Repeated Sequential Single-Cluster Auctions with Dynamic Tasks for Multi-Robot Task Allocation with Pickup and Delivery

Bradford Heap and Maurice Pagnucco

School of Computer Science and Engineering
The University of New South Wales
Sydney, NSW, 2052, Australia.
{bradfordh,morri}@cse.unsw.edu.au

Abstract. In this paper we study an extension of the multi-robot task allocation problem for online tasks requiring pickup and delivery. We extend our previous work on sequential single-cluster auctions to handle this more complex task allocation problem. Our empirical experiments analyse this technique in the domain of an environment with dynamic task insertion. We consider the trade-off between solution quality and overall planning time in globally reallocating all uncompleted tasks versus local replanning upon the insertion of a new task. Our key result shows global reallocation of all uncompleted tasks outperforms local replanning in minimising robot path distances.

1 Introduction

Consider a team of autonomous mobile robots operating as courier delivery vehicles in a large office-like environment (Fig. 1). Each robot is required to pickup from and deliver parcels to a variety of locations around the office building. Each robot may be constrained to a fixed capacity in the number of parcels it can carry at any one time and, after a parcel is picked up, it can only be delivered to its intended destination. Our goal is for the robots to be allocated and deliver all parcels as effectively and efficiently as possible.

The task allocation problem with pickup and delivery is an extension of the widely studied *multi-robot task allocation (MRTA) problem* which, in general, considers each task as a single location to visit. There are many existing approaches for solving this class of problem. However, most existing techniques require that all tasks are static and known before allocation. In many real-world situations, additional tasks are dynamically discovered during execution.

In this paper we extend our previous work on solving MRTA problems through auctioning clusters of geographically close tasks [6]. We propose an algorithm for the formation of task clusters based on pickup and delivery locations. We apply this extended technique to a scenario with dynamically inserted tasks. We compare these results to a scenario where all tasks are known at the outset. Finally, we consider the trade-off between solution quality and overall planning

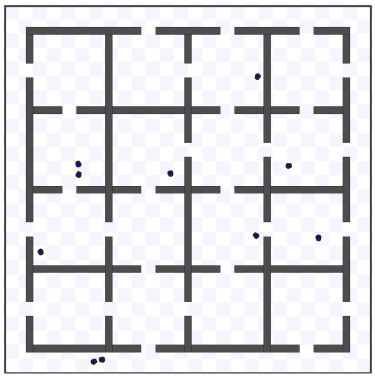

Fig. 1: A simulation of robots operating in an office-like environment.

time in globally reallocating all uncompleted tasks versus local replanning when a robot is assigned a previously unknown task.

2 Problem Definition

Multi-robot routing is considered the standard testbed for MRTA problems [2]. We expand the problem formalisation given by Koenig *et al.* [11] to include tasks with pickup and delivery. Given a set of robots $R = \{r_1, \ldots, r_m\}$ and a set of tasks $T = \{t_1, \ldots, t_n\}$. A partial solution to the MRTA problem is given by any tuple $\langle T_{r_1}, \ldots, T_{r_m} \rangle$ of pairwise disjoint task subsets:

$$T_{r_i} \subseteq T \text{ with } T_{r_i} \cap T_{r_{i'}} = \emptyset,\ i \neq i',\ \forall i = 1, \ldots, m \tag{1}$$

Each task subset T_{r_i} is then assigned to a single robot $r_i \in R$. To determine a complete solution we need to find a partial solution where all tasks are assigned to task subsets:

$$\langle T_{r_1} \ldots T_{r_m} \rangle \text{ with } \cup_{r_i \in R} T_{r_i} = T \tag{2}$$

For tasks with pickup and delivery, the structure of each task t is a tuple $t = \langle l_p, l_d \rangle$ of a pickup location l_p and a delivery location l_d. We consider a robot to be executing a task once it has visited its pickup location and until the point it has reached its delivery location. Robots may have capacity constraints in the number of tasks they are able to execute at any moment in time. This is representative of real robots which may have a fixed maximum number of items they can carry.

