Mechanism Selection for Multi-Robot Task Allocation

Eric Schneider[1,2(✉)], Elizabeth I. Sklar[1], and Simon Parsons[1]

[1] Department of Informatics, King's College London, London, UK
{eric.schneider,elizabeth.sklar,simon.parsons}@kcl.ac.uk
[2] Department of Computer Science, University of Liverpool, Liverpool, UK

Abstract. The work presented here investigates how environmental features can be used to help select a task allocation mechanism from a portfolio in a multi-robot exploration scenario. In particular, we look at clusters of task locations and the positions of team members in relation to cluster centres. In a data-driven approach, we conduct experiments that use two different task allocation mechanisms on the same set of scenarios, providing comparative performance data. We then train a classifier on this data, giving us a method for choosing the best mechanism for a given scenario. We show that selecting a mechanism via this method, compared to using a single state-of-the-art mechanism only, can improve team performance in certain environments, according to our metrics.

Keywords: Multi-robot team · Auction mechanism · Task allocation

1 Introduction

We are studying task allocation in multi-robot teams, known as the *multi-robot task allocation* (MRTA) problem. This is an issue of critical importance in the deployment of such teams, and an issue that must be addressed if the future potential of autonomous robots is going to be fullfilled. A popular approach to the problem is to apply market-based methods, such as auctions [3]. In this approach, tasks are offered to the team members and team members bid against each other for the tasks. A typical bidding strategy has bids derived from the distance of the robot from the site at which the task needs to be carried out. Allocating tasks to the robot making the lowest bid can then ensure an allocation which reduces the total distance travelled by the team.

Many market-based mechanisms have been suggested for the task allocation problem. These mechanisms vary considerably in the trade-offs that they make between computation time and space, and the quality of solutions that they deliver, measured by metrics such as the total distance covered by the team while completing a set of tasks. In addition, the performance of mechanisms seems to be greatly affected by the environments in which they are deployed. In some environments, a simple, greedy mechanism which might not be expected to perform well in the general case may, in fact, perform competitively with more

Y. Gao et al. (Eds.): TAROS 2017, LNAI 10454, pp. 421–435, 2017.
DOI: 10.1007/978-3-319-64107-2_33

sophisticated mechanisms, with the advantage of scaling better. Prior work has shown evidence that this is the case in both simulated and physical experiments [22–24].

In particular, in our earlier work, we showed [23,24] that while the sequential single-item auction [12] performs better than the parallel single-item auction [12] when the allocation is carried out with robots clustered together geographically, this advantage diminishes as robots are distributed over space and tasks are distributed over space and time. Based on this observation, in this paper we propose a portfolio-based approach to the MRTA problem. Given a set of task allocation mechanisms and a set of environmental features that we can measure, we would like to be able to classify a previously unseen environment in order to choose a task allocation mechanism that performs well in it.

By "environments", we refer here to the spatial arrangements and distributions of robots and tasks. It seems appropriate to apply the tools and techniques of cluster analysis to these environments. We need to consider environmental obstacles like walls, so it seems natural to model environments as graphs, where nodes may be robots and/or task locations, and edges are paths computed by a path planner (e.g., A* [8]) between these nodes, around obstacles. The graphs we can construct in this way resemble something like road networks, and that suggests an approach to characterising different environments.

The distribution of sites over road-network-like graphs is a well-studied research area in Geographic Information Systems (GIS). One particular class of problem from GIS that is useful to apply here is *location-allocation* or *facility location*, which seeks to determine the ideal locations for "facilities" and allocates "demand points" to them in a way that minimises some measure of overall cost or maximises some overall utility. Examples of facilities and demand points might be warehouses and customers, or police stations and potential crime scenes, respectively. In the family of facility location problems [19], the p-median problem seems most suitable here, with p representing the number of facilities one wishes to locate.

In our case we can think of robot team members as facilities and task locations as demand points. If we can solve such a facility location problem for one of our scenarios, where the number of facilities is equal to the number of team members, we might find an ideal set of team starting locations for a simple, greedy mechanism like the parallel single-item auction. Our hypothesis is that if actual robot start locations are close to ideal facility locations, then the parallel single-item auction will lead to competitive performance. Conversely, if actual robot start locations are far away from ideal facility locations, then a more sophisticated mechanism like the sequential single-item auction is a better choice. Furthermore, it will be possible to select the best mechanism for specific sets of start and facility locations based only on knowledge about those locations. This paper provides an empirical test of this hypothesis and finds that, at least for some sets of locations, we can use machine learning to identify the best mechanism to use. We show that this approach can produce significant improvements in team performance.

