Priority Basis Task Allocation for Drone Swarms

Kolitha Warnakulasooriya, Ryan G Benton, Aviv Segev

School of Computing, University of South Alabama, 150 Student Services Drive, Mobile, Alabama 36688 USA
hkw2021@jagmail.southalabama.edu, {rbenton,segev}@southalabama.edu

Abstract

Drones are a special category of robots that can be categorized under unmanned aerial vehicles. A swarm of drones is a collection of drones working interactively, and each drone has a specific task to accomplish. Some drone swarm applications are expected not only to complete the tasks but also to persist in the priority of the task execution order. Existing task allocation algorithms that are developed for drone swarms mostly focus on efficient task-sharing rather than maintaining task execution order. We describe a method to share a set of given prioritized tasks among drone robots in a drone swarm based on an auction-based algorithm called contract net protocol. We ensure drones acting on the field endeavor to follow the given priority and ensure our method is capable of utilizing the resources of drones to get the maximum outcome. We demonstrate our results using a multi-agent simulation platform and compare them with the recently developed task prioritization approach. Our approach outperforms leading drone swarm algorithms.

Introduction

"Drone" is a common term in society for identifying autonomous or remote-controlled flying machines which can be generalized under the unmanned aerial vehicles (UAV) category (D'Andrea 2014). Nowadays, drones are usually used for personal purposes such as photography and cinematography (Ayodeji et al. 2016), commercial purposes such as crop spraying and planting in agriculture and farming (Klauser and Pauschinger 2021), and military purposes like surveillance (Semsch et al. 2009), attacks, and combat field operations (Haulman and Daniel 2003). Speeding up mission completion, more robustness, improving the quality of the solution, and optimizing the power sources are advantages of drone swarm applications rather than single drone applications (Alkouz and Bouguettaya 2021).

(Bahgeci and Sahin 2005; Brambilla et al. 2013) define a swarm of robots as a large number of relatively simple physically embodied agents that can be designed such that a desired collective behavior emerges from the local interactions among agents and between the agents and the environment. A swarm of drones can be considered a swarm of robots according to the above definition. Each drone in the swarm has been assigned to complete a small specific task, and a drone swarm can be controlled either semi-autonomously or fully autonomously (Tahir et al. 2019).

In the future, drones will be assigned the delivery job. Then it will be fast and cheap. Now, since researchers and developers focus on delivering multiple items using a single drone, task allocation algorithms are important (Vazhavelil and Sonowal 2021). Usually, task-allocation algorithms are focused on the completion of tasks rather than following the order of execution, and hence the order of task execution is random and unknown (Rizk, Awad, and Tunstel 2019). Nonetheless special scenarios need to prioritize the execution order of tasks. As an example, when a medical supply is delivered, some emergency supplies need to be delivered quickly.

Mission A and mission B are two tasks in two different locations. Location L_A is the target location of mission A, and location L_B is the target location of mission B. The impact of mission A is less than the impact of mission B, although the distance to L_B is greater than the distance to L_A. According to the impact, the drone controller decides that mission A has higher priority than mission B; hence drones should give their priority to accomplishing mission B after accomplishing mission A. Even so, there is a lack of prioritized task-sharing algorithms for drones in the literature. In this paper, we propose a new approach called Multi-Robot Priority Task Allocation (MRPTA) by improving the current CNET protocol algorithm to maintain the task execution order.

The rest of the paper will be organized as follows. In the next section we discuss the related work for this approach. Then we present our task distribution approach and the next section describes algorithmic implementations. Then we present our experiments and the next section displays our findings. Finally we focus on our conclusions.

Related Work

The multi-robot task allocation problem is a key research problem in multi-robot applications, allocating the most suitable task to the best robot out of a set of robots (Liu et al. 2020). There are multiple algorithms developed to allocate tasks within the robot agents. Consensus-based bundle algorithm (CBBA) and Contract Net Protocol (CNET) are the most common auction-based algorithms. Simultaneously meta-heuristic based approaches are major types of task allocation algorithms (Liu et al. 2020; Notomista et al. 2019). All the above developed algorithms are motivated to answer multi-robot task allocation problems and consider allocating tasks to every robot in the multi-robot application rather than maintaining execution priority. According to (Notomista et al. 2019; Roldan, Cerro, and Barrientos 2018; Emam et al. 2020; Hoeing et al. 2007; Gerkey and Matarić 2004; Garapati et al. 2017), researchers maintain the priority of task execution within the robot scope after tasks are all allocated. They focused on minimizing the energy consumption and the time constraint of robots by prioritizing tasks by the robot itself. There was no consideration of the user expected task execution order. Therefore, there was no consideration of the actual expectancy of the user. We used modified contract net protocol (Zhen et al. 2021; Lemaire, Alami, and Lacroix 2004) as the basis of our approach. A drone swarm is a distributed system. Thus, we use multi-agent based protocol to share tasks among robots. Since the drones' capabilities are heterogeneous, mission tasks in the field can only be accomplished by a drone that is capable of accomplishing it.

Contract Net Protocol (CNET) was introduced by (Smith 1980) as a task-sharing protocol for distributed problem solvers, and it eases cooperative task-sharing through effective communication among the processor nodes. Smith emphasized a key issue in a decentralized distribution system, finding the suitable idle processor node to execute the task, which is called a connection problem. Smith resolved the connection problem by introducing an algorithmic solution. Two major aspects are covered in this protocol such that distributing a balanced computational load to every node in the net is called resource allocation and selecting appropriate nodes to execute tasks is called the focus.

