Indonesian Journal of Electrical Engineering and Computer Science
Vol. 28, No. 2, November 2022, pp. 934~945
ISSN: 2502-4752, DOI: 10.11591/ijeecs.v28.i2.pp934-945

An enhanced approach for solving winner determination problem in reverse combinatorial auctions

Jawad Abusalama[1,3], Sazalinsyah Razali[1], Yun-Huoy Choo[2]
[1]Centre for Robotics and Industrial Automation, Faculty of Information and Communication Technology, Universiti Teknikal Malaysia Melaka, Durian Tunggal, Malaysia
[2]Center for Advanced Computing Technology (C-ACT), Faculty of Information and Communication Technology, Universiti Teknikal Malaysia Melaka, Durian Tunggal, Malaysia
[3]College of Engineering and Computer Science, Mustaqbal University, Buraydah, Qassim, Saudi Arabia

Article Info

Article history:

Received Sep 30, 2021
Revised Jun 30, 2022
Accepted Aug 19, 2022

Keywords:

Combinatorial auction
Coordination
NP-complete problem
Reverse combinatorial auction
Winner determination problem

ABSTRACT

When a disaster occurs, the single agent does not have complete knowledge about the circumstances of the disaster. Therefore, the rescue agents should coordinate with each other to perform their allocated tasks efficiently. However, the task allocation process among rescue agents is a complex problem, which is NP-complete problem, and determining the rescue agents that will perform the tasks efficiently is the main problem, which is called the winner determination problem (WDP). This paper proposed an enhanced approach to improve rescue agents' tasks allocation processes for WDP in reverse combinatorial auctions. The main objective of the proposed approach is to determine the winning bids that will perform the corresponding tasks with minimum cost. The task allocation problem in this paper was transformed into a two-dimensional array, and then the proposed approach was applied to it. The main contribution of the proposed approach is to shorten the search space size to determine the winners and allocate the corresponding tasks for a combination of agents (i.e., more than two agents). The proposed approach was compared to the genetic algorithm regarding the execution time, and the results showed good performance and effectiveness of the proposed approach.

Corresponding Author:

Jawad Abusalama
Centre for Robotics and Industrial Automation, Faculty of Information and Communication Technology
Universiti Teknikal Malaysia Melaka
Durian Tunggal, Malaysia
Email: jenin2101@gmail.com, sazalinsyah@utem.edu.my

1. INTRODUCTION

In managing disasters, the coordination processes among rescue agents in multi-agent system (MAS) means the ability of rescue agents to work together to achieve some goals [1]. Coordination includes sharing for both tasks (i.e., how tasks are allocated for other individual agents) and information sharing relevant to their subtasks. It can be proactively, i.e., one agent sends information to another interested agent, or reactively, i.e., an agent sends information in response to a previously sent request [2]. In this sense, Andrea [3] suggested three processes to solve the task allocation and coordination problem among agents. First, direct supervision means that the agent can send or receive an order from another agent. The second is a mutual adjustment, which means the agent adapts to enhance the coordination processes. The third process is standardization, which means that there are rules in coordination among the agents. This paper concerns first process (i.e., direct supervision), also called centralized approaches, which means that one agent is responsible for making all decisions independently, and it is usually called a decision-making agent or supervisor agent. One of the most centralized approaches to efficiently allocating tasks is a market-based mechanism [4]. The current marked-based mechanism of the research community interest is the auctions. There are clear rules to determine the prices and resource allocation based on bids in an auction. Therefore, auctions are practical and efficient to allocate resources to agents [5]. Two main parts in an auction are the auctioneer (seller) and the bidders (buyers), where the seller wants to sell the items with the highest price available, and the buyers want to get the item with the lowest price [3].