The rest of this paper is structured as follows. Section 2 reviews the related literature. Section 3 then introduces our methodology, and Sect. 4 describes the experiments that we carried out to test our hypothesis. Section 5 presents our results; Sect. 6 discusses our results, arguing that they support our hypothesis, at least for some sets of robot start locations and some performance metrics; and Sect. 7 concludes.

2 Related Work

The use of market mechanisms in distributed computing can be considered to start with Smith's *contract net* protocol [25]. A strength of market-based approaches is their reliance only on local information and/or the self-interest of agents to arrive at efficient solutions to large-scale, complex problems that are otherwise intractable [3]. The most common instantiations of market mechanisms in multi-robot systems are auctions. Auctions are commonly used for distribution tasks, where resources or roles are treated as goods and auctioned to agents. Existing work analyses the effects of different auction mechanisms on overall solution quality [1,3,13,30]. A body of work has grown up around the *sequential single-item auction* (SSI) [12], which has been proven to create close to optimal allocations, exploiting synergies of related tasks while not suffering from the complexity issues of combinatorial auctions.

Location theory sits at the intersection of Geographic Information Systems (GIS) and economics. The *p-median* problem is one class of *location-allocation* or *facility location* problem that seeks to find optimal locations among existing sets of points that either maximize some measure of distribution utility or minimize some measure of cost [19]. Hakimi developed such problems on a graph to locate optimal switching centres for communication networks or police stations in a highway system [7]. Kariv and Hakimi showed that finding solutions to p-median problems is NP-hard on a general graph [11], but heuristics have been developed to make this more efficient [2,26].

Clustering or bundling of tasks has been investigated in the design of task allocation mechanisms. Sandholm extended Smith's Contract Net Protocol with *C-contracts* (cluster contracts), which award bundles of tasks, rather than single tasks, to agents [21]. Dias and Stentz proposed a mechanism that clusters geographically close tasks into a forest of minimum spanning trees, which may then be auctioned and potentially swapped [4]. Heap proposed sequential-single-cluster (SSC) auctions, an extension to SSI that uses an agglomerative clustering algorithm to create task bundles, which are then auctioned as in SSI [9]. Liu and Shell [16] develop a hybrid distributed-centralised approach to MRTA. The task set is first partitioned into subsets that are then solved in parallel using a centralised assignment algorithm [14].

The problem of algorithm selection and criteria for selecting an algorithm were proposed at least as early as Rice [20]. Computational or algorithm portfolios that use domain knowledge to define features of problem instances in order to select an appropriate algorithm have been investigated by Huberman et al. [10],

Gomes and Bart [6], and Leyton-Brown et al. [15]. Portfolio-based SAT solvers like SATzilla [29] and Hydra [28] have had success in selecting appropriate heuristics to solve NP-hard problems. As far as we are aware, a portfolio-based approach to market-based mechanism selection for MRTA problems, as we propose here, is novel.

3 Methodology

Here we define terms and notation and describe the software architecture used to conduct our experiments before detailing the proposed method for mechanism selection.

3.1 Definitions and System Architecture

As discussed above, our work focuses on task allocation in multi-robot teams. As in prior work [23], we consider that a set of n robots $R = \{r_0, \ldots, r_{n-1}\}$ comprises a *team* and a *mission* is a set of m tasks $T = \{t_0, \ldots, t_{m-1}\}$. A *task scenario* defines a map and the locations of tasks in T. Finally, a *parameterised environment* defines a task scenario, the starting locations of the team, and properties of tasks such as precedence ordering or arrival time. In the work presented here, all parameterised environments are, in the classification of [24], *SR-IT-SA*, that is, they comprise single-robot, independently ordered, statically allocated tasks.

Also as discussed above, this work makes use of the concept of the median of a graph. In particular, we consider the ***p*-median problem**:

> Given a graph or a network $G = (V, E)$, find $V_p \subseteq V$ such that $|V_p| = p$ and the sum of the shortest distances from the vertices in $\{V \setminus Vp\}$ to their nearest vertex in V_p is minimized [19].

We use the p-medians of a graph that spans robot start locations and task locations to determine which task allocation mechanism to employ in a given scenario.

Our work makes use of MRTeAm, a software framework for conducting multi-robot coordination experiments [23,24]. It is written on top of ROS [18] and its main components are a central *auctioneer* agent and multiple *robot controller* agents. The auctioneer is responsible for loading a task scenario and conducting auctions according to the rules of one of several types of auction mechanisms. That is, it announces tasks to, receives and aggregates bids from, and finally awards tasks to the robot controller agents. The robot controllers are responsible for computing and submitting bids using an A* path planner [8] and executing tasks once they have all been awarded.