We assume robots have perfect localisation[1] and can calculate the cost λ to travel between locations. The cost to travel between any two locations is equal across all robots. The robot cost $\lambda_{r_i}(T_{r_i})$ is the minimum cost for an individual

[1] This allows us to focus on the effectiveness of the bidding method.

robot r_i to visit all locations T_{r_i} assigned to it. There can be synergies between tasks assigned to the same robot, such that:

$$\lambda_{r_i}(\{t\}) + \lambda_{r_i}(\{t'\}) \neq \lambda_{r_i}(\{t\} \cup \{t'\}) \tag{3}$$

A positive synergy is when the combined cost for a robot to complete two tasks is lower than the individual costs for the robot to complete each task:

$$\lambda_{r_i}(\{t\} \cup \{t'\}) < \lambda_{r_i}(\{t\}) + \lambda_{r_i}(\{t'\}) \tag{4}$$

Generating a valid solution to the standard MRTA problem is not difficult. For instance, a simple approach is to assign each task in turn to a randomly selected robot. However, this approach gives no guarantees on the execution time, energy or resources used in completing the assigned tasks. Subsequently, the application of *team objectives* arises to provide additional guidance in the search for solutions to the task allocation that meet certain criteria. For instance, some common desires of a multi-robot system are minimising time spent in execution of tasks, minimising energy or fuel consumed, and/or even distribution of tasks across all robots.

Lagoudakis *et al.* discusses team objectives in detail and their application to MRTA [13]. In this work we use two commonly used team objectives:

MiniMax $\min \max_{r_i \in R} \lambda_{r_i}(T_{r_i})$ that is to minimise the maximum distance any individual robot travels.

MiniSum $\min \sum_{r_i \in R} \lambda_{r_i}(T_{r_i})$ that is to minimise the sum of the paths of all robots in visiting all their assigned locations.

The application of these two team objectives to the solving of the MRTA problem can generate vastly different allocations of tasks to robots. The MiniMax team objective can be considered as the Min-Max Vehicle Routing Problem or the Makespan problem and the MiniSum team objective can be considered as a multi-robot version of the Travelling Salesperson Problem [22].

3 Related Work

3.1 Market-based Task Allocation

Market-based distributed auction algorithms are popular in the robotics community for solving standard MRTA problems [2, 9]. An optimal allocation of tasks to robots can be determined using a single-round *combinatorial auction* [1]. However, winner determination in most combinatorial auctions is NP-complete, they have high communication costs, and are therefore very slow and generally not used in practical applications.

An alternative approach is *sequential single-item auctions* (SSI auctions) which allocate tasks over multiple rounds [13, 10]. Despite SSI auctions producing team costs that are generally sub-optimal, they have much lower communication and winner determination costs which result in a quicker allocation of tasks. A variety of improvements and extensions to SSI auctions have been studied which trade off allocation time against overall team costs [9].

A major weakness in non-optimal auction algorithms with one-shot task assignments is their inability to avoid local minima. For instance, consider the bidding rules for the MiniSum and MiniMax team objectives in standard SSI auctions, when no tasks are assigned to an individual robot the robot will always bid for the task closest to it which in many situations will not be an optimal task assignment. Once they have been assigned this task, all subsequent bids will factor in the inter-task synergies of this task and as a result solutions that are far from optimal are developed [18].

To refine a task allocation solution post-initial allocation in SSI auctions Nanjanath and Gini switch the bidding rule from MiniSum to a MiniMax like approach [16]. Adapting their previous re-auction approach [15] the initial allocation of tasks to robots is done using a standard SSI auction. Then upon each task completion all uncompleted tasks are auctioned under the requirement of minimising execution time. Robots only exchange tasks if it improves the overall team objective and once a task has been exchanged, the robots involved in the exchange replan their paths to travel.

An alternative approach to improving the solution to SSI auction solution with capacity constraints is with K-swaps [25]. K-swaps generalise Sandholm's previous work on contracts for task exchanges [17] and allow two or more robots to exchange differing numbers of tasks to reduce the overall team cost. K-Swaps has been experimentally shown to make significant improvements to task allocations, however, the algorithm trades off larger improvements with pronouncedly increased computational time. Furthermore, task swaps and transfers can also be considered with auctions that incorporate simulated annealing to avoid local minima [24]. Such approach randomly selects tasks and robots for task exchanges and calculates probabilities of the task exchange being accepted and over time can generate an optimal solution.

3.2 Task Clustering

A major computational challenge in the performance of auction mechanisms that consider inter-task synergies is their ability to handle large numbers of robots and tasks. In many auction algorithms increasing the number of tasks causes a combinatorial explosion in the number of calculations required to form task bids. Further compounding this, as the number of robots increases the communication and computational requirements for winner determination also increase. As a result the suitability of these techniques in large real-world scenarios is limited.

