Accredited Ranking SINTA 2
Decree of the Director General of Higher Education, Research, and Technology, No. 158/E/KPT/2021
Validity period from Volume 5 Number 2 of 2021 to Volume 10 Number 1 of 2026

Published online on: **http://jurnal.iaii.or.id**

JURNAL RESTI
(Rekayasa Sistem dan Teknologi Informasi)
Vol. 6 No. 6 (2022) 911 - 922 ISSN Media Electronic: 2580-0760

Ant Colony Optimization Modelling for Task Allocation in Multi-Agent System for Multi-Target

Iis Rodiah[1], Medria Kusuma Dewi Hardhienata[2], Agus Buono[3], Karlisa Priandana[4]
[1,2,3,4] Department of Computer Science, Faculty of Mathematics and Natural Sciences, IPB University
[1]rodiah_0412@apps.ipb.ac.id, [2]medria.hardhienata@apps.ipb.ac.id, [3]agusbuono@apps.ipb.ac.id, [4]karlisa@apps.ipb.ac.id

Abstract

Task allocation in multi-agent system can be defined as a problem of allocating a number of agents to the task. One of the problems in task allocation is to optimize the allocation of heterogeneous agents when there are multiple tasks which require several capabilities. To solve that problem, this research aims to modify the Ant Colony Optimization (ACO) algorithm so that the algorithm can be employed for solving task allocation problems with multiple tasks. In this research, we optimize the performance of the algorithm by minimizing the task completion cost as well as the number of overlapping agents. We also maximize the overall system capabilities in order to increase efficiency. Simulation results show that the modified ACO algorithm has significantly decreased overall task completion cost as well as the overlapping agents factor compared to the benchmark algorithm.

Keywords: task allocation, multi-agent system, multiple tasks, ACO.

1. Introduction

Multi-agent systems have been recently used in various fields due to their superiority in completing complex tasks compared to single-agent systems [1]–[4]. One of the problems in multi-agent systems is task allocation, *i.e.* the problem of allocating a group of agents in completing task to achieve the system's goal [2], [5]–[10]. Some real-world task allocation problems in multi-agent systems include the coordination and planning problems of multi-robot deployment in production process [11], coordination problems of several drones [4], [12]–[14], and multi-robot allocation problems in precision agriculture [15]–[18].

Task allocation in a multi-agent system is an optimization problem with high complexity. It is classified as an NP-hard problem with difficulty in finding an exact solution [1]. Several approaches have been used to find the best solution to solve the task allocation problem in a multi-agent system. Some of the widely used approaches are the heuristic methods, such as the Auction-based method which is inspired by the economic system [3], [19]. The advantage of this method is that it has a high scalability. However, the required computational resources increase as the scale of the problem increases [14].

Other heuristic methods that have been used to solve the task allocation problem in multi-agent systems are inspired by natural events (bio-inspired), *e.g.* Genetic Algorithm (GA) [20] and Ant Colony Optimization (ACO) algorithms [7], [18], [21]–[23]. These bio-inspired methods tend to require lower computational resources compared to other methods [14]. In general, GA can find the best solution for multi agent system's task allocation faster than other methods. However, the efficiency of the search process using the GA method decreases when the scale of the problem increases due to the increasing number of possible solutions built at the beginning of the iteration [24]. Another bio-inspired heuristic method, the ACO algorithm, uses heuristic information and learning mechanisms in the form of pheromone trails in finding the solution. Although the convergence rate at the beginning of its iteration is relatively slow, the efficiency of the search process carried out by the ACO algorithm improves as the pheromone trail increases [24].

The ACO algorithm is an optimization algorithm introduced by Dorigo [25]. It is inspired by the behavior of ants, *i.e.* using pheromone trails to find foods. Wang [21] introduced a modification of ACO algorithm to solve the task allocation problem in a multi-agent system by considering the distance factor between agents. Another study conducted by Sriatun

Accepted: 30-06-2022 | Received in revised: 23-10-2022 | Published: 27-12-2022

[23] further modified the ACO algorithm by considering the distance between the agent and the task. It was found from the study by Sriatun [23] that the efficiency of the agents' coalition is better than the basic ACO algorithm [25] and the ACO algorithm modified by Wang [21].

This study aims to further develop the method proposed by Wang [21] and Sriatun [23] by considering the condition when there is more than one task. In this study, tasks are defined as targets which need to be completed by using several agents' capabilities. Thus, "tasks" are hereinafter referred to as "targets", and we will focus our discussion on the task allocation problem in a multi-target scenario.

An example of multi-target scenario is landslide disaster scenario, where there may be more than one targets (victims) to be rescued. Different from the single-target scenario, problem in a multi-target scenario may occur when there are one or more overlapping agent(s) chosen for different targets. To overcome this problem, it is necessary to add an objective function to minimize the number of overlapping agents which must be optimized simultaneously with other objective functions (multi-objective optimization problem). Several studies have shown that the ACO algorithm can be modified to solve multi-objective optimization problems [26]–[31]. This study aims to modify the ACO algorithm to solve the task allocation problem in a multi-target-multi-agent system. The final solution was obtained by optimizing all objective functions, *i.e.* minimizing the cost of task completion, minimizing the number of overlapping agents, and maximizing the system capabilities. Simulations were conducted to compare the efficiency of the modified ACO algorithm with the existing benchmark algorithms.

2. Research Methods

2.1 Ant Colony Optimization (ACO) for Solving Task Allocation Problems in Multi-Agent Systems

The ACO algorithm is one of the heuristic methods to solve optimization problems by imitating the behavior of ant colonies, *i.e.* utilizing pheromone trails to find foods. One of the implementations of ACO is to solve the widely-known Traveling Salesman Problem (TSP). Here, ants are represented as the artificial agents who travel through all the cities that must be visited to find the shortest route by utilizing pheromone trails [25]. The flowchart of the original ACO algorithm for TSP is depicted in Figure 1. As can be seen from the figure, initially, the ants randomly choose the starting point for its solution. Then, the ants use the pheromone values and the distances between the starting point and the candidate points to select the next point for its solution.

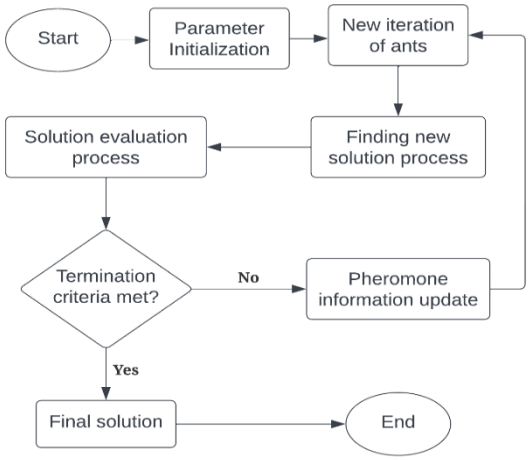

Figure 1 Flowchart of original ACO algorithm [32]

In addition to solving TSP problems, the ACO algorithm was also developed to solve other optimization problems, including the task allocation problem in multi-agent systems [21], [23]. The task allocation problem in a multi-agent system is defined as an optimization problem to find the best agent coalition to complete a target. In this scenario, a target requires one or more capabilities of agent(s) to be completed, and an agent has one or more distinct capabilities. Agent capabilities are represented in a multi-agent capability matrix, *MGK*, where each row element $g_i^{k_j}$ indicates whether agent g_i has capability k_j or not. The value of $g_i^{k_j}$ indicates the weight of capability k_j owned by agent g_i. For example, $g_i^{k_j} \in [0,10]$ and $MGK = \begin{bmatrix} 6 & 0 & 1 \\ 3 & 8 & 10 \end{bmatrix}$ indicates that there are two agents and three capabilities whose capability weight values are between zero to ten. The elements in the first line of *MGK* matrix contain information on the capability weights of the first agent, which correspond to capabilities 1, 2, and 3 [21], *i.e.* agent g_1 has capability k_1 with a value of six, capability k_2 with a value of zero, and capability k_3 with a value of three.

