SEMARAK ILMU
PUBLISHING
202103268166(003316878-P)

Journal of Advanced Research in Applied Sciences and Engineering Technology

Journal homepage:
https://semarakilmu.com.my/journals/index.php/applied_sciences_eng_tech/index
ISSN: 2462-1943

Modification of the Ant Colony Optimization Algorithm for Solving Multi-Agent Task Allocation Problem in Agricultural Application

Medria Kusuma Dewi Hardhienata[1], Karlisa Priandana[1,*], Daffa Rangga Putra[1], Mamiek Sriatun[1], Wulandari[1], Agus Buono[1], Raihani Mohamed[2]

[1] Department of Computer Science, Faculty of Mathematics and Natural Sciences, IPB University, Bogor, Indonesia
[2] Intelligent Computing RG, Department of Computer Science, Faculty of Computer Science and Information Technology, Universiti Putra Malaysia, 43400 Serdang, Selangor, Malaysia

ARTICLE INFO

Article history:
Received 16 May 2023
Received in revised form 23 September 2023
Accepted 17 November 2023
Available online 26 November 2023

Keywords:
Ant Colony Optimization; Efficiency Factor; Multi-Agent Systems; Task Allocation; UAV

ABSTRACT

This paper considers the problem of task allocation where the goal is to find a coalition of UAVs (agents) to complete on-farm agricultural tasks. In this study, Ant Colony Optimization (ACO) algorithm is employed to find the best coalition of agents. The performance of the basic ACO algorithm for solving task allocation is improved by modifying the efficiency factor. In the proposed algorithm, the efficiency factor is defined as a function that relates not only to the capability of the agents and the distance between the agents, but also to the distance between the agents and the target. To solve the task allocation problem, the capability list of the agents was also adjusted using common UAV capabilities in agricultural application. Simulation results showed that the proposed ACO algorithm with the modified efficiency factor improved the performance of basic ACO algorithm for solving task allocation problem in terms of the average total travel cost for each agent. The optimum number of ants and agents in the proposed algorithm was also analysed for robust performance. Simulation results revealed that the addition of the numbers of agents and ants increases the average efficiency of the algorithm. In this study, we have also added a function to calculate the system capability utilization. By employing such a function, simulation results show that the total resource used by the agents and total communication cost can be optimized. In addition, a simple experiment using five ground robots with a centralized control was also carried out as a proof of concept for the proposed algorithm.

1. Introduction

Task allocation can be considered a problem of allocating a group of agents to tasks [1-4]. One real-world scenario where task allocation arises is the allocation of UAVs to perform on-farm agricultural tasks. In this case, each UAV in the multi-UAV system possesses different capabilities, and tasks can only be completed using specific combinations of these capabilities. One of the algorithms that can be used to address the task allocation problem with a group of agents is Ant Colony

* *Corresponding author.*
E-mail address: karlisa@apps.ipb.ac.id

https://doi.org/10.37934/araset.34.1.90105

Optimization (ACO) [3]. ACO was initially proposed by Marco Dorigo for combinatorial optimization problems [5,6]. In the context of task allocation, ACO has been applied in various cases, such as path planning for autonomous vehicles [7], assignment of patterns for spray paint robots [8], job allocation to processors [9], and multi-robot task allocation [3]. Particularly in Wang *et al.,* [3], the ACO algorithm demonstrated the ability to effectively handle task allocation problems in multi-robot systems due to its stability and efficiency.

Another relevant study that considered the problem of task allocation in a multi-agent system can be found in Lu *et al*., [10]. The study's objective in Lu *et al*., [10] is to find the best agent coalition to tackle a complex task, taking into account the agents' capabilities and communication costs. The study in [10] modified the original ACO algorithm by adding an efficiency factor, which is a function that relates to the agent capabilities and the total distance between agents. By integrating the efficiency factor with the basic ACO algorithm, the modified algorithm could find the best agent coalition within a relatively short time [10]. Simulation results demonstrated that the algorithm in Lu *et al.,* [10] outperforms the Forward Optimal Heuristic Algorithm. While the algorithm in [10] has demonstrated relatively good performance in several cases, it is originally designed to address a general task allocation problem. Furthermore, the original ACO algorithm for task allocation in [10], has also not considered some real-world factors such as the limited number of agents, the limited capability or function of the agents, the total power available on the system, and the system's overall operational costs. Therefore, there is a high probability that the chosen agent coalition has excessive wasted resources, which might not be desirable in real-world implementation.

To address the aforementioned problem, this study adopts and further modifies the ACO algorithm proposed in Lu *et al*., [10] to solve the task allocation problem in multi-UAV systems used in agriculture. This study considers a task allocation problem where the goal is to find the best coalition of agents to solve a task [3,10-13]. A group of agents in this study is associated with a UAV team and the task is associated with the agricultural tasks that need to be completed. However, different from Lu *et al.,* [10], the capabilities of the agents are modified based on the possible UAV capabilities in solving some real agricultural-related tasks [14-17]. To improve the performance of the algorithm in Lu *et al.,* [10] for solving task allocation problem in agriculture, the efficiency factor is also modified. This is the main contribution of the paper. While the efficiency factor in Lu *et al*., [10] only considers the distance between agents, the distance between the agent and the task needs to also be considered to allocate the task effectively. Thus, the efficiency factor in this study includes the distance between the agent and the task.

Simulations were conducted to demonstrate that this modified ACO algorithm can improve the performance of the basic ACO algorithms for solving task allocation problems, particularly in terms of the average total travel cost for each agent. The impact of varying different numbers of agents for solving the problem was also analysed. Furthermore, to increase the capability utilization of each agent, an additional function to calculate the system capability utilization was added to the ACO. As a proof of concept for the proposed algorithm, a simple experiment using five ground robots with a centralized control was carried out.

This paper is arranged in four sections. Section II describes an overview of Ant Colony Optimization algorithm and the modified ACO algorithm proposed in this study. Section III presents the results and discussions, starting from the simulation setup for task allocation problem in agriculture and followed by the simulation results. This section also describes the implementation of the modified ACO using real robots. Finally, section IV concludes the findings from this study and highlight the future research direction.

2. Methodology
2.1 Ant Colony Optimization for Task Allocation Problem

The idea of the ant colony algorithm is inspired by the behaviour of ants when searching for food [11]. The ants collaborate together to construct a solution by establishing an indirect communication mechanism called stigmergy. To guide other ants to the source of food, each ant deposits a chemical substance called pheromone along the travelled path (pheromone trails). Over time, these pheromone trails evaporate. The path that is constantly travelled by the ants, will have higher pheromone concentration compared to the paths that are rarely travelled by the ants. The paths frequently travelled by the ants will have a higher pheromone concentration compared to less-travelled paths. This mechanism is expected to lead most ants to eventually follow the shortest path to reach the food source. ACO adopts this behaviour to solve optimization problems. In ACO, the ants travelled along the graph to find the optimum solution [3]. The first problem that was solved utilizing ACO was Traveling Salesman Problem (TSP) [11]. The goal of TSP is to find the shortest tour for the salesman to visit a set of cities. In ACO, each ant represents an individual salesman. Each ant travels from the starting city, visits all other cities once and only once, and then returns to the starting city.