Forming clusters of tasks has been explored by a number of researchers as a method to reduce the combinatorial explosion of increasing task counts. In early work on market-based task allocation Sandholm expanded the *Contract Net Protocol* (CNP) [20] by introducing *C-contracts* which replace the CNP's standard one item contract with a contract for a cluster of tasks all of which the contracted robot must complete. Sandholm shows that allocating clusters of tasks to robots can avoid some local minima that single item contracts become stuck in; although, C-contracts can get stuck in different local minima [17]. In early work on multi-robot auctions, Dias and Stentz developed a clustering

algorithm that connects geographically close tasks under the assumption that two tasks that are close have high inter-task synergies, robots then exchange clusters of tasks through an auction method which experimentally is shown to perform better than single task auctions [3].

Many subsequent clustering approaches for MRTA problems use the distance between tasks as a metric for cluster formation. Sariel and Balch discuss the allocation of clusters of tasks to robots where in some situations optimal MRTA solutions can be generated that are unable to be formed using single item auctions, however, in other situations clustering performs worse [18]. Zlot and Stentz use K-means clustering to form clusters of geographically close tasks. To determine an ideal cluster, the value of k is incremented from 2 to the total number of tasks with the value of k generating clusters with the largest relative improvement over the previous value being used [26].

In our previous work [6, 8], we expanded upon SSI auctions to develop *sequential single-cluster auctions* (SSC auctions). In SSC auctions a clustering algorithm forms fixed clusters of geographically close tasks which robots subsequentally bid on using an SSI-like auction technique. Auctioning clusters of tasks reduces the numbers of bids required and thus reduces the communication overhead. This work also demonstrates that repeatedly forming clusters with different task memberships allows robots to consider many combinations of inter-task synergies that are not considered during SSI auctions which only allocate tasks once.

3.3 Dynamic Task Insertion

Despite a large body of work on auction-based algorithms for MRTA problems, few papers have considered the effects of dynamically appearing tasks in the problem domain. While it can be argued that algorithms that continually change task allocations *could* handle dynamically inserted tasks, this has little experimental grounding. An important consideration in the handling of dynamic tasks is deciding how much of the existing task allocation to modify. This can range from an individual robot locally replanning its task execution plan, through to all robots running auctions for a global reallocation of all uncompleted tasks.

Previous work by Schoenig and Pagnucco has considered SSI auctions with dynamically inserted tasks and compared the costs of robots bidding only for the new task versus a full new auction of all uncompleted tasks [19]. Their results show, despite a large trade-off in computational time, a global reallocation of tasks produces lower team costs than local replanning. Zlot *et al.* consider MRTA problems in an exploration domain in which a robot generates additional tasks for allocation after each task completion. These tasks are sequentially offered for auction to other robots, however, if no buyer is found the generating robot retains the task [27]. Viguria, Maxa and Ollero's approach of repeatedly auctioning subsets of uncompleted tasks allows it to handle dynamically inserted tasks [23]. This approach sits between local replanning and global reallocation in that robots only offer for auction tasks that they specifically consider to be of high cost. Additionally, these approaches avoid the problem of never completing any

tasks through ensuring that the currently executing task is never offered for reallocation.

3.4 Tasks with Pickup and Delivery

Little work has focused on distributed auctions for MRTA problems with pickup and delivery. While it can be argued that many existing techniques for single point locations should continue to work for tasks with pickup and delivery this has almost no experimental grounding. Despite this, a large body of work exists in the field of transport logistics.

Fischer, Müller and Pischel apply the CNP to transportation scheduling with fixed time windows [5]. In this work trucks bid for tasks from a central controller and can also make one-for-one swaps with other trucks before they begin to execute their plans. During the execution of plans, the trucks may face traffic delays and as such they can locally replan their routes or auction their uncompleted tasks. Their results show that global reallocation of uncompleted tasks provides a large reduction in distance travelled.

Kohout and Erol argue that Fischer, Müller and Pischel's generation of an initial allocation is poor and therefore global reallocation will produce much better results than local replanning [12]. In their analysis they study problems where multiple items can be transported together and additional jobs are announced sequentially. When a new job is announced each vehicle bids for the job according to the cost of completing the additional job relative to their existing commitments. To avoid problems where inserting additional tasks has large impacts on the completion time of other tasks, upon each task insertion, already scheduled tasks are permitted to be reallocated to other vehicles. In their empirical analysis they compare this approach to Solomon's insertion heuristic which is a popular operations research based approach [21]. Overall they show that their distributed approach is statistically equivalent to the centralised operations research approach.