Similar to TSP, the selection of agents' coalition in a multi-agent system can be viewed as a graph trajectory search problem. For example, assume that there are V agents in a multi-agent system: $G=\{g_1, g_2, g_3, \ldots, g_V\}$. A target w requires R capabilities to be completed: $K_w=\{k_1, k_2, k_3, \ldots, k_R\}$. The group of agents that have capability k_r to complete w is denoted as $G_w^{k_r}$. Since w requires R capabilities to be completed, there will be R groups of agents. From each group of agents, one agent $g_i \in G_w^{k_r}$ is then selected with $i \in [1, V]$ and $r = \{1, 2, 3, \ldots, R\}$. To find the solution using ACO approach, each agent g_i in each group of agents $G_w^{k_r}$ is represented as a *node* and the connecting path between agents in different groups is represented as the *edge*. The task allocation process is illustrated in Figure 2.

DOI: https://doi.org/10.29207/resti.v6i6.4201

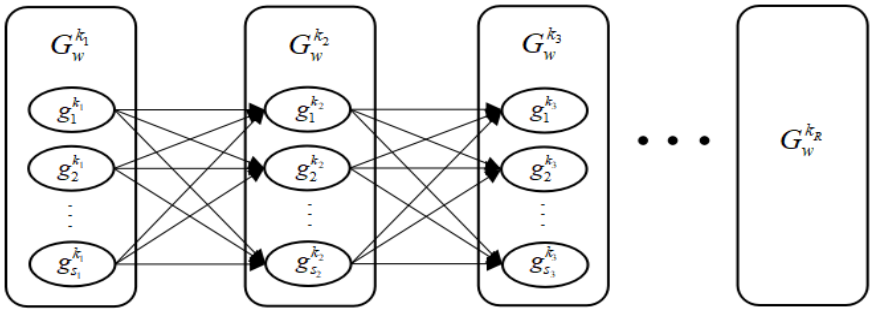

Figure 2. Illustration of a multi-agent system task allocation problem where s_r is the number of agents with capabilities k_r (r={1,2,,…,R}) [21]

Wang [21] modified the basic ACO algorithm to solve the task allocation problem in multi-agent systems, *i.e.* the problem of resource allocation in cloud computing. The algorithm is called the Collective Path Ant Colony Optimization (CPACO) algorithm [21]. In [21], the problem occurs in a dynamic and rapidly changing environment so that the chosen coordination type is the decentralized coordination. Here, the coordination process is distributed among all agents so that the agents need to communicate with each other in determining the best agent coalition. Thus, to produce the optimal system performance, modifications are carried out by adding the weight of agents' capabilities and the communication cost between agents [21].

The study by Sriatun [23] further developed the CPACO algorithm to solve the task allocation problem of a multi-agent system in a landslide disaster scenario. In this scenario, the victim (target) must be rescued by a multi-robot system. To produce the best agents (robots) coalition, the CPACO algorithm is modified by adding a travel cost factor, *i.e.* the distance between the chosen agents/robots and the target. Thus, the modified algorithm in [23] considers not only the weight of agents' capabilities and the communication cost between agents, but also travel cost between the agents and the target. The modified CPACO algorithm by Sriatun [23] is hereinafter called the CPACO-S algorithm.

In the CPACO and CPACO-S algorithms, the ants in the colony perform a solution-finding process based on the transition probability as in any general ant algorithms. The m^{th} ant will move from agent g_i to agent g_j at time t with the probability calculated by Equation (1) as follows [21], [23]:

$$p_{ij}(t) = \begin{cases} \dfrac{[\tau_{ij}(t)]^{\alpha}[\eta_{ij}(t)]^{\beta}}{\sum_{h\in\mathbf{N}}[\tau_{ij}(t)]^{\alpha}[\eta_{ij}(t)]^{\beta}}, & h \in \mathbf{N}, \\ 0, & h \notin \mathbf{N} \end{cases} \quad (1)$$

where $\tau_{ij}(t)$ is the value of pheromone at the path *i-j* from agent g_i to agent g_j at time t, and α is a parameter to adjust the effect of pheromone value τ ($\alpha \geq 0$). The notation $\eta_{ij}(t)$ is a heuristic function that represents the feasibility for the transition from agent g_i to agent g_j at time t based on some known information, and β is a parameter to set the effect of the feasibility value η ($\beta \geq 1$). In Equation (1), h is a member of the set **N** which contains all the agents that belongs to the next group of agents, *i.e.* agents that have one same capability that is still required to be fulfilled for completing the target. In other words, the group of agents for capabilities that have not been "visited" by the m^{th} ant. At the beginning of the iteration, the initial pheromone value is defined as $\tau_0 = 1/aL_{agen}$ where a is the number of nodes (agents) and L_{agen} is the total distance of each agent to all other agents.

The CPACO algorithm modified the heuristic function $\eta_{ij}(t)$ of the basic ACO algorithm by using the weighted capabilities of each agent in the numerator and the communication costs in the denominator as depicted in Equation (2) [21]:

$$\eta_{ij}(t) = \frac{\omega_a^1 g_i^{k_a} + \omega_b^1 g_j^{k_b}}{\omega^2 d_{ij}}. \quad (2)$$

Here, $\omega_r^1 \in [0,1]$, which is the weight of the capability factor to represent the importance of the capability k_r which required for target w. The value of communication cost, $\omega^2 d_{ij}$, is proportional to the distance between agent g_i and agent g_j (d_{ij}) with a communication weight factor (ω^2).

In CPACO-S algorithm, the heuristic function in (2) is further developed by adding the travel cost from agents to target as shown in Equation (3) [23]:

$$\eta_{ij}(t) = \frac{\omega_a^1 g_i^{k_a} + \omega_b^1 g_j^{k_b}}{\omega^2 d_{ij} + \omega^2 d_{iw} + \omega^2 d_{jw}} \quad (3)$$

Here, the travel cost is proportional to the distance between agent g_i and agent g_j to the target w (d_{iw}, d_{jw}) with a factor of ω^2.

In both CPACO and CPACO-S, an iteration is completed when all ants have reached the last group of agents, that is, the last capability required by the target. Then, the best agent coalition is determined by evaluating the efficiency values of all candidate solutions formed by each ant. The efficiency value for the CPACO algorithm is shown in Equation (4) as follows [21]:

DOI: https://doi.org/10.29207/resti.v6i6.4201

$$\varepsilon_m = \frac{\sum_{r=1}^{R} \omega_r^1 g_i^{kr}}{\sum_{i=1}^{R-1} \omega^2 d_{ij}}, j = i + 1. \quad (4)$$

The numerator in Equation (4) is the sum of the weighted capabilities of all selected agents as the candidate solution by the m^{th} ant. The denominator in Equation (4) is the sum of the communication costs between agents in the same candidate solution. Note that the target requires *R* number of capabilities to be completed, which corresponds to the number of agents in the candidate solution (number of agents in candidate solution set <= *R*).

In the CPACO-S algorithm, the efficiency value of the CPACO algorithm is modified by adding the travel cost between the agents in the candidate solution and the target, as shown in Equation (5) [23]:

$$\varepsilon_m = \frac{\sum_{r=1}^{R} \omega_r^1 g_i^{kr}}{\sum_{i=1}^{R-1} \omega^2 d_{ij} + \sum_{i=1}^{R} \omega^3 d_{iw}}, j = i + 1. \quad (5)$$

The numerator in Equation (5) is the sum of the weighted capabilities of all agents in the candidate solution by the m^{th} ant. The denominator in Equation (5) is the sum of the total communication costs between agents and the total travel costs between agents in the candidate solution and the target.

