2020 17th International Conference on Ubiquitous Robots (UR)
June 22-26, 2020. Kyoto, Japan

Ant Colony Optimization based Multi-Robot Planner for Combined Task Allocation and Path Finding

Agha Ali Haider Qizilbash
Fraunhofer IPA
Stuttgart, Germany
qizilbash_ali@yahoo.com

Christian Henkel
University of Stuttgart, Germany
Stuttgart, Germany
post@henkelchristian.de

Sanaz Mostaghim
Otto-von-Guericke-University Magdeburg
Magdeburg, Germany
sanaz.mostaghim@ovgu.de

***Abstract*—Nature has inspired many solutions to the problems in computer science and recently in the field of robotics as well. Ant based algorithms have been successful in solving the NP hard problems such as traveling salesman problem. In the field of multi-robots it has been used to solve path finding and task allocation problems. In industrial warehouse applications, these problems are often combined, when for example multiple robots need to pick-up objects from one location and drop-off at the other. Multiple mobile robots need to perform these task optimally and simultaneously being on the floor without collisions. In this paper, we address this problem keeping the objective of being able to obtain collision free paths for all robots in a map, assigned for all given pick-up and drop-off tasks among themselves with an optimized minimal total distance traveled by the robots. We propose a multi-robot planner inspired from ant colony optimization to solve this combined problem. This planner finds collision free paths to all tasks to be done using a spread of ants from each robot. Ignoring the ones with collisions from other ants in their determined paths, the planner rates the tasks according to the total distance traveled. Using this rating system through multiple iterations, the planner eventually selects the best task allocations with paths for all robots among given iterations. This planner or as we call it, Ant Colony Optimization based Multi-Robot Planner for Combined Task Allocation and Path Finding Ant Colony Optimization based Multi-Robot Planner for Combined Task Allocation and Path Finding (ACTF) for pick-up and drop-off tasks is presented in this paper and has been tested against other similar planners producing promising results.**

***Index Terms*—multi-robot systems, ant colony optimization, combined task allocation and path finding, swarm robotics, multi-robot material handling, multi-robot intelligence**

I. Introduction

A. Motivation

The field of mobile robotics is advancing and it has brought new applications to the fields of swarm robotics and multi-robot/multi-agent optimization. With ongoing advancements, these fields have brought us a multi-copter Olympics logo display in the opening ceremony of 2018 winter Olympics in South Korea or the automated warehouse robots in Amazon. These not only need less human attention but also have a combined intelligence rather than being an isolated robot. The work from Tilley, [1] suggests that there is strong progress and market growth due to increase in multiple mobile robots, the coordination and their combined intelligence is of immense importance. Initially, the robots took their place on the conveyor belts in production lines, and then mobile robots made their way into transporting goods in warehouses. With the increasing use of multiple mobile robots into one place, challenges arise within the field of multiple robots coordination. Markus and Daniel [2] talk about how advanced robotics is taking over all types of industries. It also talks about how Automated Guided Vehicles (AGV) contribute towards the "The Factory of the Future" concept and will transform every field by 2025. Two of the major planners which were designed and being used for simulation of multi-robots for the Combined Task Allocation and Path Finding (CTAPF) at the company mentioned did not provide a good result and/or in the computationally good time. Therefore, we suggested another approach to solve this problem. This new planner algorithm is inspired by Ant Colony Optimization (ACO) and it's success in the individual problems of task allocation and path finding.

B. Goals of the Paper

One of the aims of this research is to advance the multi-robot coordination and intelligence in general. Another aim of this paper is to address the ever popular Pickup-and-Delivery Problem (PDP) applications for multi-robot. Goals of this paper is as follows:

Nature inspired Algorithm We are very much inspired by the outcomes obtained by simple rules that come from nature, such as unique multi-agent cohesion is achieved by simple rules of maintaining distance by every agent. A goal of this paper is to see if nature-inspired algorithm can be used to provide a solution for CTAPF.

Minimal Total Distance One of the most significant factor for any warehouse robots application is to reduce the total travel of all robots while achieving the desired tasks. In this paper our goal is to obtain a planner that gives an assignment and collision free paths while keeping the distance travelled by all robots to minimum.

