Auction-based task allocation for multi-robot teams in dynamic environments

Eric Schneider[1], Elizabeth I. Sklar[1,2], Simon Parsons[1], and A. Tuna Özgelen[2]

[1] Dept of Computer Science, University of Liverpool, UK
[2] Dept of Informatics, King's College London, UK
[3] Dept of Computer Science, Graduate Center, City University of New York, USA
{eric.schneider,s.d.parsons}@liverpool.ac.uk, elizabeth.sklar@kcl.ac.uk,
aozgelen@gc.cuny.edu

Abstract. There has been much research on the use of auction-based methods to provide a distributed approach to task allocation for robot teams. Team members bid on tasks based on their locations, and the allocation is based on these bids. The focus of prior work has been on the optimality of the allocation, establishing that auction-based methods perform well in comparison with optimal allocation methods, with the advantage of scaling better. This paper compares several auction-based methods not on the *optimality* of the allocation, but on the *efficiency* of the execution of the allocated tasks giving a fuller picture of the practical use of auction-based methods. Our results show that the advantages of the best auction-based methods are much reduced when the robots are physically dispersed throughout the task space and when the tasks themselves are allocated over time.

1 Introduction

This paper is concerned with the *multi-robot routing* problem. This is a task allocation problem in which tasks require robots to reach particular *target* points, and each target must be visited by one robot only. Multi-robot routing is a standard task [9] that is part of problems such as search-and-rescue and humanitarian demining, and it is a common test domain for robot team coordination [2]. The problem is also computationally hard. The number of ways of allocating m target points to n robots is a Stirling number of the second kind:

$$S(m,n) = \frac{1}{n!}\sum_{j=0}^{n}(-1)^{n-j}\binom{m}{n}j^{m}$$

and the size of the problem quickly defeats attempts to use standard optimization techniques like integer programming [23].

The computational explosion in the multi-robot routing problem has led to researchers looking for more efficient solutions. Lagoudakis et al. [12] began an interesting line of work in using *auction mechanisms* to allocate the target points

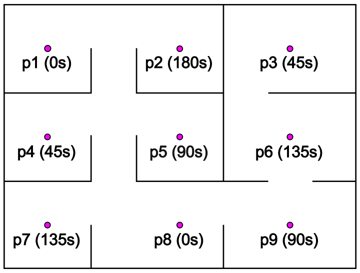


Fig. 1. The experimental environment — an office-like floor plan and set of target points. In dynamic allocation scenarios, points become available for allocation at the times indicated in the figure.

in a multi-robot routing problem. This work showed that multi-round auctions, where each round consisted of bids for single tasks, had a performance that was experimentally determined to be close to that of an optimal allocation. This initial paper was followed by work on the way that rules for determining bids affected the performance of the auction mechanism, establishing bounds on the performance of different rules [13], and empirically comparing the performance of the auctions against an optimal allocation under different bidding rules [23]. Later work considered the *sequential single-item auction* [7,8] as an alternative to *combinatorial auctions* for multi-robot routing, again showing that the total path cost — the total distance travelled by all the robots in the team — is close to optimal.

Overall, this prior work shows the power of auction mechanisms in task allocation in this domain, and in this paper we build upon it. Our extension is twofold. First, we evaluate the effectiveness of auction mechanisms not only in terms of the total distance that the team expects to travel given the task allocation — what we might consider the theoretical cost of executing the tasks — but also in terms of a set of metrics that measure how the tasks are executed in practice. Second, we not only deal with static allocation — where the target points are all known at the beginning of the allocation — but also dynamic allocations, where target points are given to the team over time.

These two factors reveal some of the trade-offs between auction mechanisms in different scenarios, and show that there are situations in which the sequential single-item auction is no more efficient in practice than mechanisms which theory tells us should perform much worse.

2 Methodology

The work reported here is part of a larger effort [16,21] aimed at evaluating coordination techniques in terms of their potential for application in real robot systems. This influences a number of aspects of the experiments that we conducted. We discuss these aspects in this section.