In a similar vein, Mes, van der Heijden and van Harten compare distributed auctions in MAS to hierarchical operations research approaches in dynamic environments [14]. In this work tasks are arrive sequentially and trucks can only carry one task at a time. Each truck bid calculation for a task considers the time required to do the job and any waiting time between jobs before and after. During execution trucks can also swap future task commitments between each other to improve the overall solution. In the comparison to the operations research approaches the distributed auction approach performs substantially better in highly dynamic environments.

4 Repeated SSC Auctions with Dynamic Tasks

We now formally explain SSC auctions and provide a simple extension to handle dynamically appearing tasks. SSC auctions assign fixed clusters of tasks to robots over multiple bidding rounds. In Fig. 2 an algorithm describing this process is

function SSC-Auction (U,K_{r_i}, r_i, R)
Input: U: the set of clusters to be assigned
K_{r_i}: the set of clusters presently assigned to robot r_i
r_i: the robot
R: the set of robots
Output: K_{r_i}: the set of clusters assigned to the robot

1: **while** ($U \neq \emptyset$)
2: /* *Bidding Stage* */
3: $\beta_{min} \leftarrow \infty$
4: **for each** cluster $C \in U$
5: $\beta_{r_i}^C \leftarrow$CalcBid(K_{r_i},C);
6: $\beta_{min} \leftarrow \min(\beta_{min}, \beta_{r_i}^C)$;
7: Send(β_{min}, R) | $B \leftarrow \bigcup_i$Receive($\beta_{r_i}^C$, R);
8: /* *Winner-Determination Stage* */
9: $(r', C) \leftarrow \arg\min_{(r' \in R, C \in U)} B$;
10: **if** $r_i = r'$ **then**
11: $K_{r_i} \leftarrow K_{r_i} \cup C$;
12: $U \leftarrow U \backslash C$;

Fig. 2: Algorithm for Sequential Single-Cluster Auctions [8].

given. During the bidding stage (Lines 2-7) the robot calculates bids for every unassigned task cluster and submits its single lowest bid for any one cluster to all other robots. Each bid is a triple $\beta = \langle r_i, C_j, b_\lambda \rangle$ of a robot r_i, a task cluster C_j and a bid cost b_λ. Each bid calculation requires robots to provide a solution to the travelling salesperson problem taking into account the tasks they already have allocated and the tasks in the cluster for which they are calculating a bid. Because this problem is NP-hard, robots may use the cheapest-insertion heuristic to provide a close approximation to the optimal solution. At the conclusion of each bidding round (Lines 8-12), one previously unassigned task cluster $C = \{t_1, \ldots, t_o\}$ is assigned to the robot that bids the least for it so that the overall team cost increases the least. After all task clusters $K = \{C_1, \ldots, C_k\}$ are allocated, each robot seeks minimise the distance travelled to complete all its allocated tasks. To achieve this, robots do not have to do all tasks in a cluster sequentially. When a robot is awarded a new cluster, the robot adds the tasks in this new cluster to its existing task assignment and replans its path to travel.

Before the SSC auction algorithm begins each task is assigned to one, and only one cluster, and clusters can be of varying sizes. All robots are informed of all tasks and all clusters before calculating bids. For the initial task allocation, either a centralised task manager or a single robot generates task clusters and subsequentally informs all robots of the details of each cluster. During repeated auctions each robot individually forms clusters of its uncompleted tasks. After removing any clusters containing tasks that are currently being executed, each robot informs all other robots of their available clusters for auctioning (a for-

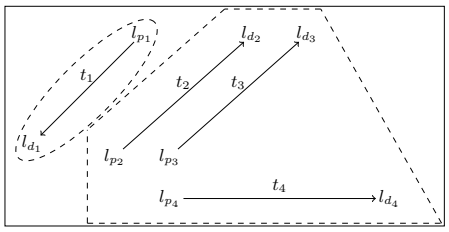

Fig. 3: Four tasks clustered into two pickup task clusters.

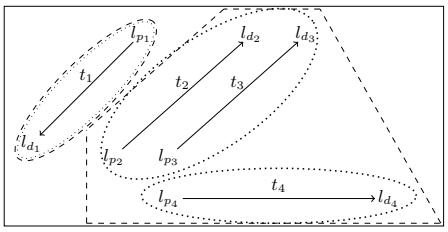

Fig. 4: Formation of three final task clusters based on delivery locations.

malisation of this algorithm is given in [8]). While the auction for uncompleted task clusters runs, all robots in parallel, continue to complete their currently executing tasks.