978-1-7281-5715-3/20/$31.00 ©2020 IEEE

II. Background

A. Problem

Combined Task Allocation and Path Finding problem for Multi-Robots caters to the issue where an optimal task allocation for multi-robots would result in collisions in path and alternatively optimal path finding for multiple robots would not yield optimal task allocation. Therefore, we need to consider both the problems simultaneously. Some significant work in this field has been done as Multi-Robot Task Allocation (MRTA) with real time path planning as suggested by Woosley [3]. Where a set of mobile robots are introduced in a grid environment with knowledge of boundary and tasks. They avoid obstacles and other robots in their paths dynamically. Initially tasks are allocated as a simple Travelling Salesman Problem (TSP) and then using laser scan data generating obstacle data and detouring. Moving robots are avoided by also using the laser scan data and using simple approaches like stop, go and, in case of head-on collision, detour to avoid the collisions. A detailed research has been done on this by Zheng and Zhao [4], where they propose a centralized approach for CTAPF. In this approach tasks are assigned assuming all tasks to be done by one robot. Randomly segmentation of tasks schemes are done and peak segments are identified. Using optimization in peak segments, reallocation of tasks is done and final tasks schemes for all robots are obtained.

Conflict-Based Search (CBS) is a recent development in CTAPF. CBS is a two level search. low level search constructs paths for all agents whereas high level search finds conflicts and give solutions to it. Hoenig, along with Amazon Robotics, introduces an enhanced version of CBS [5] for CTAPF. This Enhanced Conflict-Based Search Algorithm (ECBS) use focal and open lists for a focal search within the low-level and high-level searches. It provides a stronger guarantee for an optimal result along with a lower cost comparative to the usual CBS. Another advance version of CBS is given by Henkel and Abbenseth [6] in which they introduce Task Conflict-Based Search Algorithm (TCBS). Initially TCBS ignores the collisions and only considers the optimal path finding for each agent for given task allocations. Then uses CBS approach of conflicts for collisions. In comparison to other approaches, it concludes a better solution quality when solving these problem combined.

Ant Colony Optimization was also previously used for a similar simultaneous problem by Kulatunga [7]. In this paper they directly equate an ant touring with a robot in the map and propose an approach for solving the simultaneous problem. First tasks are assigned randomly for every robot. Then, until all remaining tasks are assigned, probabilities of remaining tasks are calculated and robot picks up the best task among them. These probabilities are based on Simultaneous Path and Motion Planning (SiPaMoP) approach with distances. The term SiPaMoP is termed by Liu and Kulatunga [8], where they talk about rising number of constraints with rising number of autonomous vehicles in an environment. They presented an approach on SiPaMoP to find collision-free path plan to the respective assignments. This ACO approach on simultaneous problem, gives slight improved results in comparison with selected approaches and, therefore, the solution quality can not be judged fairly. For small size problems, this approach has been compared with exhaustive search mechanism, and computation time of this approach is compared with computation time of Simulated Annealing approach. Along with this targeted comparison success, this algorithm has some short comings. Such as in this algorithm all robots depict as ants and start from the same location (depot) which makes it not very useful with variable robot start points. The first task to all robots is always assigned at random which is one more fixed entity where algorithm can be certainly improved.

B. Ant Colony Optimization

Ant colony optimization, as the name suggests, has been derived from ants, particularly the foraging behavior of ants. Foraging behavior of animals is the searching of resources, in this case, food. As ants do not have a developed sense of sight, they cannot use it for navigation. Having only smell, touch and hear, ants use pheromone, a chemical substance which has strong scent, to follow path. They leave pheromone as they move in search of food and also as they return back home. The foraging behavior of ants is simply broken down into steps. Firstly, few ants set out in search of food. While finding they might be taking path decisions based of food smell or randomness which can also be seen in the first part of the figure 1, which comes from Wahab and Nefti-Meziani [9]. In this figure, F represents food and N represents nest, while a and b are directions to food and to nest respectively. Once they find their food; they will return back to the nest using the same path they took to the food owing to their memory. During their move they lay down pheromone to show the path they followed.