2.1 System Architecture

Our multi-robot system is fully distributed in the sense that the controllers for the robots are independent agents running as separate processes. No controller has the ability to tell the controller of another robot what to do. The coordination of the team is through a central *auction manager*, which holds a list of target points and communicates the start of an auction and awards target points to *robot controllers*.[4] Each controller sends in bids, the auction manager determines the winner of the auction and allocates target points accordingly. Since the individual controllers only have access to their local information, their bids and the allocation don't explicitly attempt to optimize across all robots.

Our software architecture is agnostic about whether the team executes its tasks on real robot hardware or in simulation. That is, the robot controllers are capable of receiving information either from real robots operating in our lab, or from simulated robots operating on a model of the lab, and the same decisions are made and the same control signals output in both cases. For the work reported here, we ran the experiments using the Stage simulator [4]. Previous work [16] suggests that for our setup, there is close agreement between simulated results and results obtained on real robot hardware.

Our hardware platform is the Turtlebot 2, which has a differential drive base and a colour/depth-sensing Microsoft Kinect camera. The ROS [17] navigation stack provides communication, localisation and path planning capabilities. Our robot controller software works with ROS to implement bidding, task selection and collision avoidance behaviours using a finite state machine. Our simulated Turtlebot has the same properties as its physical counterpart (size, shape, velocity, acceleration, etc.). In the simulation experiments reported here, localisation is provided by Stage. An on-board sensor that approximates the Kinect camera is used for collision avoidance.

The environment in which our robots, both real and simulated, operate is an office-like environment with rooms opening off a central hallway. A sample layout is shown in Figure 1. It is a smaller environment, in terms of the number of rooms, than that studied by [8] (and in similar work by the same group of authors), a fact determined by the maximum size of the physical area available in our lab for the real robots and our desire to be able to run parallel experiments in both simulation and on the real robots (though here we only report work in simulation).

2.2 Metrics

To measure the performance of the robot team in practice — and hence the performance of the coordination mechanisms — we consider a number of metrics applicable to the performance of individual robots and the team as a whole.

[4] Though the bidding process and winner determination are managed centrally, there is no centralized control in the usual sense. The auction could also be distributed among the robots as in [3].

In any work with robots, an important consideration is power consumption. This is the fundamental scarce resource that a robot possesses. Robot batteries only last for a limited time, and so, all other things being equal, we prefer solutions to the multi-robot routing problem that minimize battery usage. As in [7,8,12,13,23], therefore, we measure the *distance travelled* by the robots in executing a set of tasks — both individually and as a group — since this is a suitable proxy for power consumption.

It is important to note that distance is not computed by looking at the shortest distances between the target points, but is (as closely as we can establish) the actual distance moved by the robots during task execution. We collect regular position updates from the simulator, compute the Euclidean distance between successive positions, and sum these up.

In many multi-robot team activities that relate to the routing problem, time to complete a set of tasks is also important. Time is important in exploration tasks — in search and rescue activities, in patrolling, and possibly in demining[5] — and so we measure *run time*, the time between the start of an experiment and the point at which the last robot on the team completes the tasks allocated to it. Also relevant is the *deliberation time*, the time that it takes for the tasks to be allocated amongst the robots. Deliberation time matters because it feeds into the overall time required to complete a set of tasks, but also because it allows us to establish how different allocation mechanisms compare in terms of the computational effort and communication resources required to run them. *Execution time*, is run time minus deliberation time, the time that the robots take to execute the set of tasks they are given.

We also measure *idle time*, the amount of time that robots sit idly during the execution of a set of tasks. We compute the idle time for a robot as the time that elapses between when that robot completes its last task and when all robots on the team have completed all of their assigned tasks. This gives us a way of quantifying how equally tasks are distributed among robots, and it also suggests the extent to which resources are being wasted by a particular allocation. Although a mismatch between the number of tasks and the number of robots means that idle time can be inevitable, idle time represents the use of precious power that is not being directed towards the completion of the tasks.