3.2 Task Allocation Mechanisms

The portfolio method proposed here considers two auction mechanisms that we have investigated in previous work. In a sequential single-item auction (SSI) [12], unallocated tasks are advertised to all robots at once. Each robot bids on the task with the lowest path cost and each task is allocated to the robot that made the lowest bid for that task. The winning task is removed from the set of tasks to be advertised in the next round and the process is repeated until all tasks have been allocated. In a parallel single-item auction (PSI) [12], allocation starts like SSI but all robots bid on all points from their current locations. All the tasks are allocated in one round, with each task going to the lowest bidding robot that bid on it. In later sections we also refer to SEL, which represents our portfolio selection method.

3.3 Classification of Mechanism Selection Methods

The method we propose works as follows. On a map (shown in Figs. 1 and 2), we generate a large number of parameterised environments in which task and robot starting locations are randomly chosen over a uniform distribution with some buffer distance from walls (and robots from each other). For each environment, we conduct an experimental run with both PSI and SSI mechanisms. This generates a pair of results with the same starting conditions but different performance outcomes. From these results, we create a training instance for each environment by recording properties of that environment as training features (Table 1) and the winning mechanism, for some performance metric we wish to optimise, as a label. Finally, after balancing the training set and selecting features, we train a (binary) classifier to predict a winning mechanism.

We can now, in previously unseen environments (i.e., arrangements of task and robot starting locations), query our classifier at runtime to select the mechanism that is predicted to perform best in that environment.

3.4 Features and Training

The features used to build our training sets are defined in Table 1. For a given parameterised environment, there are three main steps to produce a training instance:

1. *Building the graph*: ROS's global planner[1] is invoked to construct a path between each pair of task locations. The result is a complete graph whose nodes are task locations and whose edges are paths planned between them. The graph is complete for the sake of simplifying the calculation in step 2, but completeness comes at a price: the path planner is invoked $O(m^2)$ times. This cost is important to consider, as the graph ultimately needs to be constructed at runtime.

[1] http://wiki.ros.org/global_planner.

Table 1. Environmental properties recorded as training features

Feature	Description
total distance to assigned medians	Sum of all robots' distances to their (SSI-) assigned medians
total distance to all medians	Sum of all robots' distances to all medians
maximum distance to assigned median	Maximum distance of any robot to its (SSI-)assigned median
maximum distance to any median	Maximum distance of any robot to any median
minimum distance to assigned median	Minimum distance of any robot to its (SSI-)assigned median
minimum distance to any median	Minimum distance of any robot to any median
assigned median distance spread	*maximum distance to assigned median* minus *minimum distance to assigned median*
total median distance spread	*maximum distance to any median* minus *minimum distance to any median*
greedy median count spread	Max. number of p-medians greedily (PSI-)assigned to any one robot minus min. number of the same
team diameter	Longest distance between any two team members

2. *Finding the medians*: The task graph is represented as a weighted adjacency matrix as input for a p-median solver. We use the Teitz-Bart method [26] from Xiao [27]. The result is a list of $p = n$ nodes, coincident with task locations, which we hypothesise to be ideal start locations for the robot team members when using PSI.
3. *Assigning medians to robots*: Deciding which median should be assigned to which robot is a task allocation problem in itself. Here we use both the PSI and SSI mechanisms to compute possible assignments. With all data in the memory of a single process (the auctioneer), no messages need be communicated among separate agents. Once an assignment of medians to robots has been computed, we can calculate the distance of each robot's start location to its assigned median(s). Examples of assignments are shown in Figs. 1 and 3. Assignments in Fig. 3 are shown as straight line paths for clarity of illustration. It should be mentioned that different robot-to-median assignments will produce different distance measurements (Table 1).