Contract net protocol is mentioned as an auction-based algorithm because task sharing happens through a bidding process. Contract Net Protocol defines two different roles of nodes. One role is the manager role, and the other one is the contractor role. There are small processing units in the contract net called nodes, and every node in the net should belong to one of these roles. A manager is responsible for monitoring the execution of tasks on the net and is also responsible for executing tasks assigned to it. A contractor is responsible for the execution of the tasks. Mutual task selection and allocation is done through an end-to-end negotiation discussion between the manager and every other contractor on the net. Most of the contract net applications nodes are capable of acting as either a manager or a contractor or both roles, and roles are assigned randomly among nodes (Lemaire, Alami, and Lacroix 2004; Ota 2006).

Contract net protocol is a negotiation based protocol which has four basic stages (Liang and Kang 2016). When the manager receives the expected task list, it starts announcing the task list to other nodes on the net. This stage is called the task announcement stage. Tasks are introduced to other contractor nodes via a specific template called the task template. The task template contains detailed information about the task. This template is understandable by every node on the net. Then these announcements are ranked according to the time that the task was created. This order depends on the created time of the announcement message. A contractor assesses the tasks according to the cost of task completion and decides which task can be fulfilled with the existing resources, which is called task bidding. After the task bidding process, every contractor has their bids. Then contractors communicate back to the manager with the message containing the bids. This stage is called the winning bids stage. The managers evaluate the bids and award contracts to the most suitable contractor (Smith 1980). This last stage is called the signing contracts stage. (Sandholm 1993) implemented the contract net protocol based on the marginal cost calculation of the delivery routing project. Marginal cost is the cost of delivery of a task. Winning bids depend on the maximum price of the announcement and the delivery cost. If the maximum price of the announcement is higher than the delivery cost, then the bid is considered as won.

(Zhen et al. 2021) improved the control net protocol, applied it to a heterogeneous drone swarm, and focused more on cooperative task allocation of a swarm of UAVs. (Zhen et al. 2021) modified the contract net protocol which is more efficiently assigning heterogeneous targets to a military drone swarm. Their modification is based on the time sequences of mission completion and balancing load rates of allocation. These modifications assure that no robot is overloaded by tasks nor stays in idle stage without assigning a single task. The algorithm of (Zhen et al. 2021) is based on situation assessment which has three aspects. Distance index describes the assessment of distance along the line between task location and current drone location. Angle index describes the angular-distance between task location and current drone location, and ability index follows the inherent ability of the drone to accomplish the mission.

(Lemaire, Alami, and Lacroix 2004) also introduced a new parameter called the equity factor to perform a load-balanced task distribution strategy. Equity factor (Eq) resolves two major problems in multi UAV applications. The equity factor resolves the Multiple Traveling Salesman Problem (MTSP) which is an extension of the Traveling Salesman Problem (TSP). MTSP is based on a case of a salesman traveling through multiple cities and the main achievement is that the salesman should visit each city only one time with a minimum total travel cost. Next, the equity factor resolves the extension of MTSP which is a constraint on task execution times. The equity factor evaluates the plan of tasks with respect to other robots among the drone robots in a multi UAV application like a drone swarm. The approach of (Lemaire, Alami, and Lacroix 2004) helps to evaluate task distribution based on the workload and the utility costs of the robots. Time constraints and the costs of traveling to the target are examples for the utility costs of robots.

(Das et al. 2015) came up with an algorithm called Consensus Based Parallel Auction and Execution (CBPAE) for sharing tasks among multiple robots based on CBBA algorithm according to the given priority. This approach may need continuous cooperation and communication between many robots, and thus this approach is not suitable for drone robots. (Notomista et al. 2019) came up with a way of optimizing the task sharing to the multiple robots by prioritizing tasks within the robot. They try to prioritize the tasks according to a convenient way of minimizing the energy consumption within the scope of a local robot; hence this doesn't support maintaining the whole mission order. (Gürel, Adar, and Parlaktuna 2013) also introduced a method to distribute tasks according to the given priorities for a team of mobile ground robots. We identify this approach as the Gürel approach further in this paper. The Gürel approach is tested on mobile ground robots and allocates tasks according to the distance between robots.

Our approach discusses an effective priority-based task allocation for drone swarms based on the contract net protocol (Smith 1980; Sandholm 1993; Lemaire, Alami, and Lacroix 2004). We distribute tasks using the contract net protocol and we improve the current contract net protocol by performing the task sharing according to the priority of the task.

Task Distribution

Contract net protocol (Smith 1980) has four basic main phases. In our case, we represent the phases as below. The first phase is the task announcement phase. Then the estimate calculation phase is the same as the bidding phase. The third phase is the announcement estimations phase, which is similar to the winning bids phase, and finally the signing contract phase. The first phase manager drone starts announcing tasks to the other contract drones. Then contract drones start to calculate estimations of each task and send back the estimations to the manager drone. Then the manager node starts signing the contracts and allocating tasks to the appropriate drones according to the priorities of the tasks.