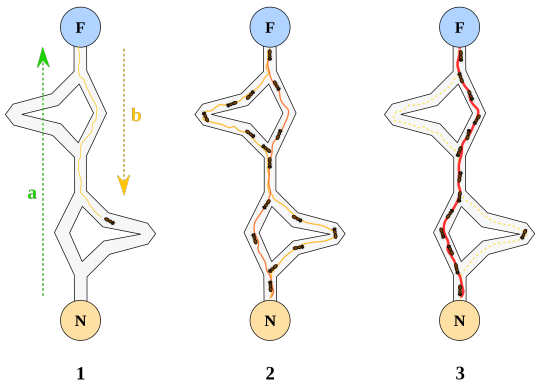

Fig. 1. Foraging Behaviour of Ants

Following the earlier ants, the next set of ants set out in search of food, which can also be seen in figure 1 in the second part. But these ants are less dependent on randomness and more attracted towards the pheromone smell, the more pheromone the higher chance that the ant follows that particular path. After getting to the food, these ants, too, return to the nest and during their move they too lay pheromone. As time

passes, the shorter and faster to travel path towards the food attains more and stronger pheromone, therefore, increasing the density of ants towards that path. Along this process the pheromone also evaporates or decays into the air with time depending on external air factors, which can also be seen in figure 1 in the third part. In other words, this results in an optimized path towards the food that is possibly shorter but definitely smoother and with fewer diversions than the other paths. This eventually also forces the ants to converge to one path and a single path is formed from the nest to the food source.

C. Other Planners

This paper addresses two other planners for the same problem which are:

- Greedy Planner is a local planner that assigns tasks to the robots using nearest neighbour search and considers only the closest tasks for assignments to robots. For path finding and collisions conflict it uses CBS. Major short coming for Greedy planner was that the robots were quickly assigned the nearest tasks.
- TCBS, as described by Henkel and Abbenseth [6], is the other planner which is also talked about in detail in II-A. This planner provides good optimal task allocations and also keeps in check the waiting cost effect of the tasks. The parameter called Waiting time of Task Completion (tW) is introduced and the comparisons will be made against it as well to keep the comparisons fair. The short comings of this planner are that it takes a lot of time due to its different level searches and conflict avoidance.

III. Implementation

A. Problem Structure

In this section, assumptions, premises and structure of the problem is defined. These problem structure and assumptions are commonly used in these types of problems [8] [6] [7]. They also make it possible to make evaluation and comparison with other planners.

- The workspace of mobile robots will be a known 2D environment with a grid map with static obstacles.
- One grid map size would be assumed as same size as one robot or one task. So that one robot or task can occupy one grid at one time.
- Loading and unloading time of a task by a robot is assumed as time taken by one unit distance travel.
- Each robot moves at an even speed in one grid size and stops at a sudden halt.
- Collisions of robots may be caused by any obstacle or any other robot.
- Each robot has 4 move-able directions in the grid.
- Each task has equal priority and all robots have equal priority over all tasks
- A task can be picked up from one location only and, if picked up, needs to be dropped to its specific drop location only.
- A robot can only cater to one task at one time. Similarly, a single task can be catered by one robot only.
- *First input:* Tasks, which should be in format of list of set of sets of pickup and drop x, y coordinates.
- *Second input:* Robots, or Agents, which should be in format of list of sets of x, y coordinates.
- *Third input (optional):* The distance function which should have 2 sets of pickup and drop x, y coordinates as inputs.