In both CPACO and CPACO-S, the efficiency values of all candidate solutions in the colony are calculated at the end of an iteration. The highest efficiency value in each iteration corresponds to the best agent coalition in that particular iteration. This efficiency value is then used as the basis for updating the best efficiency value and the best agent coalition from all iterations.

As in other general ant algorithms, in addition to updating information about the best efficiency value and the best agent coalition at the end of the iteration, the CPACO and CPACO-S algorithms also update the pheromone value. This value is updated at the end of each iteration to increase the efficiency of the solution search process. CPACO and CPACO-S algorithms use the same equation as used in other general ant algorithms to update the pheromone value, as written in Equation (6) [21], [23]:

$$\tau_{ij}^m(t+1) = (1-\rho)\tau_{ij}^m(t) + \sum \Delta\tau_{ij}^m(t). \quad (6)$$

In Equation (6), the pheromone value of the path *i-j* in the ant algorithm consists of the pheromone evaporation value, which is influenced by the degree of evaporation (ρ), and the accumulated values of pheromone addition ($\Delta\tau_{ij}(t)$) [26]. To calculate $\Delta\tau_{ij}(t)$, the value of ε_m is used as written in Equation (7) [21], [23]:

$$\Delta\tau_{ij}^m(t) = \begin{cases} Q\varepsilon_m, & \text{if } m^{th} \text{ ant go from point } i \text{ to } j \\ 0, & \text{otherwise} \end{cases}. \quad (7)$$

In Equation (7), *Q* is a constant value to determine the strength of the pheromone and ε_m is the efficiency value of ant m^{th}'s route. After updating the best efficiency value, the best agent coalition and the pheromone value, the algorithm will then proceed to the next iteration. The iterations are repeated again until the algorithm termination criteria have been met. When the termination criteria is met, a candidate solution from the ants with the best ε_m value is then selected as the best agent coalition to complete the target. Note that both the CPACO and the CPACO-S algorithms consider only a single target.

2.2 Developing ACO for Task Allocation Problems in Multi-Agent Systems for Multi-Target

This study carried out four main stages. These stages are problem identification, ACO algorithm modification, simulations as well as analysis and evaluation. The flowchart of the four stages that we carried out in this study is shown in Figure 3.

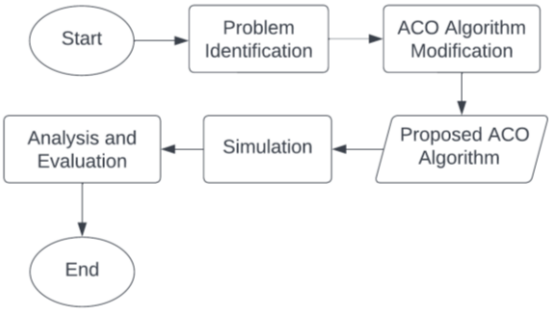

Figure 3. Flowchart of the research stages

2.2.1 Identification of Task Allocation Problem in Multi-Agent System with Multi-Target

The following assumptions were used in this study. The problem that we consider involves several heterogeneous agents that work together to complete tasks which require certain capabilities. The coordination between agents is assumed to be centralized. The process of determining the allocation of agents to task is carried out by a server or a central control system. It is assumed that the agents communicate with each other during the task completion process and therefore we consider the communication costs. The communication cost factor was also used to determine the best agent coalition in the CPACO and CPACO-S algorithms.

In addition to the communication cost between agents, we also consider the distance between the agent and the target. However, in contrast to CPACO-S, which uses the weight of the communication factor ω^2 to determine the travel cost, we consider an additional variable ω^3, which is the weight of agents' transition factor. This variable is employed to calculate the travel cost because the agents' movement towards the target during the task completion process may be influenced by other factors besides the distance between the agent and the target. Note that the weight of the agent transition factor is

DOI: https://doi.org/10.29207/resti.v6i6.4201

different from the weight of the communication factor, so it is necessary to assign a different variable.

Variables used in this study for task allocation problem are as follows:

- A set of agents in a multi-agent system: There are a number of *V* agents in a multi-agent system which is denoted as $G = \{g_1, g_2, g_3, ..., g_V\}$.
- Targets: There are a number of Z targets in a multi-agent system environment. A set of targets is written as $W = \{w_1, w_2, w_3, ..., w_z, ..., w_Z\}$.
- Capabilities to complete tasks: Each target requires a number of $R_z = R_1, R_2, \ldots, R_Z$ capabilities. The set of capabilities required for target w_z is denoted as K_{w_z} containing the capabilities k_r where *r* is the index of capabilities.
- Group of Agents: A group of agents with the capability k_r required for target w_z is denoted by $G_{w_z}^{k_r}$.
- Agent coalition: A coalition of agents for target w_z is denoted as G_{w_z}.
- A set of agent coalitions for all targets is denoted as: $G_w = \{G_{w_1}, G_{w_2}, ..., G_{w_Z}\}$.
- The collection of the best agents' coalition that is the solution to the task allocation problem in a multi-agent system with multi-targets is denoted as G_w^{best}.

A problem that arises when there is more than one target in a task allocation problem is the problem of overlapping agents, i.e. when the same agent(s) are selected to an agents coalition for different targets. For example, assume that there are two targets w_1 and w_2 that require some capabilities to complete the task, i.e. $K_{w_1}=\{k_1, k_3, k_7\}$ and $K_{w_2}=\{k_1, k_2, k_7\}$. The agents in the multi-agent system are then grouped according to the capabilities required by each target. For target w_1, the agent group formed for example are $G_{w_1}^{k_1}=\{g_2,g_5,g_6,g_8\}$, $G_{w_1}^{k_3}=\{g_1, g_2, g_4,g_7,g_8\}$ and $G_{w_1}^{k_7}=\{g_5,g_7,g_8, g_9,g_{10}\}$. For target w_2, the agent group formed are $G_{w_2}^{k_1}=\{g_2,g_5,g_6,g_8\}$, $G_{w_2}^{k_2}=\{g_3, g_5,g_6,g_8,g_{10}\}$, and $G_{w_2}^{k_7}=\{g_5,g_7,g_8, g_9,g_{10}\}$. From each group of agents, if only the weight of the agents' capability and task completion cost are considered as in the CPACO-S algorithm, then the best solution obtained are $G_{w_1}=\{g_2,g_2,g_5\}$ and $G_{w_2}=\{g_2,g_3,g_5\}$. We can see that from those agent coalitions for the two targets, agent g_2 and g_5 are selected for both targets. This condition is called overlapping agents, which has to be minimized so that the task completion process on all targets can be carried out in the shortest possible time. In order to search for a suitable solution, an objective function is added, which is used to minimize overlapping agents.

In this study, the objective functions are designed to: (1) minimize the task completion costs, (2) maximize the system capabilities, and (3) minimize overlapping agents. The task allocation problem that we consider can, thus, be considered as a multi-objective optimization problem. All objective functions must be optimized simultaneously to obtain the best solution. The final solution is represented in the form of agents' coalition that optimizes all objective functions.

2.2.2 ACO Algorithm Modification for Task Allocation Problem in Multi-Agents System with Muti-Targets

The basic ACO algorithm in this study is modified to solve the task allocation problem in a multi-agent system with multi-target. The proposed model is then referred as the Modified ACO Model. Generally, there are five elements which are specified to define a suitable ant algorithm for different optimization problem [28].

The first element is constructing a candidate solution. As mentioned earlier, the final solution for the task allocation that we consider is to form an agent coalition for each target that optimizes all objective functions. To minimize overlapping agents, finding the final solution is done by finding a solution for each target. Once the solution for one target is obtained, we calculate the best solution for the next target. Therefore, the number of ant colonies used to construct candidate solutions is as much as the number of targets. Each ant colony seeks a solution for a target by forming a candidate solution for the corresponding target.