Task Representation

Consider n number of different tasks to accomplish in domain ψ denoted as $T = \{T_1, T_2, ..., T_n\}$. Each task has its own rank or priority which is assigned by the controller. $T_i \in T$ task has assigned w rank which is represented as w_{T_i}, $0 \leq w_{T_i} \leq 1$ which is called rank factor.

$$\sum_{i=1}^{n} w_{T_i} = 1 \tag{1}$$

Task Announcement

Task announcement addresses all drone nodes in the contract net. This paper assumes that only one manager drone exists and others act as contractors within the net. This manager node advertises the ordered task list to others through a specific template. This template contains the target location and type of the task. This paper also assumes that the trajectories of the drone are drown on the two dimensional plane due to the simulation purposes. In the rest of the paper we define the mission for a drone as completing a task.

$$T_i = (x_i, y_i, z_i) \tag{2}$$

T_i is the task representation vector. (x_i, y_i) is the coordination of i-th task $T_i \in T, 1 \leq i \leq n$, the destination where the assigned drone should fly. z_i represents the type of task that can be fulfilled by the assigned drone.

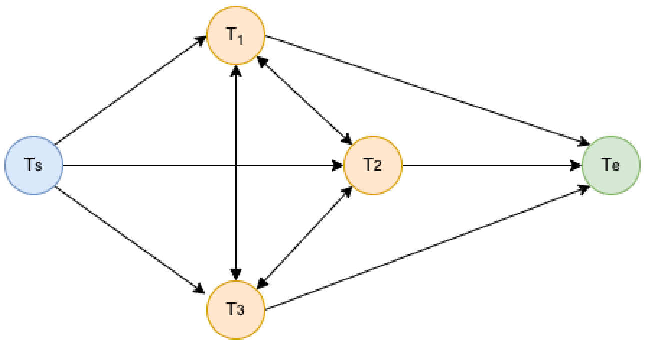

Figure 1: Task representation in graph structure n = 3

Calculate Estimates

Once the task announcement phase is completed, drones have a list of tasks to perform. Then drones assess the capability of fulfilling each task by a single drone. This assessment is named a situation assessment. Drones start to calculate situation assessments. Situation assessment is a metric for measuring the ability to complete a given task. There are three aspects considered to describe situation assessments such as the distance index, the ability index, and the angle index (Zhen et al. 2021). We only consider distance index and ability index, and we can remove angle index because the rotation of a drone around the yaw axis at the same location does not affect the situation assessment (Sandholm 1993). Situation assessment calculation will result from the capability of the mission success, time constraint to complete the mission, start location, end location, and power consumption as the basic outcomes. A robot should calculate situation assessments for every task from every possible location that the robot could start the mission.

The target field can be represented as a directed complete graph $G = (V, E)$. V represents a set of all tasks, and E represents the set of situation assessments which are calculated from one task to a certain V task. T_s and T_e are the drone starting and ending tasks respectively relevant to the drone robot. The contractor drone robot receives a list of tasks T from the manager drone and the contractor drone adds start and end tasks T_s and T_e to generate the correct vertices list. Then the new set of vertices is $V = \{T_s, T_1, T_2, ..., T_n, T_e\}$. The situation assessment between two tasks is denoted by A_{T_i,T_j} where $T_i, T_j \in T$ or $T_i = T_s$ or $T_j = T_e$. Situation assessment can be calculated with Equation 3. In this paper we assume the drone needs to visit the task that is mandatory to complete the mission successfully. Starting and end locations are T_s and T_e respectively. The robot calculates the situation assessments of every T_i task and publishes the graph as an adjacency matrix called the situation assessment matrix $R^{l \times r}$ where l is the number of tasks including start and end nodes and r is the number of tasks without including start and end nodes. Figure 1 shows an example graph representation of a task field which $n = 3$.

$$f : A_{T_i,T_j} \longrightarrow (T_i, T_j); \tag{3}$$

Signing Contracts

Consider m number of different robots sharing these n tasks. The set of the drone robots is denoted as $D = \{d_1, d_2, ..., d_m\}$. D_k represents the k-th drone robot where $D_k \in D, 1 \leq k \leq m$

$$D_k = (x_k, y_k, z_k, v_k, \theta_k) \tag{4}$$

(x_k, y_k) represents the current coordinates of the drone, and z_k describes the type of tasks that this specific k-th drone can perform. v_k and θ_k represent flying velocity and the heading angle respectively. Next, the manager drone selects the next prioritized task from ordered task set $T_o = \{T_i \in T | w_{T_i} > w_{T_{i-1}}\}$. Then the manager drone selects the task which has the lowest utility cost among the idle drones. Utility cost UC_{T_i,T_j} is a single numeric value representing the A_{T_i,T_j} that can be calculated by Equation 5 from the situation assessment. F denotes the utility function for converting situation assessments into utility costs.

$$UC_{T_i,T_j} = F(A_{T_i,T_j}) \tag{5}$$

Idle robots mean the set of drone robots has completed the given mission or not yet started. After the selected robot, the total utility cost UCT of the robot is calculated as Equation 6. The number of tasks assigned to a certain drone robot conditioned on the total utility cost UCT is lower bounded or equals to the maximum utility cost UC_{max} of the robot according to Equation 7.

$$UCT = UC_{T_s,T_1} + \sum_{i=1}^{n-1} UC_{T_i,T_{i+1}} + UC_{T_n,T_e} \tag{6}$$

$$UCT \leq UC_{max} \tag{7}$$

U_{T_s,T_1} and U_{T_n,T_e} represent the utility costs from the initial location to the first task and the situation assessment for the return to the end location from the location of the last task respectively.