4 Experiments

Our main hypothesis is that if robot start locations are close to medians (i.e., ideal "facility" locations), then PSI will perform at least as well as SSI, if not

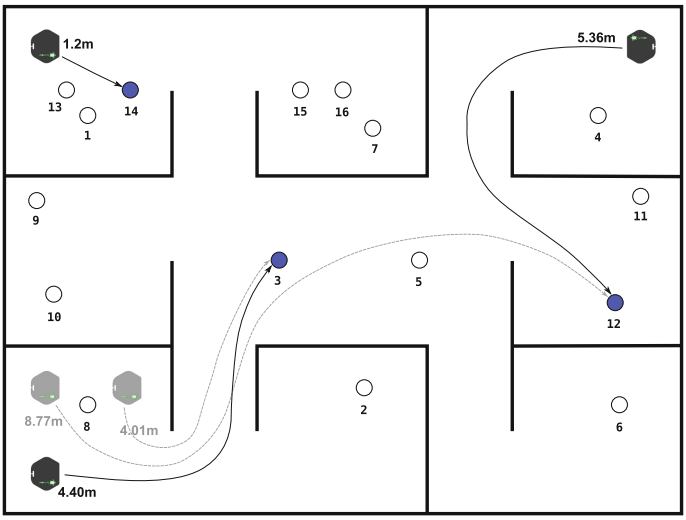


Fig. 1. The path distance of each robot to its assigned median for *clustered* (faint) and *distributed* (dark) start configurations with the same set of task locations.

better. We ran experiments to investigate if this hypothesis could be supported over a range of environments. One shortcoming of our prior work was that task and robot start locations were somewhat ad hoc, being chosen manually. To address this, in the experiments reported here, locations were randomised as described in Sect. 3.3.

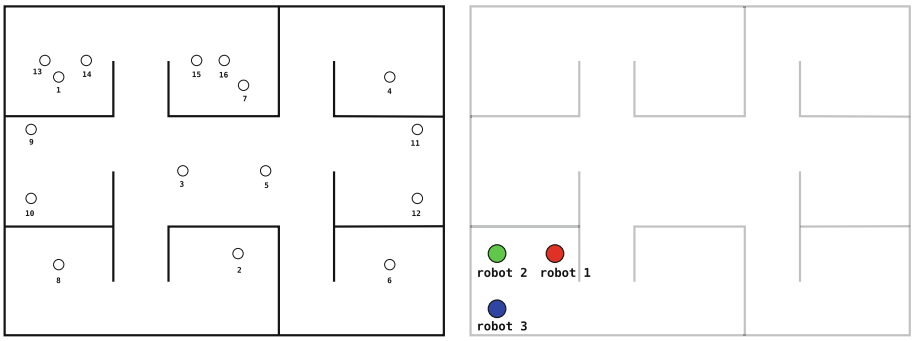
4.1 Environments

To test our mechanism selection method, two types of randomised environments were investigated:

1. In *fixed start, random task location* environments, robot start locations were fixed in the *clustered* and *distributed* arrangements shown in Figs. 1 and 2b[2] and task locations were chosen randomly.
2. In *random start, random task location* environments, both robot start locations and task locations were chosen randomly.

In all environments that we investigated, the size of the robot team ($n = 3$) and number of tasks ($m = 16$) were fixed. All experiments were conducted using the Stage simulator [5]. For *fixed start, random task location* environments, 150 environments were generated to test mechanism selection, for a total of 150 $\times$ $\{clustered|distributed\}$ starts $\times$ $\{\text{PSI}|\text{SSI}|\text{SEL}\}$ mechanisms = 900 experimental runs for each performance objective. For *random start, random task location* environments, 300 environments were generated for a total of 300 $\times$ $\{\text{PSI}|\text{SSI}|\text{SEL}\}$ mechanisms = 900 experimental runs for each performance objective.

[2] These are the same arrangements as used in [23,24].



(a) A task scenario (b) Fixed (clustered) start locations

Fig. 2. Examples of task and start locations discussed in Sect. 4.1. (a) shows an example task scenario with randomly chosen task locations. (b) shows one of the fixed (*clustered*) sets of robot starting locations.

4.2 Performance Metrics

While there are a number of ways to measure performance, we generally measure the travel distance and the time it takes for the team to reach all of the task locations. Specifically, we define four performance metrics.

Total distance travelled (meters) is the sum of the lengths of the paths travelled by team members over the course of an experiment. This is a measure of the use of resources (e.g. battery power or fuel) by the team as a whole. *Maximum robot distance* (meters) is the maximum distance travelled by any one robot. It also measures resource usage, but gives an indication of how balanced the load of a mission is across the team for a given allocation. *Execution phase time* (seconds) measures how long it takes the team, after an allocation has been computed, to travel to all task locations. *Total run time* (seconds) measures the time it takes for a mechanism to compute and conduct an allocation, plus execution phase time. Smaller values are preferred. Ideally, a task allocation mechanism will seek to perform well by minimising all of these metrics.

5 Results

We present results from three stages of our experiments. First we show some properties of the initial set of experiments that served as training data for classifiers. Second, we show classifier accuracy on held-out portions of the training data. Finally, we show the ultimate results of using these classifiers to select mechanisms from our portfolio.