The second element is the heuristic function. In this study, more than one heuristic function is used to determine the visibility value of the ant transition. Each heuristic function is affected by a different objective function. The first heuristic function is influenced by the objective function of minimizing task completion costs consisting of communication costs and travel costs as used in the CPACO-S algorithm. The heuristic function of the CPACO-S algorithm is modified to maximize the capability of the system by increasing the probability of selecting the same agent, as defined in Equation (8),

$$\eta_{ij1}^{z}(t) = \begin{cases} \dfrac{\omega_a^1 g_i^{k_a} + \omega_b^1 g_j^{k_b}}{\omega^2 d_{ij} + \omega^3 d_{iw_z}}, & g_i = g_j \\ \dfrac{\omega_a^1 g_i^{k_a} + \omega_b^1 g_j^{k_b}}{\omega^2 d_{ij} + \omega^3 d_{iw_z} + \omega^3 d_{jw_z}}, & g_i \neq g_j \end{cases} . \quad (8)$$

The numerator of Equation (8) is the sum of the capabilities possessed by agent g_i and agent g_j on two adjacent agent groups ($k_a, k_b \in K_{w_z}$), with ω_a^1 and ω_b^1 being the capability weights of k_a and k_b. The denominator in Equation (8) is the sum of communication costs and travel costs.

To increase the probability of choosing the same agent in the same group, the value of the travel cost for the same agent is only calculated once. For example, a target

DOI: https://doi.org/10.29207/resti.v6i6.4201

w_1 requires four capabilities, and one of the candidate solutions is $G_{w_1}=\{g_2,g_2,g_2,g_5\}$, then the travel cost for agent g_2 will be counted only once.

Note that for the same agent, the value of the communication cost is considered zero because the distance between the agent and itself is zero. For different agents, the heuristic function used is the same as the heuristic function in the CPACO-S algorithm, with modifications to the variables used to calculate the travel cost.

In CPACO-S, the variable used for the travel cost is the same variable for the weight of the communication factor (ω^2). In the Modified ACO Model, a new variable is used for travel costs, which is the weight of the agent transfer factor (ω^3). In this study, a second heuristic function is added to minimize overlapping agents, which is defined as follows:

$$\eta_{ij2}^{z}(t)=\frac{1}{\Psi^{\sum\psi_{jw_y}}},\psi_{jw_y}=\begin{cases}1, & g_j\in G_{w_y},(y\neq z)\\ 0, & otherwise\end{cases}, \quad (9)$$

where variables Ψ and ψ in Equation (9) are variables to determine the value of the overlapping value. The value of the variable Ψ is chosen so that the total denominator becomes relatively large to reduce the visibility value of the agent chosen in the coalition of other targets. The value of the variable Ψ is selected based on the $\eta_{ij1}^{z}(t)$ range of values. For example, if the $\eta_{ij1}^{z}(t)$ range of values is 10^{-1} – 10^{3}, then the chosen value for Ψ is 10^{4}. In addition, variable ψ_{jw_y} is used to consider the selection of agent g_j in other targets. Its value increases when agent g_j is chosen more often in other targets w_y with $y = \{1,2,\cdots,Z\}$ and $y \neq z$. Thus, the visibility value is getting smaller and allows the selection of other agents whose visibility value is relatively small but have never been chosen or are less chosen in the agent coalition for other targets.

The total ant route transition visibility from agent g_i to agent g_j for target w_z at time t ($\eta_{ij}^{z}(t)$) is calculated as follows:

$$\eta_{ij}^{z}(t)=\eta_{ij1}^{z}(t)\times\eta_{ij2}^{z}(t). \quad (10)$$

The third element is the efficiency function which is a function to measure how good a solution is. As mentioned earlier, the process of finding a solution for each target is done one-by-one. The best solution for each target is calculated using the efficiency function that we refer as the local efficiency function.

The local efficiency function is determined based on the objective function to be optimized. As there is more than one objective function that we consider, we use more than one local efficiency function to determine the best agent coalition for a target. The first local efficiency function is determined by the objective function to minimize the task completion cost as employed in the CPACO-S algorithm.

To maximize the overall capabilities of the system, the total task completion cost is calculated by considering the travel and communication costs. If an agent is selected to complete multiple capabilities, that agent is only listed once in the set of the candidate solution. The first local efficiency function is defined as follows:

$$\varepsilon_{m1}^{z}(t)=\frac{\sum_{r=1}^{R_z}\omega_r^1 g_i^{k_r}}{\sum_{i=1}^{R_z-1}\omega^2 d_{ij}+\sum_{i=1}^{R_z}\omega^3 d_{iw}},\ j=i+1. \quad (11)$$

The numerator of Equation (11) is the total number of agent capabilities of the candidate solution chosen by the m-th ant for the w_z target.

The numerator is calculated based on all capabilities k_r that are members of the capability set required to complete the task on target w_z ($\nabla k_r \in K_{w_z}$).

The denominator of Equation (11) is the total task completion cost for the target w_z of the candidate solution formed by the m-th ant by considering different agents.

The communication cost ($\omega^2 d_{ij}$) is calculated based on the total distance between agents $g_i \in G_{w_z}^{k_r}$ for $\forall k_r \in K_{w_z}$. The travel cost ($\omega^3 d_{iw}$) is calculated based on the distance between all agents $g_i \in G_{w_z}^{k_r}$ for $\forall k_r \in K_{w_z}$ to target w_z . The additional condition that if there is one agent selected for multiple capabilities, the travel cost for that agent will only be counted once.

The next objective function is designed to minimize overlapping agents. This function is defined as the second local efficiency function as follows:

$$\varepsilon_{m2}^{z}=\frac{1}{\Psi^{\sum\psi_{jw_y}}},\psi_{jw_y}=\begin{cases}1, & g_j\in G_{w_y},(y\neq z)\\ 0, & otherwise\end{cases}, \quad (12)$$

where ε_{m2}^{z} is influenced by the value of variable Ψ ; see Equation (9). When candidate solution of the m-th ant is consisted of the same agents which are selected for several different targets, the value of ε_{m2}^{z} becomes smaller.

Based on the two efficiency functions, the total local efficiency value of the m-th ant for target w_z is calculated as: $\varepsilon_m^z=\varepsilon_{m1}^z\times\varepsilon_{m2}^z$. Then after the candidate solution for the w_z target has been formed for all ants in a colony, the best solution candidate is selected based on the biggest ε_m^z value, which is the best local efficiency value for the w_z target (ε_{best}^z).

In addition to the local efficiency values, a global efficiency value (ε_{global}) is used to determine the best overall solution. The ε_{best}^z values of all targets are added, which then produce ε_{global} value in each iteration.

DOI: https://doi.org/10.29207/resti.v6i6.4201

The value of ε_{global} is then compared at the end of each iteration to obtain the best overall efficiency value (ε_{best}).The fourth element is the probability of ant transition, which is influenced by the ant transition visibility value (heuristic function) and the value of the pheromones. In this study, the probability of ants *m*-th movement from agent g_i to agent g_j in the process of finding solution for the target w_z at time t $p_{ij}^{z}(t)$ is calculated using Equation (1) with the value of visibility $\eta_{ij}^{z}(t)$ and the pheromone value $\tau_{ij}^{z}(t)$.

The last element is the pheromone update rule. Each ant colony uses a different pheromone matrix to store the accumulated pheromone values to avoid selecting the same agent for different targets. The equation used to update the pheromone value on the path between agent g_i to agent g_j for target w_z ($\tau_{ij}^{z}(t+1)$) is the same as CPACO and the CPACO-S algorithms in Equation (6) by considering the pheromone value on the path to find solution for target w_z ($\tau_{ij}^{z}(t)$). In this study, the pheromone update rule is applied on the ant paths that are formed by only the best agent coalition for target w_z. If the ant in the best agent coalition for target w_z moves from agent g_i to agent g_j, the amount of pheromone deposited at time t is calculated as $\Delta\tau_{ij}^{z}(t) = Q\varepsilon_{best}^{z}$; otherwise, the value is zero.