Proposed Algorithm

Algorithm 1 describes the task-sharing algorithm. In the first stage, the queue of tasks, the list of drones, and the list of situation assessment matrices are passed as input parameters. The queue of tasks should be filled in descending order according to the assigned ranks of the tasks. When we get the first element, we will receive the highest priority task. Then the next prioritized task is taken out of the queue of tasks. Then we perform an iterative check on the drones list to see whether there are any idle drone robots that can be found. If any idle drone robot is found, we find the related situational assessment value from the situational assessments list A and we add the situational assessment to another list called LA. Then we sort the LA list in descending order according to the utility cost value of every situational assessment.

Then we iterate through the LA list from the situational assessment which has the lowest utility cost to the highest utility cost. In each iteration, we find the related drone robot $D_{T_{sel}}$ for the selected situational assessment $A_{T_{sel}}$. Then we calculate the total utility cost D_{UCT} of the drone robot $D_{T_{sel}}$. The total utility cost UCT is the sum of the selected situational assessment utility cost $A_{T_{sel}}$ and the selected drone's total utility cost D_{UCT}. If UCT is less than or equal to the $D_{T_{sel}}$ drone's maximum utility cost $D_{UC_{max}}$, we assign the task T_i to the $D_{T_{sel}}$ drone.

Algorithm 1: Task sharing algorithm for manager

```
1:  T = Queue of tasks which needs to be completed by descending order
2:  D = List of drones
3:  A = List of situation assessment matrices
4:  while $Tqueueisnotempty$ do
5:      $T_i \leftarrow T.dequeue()$ {Get next prioritized task}
6:      $LA = empty()$
7:      for $k = 1$ to $D.size$ do
8:          $D_k \leftarrow D[k]$
9:          if $D_k.isIdle$ then
10:             $SA_{T_i,T_j} \leftarrow A[T_i.location][D_k.location]$
11:             $LA.add(SA_{T_i,T_j})$
12:         end if
13:     end for
        {Sort the list according to the utility cost}
14:     $LA_{ordered} = sort(c \leftarrow LA[c].utility\_cost)$
15:     for $j = 1$ to $LA_{ordered}.size$ do
16:         $A_{T_{sel}} \leftarrow LA_{ordered}[j]$
17:         $D_{T_{sel}} \leftarrow D.get(A_{T_{sel}})$
18:         $D_{UCT} \leftarrow D_{T_{sel}}.UCT$
19:         $D_{UC_{max}} \leftarrow D_{T_{sel}}.UC_{max}$
20:         $UCT \leftarrow A_{T_{sel}}.UCT + D_{UCT}$
21:         if $UCT \leq D_{UC_{max}}$ then
22:             $D_{T_{sel}}.add\_and\_assign(T_i)$
23:             $break$
24:         end if
25:     end for
26: end while
```

Experiments

Experiments Setup

This paper uses MASON (Luke 2019) multi-agent simulation library on top of the JAVA (Oracle 2014) virtual machine. We set the simulation field as 400×400 pixels. The number of simulations that run for each scenario depends on the number of target task points covering the whole field. We assure that more than 90% of the field area is covered from the randomly generated tasks for every scenario. We run the number of search iterations to find the unassigned task locations in every simulation. When the coverage is more than 90%, the number of search iterations grows exponentially and hence we assume that almost all the fields are covered. We run the simulations as the number of tasks over the number of drones. We define a trial case that contains a number of trial iterations. A trial iteration is one time of simulation run. We decide minimum and maximum boundaries for the drone robot count and the task count for every trial case. We select a random number of drone robots and a random number of tasks within the trial range.