To handle a new task t_n dynamically inserted into our system we must ensure that the valid complete solution to the task allocation problem $\cup_{r_i \in R} T_{r_i} = T$ continues to hold for $\cup_{r_i \in R} T \cup \{t_n\}_{r_i} = T \cup \{t_n\}$. The simplest way to meet this requirement is upon the dynamic insertion of a new task instantaneously assigning it to a robot $r_i \in R$. Depending upon the operating enviornment configuration, after the assignment the robot can either locally replan its path or can signal a repeated auction to globally reallocate and replan tasks across all robots. In our experiments a new task may be inserted immediately after a task delivery and is initially allocated to the robot that completed the delivery.

The formation of clusters of tasks that have both pickup and delivery locations is much more difficult than cluster formation in single task location problems. To generate low cost solutions to MRTA problems with pickup and delivery we need a clustering algorithm that considers the structure of tasks in the problem. In the development of our clustering approach, we considered three different approaches for generating a metric that describes each task. First, we attempted to cluster based on the midpoint of the line segment between the pickup and delivery location. However, this metric lacks any information about where either end point is located. Second, we attempted to cluster by forming vectors of pickup and delivery locations and chaining tasks that have a delivery location close to the pickup location of another task. This approach to clustering closely matches the bidding pattern of robots in auctions, and as a result, there would be no advantage over doing no clustering at all.

Our third, and successful, approach was to cluster the pickup and delivery locations of all tasks separately. First, we form clusters of tasks based on pickup locations (Fig. 3). Second, within each pickup location cluster we cluster again into smaller clusters based on delivery locations (Fig. 4). The complete set of all small clusters is then used by robots in bidding. We formulate this algorithm in Fig. 5. We begin with no final clusters set (Line 1). We initially cluster all tasks based on pickup location (Line 3). We then iterate over each of these pickup location clusters (Line 5). Within each pickup location cluster we cluster again

function TwoStepClustering (T, k_p, k_d)
Input: T: the set of tasks to be clustered
k_p: the number of pickup clusters to be formed
k_d: the number of delivery clusters to be formed per pickup cluster
Output: K: the set of clusters for auction

1: $K = \emptyset$;
2: /* *Pickup location clustering* */
3: $PickupClusters \leftarrow$ CalcPickupClusters(T,k_p);
4: /* *Delivery location clustering* */
5: **for each** pickup cluster $p \in PickupClusters$
6: $DeliveryClusters \leftarrow$ CalcDeliveryClusters(p,k_d);
7: $K \leftarrow K \cup \{DeliveryClusters\}$;

Fig. 5: Clustering of tasks with pickup and delivery.

based on delivery location (Line 6). Finally we merge each set of delivery location clusters into the final set of clusters for auction (Line 7).

Both clustering functions CalcPickupClusters and CalcDeliveryClusters consider only one side of a task's location. This allows us to use any existing clustering algorithm that clusters based on point locations, e.g., K-means clustering or single-linkage clustering. In our algorithm we need two k values k_p and k_d; one for each clustering function. For our experiments we seek to find k overall clusters for auction such that $k_p * k_d = k$. This ensures that each pickup cluster is split into equal numbers of delivery clusters. However a problem can arise, due to each cluster containing a varying number of tasks, when considering the clustering of delivery locations inside a pickup cluster. If the number of tasks in the cluster $|p|$ is less than k_d we can only form $|p|$ clusters, and as a result, in total we will have less than k clusters. Without modifying the behaviour of the algorithm used to form the pickup clusters we cannot prevent this occuring. As a result, in these situations we end up with fewer clusters than we initially sought. To mitigate the effect of this during the formation of a large number of clusters we suggest a novel solution to gradually increase the value of k_d used in remaining cluster formations:

$$k_d = k_d + \frac{k_d - |p|}{|\text{remaining pickup clusters}|} \tag{5}$$

First we calculate the difference between the requested number of clusters k_d and the number of tasks in the cluster $|p|$. We then divide this by the number of remaining pickup clusters that have yet to have delivery clusters formed inside them. We add this value to k_d and continue to the next pickup cluster for clustering.