From the description of each element in the ACO algorithm developed in this study, it can be concluded that the modifications are mainly carried out on the heuristic and efficiency functions. In summary, the pseudocode of the proposed Modified ACO is shown in Figure 4.

Pseudocode of Modified ACO Model Algorithm

Input: number of ants *M*, number of agents *a*, agents' capability k_r, agents' position [x,y], number of targets *Z*, capability collection needed of each target K_z, target's position [x,y], group of agents $G_{w_z}^{k_r}$ (based on capability needed of each target), initial value $\varepsilon_{best}^{z} = 0$ and $\varepsilon_{best} = 0$, initial pheromone value τ_0, ants colony parameter value (Q, α, β, ρ, ω_r^1, ω^2, ω^3) new heuristic constant value Ψ and number of maximum iteration *IterMax*.

For *iter* in range *IterMax*:
  Define *chosen_agent* = [ ];
  Define $\varepsilon_{global} = 0$;
  For *z* th target in range *Z*:
    For *m* th ant in range *M*:
      Choose an agent g_i from $G_{w_z}^{k_r}$ randomly;
      Write the choosen agent g_i in *colony.ant*[*m*].*tour*[1];
      For *r*+1 th in K_z
        Choose an agent g_j from $G_{w_z}^{k_r}$ based on $p_{ij}^{z}(t)$ using $\eta_{ij}^{z}(t)$ (considering *chosen_agent*) and $\tau_{ij}^{z}(t)$;
        Write the chosen agent g_j in *colony.ant*[*m*].*tour*[*c*];
      End for
      Calculate the candidate solution local efficiency using ε_m^z (considering *chosen_agent*);
      If $\varepsilon_m^z > \varepsilon_{best}^z$
        $\varepsilon_{best}^z = \varepsilon_m^z$;
        G_{w_z}= *colony.ant*[*m*].*tour*;
      End if
      chosen_agent ← G_{w_z};
      $\varepsilon_{global} = \varepsilon_{global} + \varepsilon_{best}^z$;
      Update pheromone in ant colony for target w_z using $\tau_{ij}^{z}(t+1)$;
    End for
  End for
  If $\varepsilon_{global} > \varepsilon_{best}$
    $\varepsilon_{best} = \varepsilon_{global}$
    G_w^{best} = *chosen_agent*;
  End if
End for
Output: G_w^{best} in the last iteration.

Figure 1 Pseudocode for Modified ACO Model algorithm

The final solution generated by the Modified ACO Model is an agent coalition from all targets that produce the ε_{best}. The process of finding the final solution is carried out one by one for each target. When an agent coalition is define for a target, the results affect the process of selecting the agent coalition for the next target. Thus, it is possible that the sequence of finding solutions affects the selection of an agent coalition for all targets. Testing needs to be carried out to determine the effect of finding solutions sequences on existing targets so that the best Modified ACO Model is achieved to solve the task allocation problem of multi-agent systems with multi-target.

2.2.3 Simulation

Simulations were carried out using Matlab R2015a software to test the performance of the Modified ACO Model in solving the multi-agent-multi-target task allocation problem. In the simulation, the problem was generated in a two-dimensional area with $0 \le x \le 50$ and $0 \le y \le 50$. The multi-agent system consists of ten heterogeneous agents with ten types of capabilities. To simplify the simulation, an agents' capability is represented in binary number, *i.e.* 1 represents that the agent has a certain capability whereas 0 represents that the agent has no capability of a certain type. With this binary weighting system, the heuristic function $\eta_{ijl}^{z}(t)$ in Equation (8) and the local efficiency function $\varepsilon_{ml}^{z}(t)$ in

DOI: https://doi.org/10.29207/resti.v6i6.4201

Equation (11) can be simplified into Equation (13) and Equation (14).

$$\eta_{ijl}^{z}(t) = \begin{cases} \frac{\omega_a^1+\omega_b^1}{\omega^3 d_{iw_z}}, & g_i = g_j \\ \frac{\omega_a^1+\omega_b^1}{\omega^2 d_{ij}+\omega^3 d_{iw_z}+\omega^3 d_{jw_z}}, & g_i \neq g_j \end{cases}, \quad (13)$$

$$\varepsilon_{ml}^{z}(t) = \frac{\sum_{r=1}^{R} \omega_r^1}{\sum_{i=1}^{R-1} \omega^2 d_{ij} + \sum_{i=1}^{R} \omega^3 d_{iw_z}}, \; j = i+1. \quad (14)$$

Here, the value for each agents' capability ($g_i^{k_a}, g_i^{k_b}, g_i^{k_r}$ for $r = \{1,2,\ldots,R\}$) in Equation (8) and Equation (9) is equal to one. Therefore, in Equation (13) and Equation (14), we need to only consider the importance of the agents' capability in solving the target, *i.e.* ω_a^1, ω_b^1, ω_r^1 for $r = \{1,2,\ldots,R\}$.

In the simulation, information on the agents' capabilities is written in a *MGK* capability matrix, as shown in Table 1. Meanwhile, the information on the position of each agent is shown in Table 2.

Table 1. Multi-agent capability matrix MGK

Agent	Capability									
	k_1	k_2	k_3	k_4	k_5	k_6	k_7	k_8	k_9	k_{10}
g_1	0	0	1	0	1	0	0	1	0	1
g_2	1	0	1	0	0	1	0	0	1	1
g_3	0	1	0	0	0	0	0	1	1	1
g_4	0	0	1	0	0	1	0	0	0	0
g_5	1	1	0	0	1	0	1	0	0	0
g_6	1	1	0	0	1	0	0	0	1	1
g_7	0	0	1	1	1	0	1	1	0	0
g_8	1	1	1	1	0	0	1	1	0	0
g_9	0	0	0	1	0	1	1	1	1	0
g_{10}	0	1	0	1	0	0	1	0	0	0

Table 2. Positions of agents

Agent	Position	
	x	y
g_1	10	35
g_2	20	50
g_3	30	50
g_4	40	50
g_5	50	35
g_6	50	15
g_7	40	0
g_8	30	0
g_9	20	0
g_{10}	10	15

Two simulations were carried out:

1. Simulation to analyze the process of finding the optimum solution using the proposed Modified ACO Model, and
2. Simulation to evaluate the proposed Modified ACO Model and its comparison to the benchmark algorithm, i.e. CPACO and CPACO-S.

In the proposed Modified ACO Model, the optimum solution for each target is calculated iteratively, starting from the first target, the second target, and so on. Therefore, the determination of target sequence may affect the selection of agents' coalitions. To analyze this issue, in the first simulation, the target sequence was determined using two approaches: (1) based on initial information and (2) randomly. Four tests were carried out for each approach with different number of targets, *i.e.* 5, 10, 15 and 20. Each test was simulated 30 times with random target combination.

In the second simulation, evaluation to the proposed Modified ACO Model was done by comparing this algorithm to a benchmark algorithm, namely the CPACO-S [23], which is proven to be more efficient than the original ACO algorithm and the CPACO algorithm. For each algorithm, six tests were conducted by varying the number of targets, *i.e.* 3, 5, 8, 10, 15, and 20. Each test was simulated 30 times with random target combination.

In the first and second simulations, the targets were taken randomly from the following data:

1. The data contains 30 targets with different positions and capability requirements to be completed.
2. Target positions were within the simulation area, *i.e.* $0 \leq x \leq 50$ and $0 \leq y \leq 50$.
3. None of the targets was in the exact same location as other targets or agents in the simulation area.
4. Each target requires more than one capability to be completed and there might be more than one target with the same capability requirement.

