component is again the path cost. The second component is the degree of over-qualification the agent has for this task; basically, how many abilities of this agent are not utilized in executing the task. Agents determine both which single best-fit bid to submit per round, and, during winner determination, which best-fit bid will win a round as follows: if the path cost component is less than path costs of other bids by $\epsilon_1\%$, then that is the winning bid. If the path cost is within $\epsilon_1\%$ of x other bids, then the winning bid is the one from the set of x bids which has its over-qualification component less than other bids in x by $\epsilon_2\%$. If no other over-qualification bids are within $\epsilon_2\%$, then the winning bid is simply the bid from the set of x with overall lowest path cost. So, we still give first preference to path cost, but if these bids are close, we try to "best fit" the agent

to the task it is bidding for. We chose values for $\epsilon_1 = 4\%$ and $\epsilon_2 = 1\%$. But the method was fairly sensitive to these values and it was difficult to adopt consistent values.

3) **TeamFit(TF):**One of the drawbacks of the FF and BF methods is that an agent may win tasks in the initial rounds, for which there are other qualified bidders, only to find out it is the sole bidder for some other distant task for which only it is qualified. This will likely lead to a bad allocation and so we seek to avoid this by incorporating some knowledge of the agent team into the bids that are made. Agents, being part of a team, can have knowledge of the abilities of other team members. This information can be provided once at team creation as abilities do not change. The goal here is to allocate tasks based on how "hard" it will be to allocate them to the team. If there are many tasks of a particular type and only a few agents qualified to do them, we auction and allocate those tasks first so that hill climbing can proceed on that basis. The TeamFit algorithm proceeds as follows: Before the auction begins, each robot computes the $priority(c_i)$ for all task classes in the domain and places them in a list. If any task class has only one agent qualified to execute it, its $priority$ is set to the maximum value in this list, and the list is then sorted in descending order. The idea here is that if any task class has only one agent available to execute, it should be allocated to that agent in the first rounds itself. During the auction process now, tasks are announced for bidding in order of the $priority$ of their associated task class. All the tasks in task classes that have the same $priority$ are released simultaneously, starting from the head of the list. Only after all tasks with a higher $priority$ have been auctioned and awarded, are lower $priority$ tasks released for auction. Each robot now bids only for the currently released tasks, and bids its path cost. Winners are assigned on the basis of the first, lowest path cost value. The auction still has the same number of rounds as before but the rounds move in stages according to the priority relationship among the tasks. The intuitive idea here is that by using the priorities, the team is automatically being "best fit" for the tasks.

B. Optimal Solution to Heterogeneous Task Allocation

Computing the path cost for our problem can be mapped into solving multiple traveling repairmen problems and is thus NP-complete. Consequently, computing the minimum makespan for our problem involves solving multiple NP-hard instances. We modeled this problem as an integer program(IP), which we have to omit due to space constraints. We coded the IP using ZIMPL [16] and ran it in SCIP [17] but we could only solve very small problem instances (3 agents, 10 tasks). Hence, we focused on finding faster, sub-optimal solutions.

C. Random Generation of Good Teams

To test our allocation methods, we needed a consistent method of generating heterogeneous teams. Three random teams of m agents were selected as follows: we randomly generate the skills for m agents, ensuring that the team cost is within budget B, every agent is qualified for some task, and all tasks in the task set have at least one agent qualified to execute them. For every team so generated, we compute its task coverage and store it sorted according to its coverage. We then select the set of three teams from this sorted list that had coverages distributed uniformly through the coverage spectrum, the lowest, median and highest coverages generated. We call these teams RW, R50 and RB respectively. Depending on the particular task set, required team size and budget constraints, we may have to reject many random teams to arrive at a single valid team. So our process was equipped with a timeout of 5 minutes to generate a maximum of $10,000$ teams, or return with whatever had been created within the allotted time. Clearly, the more teams we generate, the better the maximum coverage team generated.

VI. Evaluation

We envisage our allocation methods to be applied to task sets with dimensions $\mathcal{K} \leq 20$, as in the lunar task set of [18]. Here there were 8 task classes and a dimension of 7. We, therefore, generated representative task sets, selected the teams we want to test on the task sets, and used our allocation mechanisms to perform task allocation, analyzing the results.