5 Empirical Experiments

We test SSC auctions with dynamic tasks requiring pickup and delivery in a simulated 510x510 grid-based office-like environment with 16 rooms. Each room contains four doors that can be independently opened or closed to allow or restrict travel between rooms (Fig. 1). This environment has become the standard testbed in recent literature [11, 25, 16, 6, 8]. In each experiment, the doors between different rooms and the hallway are either open or closed. We test on 25 randomly generated configurations of opened and closed doors with each robot starting in a different random location. Robots can only travel between rooms through open doors and they cannot open or close doors. However, it is guaranteed there is at least one path between each room and every other room. For each configuration we test with 10 robots and 60 tasks. We use single-linkage clustering with a true path distance metric (previously discussed in [7]) for our clustering algorithm. For the initial allocation we form $k = \frac{1}{2}|T_{\text{known}}|$ clusters and where $T_{\text{known}} \in T$ is the set of known tasks at the start of the initial allocation. For repeated auctions each robot individually forms $k = \frac{1}{2}|T_{r_i}|$ clusters.

In each experiment configuration a robot may be randomly assigned a new task upon completion of a task delivery, we then compare local replanning versus global reallocation. In local replanning, when a robot is assigned a new task, the robot replans its path to complete all uncompleted tasks. In global reallocation, when a robot is assigned a new task it signals to all other robots to begin an auction of all uncompleted tasks across all robots. We compare three ratios of dynamic to static tasks, 25%, 50%, and 75% unknown at the start. We also compare our results to a baseline one-off task allocation with all tasks known. Finally, we compare the effects of different robot task capacities of 1, 3, and 5.

The mean results for the MiniMax team objective are presented in Table 1 and for the MiniSum team objective in Table 2. In considering capacity constraints, unsurprisingly, the absence of constraints produces the lowest team costs and the more restrictive the constraints the higher the cost. This directly leads to the largest reduction in team costs occurring in scenarios with highly restrictive constraints. In both team objectives, when robots are restricted to a capacity of only executing one task at a time, the global reallocation of tasks produces better results than the baseline of all tasks known. While this may initially come as a surprise it has been experimentally shown before in SSC auctions with static tasks [8] and SSI auctions with dynamic tasks [19]. The key explanation for this is that during a one-off task allocation, due to the greedy nature of SSI auctions, each robot can reach a local minima in its bidding preferences whereas in repeated auctions this is avoided. Overall, across both team objectives, global reallocation generally produced lower overall results than local replanning.

Across the MiniMax team objective results, the average advantage of global reallocation over local replanning ranges from 11.8% to 36.5%. For local replanning the best results occur when 50% of tasks were unknown. We speculate that in situations where 25% of tasks are unknown, despite being "better informed" of other tasks during planning, there may be instances where new tasks are in-

Table 1: Mean MiniMax Team Objective Results (percentage improvement of reallocation compared to replanning in brackets).

		Local Replanning		
Capacity	All Known	25% Unknown	50% Unknown	75% Unknown
1	7857	8228	7276	7275
3	3985	5430	4976	6256
5	3070	4620	4471	6250
∞	2602	4438	4418	6250
		Global Reallocation		
Capacity	All Known	25% Unknown	50% Unknown	75% Unknown
1	7857	5228 (36.5%)	5593 (23.1%)	5911 (18.7%)
3	3985	3800 (30.0%)	4389 (11.8%)	4877 (22.0%)
5	3070	3469 (24.9%)	3800 (15.0%)	5000 (20.0%)
∞	2602	3415 (23.1%)	3810 (13.8%)	5165 (17.4%)

Table 2: Mean MiniSum Team Objective Results.

		Local Replanning		
Capacity	All Known	25% Unknown	50% Unknown	75% Unknown
1	41539	41527	42025	45305
3	21245	26922	28316	33238
5	16519	22554	25720	30096
∞	10150	16613	19675	27985
		Global Reallocation		
Capacity	All Known	25% Unknown	50% Unknown	75% Unknown
1	41539	37449 (9.8%)	40278 (4.2%)	36907 (18.5%)
3	21245	25775 (4.3%)	28303 (0.0%)	26594 (20.0%)
5	16519	22639 (-0.4%)	24692 (4.0%)	22238 (26.1%)
∞	10150	17813 (-7.2%)	22207 (-13%)	20332 (27.3%)

serted late into plan execution which cause robots to travel greater distances. This hypothesis is further supported by the large improvement gains shown by global reallocation in the 25% unknown experiments. Across the global replanning results the best results occur when only 25% of tasks are unknown. We speculate that in these instances the advantage of being "better informed" of other tasks helps with the formation of new clusters and repeated auctions of tasks. Overall, in the worse case of 75% tasks unknown, on average, a robot travels a maximum of twice the distance of the baseline result.