Auctions were used widely for task allocation and coordination in MAS [6]. In this sense, a wide range of possible auctions has been identified, and the combinatorial auctions are most frequently used. In combinatorial auctions, multiple bidders can place bids for combinations of items (i.e., tasks) for only one auctioneer [7], which means that multiple bidders are required to fulfil a combination of tasks according to a given criterion. The criterion can maximize the utility (i.e., cost) or minimize the accomplished time for all tasks. The main objective of combinatorial auctions is to obtain the maximum benefit for the auctioneer by determining the appropriate set of winning bids (i.e., the winners). Determining the winning bids that will perform the corresponding tasks is regarded as one of the main problems in the combinatorial auction, which is called the winner determination problem (WDP). A combinatorial auction problem is a winner determination problem which is still one of the main challenges of combinatorial auctions [8]. Indeed, WDP in combinatorial auctions is a complex problem, and it is an NP-complete problem, which is equivalent to a weighted set packing problem [9].

The WDP problem stated as follows: Let us consider a set of m tasks, $T = 1, 2 ... m$ to be auctioned and a set of n bids, $B = B_1, B_2 ... B_n$. A bid Bj is a tuple (Ti, Cj) where Ti is a set of tasks, and Cj is the cost of Bj. Then, a Bj can place a bids $bj(S) > 0$ for any combination of $S \subseteq T$ (i.e., the winning bids). The WDP is the problem of finding winning bids that maximize the auctioneer's revenue, which is computed as the sum of the cost of the winning bids, under two constraints that each task can be allocated to at most one bidder and all tasks should be covered. The WDP can be modelled as (1).

$$Max \sum_{j=1}^{n} b_j\ (S, C) \tag{1}$$

Under the following constraints:

1) Each solution is a sequence of disjoints bids, which means that no tasks are shared. Therefore, it holds that $S_1 \cap S_2 \cap \ldots \cap S_j = \emptyset$.

2) All tasks should be covered in the solution; it holds that $S_1 \cup S_2 \cup \ldots \cup S_j = T$.

This paper suggest to improve task allocation processes in a combinatorial auction by minimizing the cost of performing the tasks as quickly as possible rather than maximizing the cost contrary to the traditional combinatorial auction, the reverse case. The roles of auctioneers and bidders in the reverse combinatorial auction are exchanged. According to [10], reverse combinatorial auctions are widely used in procurement, where the auctioneer wants to gain the tasks with minimum possible cost from an appropriate set of bidders to perform the corresponding tasks [11]. The auctioneer can hold a reverse auction to try to obtain the tasks, more formally, the WDP in a reverse combinatorial auction finds winning bids that minimize the auctioneer's revenue under the same constraints. Therefore, the WDP in the reverse combinatorial auction can be modelled as (2).

$$Min \sum_{j=1}^{n} b_j\ (S, C) \tag{2}$$

A few studies have addressed the WDP in reverse combinatorial auctions. The most popular methods that have been used for solving this problem are evolutionary algorithms (EAs) such as genetic algorithm (GA) [12]-[15], swarm intelligence (SI) algorithms such as ant colony optimization (ACO) [16]-[17], lagrangian relaxation [18]-[20] and bidtree formulation [3], [21].

The previous studies noted that the WDP in reverse combinatorial auctions had been solved in different methods as mentioned above. However, each method has drawbacks to reaching the optimal solution, such as the genetic algorithm (GA) [12]. The main drawback of this method needs comparatively many generations. In addition, computational time increases significantly with increasing the number of tasks and bidders. In swarm intelligence (SI) [16], the main drawback is the average cost of the bidder, which is considered constant (i.e., there is no significant difference in costs among items). In addition, the improved algorithm is effective for the auctioneer to find an optimal solution for small scale only. While the lagrangian relaxation method [18] can only find near-optimal solutions for minor problems without considering the large scales. Finally, the bidtree formulation method requires complex steps to prepare the bidtree structure before applying the search algorithm. In addition, the A∗ algorithm had been used for searching for the optimal solution, while one major practical drawback of the A∗ algorithm is space complexity.