A. Task Set Selection

We created task sets with $N = 8$, $K = 7$ and each class requiring $1-3$ skills as in [18]. The skills were distributed to task classes uniform randomly, and every skill had unit cost. We then created a group of 9 sets, for which we distributed the probabilities of task frequencies among the classes in the following way:

1) **Zipf**: We used a zipf-like distribution among the various classes. We generated 3 such sets Z1, Z2 and Z3.
2) **Normal**: We used a normal-like distribution among the various classes. We generated 3 such sets N1, N2 and N3.
3) **Uniform**: We used a uniform distribution among the classes. We generated 3 such sets U1, U2 and U3.

The justification for these choices was to cover a wide spectrum of possible task distributions among the classes. Zipf distributions ensure that one class has many tasks while the rest have progressively smaller frequencies. Uniform distribution was at the other end of the spectrum, with the normal distribution in between.

B. Experimental Setup

We implemented our system using an extended version of the Teambots simulator [19]. The environment was a 51 by 51 cell environment, similar to the setup of [8], had walls, was fully known, and distances were computed on a four-way grid movement setup. A single experiment was an allocation of a task set to a team using a particular allocation mechanism. For every experiment, we geographically dispersed the tasks uniform randomly throughout the environment; each task had

a small, identical duration, and agents were always distributed uniform randomly through the environment. We also geographically dispersed the agents uniform randomly throughout the environment. We conducted 20 iterations of every experiment, with different geographic dispersions of the exact same tasks and agents to eliminate that as a factor. Thus, every value reported in the results is the average of 20 iterations. The teams RW, R50 and RB were generated for two team sizes - 5 agents and 10 agents - for a total of six teams. We tested them on task sets Z1-U3 using the three SSI methods - FF, BF and TF - and a simultaneous parallel auction (P), in which all tasks are announced, bid for and allocated to the overall lowest bidder (for each task) simultaneously in one round. Parallel auctions are simple and fast, but do not take any synergies between tasks into account.

C. Results

The results are shown in figures 1-6 for the six teams. The results show that for the high and median coverage teams, RB and R50, sequential auctions methods (FF,BF and TF) vastly outperform the corresponding parallel auctions. For the lowest coverage teams RW(5) and RW(10), there is no significant difference in performance between any of the allocation methods, though TF outperforms the other methods slightly. For the high coverage teams, RB(5) and RB(10), there is no significant difference in performance between the three SSI methods, FF, BF and TF, except for the last three task sets(that have uniform distribution), U1-U3, where TF outperforms the other two methods. For the median coverage teams R50(5) and R50(10), TF significantly outperforms FF and BF across all task sets. In further analysis, we used the