3 Experiments

3.1 Experimental Setup

As described earlier, the multi-robot routing problem that we are studying involves a team of robots tasked to visit a number of target points such that one robot visits each point. A *scenario* is defined by a specific set of parameters: the number of robots on the team (n), the starting locations for the robots,

[5] One can easily imagine demining happening against the clock — in humanitarian demining [6], for example, there may be the need to demine an area in order to allow refugees to move safely away from a dangerous situation.

the number of target points to visit (m), and the locations of the target points. Thus, a scenario can be described by a tuple

$$\langle n, \{(x_0, y_0), \ldots, (x_{n-1}, y_{n-1})\}, m, \{(x'_0, y'_0), \ldots, (x'_{m-1}, y'_{m-1})\}\rangle$$

The experiments reported here measured results in four different scenarios, all of which involved $n = 3$ robots. Two scenarios used static allocations and two scenarios used dynamic allocations. All scenarios used the environment in Figure 1 and had $m = 9$ target points. In static allocations scenarios, all target points were allocated at the beginning of each experiment in a single deliberation phase followed by a single execution phase. In dynamic allocation scenarios, target points were introduced over time according to a predefined schedule. Every introduction of target points triggered a new deliberation phase (possibly interrupting an execution phase). Labels in Figure 1 indicate when each point was introduced in seconds after the start of an experiment. There were two sets of starting locations, one which clustered the robots in the lower left room (as the environment appears in Figure 1), and one in which the robots were distributed in the lower left corner, the upper left corner and the upper right corner of the environment. Experiments were conducted with each of the four scenarios using each of the four task allocation mechanisms discussed below, and each was run 10 times. Thus 160 experimental trials were conducted in all:

$$2\ \textit{start locations} \times \ \{\textit{static}\ , \textit{dynamic}\} \times 4\ \textit{allocation mechanisms} \times 10\ \textit{trials}$$

Each experiment recorded the metrics described above: distance travelled; run time (including execution time); deliberation time; and idle time.

3.2 Mechanisms tested

Our experiments involved four different mechanisms for task allocation: Round-robin (RR); Ordered single-item auction (OSI); Sequential single-item auction (SSI); and Parallel single-item auction (PSI). We describe these in turn.

In *round robin* allocation, we start with two ordered lists, one of target points and one of robots. The first target point is allocated to the first robot, the second target point to the second robot and so on. When one target point has been allocated to each robot, a second target point is allocated to the first robot. And so on. This is clearly not a particularly efficient way to approach task allocation, but it provides a baseline against which other mechanisms can be tested. We referred to this as the "greedy taxi" policy in our earlier work [16], because this policy emulates the behaviour of a taxi rank.

The approach we call the *ordered single-item auction* takes a simple step to improve on round robin allocation. In this case the target points are again placed in an ordered list, but this time each point in turn is offered to all the robots. Each robot makes a bid for the point, where the bid is the distance that the robot estimates (using an A* path planner) it will have to travel to reach the point from its current location. The point is allocated to the robot that makes the lowest bid, that robot updates its "bid-from" location to be the location of the target point it just acquired, and the next point is auctioned.

In the *sequential single-item auction* [8], all unallocated target points are presented to all the robots simultaneously. Each robot bids on the target point with the lowest cost (again computed as the A* distance to the point) and the point with the lowest bid is allocated to the robot that made the bid. The winning robot updates its "bid-from" location to that of the target point it just won and the process is repeated until all points have been allocated.

The *parallel single-item auction* (introduced as something of a strawman in [8]) starts like SSI with all robots bidding on all points from their starting locations. All the target points are allocated in one round, however, with each point going to whichever robot made the lowest bid on it.