Another problem that can be solved using ACO algorithm is task allocation problem as described by Lu *et al.,* [10]. The task allocation problem in Lu *et al.,* [10] is defined as a problem of finding a coalition of agents to complete a given task, *v*. The task *v* needs a number of agent capabilities that should be fulfilled by the agent coalition. Let a population of agents be denoted as R = $\{r_1, r_2, ..., r_i, ..., r_I\}$, where each agent in the population can have one or more capabilities. Each agent has certain properties that are represented by the tuple {RK, RL} where RK represents its capabilities and RL represents its location in a two-dimensional space [x, y]. The agents' capabilities are represented in a multi-agent capability matrix MRK where each row element $RK_i = [rk_{ij}]$ indicates whether agent r_i has capability k_j. The values of rk_{ij} reflects a particular agent capability, as well as the size of the capabilities, where $rk_{ij} \in [0, 10]$. For example, MRK = $\begin{Bmatrix} 0 & 4 & 2 & 6 & 1 \\ 8 & 0 & 3 & 7 & 0 \\ 3 & 5 & 0 & 9 & 10 \end{Bmatrix}$ indicates that there are 3 agents and 5 capabilities where the first, second and third rows of MRK indicates the capabilities of agent 1, 2, and 3 with respect to capability 1, 2, 3, 4, and 5.

Suppose a task *v* needs *N* number of capabilities, where $n \in [1, J]$. A collection of capabilities that are required to complete *v* is Kv = $\{k_1, k_2, ... k_N\}$. A collection of agents with capability k_j to complete *v* is denoted as $R_v^{k_j}$. A group of agents which has the capabilities that *v* requires is thus defined as Rv = $\{r^{k_1}, r^{k_2}, ..., r^{k_N}\}$, where $r^{k_1} \in R_v^{k_1}$, $r^{k_2} \in R_v^{k_2}$, ..., $r^{k_N} \in R_v^{k_N}$ as shown in Figure 1.

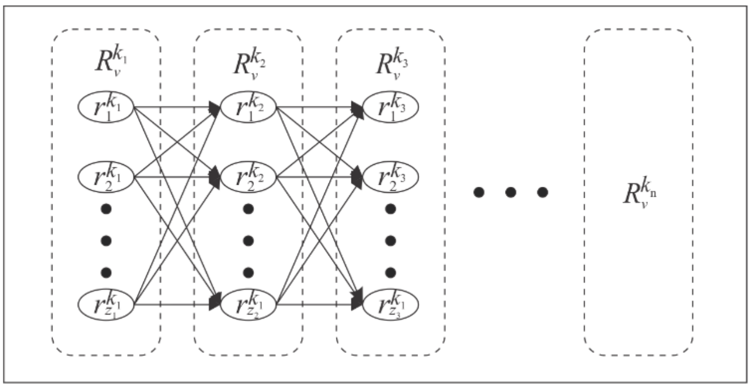


Fig. 1. The process of task allocation in [10]

Using the mathematical notations mentioned above, the ACO algorithm can be applied for solving a task allocation problem in Lu *et al.,* [10]. In TSP, a city is simulated as a node and a salesman is simulated as an ant, while in Lu *et al.,* [10], a node is a group of agents that have the same capabilities and the ants are used to find the optimal agent coalition. Let a colony of ants in a system be denoted by *S = {1, 2, 3, ... m, ..., M}*, where M is a total number of ants. Each ant finds a coalition of agents by choosing the node to travel based on a transition probability. Here, the node is the agent to be chosen to solve task *v* from a group of agents with the same capability k_j, $R_v^{k_j}$. The ants are initially placed in the agents that are listed in the first node group, that is, $R_v^{k_1}$. The transition probability of ant *m* to travel from node *a* to node *b* at time *t* is defined as follows [10]:

$$p_{ab}^{m}(t) = \begin{cases} \frac{[\tau_{ab}(t)]^{\alpha}[\eta_{ab}(t)]^{\beta}}{\sum_{s \subset allowed_m}[\tau_{as}(t)]^{\alpha}[\eta_{as}(t)]^{\beta}}; \; b \in allowed_m, \\ 0; \quad \text{otherwise}, \end{cases} \tag{1}$$

where $\tau_{ab}(t)$ is the amount of pheromone in path that connects node *a* to *b*; α is the relative importance of the pheromone trail which reflects the relative importance of $\tau_{ab}(t)$; $\eta_{ab}(t)$ is the heuristic function representing the desirability of state transition from *a* to *b*; and β is the relative importance of the heuristic function $\eta_{ab}(t)$. In Eq. (1), $allowed_m$ is all nodes that can be visited by ant *m,* that is, the agents with the next capability required by the task. Given [k_c, k_d] denotes two neighbouring capabilities, the heuristic function, $\eta_{ab}(t)$, was defined in Lu *et al.,* [10] as follows:

$$\eta_{ab}(t) = \eta_{ab}(t) = \frac{\omega_{1,c} rk_{ac} + \omega_{1,d} rk_{bd}}{\omega_2 d_{ab}} = \frac{\omega_{1,c} rk_{ac} + \omega_{1,d} rk_{bd}}{\omega_2 |rl_a - rl_b|}. \tag{2}$$

In Eq. (2), $\omega_{1,c}, \omega_{1,d} \in [0,1]$ are the capability weighting factors that represent the importance of capability k_c and k_d respectively to complete the task. As mentioned earlier, rk_{ac} indicates the size of capability k_c for agent r_a and rk_{bd} indicates the size of capability k_d for agent r_b. In Eq. (2), the distance between the two agents is $d_{ab} = |rl_a - rl_b| = |x_1 - x_2| + |y_1 - y_2|$ [10]. Here, rl_a is the coordinate of agent *a*, (x_1, y_1) and rl_b is the coordinate of agent *b*, (x_2, y_2). The notation ω_2 is the weighting factor for the distance. In summary, the numerator in Eq. (2) denotes the total capabilities of agents and the denominator is the total distance/communication cost between agents.