5.1 Training Data

Properties of the initial set of experiments that served as training data are shown in Tables 2, 3 and Fig. 3. In general, PSI was better, in all environments tested,

Table 2. Fixed start, random task locations

Metric	PSI wins	%	SSI wins	%
Team distance	**403**	**90.97**	40	9.03
Maximum robot distance	135	30.47	**308**	**69.53**
Total run time	148	33.41	**295**	**66.59**
Execution phase time	117	26.41	**326**	**73.59**

Table 3. Random start, random task locations

Metric	PSI wins	%	SSI wins	%
Team distance	**744**	**81.2**	172	18.8
Maximum robot distance	85	9.28	**831**	**90.72**
Total run time	230	25.11	**686**	**74.89**
Execution phase time	118	12.88	**798**	**87.12**

at producing allocations that led to shorter *team distances* than SSI. SSI was better, in all environments, at producing shorter *maximum robot distances*. In the *random start, random task locations* environments (Table 3), SSI greatly outperformaned PSI on the *maximum robot distance*, *total run time*, and *execution phase time* objectives. In the *fixed start, random task locations* environments (Table 2), SSI was somewhat less dominant. These results show that, while SSI performs better overall, PSI allocations do sometimes result in performance that is competitive with SSI allocations, as we have seen in previous work [23,24].

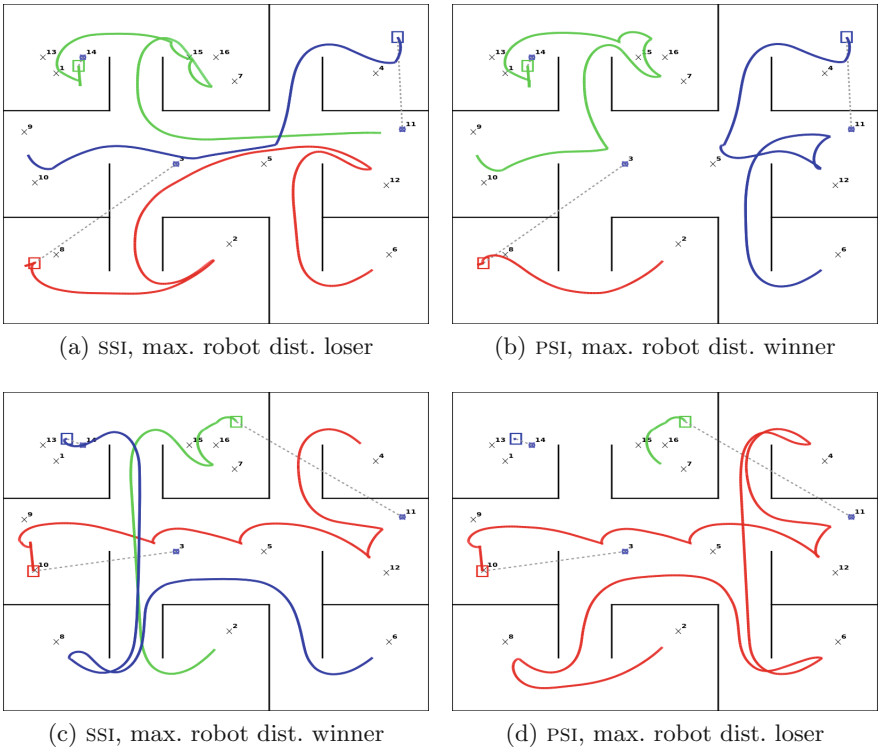
5.2 Classifier Performance

Initially, the training sets had a severe class imbalance. In the training set for the *maximum robot distance* objective, for example, SSI was the winning mechanism in 1139 cases compared to PSI's count of 220 (Tables 2 and 3). We balanced the training sets using a random undersampling method,[3] although other methods are also possible.

Table 4. Accuracy of several classifiers trained for different performance objectives.

Classifier type	Objective	Accuracy	Std. Dev.
Random forest	Execution phase time	75.22%	0.91%
SVM	Execution phase time	74.55%	1.00%
Random forest	Maximum robot distance	80.88%	1.26%
SVM	Maximum robot distance	76.80%	1.20%

[3] https://github.com/scikit-learn-contrib/imbalanced-learn.



(a) SSI, max. robot dist. loser (b) PSI, max. robot dist. winner

(c) SSI, max. robot dist. winner (d) PSI, max. robot dist. loser

Fig. 3. Trajectories and p-median assignments for two sets of *random start, random task location* environments. Robot start locations are shown as large open coloured squares, task locations are shown as $\times$ marks, and medians are shown as small closed coloured squares. (a) and (b) show an environment where the PSI allocation led to a smaller maximum robot distance. (c) and (d) show an environment in which an SSI allocation led to a smaller maximum robot distance.