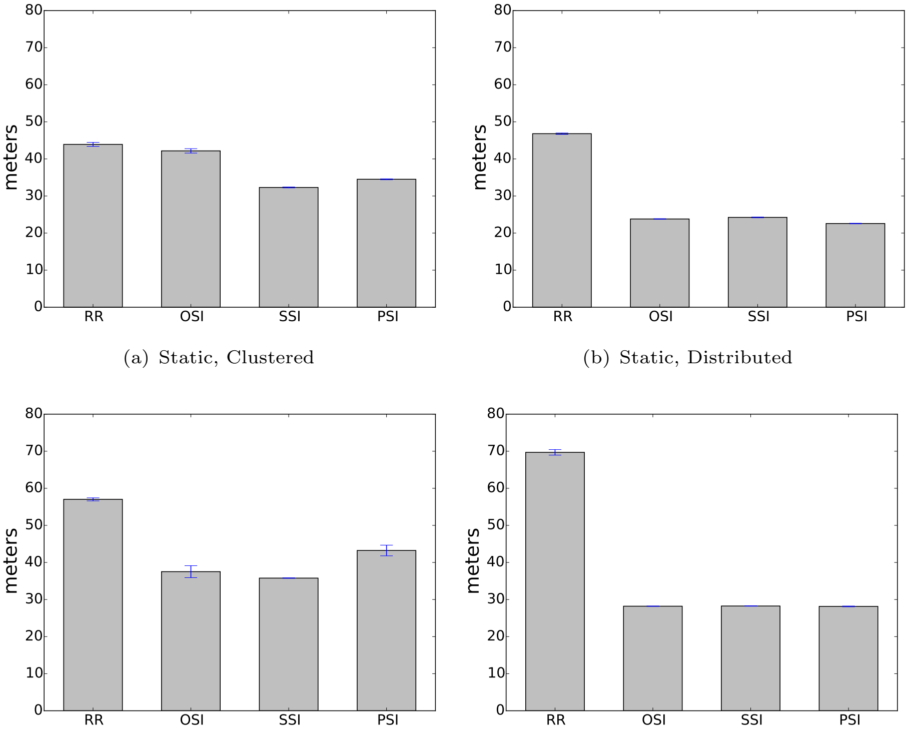
3.3 Results

The results of the experiments can be seen in Table 1, Figure 2 and Figure 3. Table 1 gives the value of the metrics for each of the four task allocation mechanisms — RR, OSI, SSI and PSI — in each of the four scenarios — static and dynamic, clustered and distributed start. We give the average value across the ten runs, and the 95% confidence intervals. Figure 2 gives the average distances travelled by the team. This is the main metric we will consider here since it is the one that SSI is looking to optimise task allocations against (that is SSI seeks to minimise distance), it is the metric that we might expect it to perform best on. Figure 3 shows the remaining metrics.

The results for the clustered static allocation in Figure 2 shows SSI as we expect it to perform given the analysis in [8], generating allocations that result in shorter total distances for the team than any of the other mechanisms. The corresponding results from Figure 3 show SSI with lower run time, execution time, and idle time. Run time and execution time are obviously strongly related to the lower distance generated by SSI, while lower idle time suggests a more even allocation of tasks to robots than other mechanisms. PSI is notably poor in this regard in the static, clustered allocation, a point first noted in [8].

When we move to consider a static allocation with a distributed start, we note from Figure 2 that SSI doesn't provide an obviously better allocation in terms of distance than either OSI or PSI. (In fact, Table 1 reveals that SSI does fractionally worse than OSI.) The fact that the robots are distributed means that the target points fall into natural clusters that are obvious to the simpler allocation mechanisms without the exhaustive search through allocations that SSI provides. SSI however, continues to do better than the other mechanisms on other metrics in Figure 3, perhaps most noticeably on idle time.

Moving on to the dynamic allocations, in terms of distance (shown in Figure 2), we see SSI performing comparably with the other mechanisms (and maybe doing a bit worse than PSI). It seems to us that the dynamic allocation is spreading the robots out in much the same way as the distributed start does — because they start moving to the initial set of target points, the robots are physically spread out by the time that later points are auctioned. The same relationships



(a) Static, Clustered

(b) Static, Distributed

(c) Dynamic, Clustered

(d) Dynamic, Distributed

Fig. 2. Average distance travelled over ten runs by each team of three robots for static and dynamic scenarios, with clustered and distributed starts. The bars are ordered RR, OSI, SSI and PSI.

holds for the distance results from the dynamic distributed start, though in this case all three auction mechanisms improve compared with RR, and across all the other metrics in Figure 3.

3.4 Discussion

The aim of these experiments is to understand the comparative performance of the allocation mechanisms across the different scenarios with the aim of establishing the suitability of the mechanisms for different kinds of multi-robot routing tasks.