A tour is finished when all ants have reached the last node group, $R_v^{k_N}$. The pheromone on each path is then updated after a tour by Eq. (3):

$$\tau_{ab}(t+1) = (1-\rho)\tau_{ab}(t) + \sum_{m=1}^{M} \Delta\tau_{ab}^{m}(t), \tag{3}$$

where ρ is the pheromone evaporation coefficient and $\Delta\tau_{ab}^{m}$ is the amount of pheromone deposited by the m^{th} ant. In this study, the equation for $\Delta\tau_{ab}^{m}$ is calculated based on [10] as follows:

$$\Delta\tau_{ab}^{m}(t) = \begin{cases} Q\varepsilon_m \text{ ; if ant } m \text{ uses agents } a \text{ and } b \text{ in its path}, \\ 0 \text{ ; otherwise}, \end{cases} \tag{4}$$

where Q is a constant that indicates the pheromone strength and ε_m is the efficiency factor of the path travelled by the m^{th} ant. The efficiency factor ε is defined as follows [10]:

$$\varepsilon = \frac{\sum_{j=1}^{N} \omega_{1,j} rk_{ij}}{\sum_{j=1}^{N-1} \omega_2 d_j}. \tag{5}$$

In Eq. (5), the numerator is the sum of the agent capabilities, rk_{ij}, that can complete the task; $\omega_{1,j} \in [0,1]$ is the capability weighting factor that represents the importance of capability k_j to complete the task; N is the number of capabilities required to complete the task. Note that each ant only chooses one agent (one node) to complete one capability for a task. The denominator represents the communication cost generated to complete the task; d_j is the path length between two adjacent nodes and ω_2 is the weighting factor to approximate the communication cost. If the termination criteria have been met, a group of agents with the highest efficiency ε will be chosen as the best agent coalition to complete the task.

2.2 First Modification: Consideration of the Distance between the Agent and the Task

This section describes the improved Ant Colony Optimization for agricultural application. It is assumed that there is only one task and its location is known by the agents. In this case, a group of agents in ACO represents the UAVs, while the task represents the field. The aim of the system is to find a coalition of UAVs with certain capabilities to be allocated to the paddy field. It is also assumed that the system knows the capabilities that are needed by the paddy field.

By considering the above scenario, the ACO algorithm can then be adopted for solving task allocation proposed by Lu *et al.,* [10]. However, in order to solve the problem in agriculture, a modification to the algorithm is performed. In this study, the same ACO algorithm proposed in Lu *et al.,* [10] is used to calculate the transition probability in Eq. (1) and update the pheromone intensity as defined in Eq. (3) and Eq. (4).

Compared to Lu *et al.,* [10], the proposed algorithm, however, differs on the heuristic function (Eq. (2)) and the efficiency factor (Eq. (5)). Thus, these are the contributions of the paper. In agricultural application, the distance between the agent (UAV) and the task (field) is a critical factor. This is because the distance between the agent and the task may relate to the cost and power needed by the UAV to reach the field. Because of this reason, the heuristic function in Eq. (2), $\eta_{ab}(t)$, is defined in this study as in the following equation:

$$\eta_{ab}(t) = \frac{\omega_{1,c} rk_{ac} + \omega_{1,d} rk_{bd}}{\omega_2 d_{ab} + \omega_3 (d_{av} + d_{bv})}, \tag{6}$$

where d_{ab} denotes the distance between agent *a* and agent *b*; d_{av} denotes the distance between agent *a* and task *v*; and d_{bv} denotes the distance between agent *b* and task *v*. As mentioned earlier, $\omega_{1,c}$, $\omega_{1,d} \in [0,1]$ are the capability weighting factors that represent the importance of capability k_c and k_d respectively to complete the task; whereas ω_2 and ω_3 are the weighting factors for the distance between agents and the distance between agent and task, respectively. Furthermore, the efficiency factor in Eq. (5) is modified as follows:

$$\varepsilon = \frac{\sum_{j=1}^{N} \omega_{1,j} rk_{ij}}{\sum_{j=1}^{N-1} \omega_2 d_j + \sum_{j=1}^{N} \omega_3 d_{r_j v}}. \tag{7}$$

Note that different from Equation 5, we add the component $\sum_{j=1}^{N} \omega_3 \, d_{r_j v}$ is added where ω_3 is the weighting factor for the distance between agent and task and $d_{r_j v}$ is the distance between agent r_j and task *v*. This modification implies that efficiency increases with the decrease of total distance, which indicates that a shorter distance between the agents and the task is preferred.

In this study, the agent capabilities are also modified using common UAV capabilities in solving agricultural-related tasks as mentioned in Grammatikis *et al.,* [14] and Kim *et al.,* [15]. Table 1 shows the capabilities of the agents represented as binary numbers, *i.e.*, 0 or 1, where 0 denotes incapability and 1 denotes capability.

Table 1
Agent capabilities in agricultural application

Capabilities of Agent	
Soil monitoring	k_1
Crop health monitoring (RGB camera)	k_2
Crop Nutrient Monitoring (multispectral camera)	k_3
Pest monitoring	k_4
Weeding surveillance	k_5
Yield and Biomass Analysis	k_6
Irrigation system monitoring	k_7
Fertilizing	k_8
Seeding	k_9
Pesticide spraying	k_{10}

A step-by-step process of the first proposed modification of ACO algorithm is shown in Figure 2. The dotted lines in Figure 2 shows the new contributions that are made for solving task allocation problem in agricultural application. When the termination criteria are met, a group of agents with maximum efficiency ε is chosen as the best agent coalition to complete the task.

2.3 Second Modification: Consideration of the Actual Capability Utilization to Perform the Task

In the original ACO algorithm for task allocation proposed by Lu *et al.,* [10], some real-world factors are not yet considered, *e.g.,* the limited number of agents, the limited capability or function of the agents, the total power available on the system, and the system's overall operational costs. Therefore, there is a high probability that the chosen agent coalition has too many wasted resources, which is not desirable in any real-world implementation.

In order to reduce the wasted/unused agents' capabilities and maximizing the system's overall performance, an additional function to calculate the system capability utilization is added. The function is adopted from the capacity utilization equation [18] as in Eq. (8). The equation shows that the total system utilization is a ratio between the actual resources that are utilized by the system and the total resources that are available in the system.

$$\text{Utilization} = \frac{\text{actual resource used}}{\text{total resource available}} \times 100\% \qquad (8)$$

In the proposed algorithm, the resource utilization is translated as the ratio between number of agent capabilities that are utilized to complete a certain task and the total agent capabilities. The greater the capability utilization, the more efficient the implementation in a multi-robot system. In this study, a formula based on Eq. (8) is added to calculate the percentage of capability utilization by one individual agent, φ_i, as follows:

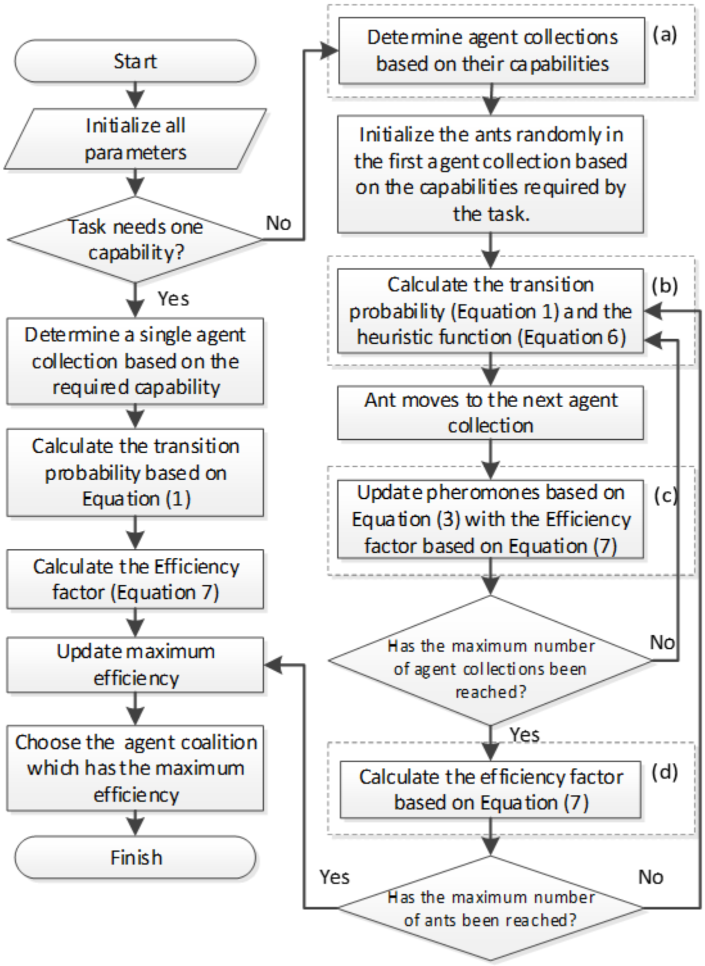


Fig. 2. Flowchart of ACO for task allocation in agricultural application

where φ_i is the capability utilization by agent *i*, C_{iv} is the number of capabilities agent *i* used to execute the task *v*, and C_i is the total number of capabilities of agent *i*.

$$\varphi_i = \frac{C_{iv}}{C_i}, \tag{9}$$

The total capability utilization for a coalition of agents to execute task *v* can be calculated by combining each capability values of the agents in the coalition, as in the following equation:

$$\varphi_{\text{task}} = \frac{1}{N_K} \sum_{i \epsilon K} \varphi_i , \tag{10}$$

where φ_{task} is the total resource utilization to execute a task, N_K is the number of chosen agents in the coalition, and *K* is the set of chosen agents in the coalition.

The capability utilization in Eq. (10) is then multiplied to the efficiency formulation in Eq. (7) as follows:

$$\varepsilon = \frac{\sum_{j=1}^{N} \omega_{1,j} rk_{ij}}{\sum_{j=1}^{N-1} \omega_2 d_j + \sum_{j=1}^{N} \omega_3 d_{r_j v}} \times \Phi_{\text{task}} . \tag{11}$$

3. Results and Discussion

3.1. Evaluation for the First Modification of Ant Colony Optimization

The proposed ACO algorithm in this study is implemented using Matlab version 9.4.0 (R2018a). To evaluate the performance of the proposed algorithm, the proposed algorithm is compared with the ACO algorithm proposed by Lu *et al.,* [10]. The task allocation problem in this paper is simulated in a two-dimensional area. Following Lu *et al.,* [10], the size of the area is in the range of $0 < x \leq 30$ and $0 < y \leq 30$. In this study, only one task/field is considered. The parameters used in this paper is shown in Table 2. In this first modification, we perform two simulations as described in the following subsections.

Table 2
ACO Parameters

Parameters	**Symbol**	**ACO in Lu *et al.,* [10]**	**Proposed ACO**
Importance of the pheromone trail	α	0.9	0.9
Importance of the heuristic function	β	2.2	2.2
Pheromone evaporation coefficient	ρ	0.5	0.5
Number of ants	M	80	80
Pheromone strength	Q	1	1
Maximum ACO iteration	Ncmax	60	60
Capability weighting factor	ω_1	[0, 1]	[0, 1]
Weighting factor for the communication cost	ω_2	0.1	0.1
Weighting factor for the distance between agent and task	ω_3	0.1	0.1

3.1.1 Simulation 1

This simulation is designed to evaluate the performance of the three algorithms in terms of the total travel cost. This performance metric is calculated after the stopping criteria for the algorithm have been reached. The total travel cost is defined as:

$$D = \sum_{i=1}^{N_K} d_{iv} , \tag{12}$$

where N_K is the number of agents in an agent coalition, and d_{iv} is the distance between the agent *i* and the task *v*. The average travel cost of each agent is also calculated, defined as:

$$D_R = \frac{D}{N_K}. \tag{13}$$

The simulation was repeated 50 times with different capabilities required by the task. The position of the agents and the tasks are initialized randomly for each simulation. We compared the performance of the proposed algorithm with the ACO in [10] in terms of the average total travel cost and average travel cost of each agent. The performance comparisons of the algorithms in Simulation 1 are shown in Table 3. Table 3 shows that the performance of the ACO in Lu *et al.,* [10] and the modified ACO is relatively similar in terms of the total travel cost. It can be observed in Table 3 that the average total travel costs for ACO in Lu *et al.,* [10] and the modified ACO are 37.28 ± 5.714 and 32.6 ± 5.642 respectively. There is also no significant difference in the average percentage of resources used for both the algorithms. However, significant different results are obtained for the average total travel cost for each agent.

The average total travel cost for each agent for the ACO in Lu *et al.*, [10] is 15.8 $\pm$ 1.782, while the total travel cost for each agent in the modified ACO is only 11.444$\pm$1.373. This result shows that, on average, each agent in the coalition for the modified ACO algorithm has lower travel cost. This result is preferable in the considered problem, as it is expected that the agent should share the load of the task more evenly. Also, by assuming that each agent has the same travel speed, it is expected that the agent can reach the task faster by having lower travel cost compared to the benchmark algorithm.

Table 3
Performance comparison of basic ACO, ACO in Lu *et al.,* [10], and modified ACO

	Average Total Travel Cost	Average Travel Cost of Each Agent
ACO in Lu *et al.,* [10]	37.28 $\pm$ 5.714	15.8 $\pm$ 1.782
Modified ACO	32.6 $\pm$ 5.642	**2.444 $\pm$ 1.373**

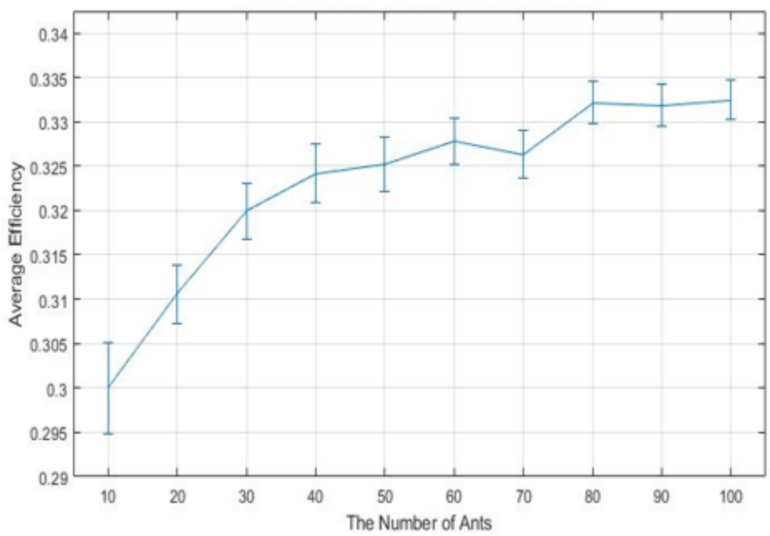
3.1.2 Simulation 2

The second simulation is conducted to observe the effect of varying the number of ants and the number of agents. The performance of the algorithm is evaluated based on the efficiency factor defined in Eq. (7). In this simulation, other variables that are not investigated are set to be constant. The simulation is repeated 50 times with different random seeds to have sufficient statistical results. For both simulations, ten agents are employed with binary MRK shown in Table 4.