2.2.4 Analysis dan Evaluation

Two algorithmic performance evaluation metrics were utilized to analyse the simulation results, those ares:

1. The total value of the task completion cost, which is the sum of the communication costs and travel costs of the agent coalitions of all targets.

   Note that the agent coalition on a target contains all the agents selected to complete the target, and that if there is one agent selected for multiple capabilities, the travel cost for that agent will only be counted once. The total task completion cost (Σtcc) is calculated using Equation (15):

   $$\sum tcc = \sum_{w_z \in W} \sum_{g_i \in G_{w_z}} (\omega^2 d_{ij} + \omega^3 d_{iw_z}), \; j = i+1, \quad (15)$$

   where W is the set of all targets to be solved and G_{w_z} is the best agent coalition for target w_z. In Equation (15), $\omega^2 d_{ij}$ represent the communication cost between two consecutive agents g_i and g_j, which is proportional to the distance between agent g_i and agent g_j (d_{ij}) with a communication weight factor (ω^2). Meanwhile, $\omega^3 d_{iw_z}$ represents the travel cost, which is proportional to the distance between agent g_i and target w_z with a factor of ω^3.
2. The overlapping value (ϑ).

DOI: https://doi.org/10.29207/resti.v6i6.4201

This evaluation metric is utilized to determine the algorithm's efficiency in minimizing the overlapping agent. The overlapping value (ϑ) is calculated using Equation (16):

$$\vartheta = z_{max}^{g} \times \frac{\sum target\ with\ overlapping\ agents}{total\ number\ of\ targets}, \quad (16)$$

where z_{max}^{g} refers to the maximum number of targets that select the same agent in the system. For example, if agent g_i is selected by three targets, agent g_j is selected by five targets, and other agents are selected by one target only, then the value of z_{max}^{g} = 5. This z_{max}^{g} is then multiplied by the ratio between the number of targets with overlapping agents and the total number of targets.

Evaluation of the simulation results was then carried out using the two performance matrices as described in Equation (15) and Equation (16).

3. Results and Discussions

3.1. Simulation 1

In the first simulation, we determined the target sequence for solution finding using two approaches. The first approach is based on the initial target information without any change in the target sequence. In other words, first target would be the first target as listed on the initial data. The second approach is by randomly determine the target sequence for the target solution, which is changed in each iteration. Each approach of determining the target solution search sequence was tested with four different number of targets, *i.e.* 5, 10, 15 and 20. Each test was simulated 30 times with different combinations of targets, taken from the data from 30 targets. The results are shown in Figure 5 and Figure 6, and the summary of the results is shown in Table 3.

Using a 95% confidence interval, Figure 5 and Table 3 show that the total average value of the task completion cost for random target sequencing is significantly lower than initial target sequencing. The total average values of task completion cost by determining the target sequence randomly are 11.08%, 17.41%, 14.28% and 20.84% lower than that of initial target sequence for 5, 10, 15 and 20 targets, respectively. Figure 6 and Table 3 show that random target sequencing returns slightly lower overlapping values compared to the initial target sequencing. As a conclusion, random target sequence determination is superior to target sequence determination based on initial information. These results also show that random target sequence determination allows better solution search so that a more optimum agent allocation can be found.

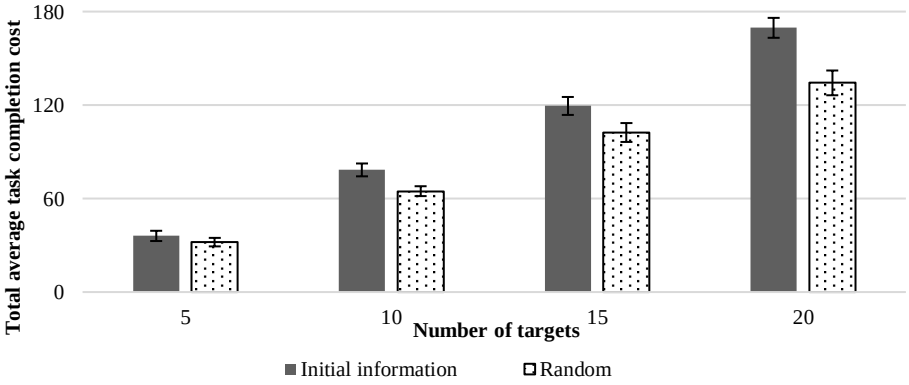

Figure 2. Results of Simulation 1: total average task completion cost

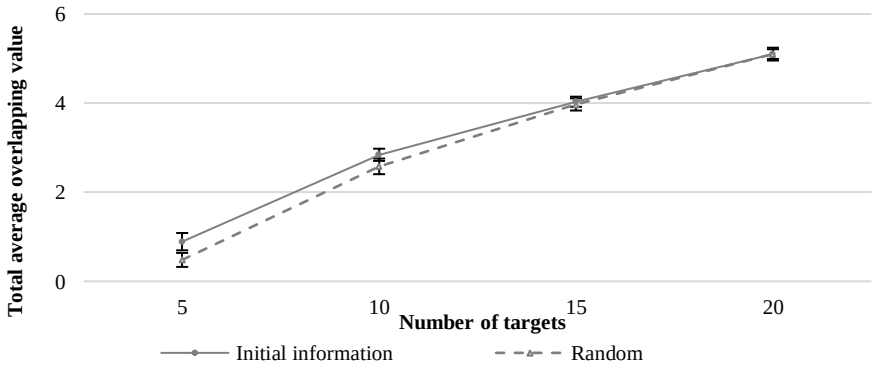

Figure 3. Results of Simulation 1: total average overlapping value

DOI: https://doi.org/10.29207/resti.v6i6.4201

Table 3. Summary of simulation 1 test results

Number of targets	Total average task completion cost		Total average overlapping value	
	Target sequence based on initial information	Random target sequence	Target sequence based on initial information	Random target sequence
5	36.03 ± 3.29	32.03 ± 2.74	0.89 ± 0.19	0.48 ± 0.16
10	78.39 ± 4.13	64.74 ± 3.17	2.84 ± 0.14	2.58 ± 0.18
15	119.44 ± 5.76	102.39 ± 6.07	4.03 ± 0.11	4.03 ± 0.14
20	169.57 ± 6.39	134.24 ± 7.94	5.10 ± 0.11	5.10 ± 0.14

3.2. Simulation 2 Scenario

In the second simulation, the proposed Modified ACO Model is compared with CPACO-S algorithm which has been proven to be superior to the basic ACO algorithm and the CPACO algorithm in terms of the efficiency of the resulting agent coalition [23]. Six tests were conducted for each algorithm using different number of targets, *i.e.* 3, 5, 8, 10, 15, and 20. The results are shown in Figure 7 and Figure 8. The summary of the test results is shown in Table 4.