B. Planner Overview

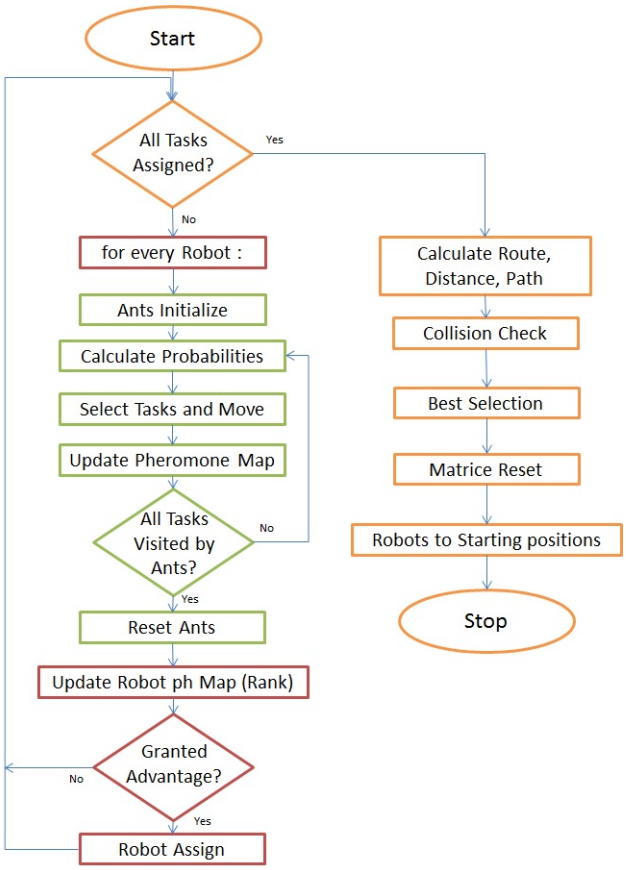

Fig. 2. ACTF Flowchart

The ACTF planner starts with the iteration loop, which is completely independent and for every robot it starts a loop in which ACO is used to rank the tasks for that particular robot. At the end of this very loop, a granted advantage is checked for, which is some relative threshold value and it decides whether the robot is assigned the most likely task. If the task is assigned then the task is removed from the list of available tasks. In any case, whether assigned or not, the same process happens for other robots and it continues back and forth for every robot multiple times until finally all tasks achieve a granted advantage to any robot which gets assigned to that robot. This can also be understood by the given pseudo algorithm as Algorithm 1. During this process, paths are also being checked for collisions and assignments of any two robots that produce a collision are removed from

consideration. The process is repeated over given number of iterations and then the best iteration in terms of Total Distance travelled by all robots is chosen to be returned by the planner.

A flowchart of one iteration of ACTF in figure 2 has been generated to follow up the flow of the planner. In the figure 2, orange color has been used for iteration processes, red color has been used for robot processes and finally green color have been used for ant processes. These color combination has been used to distinguish processes and also to easily follow the flow of the planner.

Algorithm 1: ACTF Planner

```
Main;
Input Data Processing Robot, Tasks, distance;
Robot Initialize Matrices;
for # of Iterations do
    while Unassigned Task do
        for every Robot do
            Ants Initialize;
            Ants Pick Paths and Traverse;
            Ants Route and Distance Update;
            Robot Pheromone Map Update;
            if Granted Advantage then
                Robot Assign Task;
            end
            Ants Reset;
        end
        Robot Update Positions;
    end
    Obtain Distance, Route, Path;
    Best Distance selection;
    Robot Starting Position Reset;
    Robot Matrices Reset;
end
Result: Optimized Total Distance, Route and Path
```

IV. Evaluation

A. Experimental Setup

The input parameters, which define the behaviour of the planner are:

- The Inputs: **Robots**, **Tasks**
- Algorithm Run Parameters: **# of Ants**, **# of Iteration**
- Ant Constants: **Alpha:**α**, Beta:**β
- Pheromone Update Constants: **Ph constant**, **Ph evap coefficient**

The output parameters on which above mentioned input parameters would be tested against would be:

- Total unit distance travelled by all robots **Total unit distance (Td)**
- Computation time **cT**
- Number of iterations with collisions out of 5 iterations **Coll.**
- Waiting time of task completion **tW**

In this section, chosen values or default values for the parameters mentioned above will be discussed and these will remain same in sections IV-B and IV-C until defined otherwise. For keeping it more easier for computation and evaluation, the map will be a 8 by 8 grip map with some obstacles. The default setup can also be seen in figure 6.

From our own experimentation, we have come up with some values for all the constants for these kinds of setup. The default values for algorithm run parameters will be 3 and 5 for number of ants and number of iterations, respectively. Now there is this big concern on having the number of ants as very less and this is due to the fact that in this planner, where the ants are created and spread from multiple positions (starting positions of robots) and each robot will have the a unique signature belonging to ants. So not only the number of ants will drastically increase computation time but also crowd the small map in consideration at the moment. Also, given that collisions occur with ants of different signatures will make small map a very unrealistic playground to practice a high number of ants or even a teens number of ants. Also, from our own experimentation of the planner, we decided for the values of $alpha$ and $beta$ constants. The result will be more biased towards pheromone and memory of the previous successful runs and will more likely to ignore new developments with the increase in value of α. Whereas, the result will become more biased towards smaller distance and new greedy selections with the increase in value of β. The default values for ant algorithm constants are kept as 0.5 and 1.2 for α and β respectively. The default values of pheromones update constants have kept to be 1000 and 0.4 for pheromone constant and pheromone evaporation coefficient respectively.