Overall, for the scenarios we studied, our analysis supports the results in [7,8,12,13,23], showing the effectiveness of the sequential single-item auction in finding solutions to the multi-robot routing problem when the overall distance covered is the most important performance metric. For static allocation with clustered start points, SSI generated solutions which required the team to travel

		Distance	Run time	Delib. time	Idle time
Clustered	RR	43.92 ± 1.12	262.62 ± 9.03	0.003 ± 0.0003	118.69 ± 20.95
	OSI	42.17 ± 1.23	238.17 ± 11.55	5.08 ± 0.05	70.87 ± 15.73
	SSI	32.30 ± 0.26	181.95 ± 7.08	7.87 ± 0.05	58.61 ± 10.42
	PSI	34.51 ± 0.24	405.74 ± 7.98	1.14 ± 0.03	389.80 ± 8.02
Distributed	RR	46.80 ± 0.39	233.77 ± 9.83	0.004 ± 0.0006	131.19 ± 16.49
	OSI	23.81 ± 0.15	138.93 ± 5.31	4.20 ± 0.09	97.62 ± 11.36
	SSI	24.23 ± 0.20	137.84 ± 5.41	6.43 ± 0.05	77.70 ± 10.78
	PSI	22.57 ± 0.15	169.12 ± 4.65	1.34 ± 0.02	207.94 ± 9.42

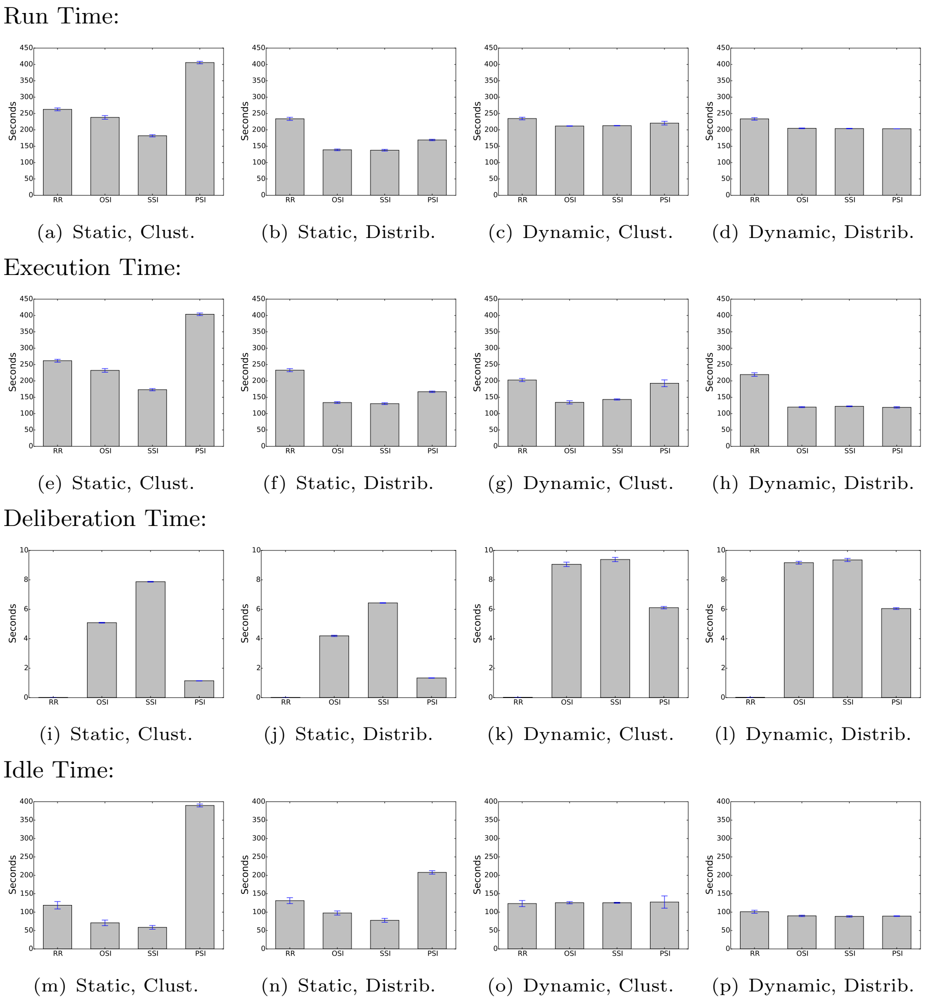
		Distance	Run time	Delib. time	Idle time
Clustered	RR	57.03 ± 0.93	234.65 ± 9.15	0.0153 ± 0.001	123.31 ± 18.69
	OSI	37.52 ± 3.64	211.79 ± 1.39	9.06 ± 0.34	125.56 ± 6.92
	SSI	35.78 ± 0.16	212.88 ± 1.61	9.38 ± 0.33	125.42 ± 3.08
	PSI	43.23 ± 3.26	220.62 ± 12.75	6.11 ± 0.18	127.4 ± 37.57
Distributed	RR	69.7 ± 1.69	233.54 ± 8.72	0.0143 ± 0.0007	101.04 ± 9.54
	OSI	28.2 ± 0.14	204.72 ± 2.43	9.17 ± 0.22	89.88 ± 4.42
	SSI	28.25 ± 0.12	204.0 ± 2.04	9.35 ± 0.24	88.45 ± 4.48
	PSI	28.14 ± 0.26	203.43 ± 0.52	6.05 ± 0.14	89.21 ± 2.08