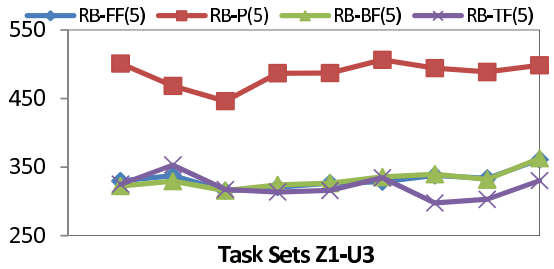

Fig. 1. Maximum Makespan vs. Task Set for RB Team of 5 agents

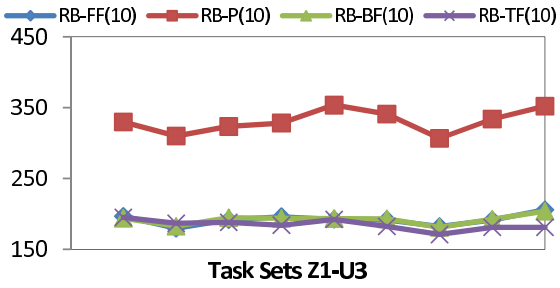

Fig. 2. Maximum Makespan vs. Task Set for RB Team of 10 agents

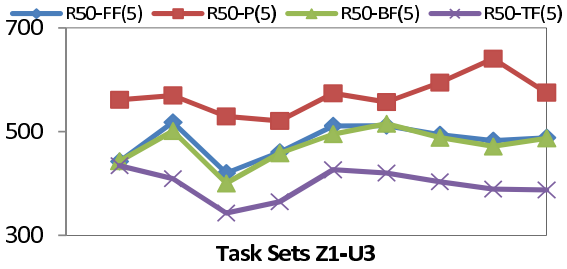

Fig. 3. Maximum Makespan vs. Task Set for R50 Team of 5 agents

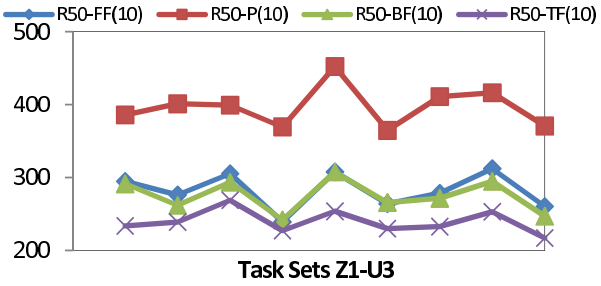

Fig. 4. Maximum Makespan vs. Task Set for R50 Team of 10 agents

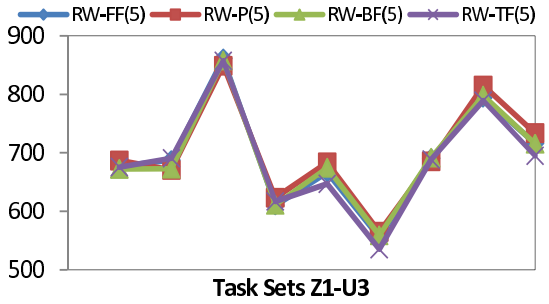

Fig. 5. Maximum Makespan vs. Task Set for RW Team of 5 agents

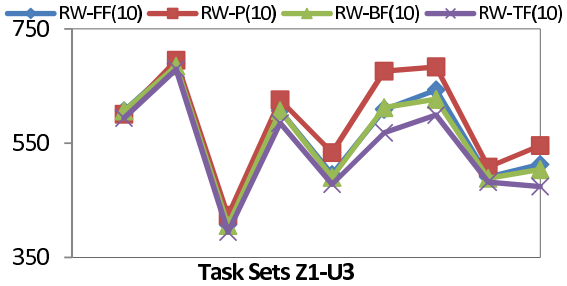

Fig. 6. Maximum Makespan vs. Task Set for RW Team of 10 agents

Wilcoxon Matched-Pair Signed-Rank test to ascertain various hypotheses. We tested the null hypotheses that FF auctions perform better than or as well as TF auctions, across all teams of both sizes 5 and 10 agents. This null hypotheses was rejected ($p < .000$ for RB, R50; $p < .001$ for RW). We also tested the null hypotheses that BF auctions perform better than or as well as TF auctions, across all teams of both sizes 5 and 10 agents. This null hypotheses was rejected ($p < .000$ for RB, R50; $p < .008$ for RW) as well.

VII. Discussion

From the results, it is clear that SSI auctions vastly outperform parallel auctions in general, because some synergies between tasks can be taken into account. For the RW team, the difference was less between SSI and parallel auctions. This is due to the fact that the RW teams were so ill suited for the tasks in terms of qualifications matching the task skill requirements that there were effectively only a few ways to allocate the tasks. The qualification matrix imposed a highly constrained allocation that could not be greatly improved by any method.

In general, TF outperformed FF and BF for all teams by statistically significant margins. For the R50 teams, TF outperformed FF and BF significantly because the method was able to organize the auctions according to priority. For the RB teams, there was not a great difference between TF and FF or BF. This is the opposite case of RW. Here, RB teams are so well-matched in terms of qualifications for the skill requirements of the task sets that they were automatically "team-fit" for the tasks by way of their inherent team makeup. So TF did not significantly improve allocation. TF did improve performance for RB(5) and, to a lesser degree, RB(10) for the uniformly distributed task sets U1-U3. This follows from the fact that TF works best when there are many choices to choose from in allocating a task, and so it allocates by priority. In zipf and normal distribution sets Z1-N3, the distribution again imposed an inherent allocation process.

Looking at absolute values in the figures, it is clear that RB teams performed much better than R50 teams, which in turn performed much better than RW teams. So using coverage as a team metric is effective. Finally, there was no difference between the FF and BF methods; myopically matching qualifications does not improve allocation when there is a large task horizon.

A key insight from these results is that, when we have a badly designed team, the specific allocation methods make little difference. When we have average teams, TF is the best choice. When we have well-designed teams, any SSI auction can perform well. So, if we have no control over the heterogeneous team we use, TeamFit auctions will provide the best results. If we can design teams of good coverage, then they virtually become independent of the specific SSI allocation method used.