The MiniSum team objective results show a much smaller benefit in global reallocation over local replanning. These differences range from a 13% increase to a 27% decrease in total distance travelled. Local replanning produces the best results when only 25% of tasks are unknown at the start. Contrarily in global reallocation, the largest improvements overall local replanning come in the highly dynamic environment of 75% of tasks being dynamically inserted. We speculate that the reason for this is due to the high numbers of repeated cluster formations exposing many different inter-task synergies during each repeated

Table 3: Mean Initial Task Allocation Computation Time (s) For MiniMax Team Objective.

Capacity	All Known	25% Unknown	50% Unknown	75% Unknown
1	268	145	62	16
3	265	145	61	16
5	264	144	61	16
∞	242	137	60	16

Table 4: Mean Overall Cumulative Task Allocation Computation Time (s) For MiniMax Team Objective.

		Local Replanning		
Capacity	All Known	25% Unknown	50% Unknown	75% Unknown
1	268	173	90	41
3	265	172	89	42
5	264	171	88	42
∞	242	157	85	41
		Global Reallocation		
Capacity	All Known	25% Unknown	50% Unknown	75% Unknown
1	268	347	276	119
3	265	245	175	106
5	264	226	176	105
∞	242	206	174	105

auction. Additionally, in scenarios with no capacity constraints local replanning outperforms global reallocation. Again, in the worst-case scenario the maximum distance result was twice that of the corresponding baseline result.

In addition to measuring the distances robots travel, we also consider the amount of computational real-time robots require to generate an initial allocation of tasks, and then the accumulated time required to repeatedly replan or reallocate tasks. The experimental timing results are from a system with a 2.8GHz Intel Core i7 CPU, 8GB RAM, running Ubuntu 11.04 x64.

Table 3 shows the mean time required to cluster and auction an initial allocation of tasks for the MiniMax team objective. The results for the MiniSum team objective are not shown, however, they are near identical. Unsurprisingly, the fewer the number of tasks known, the faster the initial allocation of tasks. In the most dynamic situation the generation of an initial allocation is 15 times quicker than the baseline. This is an important result because the quicker an initial allocation is generated, the sooner robots can begin executing tasks. We also note that the task capacity constraint has minimal influence upon the time required to generate the initial allocations.

The overall cumulative time required for robots to replan or reallocate upon the insertion of all dynamic tasks into the system is given in Table 4. In the worst-case, global reallocation of tasks is three times slower than local replanning. Of particular interest is that in all dynamic situations, local replanning was

substantially quicker in computational time than the baseline. Also in highly dynamic situations global reallocation generated solutions faster than the baseline.

Overall, taking into consideration mean distance and computational time results we can conclude that when robots seek to achieve a MiniMax team objective it is best for robots to work together and globally reallocate tasks. However, when robots seek to achieve a MiniSum team objective, except in highly dynamic environments, the small improvement offered by global reallocation is offset by much higher computational times and in many situations would be of little benefit.

6 Conclusions and Future Work

In this paper we have built upon previous work on SSC auctions and demonstrated their effectiveness in task allocation with pickup and delivery. Our empirical analysis considered the trade-off in performance between local replanning and global reallocation for dynamic task allocation. Our key result shows that global reallocation generally produces lower team costs than local replanning. However, to achieve this there is a large computational time cost.

In the consideration of future work, an ongoing challenge in cluster formation is determining suitable values for k. There are a number of more complex clustering algorithms that do not require a pre-set number of clusters to form, such as DBSCAN [4], and their usefulness in SSC auctions could be considered. Another aspect of clustering to consider is the number of items in each cluster. At present we impose no limit on the number of items, or any preference to allocating large clusters first. If a fixed maximum limit on the number of items in each cluster was imposed it could produce more even clusters which may affect the task allocation results.