Number of Tasks	Number of Drone Robots							
	[5-10]		[15-20]		[25-30]		[35-40]	
	Gürel	MRPTA	Gürel	MRPTA	Gürel	MRPTA	Gürel	MRPTA
[10-20]	TC = 8952	TC = 8961						
	K = 0.488	K = 0.629						
	T = 1423.82	T = 797.8						
	MAD = 6.08	MAD = 0.5						
[20-30]	TC = 5291	TC = 5312	TC = 5297	TC = 5297				
	K = 0.451	K = 0.459	K = 0.428	K = 0.806				
	T = 2066.81	T = 1050.35	T = 2092.32	T = 730.61				
	MAD = 8.87	MAD = 0.5	MAD = 3.35	MAD = 0.5				
[30-40]	TC = 3761	TC = 3776	TC = 3766	TC = 3762	TC = 3764	TC = 3760		
	K = 0.422	K = 0.371	K = 0.405	K = 0.68	K = 0.394	K = 0.872		
	T = 2628.68	T = 1340.98	T = 2648.32	T = 801.65	T = 2677.87	T = 723.37		
	MAD = 12.1	MAD = 0.5	MAD = 4.5	MAD = 0.5	MAD = 2.87	MAD = 0.5		
[40-50]	TC = 2918	TC = 2914	TC = 2917	TC = 2918	TC = 2911	TC = 2911	TC = 2909	TC = 2917
	K = 0.409	K = 0.322	K = 0.392	K = 0.585	K = 0.379	K = 0.791	K = 0.368	K = 0.899
	T = 3258.63	T = 1615.48	T = 3225.47	T = 915.11	T = 3183.03	T = 773.09	T = 3160.06	T = 720.76
	MAD = 13.6	MAD = 0.5	MAD = 5.7	MAD = 0.49	MAD = 3.6	MAD = 0.5	MAD = 2.64	MAD = 0.48
[50-60]	TC = 2382	TC = 2384	TC = 2384	TC = 2383	TC = 2383	TC = 2386	TC = 2383	TC = 2383
	K = 0.398	K = 0.286	K = 0.379	K = 0.517	K = 0.368	K = 0.705	K = 0.356	K = 0.835
	T = 3864.35	T = 1877.28	T = 3788.39	T = 1026.84	T = 3900.8	T = 825.66	T = 3815.31	T = 752.16
	MAD = 15.2	MAD = 0.5	MAD = 6.81	MAD = 0.5	MAD = 4.34	MAD = 0.46	MAD = 3.18	MAD = 0.5
[60-70]	TC = 2016	TC = 2016	TC = 2013	TC = 2015	TC = 2013	TC = 2015	TC = 2012	TC = 2019
	K = 0.392	K = 0.261	K = 0.373	K = 0.47	K = 0.362	K = 0.651	K = 0.349	K = 0.77
	T = 4322.58	T = 2208.06	T = 4412.38	T = 1141.86	T = 4434.47	T = 857.77	T = 4412.69	T = 790.19
	MAD = 18.6	MAD = 0.5	MAD = 8.33	MAD = 0.5	MAD = 5.07	MAD = 0.5	MAD = 3.17	MAD = 0.49
[70-80]	TC = 1745	TC = 1746	TC = 1744	TC = 1745	TC = 1743	TC = 1742	TC = 1741	TC = 1746
	K = 0.379	K = 0.24	K = 0.365	K = 0.432	K = 0.348	K = 0.59	K = 0.339	K = 0.722
	T = 4958.94	T = 2481.28	T = 4935.5	T = 1274.32	T = 4894.3	T = 961.49	T = 4956.14	T = 827.47
	MAD = 20	MAD = 0.5	MAD = 9.12	MAD = 0.5	MAD = 5.81	MAD = 0.5	MAD = 4.21	MAD = 0.38
[80-90]	TC = 1539	TC = 1537	TC = 1537	TC = 1541	TC = 1539	TC = 1538	TC = 1536	TC = 1536
	K = 0.376	K = 0.227	K = 0.352	K = 0.405	K = 0.349	K = 0.549	K = 0.34	K = 0.682
	T = 5521.99	T = 2758.9	T = 5695.66	T = 1363.3	T = 5536.08	T = 1036.69	T = 5559.7	T = 857.07
	MAD = 23.4	MAD = 0.5	MAD = 9.93	MAD = 0.5	MAD = 6.47	MAD = 0.49	MAD = 4.79	MAD = 0.49
[90-100]	TC = 1376	TC = 1375	TC = 1374	TC = 1376	TC = 1375	TC = 1376	TC = 1374	TC = 1375
	K = 0.37	K = 0.214	K = 0.371	K = 0.381	K = 0.344	K = 0.522	K = 0.327	K = 0.646
	T = 6461.12	T = 3039.57	T = 6099.42	T = 1475.78	T = 6240.51	T = 1092.36	T = 6001.95	T = 904.23
	MAD = 25	MAD = 0.5	MAD = 11.3	MAD = 0.5	MAD = 7.12	MAD = 0.5	MAD = 5.17	MAD = 0.5

Table 1: Number of trials ran for different number of targets with respect to the different number of drone robots

In this experiment, we obtained 4 different results. First is the number of trial iterations denoted by TC. The second one is the Kendall's Tau correlation value which evaluates the difference between actual and expected lists of execution orders indicated by K. Third, T mentions the mean execution time for a simulation trial run. Finally, we measure how the set of tasks is divided among drone robots. We mapped assigned tasks against the drone robot of each trial and calculated the mean absolute deviation, which measures the average distance between data points and the mean called variability. MAD shows the maximum mean absolute deviation value which we observe within the set of trials.

Kendall's Tau

We use modified Kendall's Tau method (Pusala et al. 2017) for comparing the expected ranks order with the actual ranks order of tasks. Kendall's distance is a metric for evaluating the rank correlation between two ranked sets. (Fagin, Kumar, and Sivakumar 2003) proposed a method to measure Kendall's distance between two lists by pair-wise disagree-