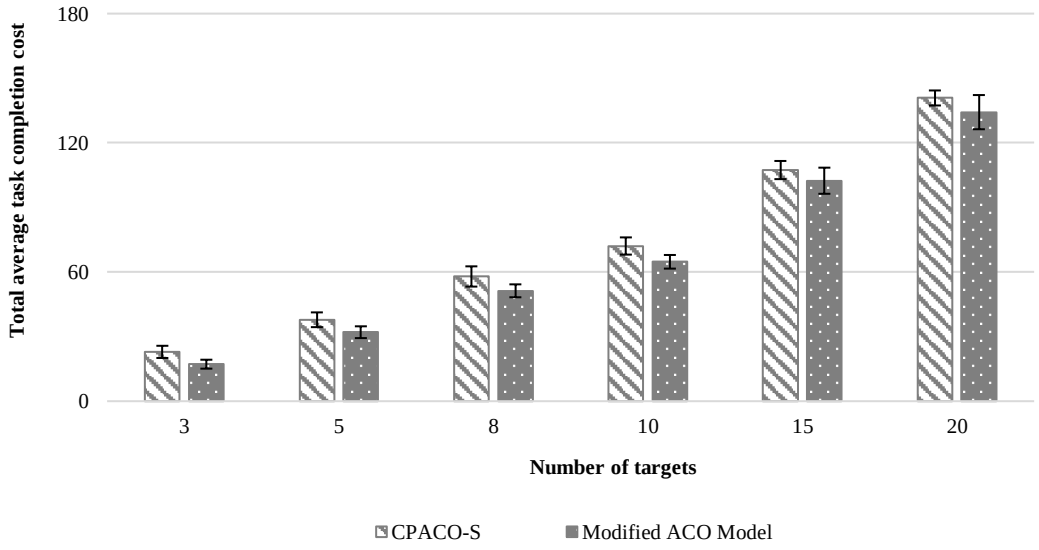

Figure 4. Results of simulation 2: total average task completion cost

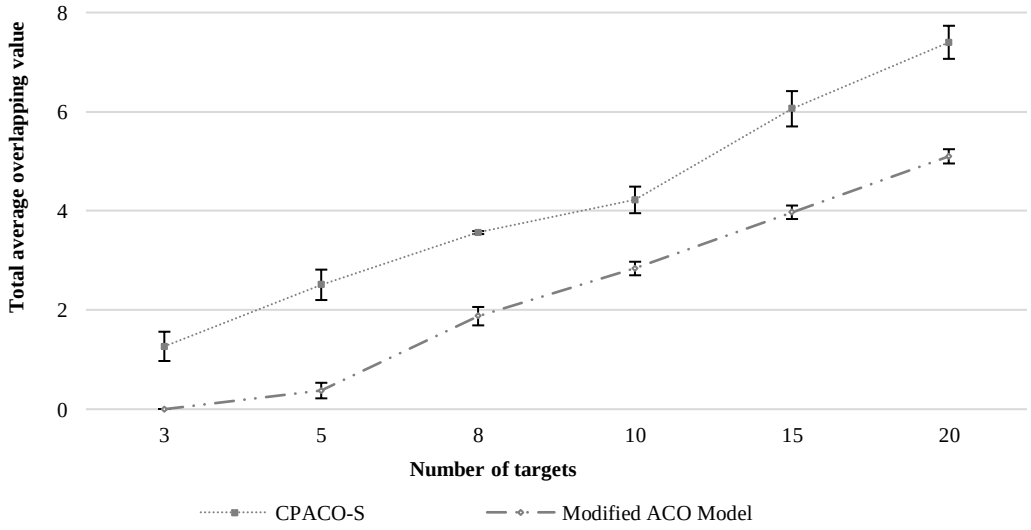

Figure 5. Results of simulation 2: total average overlapping value

DOI: https://doi.org/10.29207/resti.v6i6.4201

Table 4. Summary of simulation 2 test results

Number of targets	Total average task completion cost		Total average overlapping value	
	CPACO-S	Modified ACO Model	CPACO-S	Modified ACO Model
3	22.90 ± 2.83	17.19 ± 2.08	1.27 ± 0.29	0.00
5	37.82 ± 3.42	32.03 ± 2.74	2.51 ± 0.31	0.38 ± 0.16
8	57.93 ± 4.70	51.27 ± 3.00	3.56 ± 0.03	1.88 ± 0.19
10	72.07 ± 4.00	64.74 ± 3.17	4.22 ± 0.27	2.84 ± 0.14
15	107.34 ± 4.22	102.39 ± 6.07	6.06 ± 0.36	3.97 ± 0.14
20	140.82 ± 3.49	134.24 ± 7.94	7.40 ± 0.33	5.10 ± 0.14

Figure 7 and Figure 8 show that the Modified ACO Model in this study has a better performance than the CPACO-S algorithm in terms of the total average task completion cost and overlapping values. At 95% confidence interval, Table 4 shows that the Modified ACO Model produces a significantly lower total average task completion cost in several tests. The superiority of the Modified ACO Model is significant when the number of targets is less than the number of agents. The Modified ACO Model produces less task completion cost compared to the CPACO-S by 24.95%, 15.30%, 11.50%, 10.17%, 4.62% and 4.67% for 3, 5, 8, 10, 15 and 20 targets, respectively. When the number of targets increases, the total average task completion cost of the Modified ACO Model is not significantly better to that of the CPACO-S algorithm. This may be due to the limited number of agents available, so that the allocated agents to a specific target may not be the agents that produce the best task allocation cost. However, since our modifications maximize the agents' capability and minimize agents' overlap, the efficiency of the agent coalition generated by the proposed Modified ACO Model becomes superior to the CPACO-S algorithm.

In this study, ACO algorithm has been modified to minimize the number of overlapping agents, *i.e.* an agent selected by different targets. In addition, modifications were also made to prioritize the selection of the same agent in a agent coalition to minimize the number of agents allocated to a target. Figure 8 proves that the two modifications in the proposed Modified ACO Model have succeeded in minimizing the overlapping agents. It is evident that the Modified ACO Model produced a significantly lower average overlapping value than that of the CPACO-S algorithm at a 95% confidence interval, as also shown in Table 4. This indicates that the Modified ACO Model is more efficient in allocating agents to targets than the CPACO-S algorithm. Overall, it can be concluded that the Modified ACO Model proposed in this study performs better than the CPACO-S algorithm in solving the task allocation problem of multi-agent system with multi-target.

4. Conclusion

This study proposes a Modified ACO Model to solve the task allocation problem in a multi-agent system with multi-target. Modifications are made to minimize the number of overlapping agents, where the same agent is selected to solve some different targets. The simulation results show that the proposed Modified ACO Model has a superior performance in minimizing task completion costs by ±11.87% than the benchmark algorithm (CPACO-S). Furthermore, the simulation results also show that the Modified ACO Model performs well in minimizing overlapping agents with a lower overlapping value of ±55.11% compared to the benchmark algorithm.

Further research can be conducted to evaluate the performance of the proposed Modified ACO Model in comparison with other well-known optimization methods to solve the task allocation problems in multi-agent systems with multi-target. Some benchmark optimization methods may include: (1) the Auction-based method which is inspired by economic system and (2) Genetic Algorithm (GA), which is a bio-inspired based algorithm.

Acknowledgement

The authors would like to thank the Institute for Research and Community Service (LPPM) IPB University for their support in completing this research. Part of this research is funded by the Ministry of Research and Technology of the Republic of Indonesia (RISTEK-BRIN) through the Basic Research Program for Higher Education (PDUPT), research grant number 3626/IT3.L1/PT.01.03/P/B/2022.

Reference

[1] J. Parker, "Task allocation for multi-agent systems in dynamic environments," *12th Int. Conf. Auton. Agents Multiagent Syst. 2013, AAMAS 2013*, vol. 2, pp. 1445–1446, 2013.

[2] A. Khamis, A. Hussein, and A. Elmogy, "Multi-robot Task Allocation: A Review of the State-of-the-Art," *Stud. Comput. Intell.*, vol. 604, pp. 31–51, 2015, doi: 10.1007/978-3-319-18299-5_2.

[3] M. Irfan and A. Farooq, "Auction-based task allocation scheme for dynamic coalition formations in limited robotic swarms

with heterogeneous capabilities," *2016 Int. Conf. Intell. Syst. Eng. ICISE 2016*, pp. 210–215, 2016, doi: 10.1109/INTELSE.2016.7475122.

[4] Y. Miao, L. Zhong, Y. Yin, C. Zou, and Z. Luo, "Research on dynamic task allocation for multiple unmanned aerial vehicles," *Trans. Inst. Meas. Control*, vol. 39, no. 4, pp. 466–474, 2017, doi: 10.1177/0142331217693077.