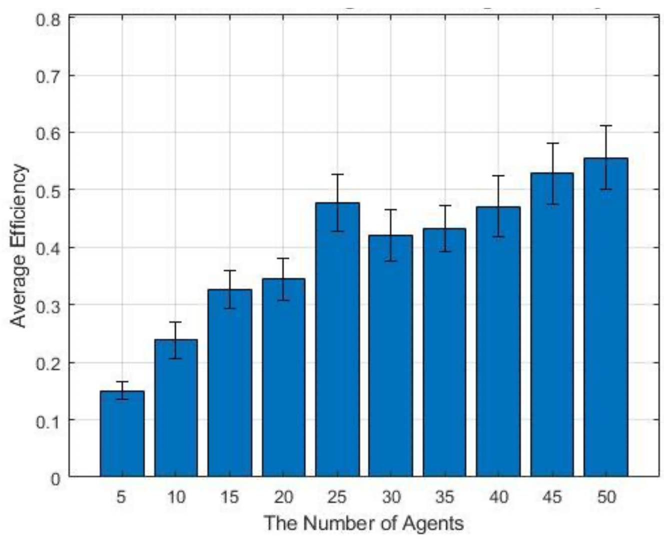
Table 4
Multi-agent capabilities matrix (MRK)

	k_1	*k2*	*k3*	*k4*	*k5*	*k6*	*k7*	*k8*	*k9*	*k10*
r1	0	1	1	0	1	0	0	1	0	0
r2	0	0	0	0	1	1	0	0	1	0
r3	1	0	1	0	0	0	0	1	1	0
r4	0	1	1	1	0	0	0	1	0	1
r5	0	0	1	0	0	1	1	1	0	1
r6	0	0	1	0	1	0	0	1	1	1
r7	0	0	0	0	1	0	1	0	1	0
r8	1	0	1	0	1	0	0	0	1	0
r9	0	1	1	1	1	0	0	1	1	0
r10	0	1	0	0	1	0	1	0	1	1

We also investigated the effect of varying the number of ants and agents in the modified ACO algorithm. Figure 3(a) shows the effect of varying the number of ants in the modified ACO algorithm at the 95% confidence interval. It can be seen from Figure 3(a) that by increasing the number of ants, the average efficiency of the modified ACO algorithm gradually increases. If the number of ants is too small, the modified ACO may not be able to reach the optimal solution. Otherwise, if the number of ants is too large, it will also increase the computational time. Nevertheless, increasing the number of ants means that more ants explore the candidate solutions, so the algorithm has a higher chance of finding the best solution. Furthermore, Figure 3(a) shows that when the number of ants is increased from 80 to 100, there is no significant difference in the average efficiency. This implies that at a certain point there is no significant impact on increasing the number of ants to the efficiency factor.



(a)



(b)

Fig. 3. The effect of varying the number of ants and the number of agents in terms of average efficiency

Figure 3(b) shows the effect of varying the number of agents in the modified ACO. In general, there is a trend that the average efficiency factor increases with the increase of the number of agents at the 95% confidence interval. This means that the increase of the number of agents will positively affect the system performance. If the number of agents is large enough there is a higher chance that the capabilities needed by the task can be fulfilled by the agent coalition. However, note that this may also come to the problem of having a high computational cost. It can be seen in Figure 3(b) that when the number of agents is 40 there is no significant improvement in the efficiency factor. Thus, choosing a moderate number of agents is preferable in this case as at certain point there is no significant impact on increasing the number of agents to the efficiency factor.

3.2 Second Modification of Ant Colony Optimization

In this study, two scenarios are simulated to evaluate the performance of the second modification and the first modification of ACO algorithm (later referred as ACO-S). In Scenario 1, the simulations were repeated 30 times using a various set of capabilities needed by the task. Simulations for Scenario 2, however, were also repeated 30 times but with a various set of robot positions.

The simulations were conducted using 10 virtual robots, 10 types of robot capabilities, and one target. Both simulations of the two scenarios use a Cartesian coordinate dimension space of 50×50 centimeters as the simulation area. In the two scenarios, the same target position (X = 30, Y = 25) and the same agent capability matrix (Table 1), referred as Matrix Agent Capabilities, were used for all test data. Here, the Matrix Agent Capabilities (MAC) is the matrix that lists all capabilities possessed by each robot. In this study, it is assumed that no robots have the same capabilities, meaning that the robots in the multi-robot system are assumed to be heterogeneous. Following Sriatun *et al.,* [19], each capability value in this study is either set to be 0 when the robot does not have certain capability or 1 when the robot has the capability. For example, as can be seen in Table 5, robot R1 has the following capabilities: C3, C5, C8, and C10.

Table 5
Matrix Agent Capabilities (MAC) of 10 robots for testing scenarios

Robot	**Capabilities [C1,C2,C3,C4,C5,C6,C7,C8,C9,C10]** [a]
R1	[0, 0, 1, 0, 1, 0, 0, 1, 0, 1]
R2	[1, 0, 1, 0, 0, 1, 0, 0, 1, 1]
R3	[0, 1, 0, 0, 0, 0, 0, 1, 1, 1]
R4	[0, 0, 1, 0, 0, 1, 0, 0, 0, 0]
R5	[1, 1, 0, 0, 1, 0, 1, 0, 0, 0]
R6	[1, 1, 0, 0, 1, 0, 0, 0, 1, 1]
R7	[0, 0, 1, 1, 1, 0, 1, 1, 0, 0]
R8	[1, 1, 1, 1, 0, 0, 1, 1, 0, 0]
R9	[0, 0, 0, 1, 0, 1, 1, 1, 1, 0]
R10	[0, 1, 0, 1, 0, 0, 1, 0, 0, 0]

[a] Capabilities that are assumed owned by each robot

3.2.1 Scenario 1

The first scenario is designed to see which algorithm produces the most optimal robot coalition (i.e., the coalition that has the highest percentage of total resource used). In the first scenario, the positions of the robots are depicted in Table 6. In this scenario, 30 simulation data is used with a variety of set capabilities required by the task.