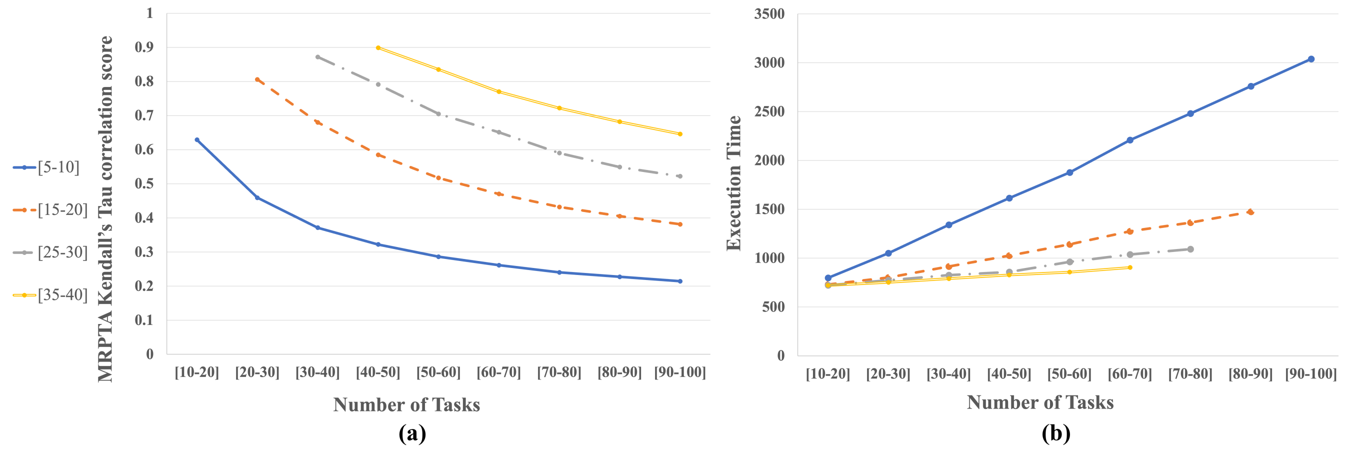

Figure 2: (a) Kendall's Tau correlation score over a different number of tasks for a different number of drone robots. (b) Execution time over a different number of tasks for a different number of drone robots

ment penalty according to the order of the items in the lists. This method considers the top k number of items in a list, although in our case we assumed k is the size of the whole list. In our case we have two lists: expected task list τ_e and actual task list τ_a. $P(\tau_e, \tau_a)$ denotes item pair from τ_e and τ_a lists. We consider the first two cases of their method. The pair of items of $(i, j) \in P(\tau_e, \tau_a)$ and Kendall's distance is $\overline{K}^{(p)}_{i,j}$,

- Case 1: Item i and j are in both the τ_e and τ_a lists respectively. If item i is ahead of item j in both the τ_e and τ_a lists $\overline{K}^{(p)}_{i,j} = 0$. If item i is ahead of item j in the list τ_e and item j is ahead of item i in τ_a $\overline{K}^{(p)}_{i,j} = 1$.
- Case 2: Both items i and j are in the τ_e although only item i is in the τ_a. If item i in τ_a is ahead of τ_e then $\overline{K}^{(p)}_{i,j} = 0$ otherwise $\overline{K}^{(p)}_{i,j} = 1$.

In our case, the user may have a facility to duplicate the rankings in the list of expected tasks.

Results

Table 1 shows the simulation results of our experiment. Our results show three major aspects. The first one is the equivalence between the order of executed tasks and the expected order of execution tasks. We compare our results with the Gürel approach experiment results. The second one is how long has the swarm taken to complete the mission. Finally, we test whether our algorithm efficiently divided the tasks among robots. Both experiments were executed in the same simulation environment with the same computational conditions. We here considered that the drone robot count is less than or equal to the target tasks count. Task locations were decided randomly and the ranks also assigned to the tasks were random. MRPTA column shows results for our experiment, and Gürel column shows results for simulation experiments of Gürel approach. Every cell shows the experiment result values for a specific drone robot's range, task range, and method.

Figure 2(a) demonstrates how Kendall's Tau correlation score changes with respect to the number of tasks and the number of drone robots. We set a number of tasks as the abscissa and Kendall's Tau correlation score as ordinate. As a result, we observe that Kendall's Tau correlation score is proportional to the number of drone robots and negative exponential to the number of tasks. When the number of drone robots is getting close to the number of tasks, Kendall's Tau correlation score has a high value which means that the expected list and the actual list are not in the same order. Figure 2(b) illustrates how the execution time changes with respect to the number of tasks and the number of drone robots. It shows that execution time is proportional to the number of tasks. When the number of drone robots is too low compared to the number of tasks, execution time appears as an unexpectedly high value. Comparatively, when the number of drone robots increases, the difference between the execution time strictly decreases.

We compared our results with the Gürel approach. In our experiment, the optimum number of drone robots was 15-20. We choose the 15-20 range for our comparison here. Figure 3(a) illustrates how the Kendall's Tau correlation score varies between the two methods. We observe that the Gürel approach has a lower and more stable Kendall's Tau correlation score compared to our method, thus practically implying that the Gürel approach more accurately maintains the expected execution order of tasks than our method. When we increase the number of tasks, the Kendall's Tau correlation score shows a decline pattern in our method. The difference between Kendall's Tau correlation scores of our method and the Gürel approach has an inverse correlation with the number of tasks. Regardless of the number of tasks, Kendall's Tau correlation score of the Gürel approach is invariable. Figure 3(b) shows how execution time varies between the two methods. As the experiment results show, the execution time of the Gürel approach is much higher than our method at every trial. Furthermore, when we increase the task count, execution time drastically increases with a higher slope com-