[5] F. Tang and L. E. Parker, "A complete methodology for generating multi-robot task solutions using ASyMTRe-D and market-based task allocation," *Proc. - IEEE Int. Conf. Robot. Autom.*, no. April, pp. 3351–3358, 2007, doi: 10.1109/ROBOT.2007.363990.

[6] M. K. D. Hardhienata, K. E. Merrick, and V. Ugrinovskii, "Task allocation in multi-agent systems using models of motivation and leadership," *2012 IEEE Congr. Evol. Comput. CEC 2012*, 2012, doi: 10.1109/CEC.2012.6256114.

[7] J. P. Wang, Y. Gu, and X. M. Li, "Multi-robot task allocation based on ant colony algorithm," *J. Comput.*, vol. 7, no. 9, pp. 2160–2167, 2012, doi: 10.4304/jcp.7.9.2160-2167.

[8] G. A. Korsah, A. Stentz, and M. B. Dias, "A comprehensive taxonomy for multi-robot task allocation," *Int. J. Rob. Res.*, vol. 32, no. 12, pp. 1495–1512, 2013, doi: 10.1177/0278364913496484.

[9] H. Liu, P. Zhang, B. Hu, and P. Moore, "A novel approach to task assignment in a cooperative multi-agent design system," *Appl. Intell.*, vol. 43, no. 1, pp. 162–175, 2015, doi: 10.1007/s10489-014-0640-z.

[10] M. Khani, A. Ahmadi, and H. Hajary, "Distributed task allocation in multi-agent environments using cellular learning automata," *Soft Comput.*, vol. 23, no. 4, pp. 1199–1218, 2019, doi: 10.1007/s00500-017-2839-5.

[11] S. S. Chiddarwar and N. R. Babu, "Multi-agent system for off-line coordinated motion planning of multiple industrial robots," *Int. J. Adv. Robot. Syst.*, vol. 8, no. 1, pp. 102–112, 2011, doi: 10.5772/10533.

[12] Yan Jin, A. A. Minai, and M. M. Polycarpou, "Cooperative real-time search and task allocation in UAV teams," in *42nd IEEE International Conference on Decision and Control (IEEE Cat. No.03CH37475)*, 2003, vol. 1, no. December, pp. 7–12. doi: 10.1109/CDC.2003.1272527.

[13] Q. Cheng, D. Yin, J. Yang, and L. Shen, "An Auction-Based Multiple Constraints Task Allocation Algorithm for Multi-UAV System," *Proc. - 2016 Int. Conf. Cybern. Robot. Control. CRC 2016*, pp. 1–5, 2017, doi: 10.1109/CRC.2016.7.

[14] H. A. Kurdi *et al.*, "Autonomous task allocation for multi-UAV systems based on the locust elastic behavior," *Appl. Soft Comput. J.*, vol. 71, pp. 110–126, 2018, doi: 10.1016/j.asoc.2018.06.006.

[15] N. Noguchi and O. C. Barawid, *Robot farming system using multiple robot tractors in Japan agriculture*, vol. 44, no. 1 PART 1. IFAC, 2011. doi: 10.3182/20110828-6-IT-1002.03838.

[16] A. T. J. R. Cobbenhagen, D. J. Antunes, M. J. G. van de Molengraft, and W. P. M. H. Heemels, "Heterogeneous multi-agent resource allocation through multi-bidding with applications to precision agriculture∗," *IFAC-PapersOnLine*, vol. 51, no. 23, pp. 194–199, 2018, doi: 10.1016/j.ifacol.2018.12.034.

[17] M. Davoodi, J. M. Velni, and C. Li, "Coverage control with multiple ground robots for precision agriculture," *Mech. Eng.*, vol. 140, no. 6, pp. 4–8, 2018, doi: 10.1115/1.2018-jun-4.

[18] R. Cao *et al.*, "Task assignment of multiple agricultural machinery cooperation based on improved ant colony algorithm," *Comput. Electron. Agric.*, vol. 182, p. 105993, Mar. 2021, doi: 10.1016/J.COMPAG.2021.105993.

[19] A. Hussein and A. Khamis, "Market-based approach to Multi-robot Task Allocation," in *2013 International Conference on Individual and Collective Behaviors in Robotics (ICBR)*, Dec. 2013, pp. 69–74. doi: 10.1109/ICBR.2013.6729278.

[20] J. Yang and Z. Luo, "Coalition formation mechanism in multi-agent systems based on genetic algorithms," *Appl. Soft Comput.*, vol. 7, no. 2, pp. 561–568, Mar. 2007, doi: 10.1016/j.asoc.2006.04.004.

[21] L. Wang, Z. Wang, S. Hu, and L. Liu, "Ant Colony Optimization for task allocation in Multi-Agent Systems," *China Commun.*, vol. 10, no. 3, pp. 125–132, Mar. 2013, doi: 10.1109/CC.2013.6488841.

[22] P. C. Pendharkar, "An ant colony optimization heuristic for constrained task allocation problem," *J. Comput. Sci.*, vol. 7, pp. 37–47, 2015, doi: 10.1016/j.jocs.2015.01.001.

[23] M. Sriatun, "Modifikasi ant colony optimization untuk menyelesaikan masalah task allocation dalam skenario bencana tanah longsor mamik sriatun," 2019.

[24] H. Cui, X. Liu, T. Yu, H. Zhang, Y. Fang, and Z. Xia, "Cloud Service Scheduling Algorithm Research and Optimization," *Secur. Commun. Networks*, vol. 2017, no. Dc, pp. 1–7, 2017, doi: 10.1155/2017/2503153.

[25] M. Dorigo, V. Maniezzo, and A. Colorni, "Ant System : An Autocatalytic Optimizing Process Technical Report 91-016," *Leonardo*, pp. 1–21, 1991.

[26] J. Li and W. Zhang, "Solution to Multi-objective Optimization of Flow Shop Problem Based on ACO Algorithm," in *2006 International Conference on Computational Intelligence and Security*, Nov. 2006, pp. 417–420. doi: 10.1109/ICCIAS.2006.294166.

[27] I. Alaya, C. Solnon, and K. Ghédira, "Ant colony optimization for multi-objective optimization problems," *Proc. - Int. Conf. Tools with Artif. Intell. ICTAI*, vol. 1, pp. 450–457, 2007, doi: 10.1109/ICTAI.2007.108.

[28] S. K. Chaharsooghi and A. H. Meimand Kermani, "An effective ant colony optimization algorithm (ACO) for multi-objective resource allocation problem (MORAP)," *Appl. Math. Comput.*, vol. 200, no. 1, pp. 167–177, 2008, doi: 10.1016/j.amc.2007.09.070.

[29] B. Yagmahan and M. M. Yenisey, "Ant colony optimization for multi-objective flow shop scheduling problem," *Comput. Ind. Eng.*, vol. 54, no. 3, pp. 411–420, 2008, doi: 10.1016/j.cie.2007.08.003.

[30] D. Angus and C. Woodward, "Multiple objective ant colony optimisation," *Swarm Intell.*, vol. 3, no. 1, pp. 69–85, 2009, doi: 10.1007/s11721-008-0022-4.

[31] Y. Li, H. Soleimani, and M. Zohal, "An improved ant colony optimization algorithm for the multi-depot green vehicle routing problem with multiple objectives," *J. Clean. Prod.*, vol. 227, pp. 1161–1172, 2019, doi: 10.1016/j.jclepro.2019.03.185.

[32] K. Khurshid, S. Irteza, and A. A. Khan, "Application of ant colony optimization based algorithm in MIMO detection," *2010 IEEE World Congr. Comput. Intell. WCCI 2010 - 2010 IEEE Congr. Evol. Comput. CEC 2010*, 2010, doi: 10.1109/CEC.2010.5586173.

DOI: https://doi.org/10.29207/resti.v6i6.4201