Table 6
Position data of 10 virtual robots for Scenario 1

Robot	R1	R2	R3	R4	R5	R6	R7	R8	R9	R10
X Position	10	20	30	40	50	50	40	30	20	10
Y Position	35	50	50	50	35	15	0	0	0	15

The simulation results of the two algorithms in the first scenario are measured based on the percentage of total average resource used, the total average communication costs between robots in the best robot coalition, and the average total distance between the robots' position in the coalition to the target. Figures 4 and 5 show the simulation results for Scenario 1.

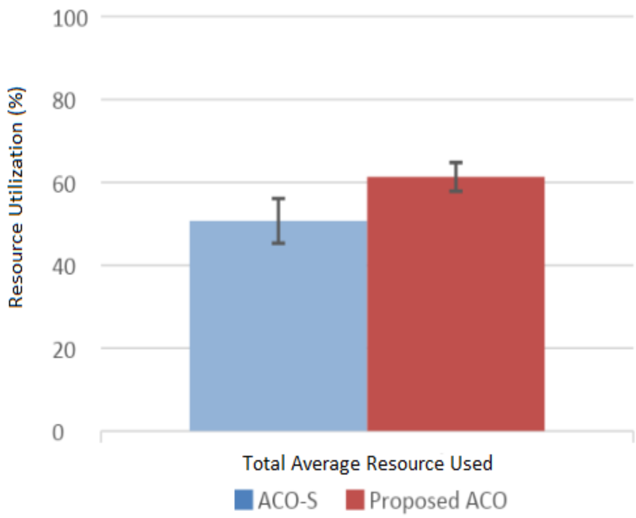


Fig. 4. Total Average of resource-used in the multi-robot systems for Scenario 1

It can be seen from Figure 4 that, by using a 95% confidence interval, the proposed ACO algorithm produces a higher total average resource used compared to the ACO-S algorithm. In terms of the total average communication cost and travel distance, the performance of the proposed ACO algorithm and the ACO-S algorithm is not statistically significant (see Figure 5).

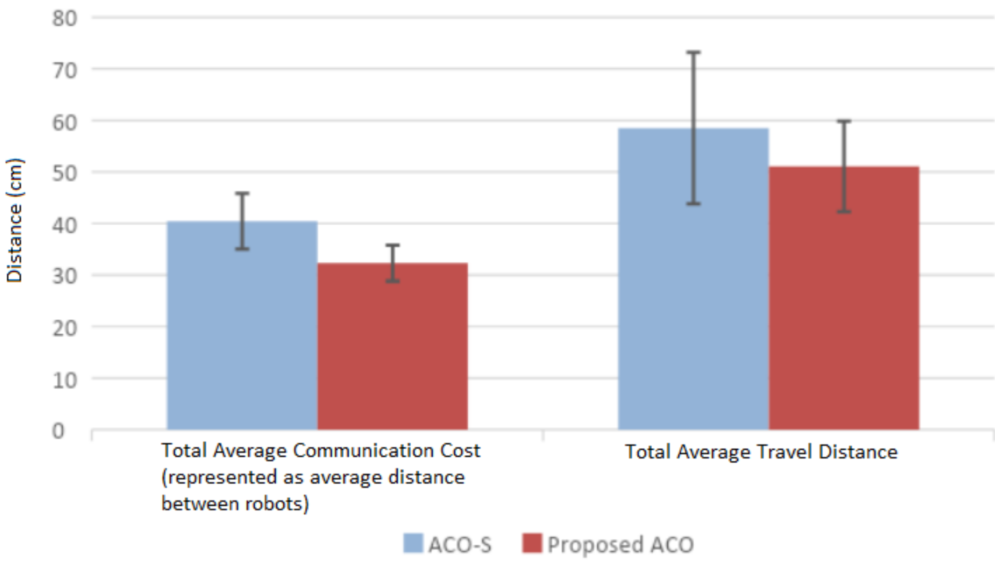


Fig. 5. Total average communication cost and travel distance for Scenario 1

3.2.2 Scenario 2

The second scenario is conducted to see the performance of the proposed algorithm in terms of the total communication costs used by the robots and the total distance between the robots to the task. In this scenario, 30 simulation data were used with different robot positions that were manually determined for the first four test data and randomly generated for the rest of the data. The target position and the robot capabilities (Table 4) are set following the first scenario. Specifically for this scenario, the capabilities required by the task are set as C1, C3, C4, C7, and C9. The simulation results from the second scenario can be seen in Figure 6 and Figure 7.

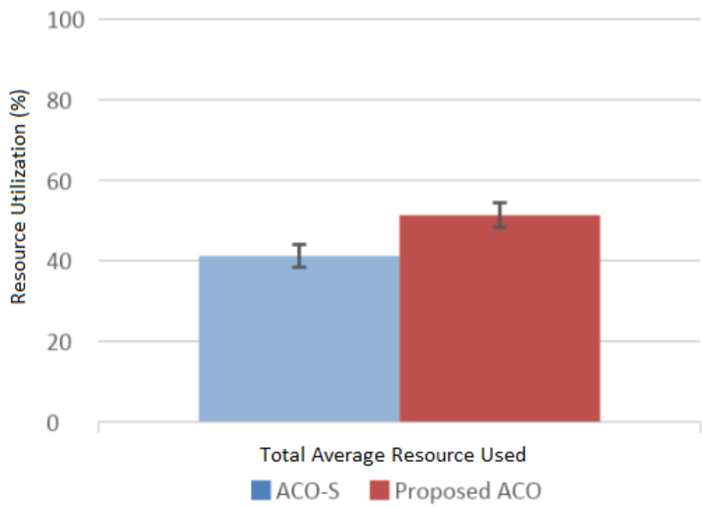


Fig. 6. Total Average resource used in the multi robot systems for Scenario 2

In Figure 6, it can be seen that the percentage of total average resource used of the proposed ACO algorithm is about 10% higher than ACO-S. Using a 95% confidence interval, it can be concluded that the proposed ACO has produced a significantly higher total average of resource used than the ACO-S algorithm. Furthermore, it can be seen from Figure 7 that the total average communication cost from the proposed ACO is also significantly lower than the ACO-S algorithm.