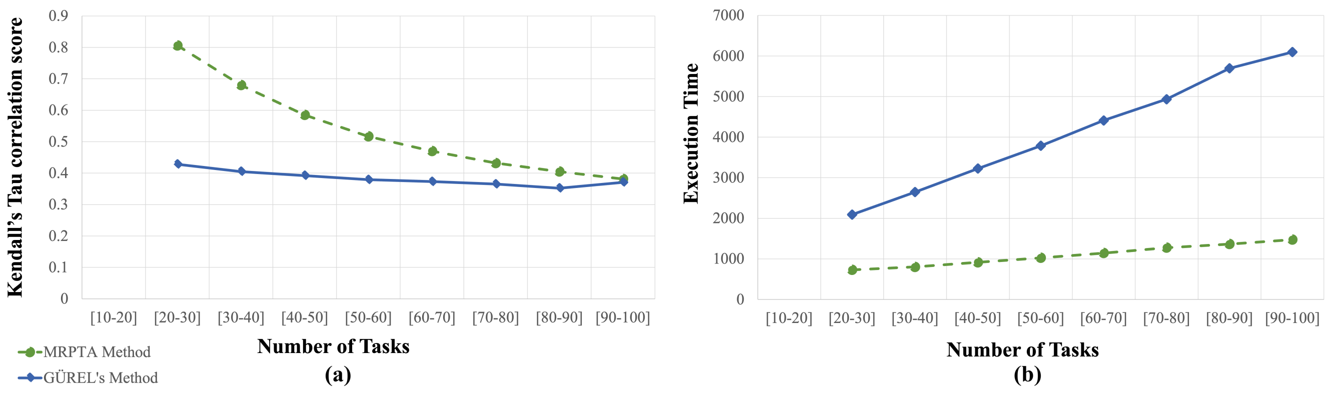

Figure 3: (a) Kendall's Tau correlation score over the Gürel approach and MRPTA method for a different number of tasks with a different number of drone robots. (b) Execution time over the Gürel approach and MRPTA method for different number of tasks with a different number of drone robots

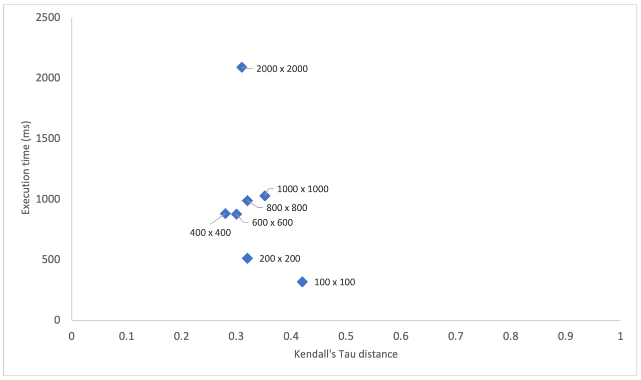

Figure 4: Kendall's Tau correlation score and execution time with respect to the field size

pared to our method. Our method is more capable of keeping the stability of the execution time over the task count. Almost all the drone robots are assigned at least one task. We evaluate how tasks were shared among the drones. We map the robot and number of assigned tasks and calculated MAD using the robot-task count map for every trial case. Then we filter out the maximum MAD value and mean value for the respective MAD value for every robot-task combination. When the difference between MAD and mean values gets closer to zero, it shows that tasks are not uniformly distributed among drone robots. The Gürel approach always results in lower values than our method, thus showing us that our method is better at distributing tasks uniformly among drone robots. Hence, we show that our method tries to share tasks optimally and equally among drone robots.

We increase the field size and observe the difference in Kendall's Tau correlation score as shown in Figure 4. For this, we select 15-20 drone robots with 80-90 tasks. We realize that increasing the field size does not have a sufficient effect on the Kendall's Tau correlation score though we see a slight decline in the Kendall's Tau correlation score. Although we are enlarging the field size, according to our experiment Kendall's Tau correlation score does not show a consistent pattern. The field size does not show an adequate correlation with the Kendall's Tau correlation score. We observe a correlation between execution time and field size also reflected in Figure 4. Targets are spread widely in larger fields, and hence traversal time also is high. Therefore the execution time increases according to the field size.

Conclusion

In this paper, we propose a novel approach for the priority-based task allocation strategy for drone swarms based on contract net protocol. Our goal was to develop an algorithm for drone swarms. Then the swarm of drones performs task execution and collectively follows the user's expected order. We evaluated this method on the multi-agent simulation platform, and ranks were assigned to the tasks according to the given priorities. We evaluated how the drone swarm maintains the user expected execution order of tasks, which changes drone counts over to the task count. We realized that this approach is suitable for drone swarm applications that have considerably higher numbers of targets to achieve with the limited number of drones. However, missions consume a considerable amount of time, which is a drawback to performance. This approach is cost-effective because the amount of resources is considerably low. This approach is not beneficial for time-critical and large drone swarm applications. Additionally, we implemented this method in real-world drone swarm applications and evaluated our simulation results in the real world in the future. Proposed future works include developing and improving this algorithm to perform the dynamic task allocation problem and improving this for time-critical applications

References

Alkouz, B.; and Bouguettaya, A. 2021. Provider-centric Allocation of Drone Swarm Services. In *2021 IEEE International Conference on Web Services (ICWS)*, 230–239.

Ayodeji, A., Opeyemi; Stephen, P., D.; Glyn, T., T.; and Peter, S. 2016. The Multimodal Edge of Human Aerobotic Interaction. In *International Conferences Interfaces and Human Computer Interaction, Game and Entertainment Technologies and Computer Graphics, Visualization, Computer Vision and Image Processing*, 243–248.

Bahgeci, E.; and Sahin, E. 2005. Evolving aggregation behaviors for swarm robotic systems: a systematic case study. In *Proceedings 2005 IEEE Swarm Intelligence Symposium, 2005. SIS 2005.*, 333–340.

Brambilla, M.; Ferrante, E.; Birattari, M.; and Dorigo, M. 2013. Swarm robotics: a review from the swarm engineering perspective. *Swarm Intelligence*, 7(1): 1–41.

D'Andrea, R. 2014. Guest Editorial Can Drones Deliver? *IEEE Transactions on Automation Science and Engineering*, 11(3): 647–648.