Table 1. Team metrics for the static (top) and dynamic (bottom) allocations. Time is in seconds. Distance is in metres. The values given are means with 95% confidence intervals.

the smallest overall combined distance on average. This means that the solutions generated by SSI were executed quickly in comparison to those generated by the other allocation mechanisms. In the remaining scenarios, the performance of SSI in terms of overall distance diminished in comparison to the other mechanisms.

SSI also performs well in terms of idle time. An individual robot accumulates idle time when it finishes visiting its allocated target points before other robots finish visiting theirs, so across the team it is a measure of wasted resource. For the static allocation scenarios, SSI has the lowest idle time of all the mechanisms. The cost for this performance can be seen in the deliberation times. SSI, which requires bids from all robots for all unallocated points on all rounds, involves more bidding than any of the other approaches, and this translates into the longest time spent in the allocation process (deliberation time). Only OSI in the dynamic allocations comes close to the deliberation time required by SSI.

In the scenarios we consider here, the deliberation times are never more than a small fraction of the total time to execute the set of tasks, but it is easy to imagine cases where deliberation time is more significant. The total number of bids to allocate m tasks to n robots is one bid from each robot for m tasks in the first round, one bid from each robot for $m - 1$ tasks in the second round, and so



Run Time:

(a) Static, Clust. (b) Static, Distrib. (c) Dynamic, Clust. (d) Dynamic, Distrib.

Execution Time:

(e) Static, Clust. (f) Static, Distrib. (g) Dynamic, Clust. (h) Dynamic, Distrib.

Deliberation Time:

(i) Static, Clust. (j) Static, Distrib. (k) Dynamic, Clust. (l) Dynamic, Distrib.

Idle Time:

(m) Static, Clust. (n) Static, Distrib. (o) Dynamic, Clust. (p) Dynamic, Distrib.

Fig. 3. The remaining metrics for the teams. In order the rows give run time execution time, deliberation time, and idle time. The bars are ordered RR, OSI, SSI and PSI.

on,[6] and it is conceivable that this could become big enough to be problematic. For example, consider the case of one hundred robots — as in the Centibots project [10] — which have to allocate 500 target points. In such a case the SSI auction would require over 12 million bids, a 12,000-fold increase over what is required in our experiments, and enough to make deliberation time a significant contributor to the time for task completion. In addition, since each bid has to be transmitted wirelessly — either to a centralised auction manager, or to all

[6] Giving a total of $n\left(\frac{m(m+1)}{2}\right)$ bids.

other robots in a distributed auction — the number of messages can be a factor in robot deployments where communication bandwidth is limited [19].

4 Related work

As mentioned above, there is much related work on auction mechanisms for task allocation in robotics. Some of this has already been mentioned, and here we take a look at the relevant work that we have not yet covered. The use of market mechanisms in distributed computing can be considered to start with Smith's contract net protocol [22], and this was followed by Wellman and Wurman's [24] *market-aware agents*. A primary strength of market-based approaches is their reliance only on local information or the self-interest of the agents to arrive at efficient solutions to large-scale, complex problems that are otherwise intractable [2].