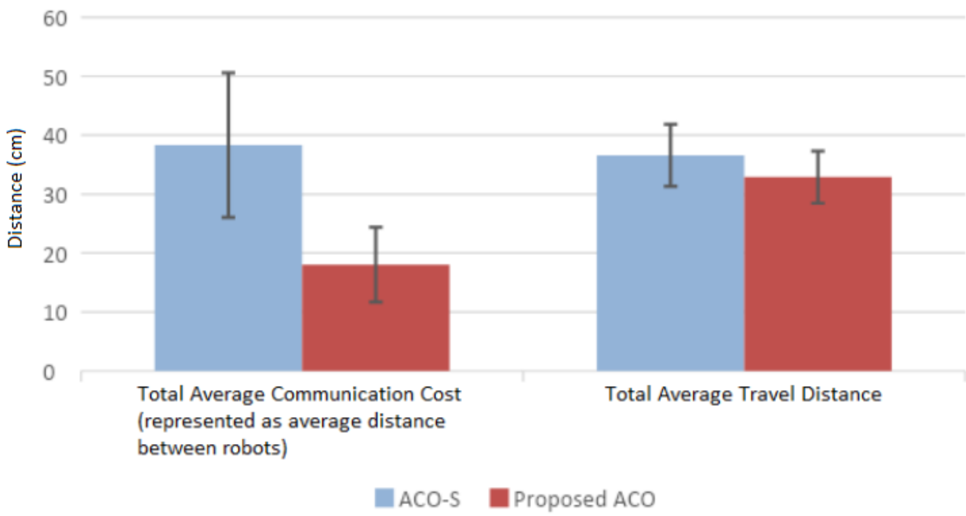


Fig. 7. Total average communication cost and travel distance from Scenario 2

3.3 Overall Evaluation

Table 7 summarizes the overall simulation results. It can be seen from Table 7 that the proposed ACO algorithm produced a relatively higher average of total resource used, which is 10.62% higher than ACO-S in the first scenario and 10.11% higher in the second scenario. The proposed ACO algorithm has also succeeded in reducing the total average communication costs between the robots by ±20% and ±52% in Scenario 1 and Scenario 2 respectively. Simulation results also indicate that the average total travel distance obtained by the proposed ACO algorithm has been reduced by ±12% and ±10% in Scenario 1 and Scenario 2 respectively.

Table 3
Simulation results summary

		ACO-S	Second modification of ACO
Resource used (%)	Scenario 1	50.70 ± 5.40	61.32 ± 3.47
	Scenario 2	41.18 ± 2.83	51.29 ± 3.07
Communication costs (cm)	Scenario 1	40.45 ± 14.68	32.29 ± 8.76
	Scenario 2	38.32 ± 12.28	18.04 ± 6.34
Travel distance (cm)	Scenario 1	58.46 ± 7.39	51.07 ± 5.04
	Scenario 2	36.61 ± 5.26	**2.89 ± 4.42**

3.4 The Implementation on Multi-Robot System

To see how the proposed algorithm can be implemented in the real world, the algorithm is also tested with real robots. Figure 8 shows the design of multi-robot implementation for this study. It can be seen that the robots in the multi-robot system will first send the information of their position and capabilities to the server via a Wi-Fi network. The robots will then go idle while the server begins to decide the best coalition of robots to execute the task using the proposed ACO algorithm. When the server has found the best coalition of robots, the server sends this command to the multi-robot system. The robots that are chosen to by the server to execute the task will then move towards the position of the task. On the other hand, robots that are not chosen will stay in their positions.

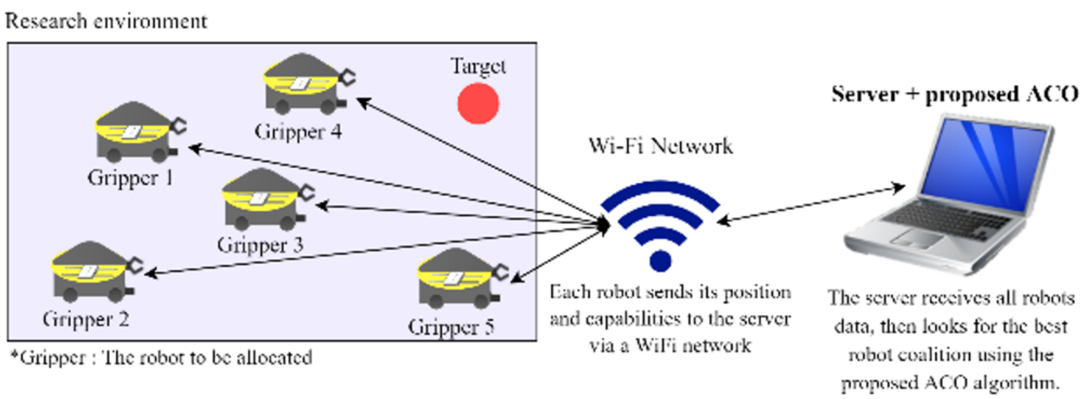
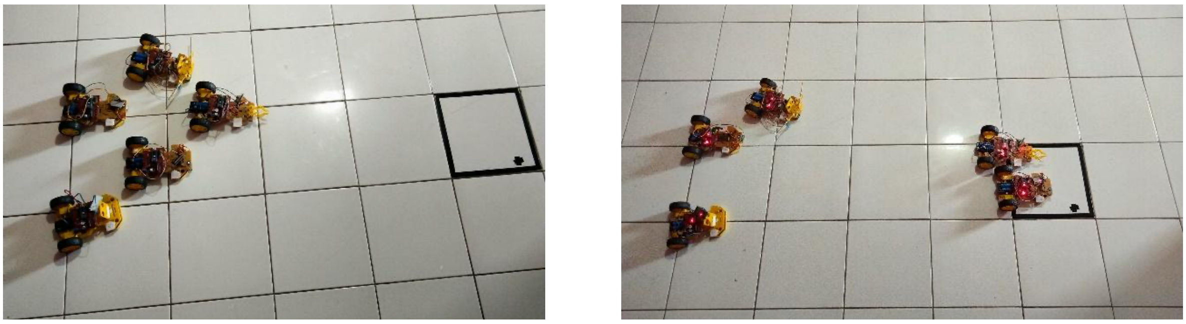


Fig. 8. The design of multi-robot system implementation

The communication between the server and the robots (as clients) are set in parallel to make the process more efficient. In this study, the computation of the proposed ACO algorithm is conducted in the server. Figure 9 (a) shows the initial setting for the experiment using five gripper robots as the clients and a target that is marked with black tape. In this experiment, the robots have successfully communicated with the server. It can be observed that the robots chosen by the proposed ACO algorithm have moved towards the target as expected. Figure 9 (b) shows the results after the proposed ACO algorithm has been executed. After the robots receive the information of the best coalition from the server, the chosen robots are able to move forward to the location of the target, while the others remain in their initial position.



(a) (b)

Fig. 9. Experiment of multi-robot system

4. Conclusion

This study has proposed a modification of Ant Colony Optimization (ACO) algorithm for task allocation problem in agriculture. Two modifications on the algorithm were proposed, i.e.,

i. Consideration of the Distance between the Agent and the Task
ii. Consideration of the Actual Capability Utilization to Perform the Task. Simulation Results have shown that, by employing the first modification, the average total travel cost of each agent can be reduced compared to the benchmark algorithm. By performing the second modification, it was found that in some scenarios, the algorithm is more optimal in terms of the total average resource used and the total communication cost compared to the ACO algorithm that we have modified earlier.