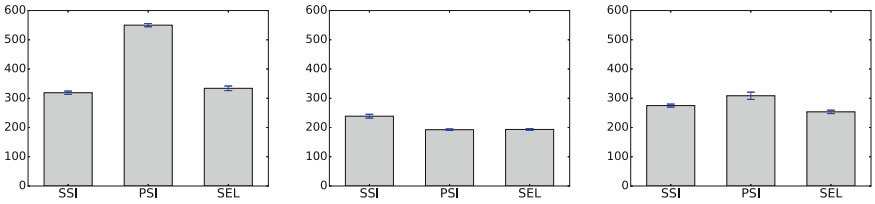
We used the scikit-learn [17] library to select features and train classifiers. Various types of classifier were investigated, including decision trees, k-nearest neighbours, random forests and support vector machines. Table 4 shows the average accuracy of some of these classifiers on held-out data over 10-fold cross validation.

As a result of these experiments, we selected the random forest classifier for the remainder of the work presented here. The features selected for the random forest classifier trained to optimise the *execution phase time* objective discussed in the next section were: {*maximum distance to assigned median*, *assigned median distance spread*, *team diameter*, and *greedy median count spread*}. Features selected for the random forest classifier trained to optimise the *maximum robot distance* objective were: {*total distance to all medians*, *maximum*

distance to any median, *team diameter*, and *greedy median count spread*}. Hyperparameters of the classifiers were tuned using a grid search.[4]

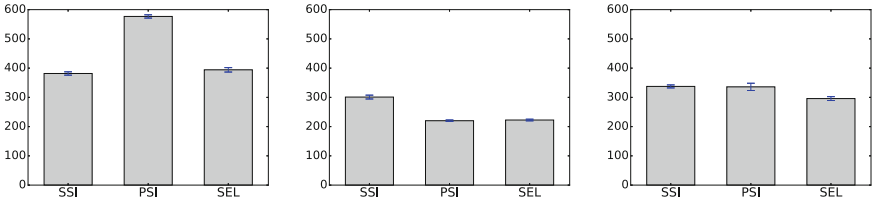
5.3 Mechanism Selection Results

Having trained a classifier, we then used it in experiments to see if a method which uses initial locations to pick an allocation mechanism using this classifier, a method we called SEL, can outperform either SSI or PSI. Some of the ultimate results of these experiments are shown in Figs. 4, 5, 6 and 7. Each figure shows the average value measured for a particular metric and mechanism with 95% confidence intervals. Figures 4, 5 and 6 show results for *fixed start, random task locations* environments. Figure 7 shows results for *random start, random task locations* environments. We trained classifiers for both the *maximum robot distance* and *execution phase time* performance objectives, but only the results for the *execution phase time* objective are shown here.



(a) Clustered (b) Distributed (c) Combined

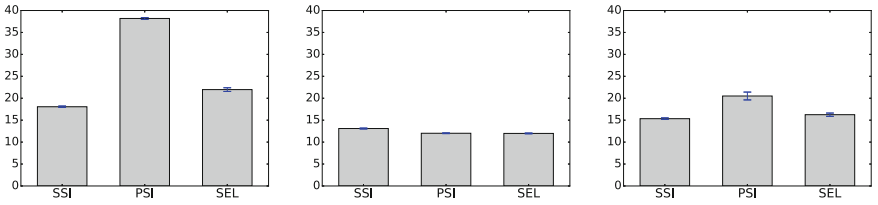
Fig. 4. Execution phase time for clustered starts (a), distributed starts (b), and the combination of clustered and distributed starts (c). Units are seconds.



(a) Clustered (b) Distributed (c) Combined

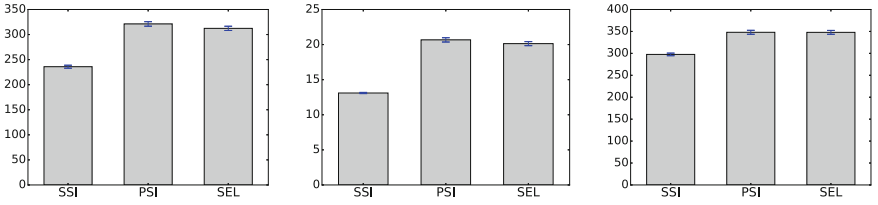
Fig. 5. Total run time for clustered starts (a), distributed starts (b), and the combination of clustered and distributed starts (c). Units are seconds.