References

1. Berhault, M., Huang, H., Keskinocak, P., Koenig, S., Elmaghraby, W., Griffin, P.M., Kleywegt, A.J.: Robot exploration with combinatorial auctions. In: IROS. pp. 1957–1962 (2003)
2. Dias, M.B., Zlot, R., Kalra, N., Stentz, A.: Market-based multirobot coordination: A survey and analysis. Proceedings of the IEEE 94(7), 1257–1270 (2006)
3. Dias, M., Stentz, A.: Opportunistic optimization for market-based multirobot control. In: Intelligent Robots and Systems. vol. 3, pp. 2714–2720. IEEE (2002)
4. Ester, M., Kriegel, H.P., Sander, J., Xu, X.: A density-based algorithm for discovering clusters in large spatial databases with noise. In: KDD. pp. 226–231 (1996)
5. Fischer, K.: Cooperative transportation scheduling: An application domain for dai. Applied Artificial Intelligence 10(1), 1–34 (1996)
6. Heap, B., Pagnucco, M.: Sequential single-cluster auctions for robot task allocation. In: Australasian Conference on Artificial Intelligence. pp. 412–421 (2011)
7. Heap, B., Pagnucco, M.: Analysis of cluster formation techniques for multi-robot task allocation using sequential single-cluster auctions. In: Australasian Conference on Artificial Intelligence. pp. 839–850 (2012)

8. Heap, B., Pagnucco, M.: Repeated sequential auctions with dynamic task clusters. In: AAAI (2012)
9. Koenig, S., Keskinocak, P., Tovey, C.A.: Progress on agent coordination with cooperative auctions. In: AAAI (2010)
10. Koenig, S., Tovey, C.A., Lagoudakis, M.G., Markakis, E., Kempe, D., Keskinocak, P., Kleywegt, A.J., Meyerson, A., Jain, S.: The power of sequential single-item auctions for agent coordination. In: AAAI. pp. 1625–1629 (2006)
11. Koenig, S., Tovey, C.A., Zheng, X., Sungur, I.: Sequential bundle-bid single-sale auction algorithms for decentralized control. In: IJCAI. pp. 1359–1365 (2007)
12. Kohout, R.C., Erol, K.: In-time agent-based vehicle routing with a stochastic improvement heuristic. In: AAAI. pp. 864–869 (1999)
13. Lagoudakis, M.G., Markakis, E., Kempe, D., Keskinocak, P., Kleywegt, A.J., Koenig, S., Tovey, C.A., Meyerson, A., Jain, S.: Auction-based multi-robot routing. In: Robotics: Science and Systems. pp. 343–350 (2005)
14. Mes, M., van der Heijden, M., van Harten, A.: Comparison of agent-based scheduling to look-ahead heuristics for real-time transportation problems. European Journal of Operational Research 181(1), 59–75 (2007)
15. Nanjanath, M., Gini, M.L.: Dynamic task allocation for robots via auctions. In: ICRA. pp. 2781–2786 (2006)
16. Nanjanath, M., Gini, M.L.: Repeated auctions for robust task execution by a robot team. Robotics and Autonomous Systems 58(7), 900–909 (2010)
17. Sandholm, T.: Contract types for satisficing task allocation: I theoretical results. In: Proceedings of the AAAI spring symposium: Satisficing models. pp. 68–75 (1998)
18. Sariel, S., Balch, T.R.: Efficient bids on task allocation for multi-robot exploration. In: FLAIRS Conference. pp. 116–121 (2006)
19. Schoenig, A., Pagnucco, M.: Evaluating sequential single-item auctions for dynamic task allocation. In: Australasian Conference on Artificial Intelligence. pp. 506–515 (2010)
20. Smith, R.G.: The contract net protocol: High-level communication and control in a distributed problem solver. IEEE Trans. Computers 29(12), 1104–1113 (1980)
21. Solomon, M.M.: Algorithms for the vehicle routing and scheduling problems with time window constraints. Operations research 35(2), 254–265 (1987)
22. Tovey, C., Lagoudakis, M., Jain, S., Koenig, S.: The generation of bidding rules for auction-based robot coordination. Multi-Robot Systems. III, 3–14 (2005)
23. Viguria, A., Maza, I., Ollero, A.: Set: An algorithm for distributed multirobot task allocation with dynamic negotiation based on task subsets. In: ICRA. pp. 3339–3344 (2007)
24. Zhang, K., Jr., E.G.C., Shi, D.: Centralized and distributed task allocation in multi-robot teams via a stochastic clustering auction. TAAS 7(2), 21 (2012)
25. Zheng, X., Koenig, S.: K-swaps: Cooperative negotiation for solving task-allocation problems. In: IJCAI. pp. 373–379 (2009)
26. Zlot, R., Stentz, A.: Complex task allocation for multiple robots. In: ICRA. pp. 1515–1522 (2005)
27. Zlot, R., Stentz, A., Dias, M.B., Thayer, S.: Multi-robot exploration controlled by a market economy. In: ICRA. pp. 3016–3023 (2002)