VIII. Conclusion

We have presented a heterogeneous task model and a metric, task coverage, for generating good heterogeneous teams. We then compared various auction allocation mechanisms, specifically focusing on sequential, single item auctions for heterogeneous tasks. We tested three different bidding mechanisms, FirstFit, BestFit and TeamFit and show that sequential auctions perform well, and our TeamFit method works well for most teams. We intend to further research auctions for heterogeneous task allocation, including exploring solutions that approach the optimal solutions provided by combinatorial auctions.

References

[1] R. Murphy, "Trial by fire [rescue robots]," *Robotics & Automation Magazine, IEEE*, vol. 11, no. 3, pp. 50– 61, Sept 2004.

[2] P. E. Rybski, S. A. Stoeter, M. D. Erickson, M. Gini, D. F. Hougen, and N. Papanikolopoulos, "A team of robotic agents for surveillance," in *In Proc. of the Int'l Conf. on Autonomous Agents*, 2000, pp. 9–16.

[3] P. S. Schenker, T. L. Huntsberger, P. Pirjanian, and E. T. Baumgartner, "Planetary rover developments supporting mars exploration, sample return and future human-robotic colonization," in *Autonomous Robots*, vol. 14, 2003, pp. 103–126.

[4] M. A. Potter, L. Meeden, and A. C. Schultz, "Heterogeneity in the coevolved behaviors of mobile robots: The emergence of specialists." in *IJCAI*, 2001, pp. 1337–1343.

[5] T. Balch, "Measuring robot group diversity," in *Robot Teams: From Diversity to Polymorphism*, T. Balch and L. E. Parker, Eds. Natick, Massachussetts.: A.K. Peters, 2002.

[6] J. C. Bongard, "The legion system: A novel approach to evolving heterogeneity for collective problem solving." in *EuroGP*, 2000, pp. 16–28.

[7] B. P. Gerkey and M. J. Mataric, "Murdoch: Publish/subscribe task allocation for heterogeneous agents," in *Proceedings of the Seventeenth National Conference on Artificial Intelligence and Twelfth Conference on on Innovative Applications of Artificial Intelligence, July 30 - August 3, 2000, Austin, Texas, USA*, 2000, p. 1070.

[8] C. Tovey, M. G. Lagoudakis, S. Jain, and S. Koenig, "The generation of automatic bidding rules for auction-based robot coordination." in *Multi-Robot Systems: From Swarms to Intelligent Automata*, L.E.Parker, F. Schneider, and A.Schultz, Eds. Springer, 2005, pp. 3–14.

[9] R. Zlot, A. Stentz, M. B. Dias, and S. Thayer, "Multi-robot exploration controlled by a market economy." in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2002, pp. 3016–3023.

[10] J. Fakcharoenphol, C. Harrelson, and S. Rao, "The k-traveling repairman problem," in *SODA*, 2003, pp. 655–664.

[11] S. C. Botelho and R. Alami, "M+ : a scheme for multi-robot coopertaion through negotiated task allocation and achievement," in *ICRA, IEEE, Detroit, Michigan*, 1999.

[12] S. Koenig, C. A. Tovey, M. G. Lagoudakis, E. Markakis, D. Kempe, P. Keskinocak, A. J. Kleywegt, A. Meyerson, and S. Jain, "The power of sequential single-item auctions for agent coordination," in *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, 2006, pp. 1625–1629.

[13] M. G. Lagoudakis, E. Markakis, D. Kempe, P. Keskinocak, A. Kleywegt, S. Koenig, C. Tovey, A. Meyerson, and S. Jain, "Auction-based multi-robot routing," in *Proceedings of Robotics: Science and Systems*, June 2005.

[14] R. Zlot, "An auction-based approach to complex task allocation for multirobot teams," Ph.D. dissertation, Carnegie Mellon University, 2006.

[15] D. S. Johnson and L. McGeoch, "Experimental analysis of heuristics for the stsp," in *The Traveling Salesman Problem and its Variations*, Gutin and Punnen, Eds. Kluwer Academic Publishers, 2002, pp. 369–443.

[16] T. Koch, "Rapid mathematical programming," Ph.D. dissertation, Technische Universität Berlin, 2004, zIB-Report 04-58.

[17] T. Achterberg, "Constraint Integer Programming," Ph.D. dissertation, Technische Universität Berlin, 2007.

[18] G. Thomas, A. Howard, A. B. Williams, and A. Moore-Alston, "Multi-robot task allocation in lunar mission construction scenarios," in *Proceedings of the IEEE Intl. Conf. on Systems, Man and Cybernetics*, 2005.

[19] T. Balch, "Teambots," in *http://www.teambots.org*, 2000.