In the figure 6, the yellow circle represents the starting point of the robot, the green triangle represents the task pickup location the pointing arrow head points towards the drop off location. The result of this default setup with default values was obtained in 0.126 seconds and the optimized plan total distance will be 49 units, while the waiting cost of all the tasks summed will be 59 units. Task allocation is such that the first robot is assigned Task 4 and 3 respectively, while second robot is assigned task 1 and third robot is assigned to the task 2. Out of 5 iterations 3 iterations detected collisions and were obviously ignored for further processing. Mode method is used to obtain values and all the values you will see in figures in this section will be mode averages of set of ten experiments. This is mainly to keep integrity of a single run of the experiment. Fixed randomness or combining averages can ruin single run importance and hide some of the short comings of the planner or vice versa.

B. Results

1) Inputs Evaluation: In this section, ACTF planner will be evaluated with varying number and varying positions of robots and tasks. Analysis reveals that the computation time is largely dependant and number of robots and number of tasks than any other parameter. Therefore, comparison has been constructed Computation Time (cT) in the figure 3. It can be observed that the cT is widely consistent even with different positions for robots or tasks as long as the number remains

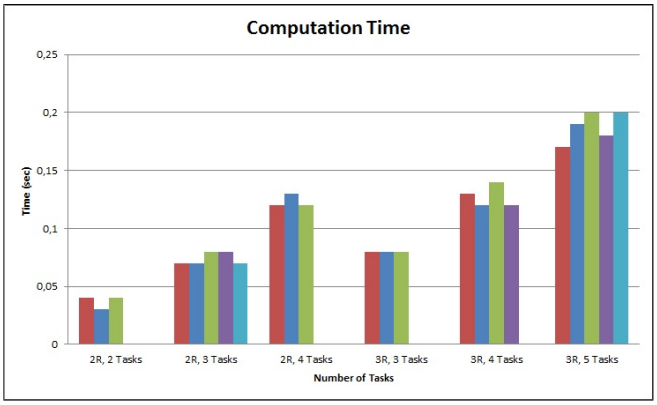

Fig. 3. Evaluation of Computation Time with increasing number of Tasks for 2 and 3 Robots

same. The increasing proportionality of cT to the number of tasks or number of robots can also be observed.

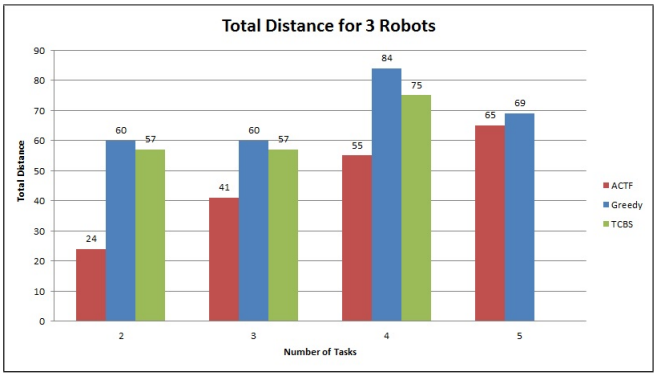

Fig. 4. Evaluation of Total unit distance for 3 Robots

This evaluation is also run against Greedy planner and TCBS to compare Td against increasing number of robots or tasks. In the figure 4, comparison of Td is done of all the three planners with respect to increasing number of tasks by using same 3 robots. It can be observed that TCBS is slightly better than greedy in the first three cases, whereas in the forth TCBS is unable to provide a result. ACTF simply dominates these set of experiments with a huge minimization of Td in the first case and a considerable minimization in the rest.

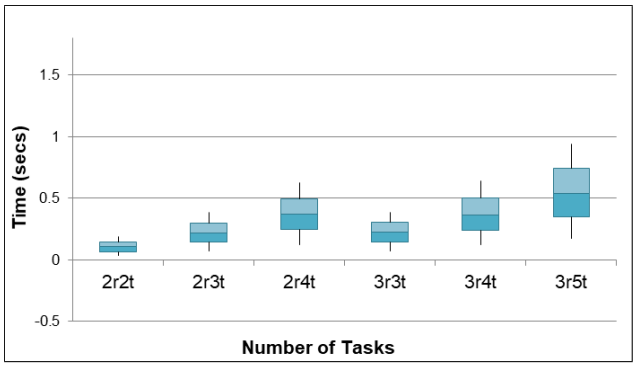

Fig. 5. Evaluation of Computation Time for Increasing Tasks

In figure 5, it can be observed that ACTF is not only better in computation time in comparison to TCBS, as was also discussed in section II-C, but also follows a linear increase in computation time. ACTF has taken under 1 second in all cases. Which also makes it very compatible for further decentralized applications where robots, due to communication with each other, will be in a known environment with other robots and every robot can easily process a CTAPF solution quickly.