The most common instantiations of market-based approaches in multirobot systems are auctions. Auctions are commonly used for distribution tasks, where resources or roles are treated as commodities and auctioned to agents. A significant body of work analyzes the effects of different auction mechanisms, [1,11,25], bidding strategies [18], dynamic task re-allocation or swapping [5] and levels of commitment to the contracts [14] on the overall solution quality.

In domains where there is a strong synergy between items, single-item auctions can result in sub-optimal allocations [1]. In the domain of interest here, multi-robot exploration, a strong synergy exists between target points that robots need to explore. Combinatorial auctions remedy this limitation by allowing agents to bid on bundles of items, and minimize the total travel distance because they take the synergies between target points into account [7]. Combinatorial auctions suffer, however, from the computational costs of bid generation and bundle valuation by the agents, and winner determination by the auction mechanism itself, all of which are NP-hard [8].

Of all the literature on auctions in multi-robot teams, [15] and [20] are the most closely related to our work. Both evaluate SSI in simulation and so could argue their work evaluates the practical cost of solutions generated using SSI (and [15] provides some results from real robots). However, the focus of both [15] and [20] is on finding optimal mechanisms for dynamic task reallocation during execution rather than, like our work, attempting to characterise auction mechanisms in terms of the tasks for which they are best fitted. In addition, neither consider the range of metrics that we do and so are unable, for example to comment on the load balancing that measuring idle time exposes.

5 Summary

This paper has studied the performance of a number of auction mechanisms on a version of the multi-robot routing problem. It has gone further than any work we are aware of in testing auction mechanisms across a range of performance metrics, in a way that evaluates how the mechanisms would work on a fleet of robots, and compares how mechanisms perform in dynamic versus static scenarios.

The main result is that while the sequential single-item (SSI) auction broadly outperforms other single item auctions in static scenarios where the robots start clustered together, its advantage is diminished when robots are distributed, and is also reduced when the target points are allocated dynamically. Given that we have only studied one specific set of target points in each of the static and dynamic scenarios, this result is tentative, and we are continuing to run experiments, gathering data on different configurations of target points, to ensure that our results are not an artefact of a specific configuration.

Finally, we are also running experiments with real robots on exactly the same scenarios studied here, but as yet do not have enough results to report.

Acknowledgements

This work was partially funded by National Science Foundation grants #IIS-1116843 and #IIS-1338884, and by a University of Liverpool Research Fellowship.

References

1. M. Berhault, H. Huang, P. Keskinocak, S. Koenig, W. Elmaghraby, P. M. Griffin, and A. Kleywegt. Robot Exploration with Combinatorial Auctions. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2003.
2. M. B. Dias, R. Zlot, N. Kalra, and A. Stentz. Market-based multirobot coordination: A survey and analysis. *Proceedings of the IEEE*, 94(7):1257–1270, 2006.
3. P. Ezhilchelvan and G. Morgan. A dependable distributed auction system: Architecture and an implementation framework. In *Proceedings of the 5th International Symposium on Autonomous Decentralized Systems*, 2001.
4. B. Gerkey, R. T. Vaughan, and A. Howard. The Player/Stage Project: Tools for Multi-Robot and Distributed Sensor Systems. In *Proceedings of the 11th International Conference on Advanced Robotics*, 2003.
5. M. Golfarelli, D. Maio, and S. Rizzi. A Task-Swap Negotiation Protocol Based on the Contract Net Paradigm. Technical Report 005-97, DEIS, CSITE - Università di Bologna, 1997.
6. M. K. Habib. Humanitarian Demining: Reality and the Challenge of Technology. *International Journal of Advanced Robotic Systems*, 4(2):151–172, 2007.
7. S. Koenig, P. Keskinocak, and C. Tovey. Progress on agent coordination with cooperative auctions. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2010.
8. S. Koenig, C. Tovey, M. Lagoudakis, D. Kempe, P. Keskinocak, A. Kleywegt, A. Meyerson, and S. Jain. The Power of Sequential Single-Item Auctions for Agent Coordination. In *Proceedings of National Conference on Artificial Intelligence*, 2006.
9. S. Koenig, C. A. Tovey, X. Zheng, and I. Sungur. Sequential bundle-bid single-sale auction algorithms for decentralized control. In *Proceedings of the International Joint Conference on Artificial Intelligence*, 2007.