Das, G. P.; McGinnity, T. M.; Coleman, S. A.; and Behera, L. 2015. A distributed task allocation algorithm for a multi-robot system in healthcare facilities. *Journal of Intelligent & Robotic Systems*, 80(1): 33–58.

Emam, Y.; Mayya, S.; Notomista, G.; Bohannon, A.; and Egerstedt, M. 2020. Adaptive Task Allocation for Heterogeneous Multi-Robot Teams with Evolving and Unknown Robot Capabilities. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE.

Fagin, R.; Kumar, R.; and Sivakumar, D. 2003. Comparing Top k Lists. In *Proceedings of the Fourteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '03, 28–36. USA: Society for Industrial and Applied Mathematics. ISBN 0898715385.

Garapati, K.; Roldán, J. J.; Garzón, M.; del Cerro, J.; and Barrientos, A. 2017. A Game of Drones: Game Theoretic Approaches for Multi-robot Task Allocation in Security Missions. In *ROBOT 2017: Third Iberian Robotics Conference*, 855–866. Springer International Publishing.

Gerkey, B. P.; and Matarić, M. J. 2004. A Formal Analysis and Taxonomy of Task Allocation in Multi-Robot Systems. *The International Journal of Robotics Research*, 23(9): 939–954.

Gürel, U.; Adar, N.; and Parlaktuna, O. 2013. Priority-based task allocation in auction-based applications. In *2013 IEEE INISTA*. IEEE.

Haulman; and Daniel, L. 2003. U.S. Unmanned Aerial Vehicles in Combat, 1991-2003. https://apps.dtic.mil/sti/citations/ADA434033. Accessed: 2022-02-13.

Hoeing, M.; Dasgupta, P.; Petrov, P.; and O'Hara, S. 2007. Auction-based multi-robot task allocation in COMSTAR. In *Proceedings of the 6th international joint conference on Autonomous agents and multiagent systems - AAMAS '07*. ACM Press.

Klauser, F.; and Pauschinger, D. 2021. Entrepreneurs of the air: Sprayer drones as mediators of volumetric agriculture. *Journal of Rural Studies*, 84: 55–62.

Lemaire, T.; Alami, R.; and Lacroix, S. 2004. A distributed tasks allocation scheme in multi-UAV context. In *IEEE International Conference on Robotics and Automation, 2004. Proceedings. ICRA '04. 2004*. IEEE.

Liang, H.; and Kang, F. 2016. A novel task optimal allocation approach based on Contract Net Protocol for Agent-oriented UUV swarm system modeling. *Optik*, 127(8): 3928–3933.

Liu, R.; Seo, M.; Yan, B.; and Tsourdos, A. 2020. Decentralized task allocation for multiple UAVs with task execution uncertainties. In *2020 International Conference on Unmanned Aircraft Systems (ICUAS)*. IEEE.

Luke, S. 2019. Multiagent Simulation And the MASON Library. https://cs.gmu.edu/~eclab/projects/mason/. Accessed: 2022-02-12.

Notomista, G.; Mayya, S.; Hutchinson, S.; and Egerstedt, M. 2019. An Optimal Task Allocation Strategy for Heterogeneous Multi-Robot Systems. In *2019 18th European Control Conference (ECC)*. IEEE.

Oracle, C. 2014. JDK 8. https://openjdk.java.net/projects/jdk8/. Accessed: 2022-02-12.

Ota, J. 2006. Multi-agent robot systems as distributed autonomous systems. *Advanced Engineering Informatics*, 20(1): 59–70.

Pusala, M. K.; Benton, R. G.; Raghavan, V. V.; and Gottumukkala, R. N. 2017. Supervised approach to rank predicted links using interestingness measures. In *2017 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*. IEEE.

Rizk, Y.; Awad, M.; and Tunstel, E. W. 2019. Cooperative Heterogeneous Multi-Robot Systems: A Survey. *ACM Comput. Surv.*, 52(2).

Roldan, J. J.; Cerro, J. D.; and Barrientos, A. 2018. Should We Compete or Should We Cooperate? Applying Game Theory to Task Allocation in Drone Swarms. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE.

Sandholm, T. 1993. An Implementation of the Contract Net Protocol Based on Marginal Cost Calculations. In *Proceedings of the Eleventh National Conference on Artificial Intelligence*, AAAI'93, 256–262. AAAI Press. ISBN 0262510715.

Semsch, E.; Jakob, M.; Pavlicek, D.; and Pechoucek, M. 2009. Autonomous UAV Surveillance in Complex Urban Environments. In *2009 IEEE/WIC/ACM International Joint Conference on Web Intelligence and Intelligent Agent Technology*, volume 2, 82–85.

Smith. 1980. The Contract Net Protocol: High-Level Communication and Control in a Distributed Problem Solver. *IEEE Transactions on Computers*, C-29(12): 1104–1113.

Tahir, A.; Böling, J.; Haghbayan, M.-H.; Toivonen, H. T.; and Plosila, J. 2019. Swarms of Unmanned Aerial Vehicles — A Survey. *Journal of Industrial Information Integration*, 16: 100106.

Vazhavelil, T.; and Sonowal, A. 2021. The Future of Delivery with Drones: Contactless, Accurate, and High-Speed - Wipro.

Zhen, Z.; Wen, L.; Wang, B.; Hu, Z.; and Zhang, D. 2021. Improved contract network protocol algorithm based cooperative target allocation of heterogeneous UAV swarm. *Aerospace Science and Technology*, 119: 107054.