C. Comparison

Using a greedy approach planner and a conflict based planner as TCBS which is introduced in [6], comparisons of a few cases have been done which is defined on wards:

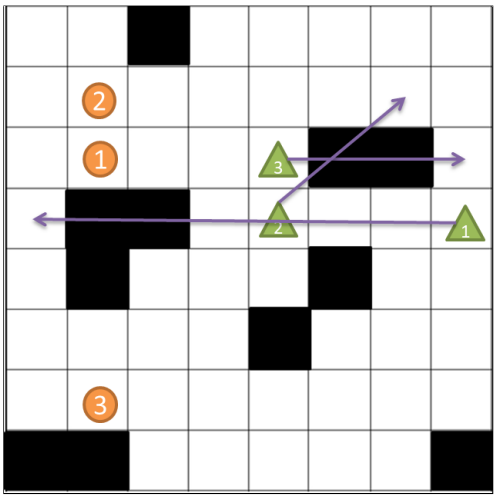

Fig. 6. Comparison Setup 1

Planner	Td	cT	tW	Task Assignments
Greedy	60	1.66	41	[3],[2],[1]
TCBS	60	8.19	40	[2],[3],[1]
ACTF	**39**	0.07	49	**[3, 2],[1],[]**

TABLE I
COMPARISON OF DIFFERENT PLANNERS FOR COMPARISON SETUP 1

1) Setup 1: In this setup, three robots have been introduced, which are at the default positions, and three tasks are placed out of default positions as seen in figure 6. The yellow circle represents the starting point of the robot, the green triangle represents the task pickup location the pointing arrow head points towards the drop off location. It can be observed from the table I that ACTF gives a good optimal result for Td for the setup in comparison to both other planners. The considerable efficiency in cT is also noteworthy. The most unique result you can see is that the assignments of Tasks are done with no regards to distributing the tasks among robots but preferring more on an optimal result driven task division. In this particular result the third robot according ACTF would not move and not assigned any task to save the unit distance in Td.

2) Setup 2: In this setup, three robots, and four tasks among the default positions, have been introduced where one

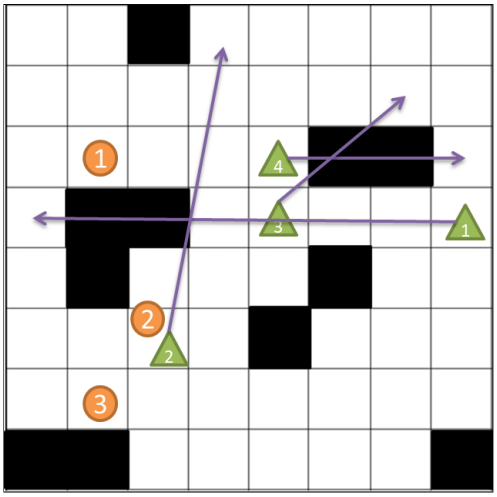

Fig. 7. Comparison Setup 2

Planner	Td	cT	tW	Task Assignments
Greedy	83	1.47	59	[3],[2, 1],[4]
TCBS	None	∞	None	None
ACTF	**49**	0.13	49	**[4, 3],[1],[2]**

TABLE II
COMPARISON OF DIFFERENT PLANNERS FOR COMPARISON SETUP 2

task and one robot starts at the same exact position, which can be seen in figure 7. As you can see in the table II that once again TCBS finds more conflicts that it can manage, providing no result. The assignment of greedy for the second robot starts with the second task which is at the same location and in first glance, would be more sensible thing to do. Whereas, ACTF gives the second task to the third robot and second robot is assigned with first task, meanwhile first robot is assigned forth and third task respectively, saving 34 unit distance in Td. This also stands to a unique solution which is achieved by using ACO to rank tasks for this problem.