[4] http://scikit-learn.org/stable/modules/grid_search.html.



(a) Clustered (b) Distributed (c) Combined

Fig. 6. Maximum robot distance for clustered starts (a), distributed starts (b), and the combination of clustered and distributed starts (c). Units are meters.



(a) Exec. Phase Time (b) Max. Robot Distance (c) Total Run Time

Fig. 7. Mechanism selection results for *random start, random task locations* environments. Time units are seconds. Distance units are meters.

6 Discussion

Examining the full range of environments we investigated, the performance improvements of our mechanism selection method are mixed.

In *random start, random task location* environments, the classifier we trained to predict which of the two mechanisms would minimise the *execution phase time* objective did not perform better, on average, than SSI. It led to execution phase times and total run times that were only slightly lower than PSI (Fig. 7a and c). It also did not lead to significantly shorter maximum robot distances (Fig. 7b), but then it was not trained to do so.

In *fixed start, random task location* environments, we did observe a significant performance improvement in *execution phase time* and *total run time* when combining results from the *clustered* and *distributed* fixed start locations (Fig. 4c and 5c). SSI led to an average execution phase time of 274.95 ± 10.04 s while SEL reduced that to 252.10 ± 11.23 s. (Both figures are 95% confidence intervals; an independent t-test yields a t-statistic of 2.99 at $p = 0.0029$.)

These early results are encouraging, but the current SEL method has room for improvement. In addition to reducing execution phase time for randomised task locations with fixed start locations, we would like it to do so for randomised robot starting locations as well. We can make two general kinds of improvements.

First, to improve classifier accuracy we might devise more descriptive environmental features than those listed in Table 1. We might also develop better methods of producing or preparing the training data, train different kinds of classifiers, or learn weights that reward the contributions of these (or other) features. Secondly, we can improve the time it takes to perform the selection method itself. As described in Sect. 3.4, the current method requires the construction of a complete graph with a number of edges that scales quadratically with the size of the mission. A simply connected, rather than complete, graph may suffice, so that building the graph scales linearly with the size of the mission.

7 Summary

We have developed a method of selecting an auction-based task allocation mechanism from a portfolio of mechanisms to address the multi-robot task allocation (MRTA) problem. We have shown that, in some environments, this method can provide an allocation that leads a robot team to execute its mission in significantly less time than that of a single, state-of-the-art mechanism. While these early results are encouraging, the method can be improved in several ways. We are working to improve its performance for more general cases of a known environment, in particular for randomised robot start locations. We are also working to generalise the approach to consider more mechanisms and richer environments—combining it with our previous work on multi-robot, precedence-ordered, and dynamically appearing tasks [23,24]—and on other maps.

References

1. Berhault, M., Huang, H., Keskinocak, P., Koenig, S., Elmaghraby, W., Griffin, P.M., Kleywegt, A.: Robot exploration with combinatorial auctions. In: Proceedings of the International Conference on Intelligent Robotics and Systems (IROS) (2003)
2. Densham, P.J., Rushton, G.: A more efficient heuristic for solving large p-median problems. Pap. Reg. Sci. **71**(3), 307–329 (1992)
3. Dias, M.B., Zlot, R., Kalra, N., Stentz, A.: Market-based multirobot coordination: a survey and analysis. Proc. IEEE **94**(7), 1257–1270 (2006)
4. Dias, M.B., Stentz, A.: Opportunistic optimization for market-based multirobot control. In: Proceedings of the International Conference on Intelligent Robots and Systems (IROS), vol. 3, pp. 2714–2720 (2002)
5. Gerkey, B., Vaughan, R.T., Howard, A.: The player/stage project: tools for multi-robot and distributed sensor systems. In: Proceedings of the 11th International Conference on Advanced Robotics (2003)
6. Gomes, C.P., Selman, B.: Algorithm portfolios. Artif. Intell. **126**(1–2), 43–62 (2001)
7. Hakimi, S.L.: Optimum locations of switching centers and the absolute centers and medians of a graph. Oper. Res. **12**(3), 450–459 (1964)