To further examine the effectiveness of the algorithms, the proposed ACO algorithm can be compared with other optimization techniques such as Genetic Algorithm and Constrained Particle Swarm Optimization. For future research, the robots can also be further developed to move in parallel and integrated with obstacle avoidance algorithms to become more adaptive. Future research directions would be to develop these features so that the systems can be implemented to solve wider task allocation problems situated in dynamic and uncertain environments.

Acknowledgement
This work is supported by the Ministry of Education, Culture, Research, and Technology of Indonesia through the Directorate of Research and Innovation, IPB University, research grant *Penelitian Dasar Unggulan Perguruan Tinggi* (PDUPT) No. 001/E5/PG.02.00.PL/2023 and 15853/IT3.D10/PT.01.02/P/T/2023. We would like to thank Made Widhi Surya Atman, Azwirman Gusrialdi from Intelligent Networked Systems Group, Faculty of Engineering and Natural Sciences, Tampere University, Finland, for giving constructive input and feedback for this study. The authors would also like to thank the committee of International Symposium on Computer Science for Smart Agriculture, Education and Medical (CSSAEM): Perspective from Indonesia and Malaysia for the opportunity to present this research article.

References

[1] Hardhienata, Medria KD, V. Ugrinovskii, and Kathryn E. Merrick. "Task allocation under communication constraints using motivated particle swarm optimization." In *2014 IEEE Congress on Evolutionary Computation (CEC)*, pp. 3135-3142. IEEE, 2014. https://doi.org/10.1109/CEC.2014.6900560
[2] Khani, Maryam, Ali Ahmadi, and Hajar Hajary. "Distributed task allocation in multi-agent environments using cellular learning automata." *Soft computing* 23 (2019): 1199-1218. https://doi.org/10.1007/s00500-017-2839-5

[3] Wang, Jianping, Yuesheng Gu, and Xiaomin Li. "Multi-robot task allocation based on ant colony algorithm." *J. Comput.* 7, no. 9 (2012): 2160-2167. https://doi.org/10.4304/jcp.7.9.2160-2167

[4] Coltin, Brian, and Manuela Veloso. "Mobile robot task allocation in hybrid wireless sensor networks." In *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 2932-2937. IEEE, 2010. https://doi.org/10.1109/IROS.2010.5649893

[5] Dorigo, Marco, Vittorio Maniezzo, and Alberto Colorni. "Ant system: optimization by a colony of cooperating agents." *IEEE transactions on systems, man, and cybernetics, part b (cybernetics)* 26, no. 1 (1996): 29-41. https://doi.org/10.1109/3477.484436

[6] Mohan, B. Chandra, and R. Baskaran. "A survey: Ant Colony Optimization based recent research and implementation on several engineering domain." *Expert Systems with Applications* 39, no. 4 (2012): 4618-4627. https://doi.org/10.1016/j.eswa.2011.09.076

[7] Escario, Jose B., Juan F. Jimenez, and Jose M. Giron-Sierra. "Ship maneuvering planning using swarm intelligence." In *OCEANS 2011 IEEE-Spain*, pp. 1-9. IEEE, 2011. https://doi.org/10.1109/Oceans-Spain.2011.6003449

[8] Fu, Yongling, and Wanqin Luo. "Optimal task assignment for serial-parallel hybrid robots cooperationvia ant colony optimization." In *2009 9th International Conference on Electronic Measurement & Instruments*, pp. 3-795. IEEE, 2009. https://doi.org/10.1109/ICEMI.2009.5274178

[9] Pendharkar, Parag C. "An ant colony optimization heuristic for constrained task allocation problem." *Journal of Computational Science* 7 (2015): 37-47. https://doi.org/10.1016/j.jocs.2015.01.001

[10] Wang, Lu, Zhiliang Wang, Siquan Hu, and Lei Liu. "Ant colony optimization for task allocation in multi-agent systems." *China Communications* 10, no. 3 (2013): 125-132. https://doi.org/10.1109/CC.2013.6488841

[11] Dorigo, Marco, and Luca Maria Gambardella. "Ant colony system: a cooperative learning approach to the traveling salesman problem." *IEEE Transactions on evolutionary computation* 1, no. 1 (1997): 53-66. https://doi.org/10.1109/4235.585892

[12] Wooldridge, Michael. *An introduction to multiagent systems*. John wiley & sons, 2009.

[13] Hussein, Ahmed, Mohamed Adel, Mohamed Bakr, Omar M. Shehata, and Alaa Khamis. "Multi-robot task allocation for search and rescue missions." In *Journal of Physics: Conference Series*, vol. 570, no. 5, p. 052006. IOP Publishing, 2014. https://doi.org/10.1088/1742-6596/570/5/052006

[14] Radoglou-Grammatikis, Panagiotis, Panagiotis Sarigiannidis, Thomas Lagkas, and Ioannis Moscholios. "A compilation of UAV applications for precision agriculture." *Computer Networks* 172 (2020): 107148. https://doi.org/10.1016/j.comnet.2020.107148

[15] Kim, Jeongeun, Seungwon Kim, Chanyoung Ju, and Hyoung Il Son. "Unmanned aerial vehicles in agriculture: A review of perspective of platform, control, and applications." *Ieee Access* 7 (2019): 105100-105115. https://doi.org/10.1109/ACCESS.2019.2932119

[16] Tian, Haoxin, Zhenjie Mo, Chenyang Ma, Junqi Xiao, Ruichang Jia, Yubin Lan, and Yali Zhang. "Design and validation of a multi-objective waypoint planning algorithm for UAV spraying in orchards based on improved ant colony algorithm." *Frontiers in Plant Science* 14 (2023): 1101828. https://doi.org/10.3389/fpls.2023.1101828

[17] Kaamin, Masiri, Nor Baizura Hamid, Noorul Hudai Abdullah, Mardiha Mokhtar, Siti Nooraiin Mohd Razali, Muhammad Naim Abu Bakar, Mohamad Ridhwan Zahri, and Nur Thaqifah Aina Mohd Zaidi. "Application of UAV and CSPI Matrix for Roof Inspection at Dewan Sultan Ibrahim and Dewan Tunku Mahkota Ismail UTHM." *Journal of Advanced Research in Applied Sciences and Engineering Technology* 28, no. 1 (2022): 106-115. https://doi.org/10.37934/araset.28.1.106115

[18] Corrado, Carol, and Joe Mattey. "Capacity utilization." *Journal of Economic Perspectives* 11, no. 1 (1997): 151-167. https://doi.org/10.1257/jep.11.1.151

[19] Sriatun, Mamiek, Buono, Agus & Hardhienata, Medria Kusuma Dewi. "Modification of the Ant Colony Optimization for solving Task Allocation Problem in a Landslide Disaster Scenario." B.S Thesis, Department of Computer Science, Faculty of Mathematics and Natural Sciences, IPB University, Indonesia. (2019).