10. K. Konolige, D. Fox, C. Ortiz, A. Agno, M. Eriksen, B. Limketkai, J. Ko, B. Morisset, D. Schulz, and B. Stewart. Centibots: Very large scale distributed robotic teams. In *Experimental Robotics IX*, pages 131–140. Springer, 2006.
11. S. Kraus. Automated negotiation and decision making in multiagent environments. In *MultiAgent Systems and Applications*. Springer-Verlag, 2001.
12. M. Lagoudakis, M. Berhault, S. Koenig, P. Keskinocak, and A. Kelywegt. Simple auctions with performance guarantees for multi-robot task allocation. In *Proceedings of the International Conference on Intelligent Robotics and Systems (IROS)*, 2004.
13. M. Lagoudakis, V. Markakis, D. Kempe, P. Keskinocak, S. Koenig, A. Kleywegt, C. Tovey, A. Meyerson, and S. Jain. Auction-based multi-robot routing. In *Proceedings of Robotics: Science and Systems Conference*, 2005.
14. M. Mataric, G. Sukhatme, and E. Ostergaard. Multi-robot task allocation in uncertain environments. *Autonomous Robots*, 14(2-3):255–263, 2003.
15. M. Nanjanath and M. Gini. Repeated auctions for robust task execution by a robot team. *Robotics and Autonomous Systems*, 58(7):900–909, 2010.
16. A. T. Özgelen, E. Schneider, E. I. Sklar, M. Costantino, S. L. Epstein, and S. Parsons. A first step toward testing multiagent coordination mechanisms on multi-robot teams. In *Proceedings of the Workshop on Autonomous Robots and Multirobot Systems (ARMS) at Autonomous Agents and MultiAgent Systems (AAMAS)*, 2013.
17. Morgan Quigley, Ken Conley, Brian P. Gerkey, Josh Faust, Tully Foote, Jeremy Leibs, Rob Wheeler, and Andrew Y. Ng. Ros: an open-source robot operating system. In *ICRA Workshop on Open Source Software*, 2009.
18. S. Sariel and T. Balch. Efficient bids on task allocation for multi-robot exploration. In *Proceedings of the Nineteenth Florida Artificial Intelligence Research Society Conference*, 2006.
19. A. V Savkin. The problem of coordination and consensus achievement in groups of autonomous mobile robots with limited communication. *Nonlinear Analysis: Theory, Methods & Applications*, 65(5):1094–1102, 2006.
20. A. Schoenig and M. Pagnucco. Evaluating sequential single-item auctions for dynamic task allocation. In *AI 2010: Advances in Artificial Intelligence*. Springer, 2011.
21. E. I. Sklar, A. T. Özgelen, J. P. Muñoz, J. Gonzalez, M. Manashirov, S. L. Epstein, and S. Parsons. Designing the HRTeam Framework: Lessons Learned from a Rough-and-Ready Human/Multi-Robot Team. In *Proceedings of the Workshop on Autonomous Robots and Multirobot Systems (ARMS) at Autonomous Agents and MultiAgent Systems (AAMAS)*, Taipei, Taiwan, May 2011.
22. R. G. Smith. The Contract Net Protocol: High-Level Communication and Control in a Distributed Problem Solver. In Alan H. Bond and Les Gasser, editors, *Distributed Artificial Intelligence*. Morgan Kaufmann Publishers Inc., 1988.
23. C. Tovey, M. G. Lagoudakis, S. Jain, and S. Koenig. Generation of Bidding Rules for Auction-Based Robot Coordination. In *Proceedings of the 3rd International Multi-Robot Systems Workshop*, March 2005.
24. M. P. Wellman and P. R. Wurman. Market-aware agents for a multiagent world. *Robotics and Autonomous Systems*, 24:115–125, 1998.
25. R. Zlot, A. Stentz, M. B. Dias, and S. Thayer. Multi-robot exploration controlled by a market economy. In *Proceedings of the IEEE Conference on Robotics and Automation*, 2002.