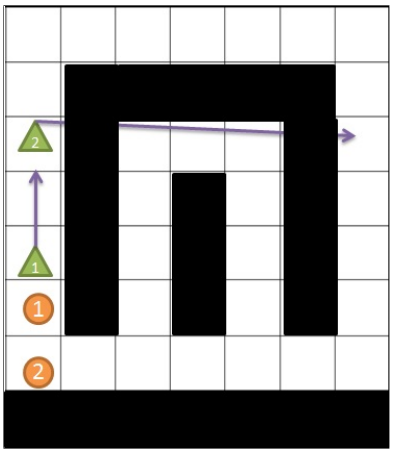

Fig. 8. Comparison Setup 3

3) Setup 3: In this setup, we have used a different map in which there are less options in the path taken to complete the task, as seen in figure 8. It can be observed from the table III that ACTF gives a good optimal result for Td for

Planner	Td	cT	tW	Task Assignments
Greedy	37	1.2	41	[1],[2]
TCBS	38	3.34	30	[2],[1]
ACTF	**27**	0.11	32	**[1, 2],[]**

TABLE III
COMPARISON OF DIFFERENT PLANNERS FOR COMPARISON SETUP 3

the setup in comparison to both other planners in this case as well. Greedy planner quickly assigns nearest tasks and, therefore, the first one is assigned first task and second robot is assigned second task respectively. On the other hand, to conserve some tW, TCBS assigns second task to first robot and first task to second robot respectively. Whereas, ACTF, looking at the advantage that both tasks lie on the same path, assigns both the tasks in respective order tot he first robot.

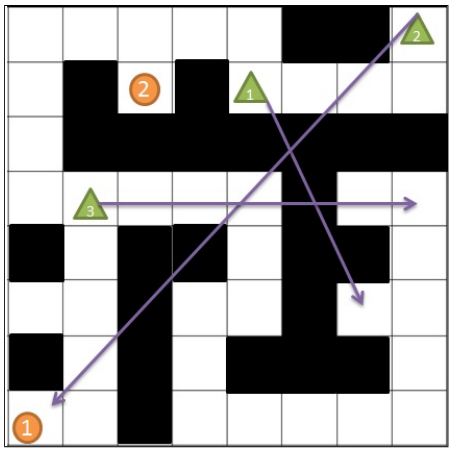

Fig. 9. Comparison Setup 4

Planner	Td	cT	tW	Task Assignments
Greedy	130	1.22	146	[3, 1],[2]
TCBS	125	25.3	130	[3],[2, 1]
ACTF	**101**	0.13	131	**[1],[2, 3]**

TABLE IV
COMPARISON OF DIFFERENT PLANNERS FOR COMPARISON SETUP 4

4) Setup 4: In this setup, we have used a different map in which there are less options in the path taken to complete the task, as seen in figure 9. It can be observed from the table IV that ACTF gives a good optimal result for Td for the setup in comparison to both other planners in this case as well. All three planners does assign second task, which has the longest drop off, to the second robot, which is an optimal assignment to do. But in addition, Greedy planner assigns both remaining tasks to the first robot, which costs it a lot of Td. On the other hand, TCBS assigns third task to first robot, whereas ACTF assigns first task to first robot, both due their optimization priorities. As a result, TCBS saves 1 unit in tW, whereas ACTF saves a considerable difference in Td.

D. Discussion

Empty Assignment: In the results of table I, where ACTF provides an empty assignment for the third robot for optimal Td. This is one of the most unique result of this paper, which

is achieved by using ACO for ranking the tasks and paths in this planner. This result is in accordance to the goals of the paper, mentioned in the section I-B, as we have succeeded to obtain unique and optimal assignment in CTAPF by using simple algorithms inspired by ants. Due this nature inspired algorithm, it completely cuts out the third robot out of results because of increasing distance on its runs. It can also be visually observed in figure 6 that third robot is visually far away from the tasks. Whereas, it is not a common practice and we see in the assignments column of table I, that every robot assignment to a task is prioritized by other planners.

Simple Ranking: As mentioned in goals of the paper in section I-B that we inspire the unique outcomes by employing simple rules inspired from nature. In this planner we used ACO to rank the tasks for every robot and this has led us to a very computation friendly planner. As we can see the cT in tables I, II, III and IV that ACTF provides the results in very short time. This is a successful result from our side. This is very useful if we further wish to employ a planner to every robot in a decentralized approach.

Waiting time of Task Completion: As we can see in the table I that the assignments of Greedy and TCBS are exactly the same along with all other parameters. ACTF here also gives a better result in terms of Total unit distance but the waiting cost of ACTF is higher than the other planners. This is due that currently ACTF does not cater to any optimization for the tW parameter. It can be said for this case that ACTF gives an optimal assignment of Td on the cost of tW.