8. Hart, P., Nilsson, N., Raphael, B.: A formal basis for the heuristic determination of minimal cost paths. IEEE Trans. Syst. Sci. Cybern. **4**(2), 100–107 (1968)
9. Heap, B.: Sequential single-cluster auctions for multi-robot task allocation. Ph.D. thesis, The University of New South Wales, November 2013
10. Huberman, B.A., Lukose, R.M., Hogg, T.: An economics approach to hard computational problems. Science **275**(5296), 51–54 (1997)
11. Kariv, O., Hakimi, S.L.: An algorithmic approach to network location problems. ii: the p-medians. SIAM J. Appl. Math. **37**(3), 539–560 (1979)
12. Koenig, S., Tovey, C., Lagoudakis, M., Kempe, D., Keskinocak, P., Kleywegt, A., Meyerson, A., Jain, S.: The power of sequential single-item auctions for agent coordination. In: Proceedings of National Conference on Artificial Intelligence (2006)
13. Kraus, S.: Automated negotiation and decision making in multiagent environments. In: Luck, M., Mařík, V., Štěpánková, O., Trappl, R. (eds.) ACAI 2001. LNCS, vol. 2086, pp. 150–172. Springer, Heidelberg (2001). doi:10.1007/3-540-47745-4_7
14. Kuhn, H.W.: The Hungarian method for the assignment problem. Naval Res. Logistics Q. **2**(1–2), 83–97 (1955)
15. Leyton-Brown, K., Nudelman, E., Andrew, G., McFadden, J., Shoham, Y.: A portfolio approach to algorithm selection. In: International Joint Conference on Artificial Intelligence (IJCAI), vol. 1543, pp. 1542–1543 (2003)
16. Liu, L., Shell, D.A.: Large-scale multi-robot task allocation via dynamic partitioning and distribution. Auton. Robots **33**(3), 291–307 (2012)
17. Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., Duchesnay, E.: Scikit-learn: machine learning in Python. J. Mach. Learn. Res. **12**, 2825–2830 (2011)
18. Quigley, M., Conley, K., Gerkey, B.P., Faust, J., Foote, T., Leibs, J., Wheeler, R., Ng, A.Y.: ROS: an open-source robot operating system. In: ICRA Workshop on Open Source Software (2009)
19. Reese, J.: Solution methods for the p-median problem: an annotated bibliography. Networks **48**(3), 125–142 (2006)
20. Rice, J.R.: The algorithm selection problem. Adv. Comput. **15**, 65–118 (1976)
21. Sandholm, T.: Contract types for satisficing task allocation: I theoretical results. In: Proceedings of the AAAI Spring Symposium: Satisficing Models, pp. 68–75 (1998)
22. Schneider, E., Balas, O., Özgelen, A.T., Sklar, E.I., Parsons, S.: Evaluating auction-based task allocation in multi-robot teams. In: Workshop on Autonomous Robots and Multirobot Systems (ARMS) at Autonomous Agents and MultiAgent Systems (AAMAS), Paris, France, May 2014
23. Schneider, E., Sklar, E.I., Parsons, S.: Evaluating multi-robot teamwork in parameterised environments. In: Alboul, L., Damian, D., Aitken, J.M.M. (eds.) TAROS 2016. LNCS (LNAI), vol. 9716, pp. 301–313. Springer, Cham (2016). doi:10.1007/978-3-319-40379-3_32
24. Schneider, E., Sklar, E.I., Parsons, S., Özgelen, A.T.: Auction-based task allocation for multi-robot teams in dynamic environments. In: Dixon, C., Tuyls, K. (eds.) TAROS 2015. LNCS, vol. 9287, pp. 246–257. Springer, Cham (2015). doi:10.1007/978-3-319-22416-9_29
25. Smith, R.G.: The contract net protocol: high-level communication and control in a distributed problem solver. In: Bond, A.H., Gasser, L. (eds.) Distributed Artificial Intelligence. Morgan Kaufmann Publishers Inc. (1988)
26. Teitz, M.B., Bart, P.: Heuristic methods for estimating the generalized vertex median of a weighted graph. Oper. Res. **16**(5), 955–961 (1968)

27. Xiao, N.: GIS Algorithms. SAGE Publications, Thousand Oaks (2015)
28. Xu, L., Hoos, H.H., Leyton-Brown, K.: Hydra: automatically configuring algorithms for portfolio-based selection. In: Proceedings of the Twenty-Fourth AAAI Conference on Artificial Intelligence, pp. 210–216. AAAI Press (2010)
29. Xu, L., Hutter, F., Hoos, H.H., Leyton-Brown, K.: Satzilla: portfolio-based algorithm selection for sat. J. Artif. Intell. Res. **32**, 565–606 (2008)
30. Zlot, R., Stentz, A., Dias, M.B., Thayer, S.: Multi-robot exploration controlled by a market economy. In: Proceedings of the IEEE Conference on Robotics and Automation (2002)