V. Conclusion

In this paper, we present a new planner for Combined Task Allocation and Path Finding (CTAPF) for multiple robots for Pickup-and-Delivery Problem (PDP) applications, which is called Ant Colony Optimization based Multi-Robot Planner for Combined Task Allocation and Path Finding (ACTF). In this planner we use Ant Colony Optimization (ACO) to rate (or rank) the tasks for every robot and then only assign tasks using granted advantage of a task over a robot. This ranking is done over multiple iterations so to remove any initial randomness bias and to avoid initially undetected collisions. This planner produces very promising results for a minimum travel Total unit distance and Computation Time.

Minimum Travel Distance for Multi-Robots: A major goal for this paper was to obtain minimum possible travel Td for multiple robots for a given set of tasks in a known environment for known PDP tasks. This paper successfully reached this goal and provides promising results in comparison to two other planners which are used for the same problem.

A New Approach for CTAPF: Another goal of this paper was to obtain a different kind of planner for CTAPF, other than tasks selection planners and Conflict-Based Search (CBS) planners. This planner has totally different approach to solving CTAPF and uses nature inspired algorithm to solve the problem. This planner also succeeds in this goal as we use very simplistic flow of ants to rate the tasks and assign them to robots given a granted advantage. Eventually producing promising results and in less time than other planners.

VI. Future Work

Multi-Objective Optimization: One of the major future work which is quite evident from the paper is multi-objective optimization. This work revolved around the optimization of Total Distance travelled by the multi-robots Td, as do most optimization in the path finding problems. The other possibility in the same combined problem is the completion time of the tasks. Both these objectives are realistic in the warehouse and logistics scenarios where different tasks might have different priorities and, therefore, it would be useful to apply a multi-objective approach on the same combined problem to obtain optimal distance along with optimal task completion time with or without priorities.

Multi-Robot Simulation: Aside from theoretical improvement in this planner, simulation in an environment could be done which is already under progress. In this simulation environment, the map and robots are defined in Gazebo and Stage simulators. As ACTF has been adjusted to already existing framework at the company mentioned, it can easily be used in comparison to other planners and be used in multi-robot simulation with Gazebo and Stage simulator along with Robot Operating System (ROS).

References

[1] J. Tilley, "Multiple Mobile Robot Web." [Online]. Available: https://www.mckinsey.com/business-functions/operations/our-insights/automation-robotics-and-the-factory-of-the-future

[2] L. Daniel, Küpper; Markus, "Advanced Robotics in Japan," 2014. [Online]. Available: https://www.bcg.com/publications/2019/advanced-robotics-factory-future.aspx

[3] B. Woosley and P. Dasgupta, "Multirobot task allocation with real-time path planning," *FLAIRS 2013 - Proceedings of the 26th International Florida Artificial Intelligence Research Society Conference*, pp. 574–579, 2013.

[4] T. Zheng and X. Zhao, "Research on optimized multiple robots path planning and task allocation approach," *2006 IEEE International Conference on Robotics and Biomimetics, ROBIO 2006*, pp. 1408–1413, 2006.

[5] W. Honig, S. Kiesel, A. Tinka, J. W. Durham, and N. Ayanian, "Conflict-based search with optimal task assignment," in *Proceedings of the International Joint Conference on Autonomous Agents and Multiagent Systems, AAMAS*. International Foundation for Autonomous Agents and Multiagent Systems, 2018, vol. 1, pp. 757–765. [Online]. Available: www.ifaamas.org

[6] C. Henkel, J. Abbenseth, and M. Toussaint, "The Combined Task Allocation and Path Finding Problem," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, Macau, 2019.

[7] A. K. Kulatunga, D. K. Liu, G. Dissanayake, and S. B. Siyambalapitiya, "Ant colony optimization based simultaneous task allocation and path planning of autonomous vehicles," *2006 IEEE Conference on Cybernetics and Intelligent Systems*, 2006.

[8] D. K. Liu, X. Wu, A. K. Kulatunga, and G. Dissanayake, "Motion coordination of multiple autonomous vehicles in dynamic and strictly constrained environments," *2006 IEEE Conference on Cybernetics and Intelligent Systems*, 2006.

[9] M. N. Ab Wahab, S. Nefti-Meziani, and A. Atyabi, "A comprehensive review of swarm optimization algorithms," *PLoS ONE*, vol. 10, no. 5, 2015.