As mentioned in the literature, few algorithms have solved WDP in a reverse combinatorial auction, where the Bidtree structure and GA algorithm were the most popular methods in solving this problem. The Bidtree was introduced by Andrea [3] and Lopez *et al.* [21], which is a binary tree that allows for searching bids based on item content. Bidtrees are used to determine how bids are considered during searching and ensure each bid combination is tested at least once. The problem with this approach is required many complex steps before preparing the bidtree structure. As well as the Bidtree method was fit to find the optimal solution for only two combinations of bidders, which means that the bidtree method will fail to find the optimal solution if the optimal solution is combined with more than two bidders [22]. On the other hand, in the GA algorithm, when the number of tasks and bidders are exposed to mutation is large there is often an exponential increase in search space size, and this makes it extremely difficult to use the technique on problems such as WDP in reverse combinatorial auction [23].

Considering the drawbacks of the previous method in solving WDP in reverse combinatorial auctions, this paper proposed an enhanced approach to improve task allocation processes after formulating the WDP in a reverse combinatorial auction to a 2D Array. The 2D Array $A_{m\times n}$ has *m* rows and *n* columns, where *m* represents the tasks and *n* represents the bids. The main contribution of the proposed approach is to shorten the search space size in determining the winning bidders efficiently. In addition, the proposed approach will be efficient in allocating the corresponding tasks for a combination of bidders (i.e., more than two bidders) in contrast to some existing methods such that bidtree method.

2. THE PROPOSED APPROACH AND METHOD

2.1. An enhanced approach for solving WDP in reverse combinatorial auctions

In this subsection, the steps of the proposed approach were discussed and explained. However, before starting the approach steps, the following example explains the problem description and formulation. Suppose we have nine tasks with the following identification: Task1, Task2…Task9, and assume that the auctioneer has received twenty bids for the mentioned tasks as shown in Figure 1.

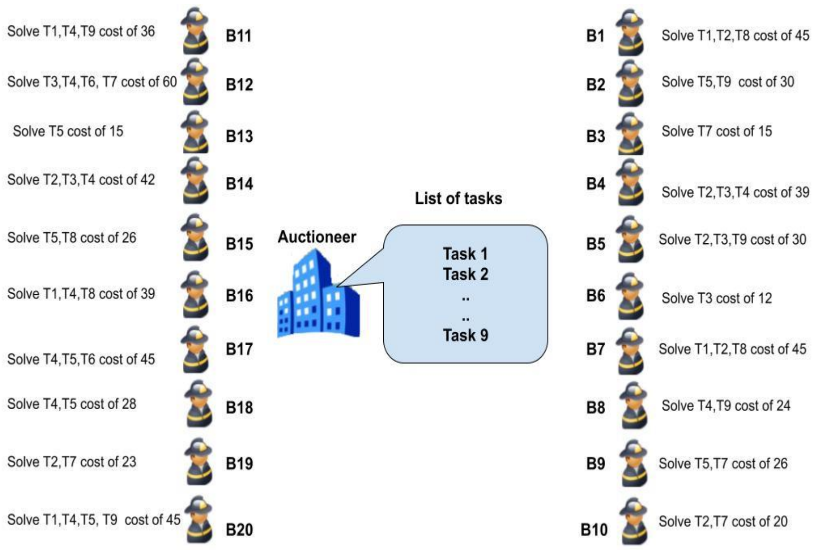

Figure 1. An illustrative example

As we explained earlier in this paper, the main objective of the WDP in reverse combinatorial auctions is to obtain the maximum benefit for the auctioneer by determining the appropriate set of winning bids with minimum cost [24]. To reach this objective, the bids and their corresponding tasks have been formulated into 2D Array A_{m*n} as shown in Table 1, taking into consideration that each sell will be 1 if the bidder has solved the corresponding task or 0 otherwise. After preparing the 2D array, the overall steps of the proposed approach were outlined below, while the flowchart of the proposed approach was described in Figure 2.

Table 1. Formulated array

Bids / Tasks	B1	B2	B3	B4	B5	B6	B7	B8	B9	B10	B11	B12	B13	B14	B15	B16	B17	B18	B19	B20
Task1	1	0	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0	0	0	1
Task2	1	0	0	1	1	0	0	0	0	1	0	0	0	1	0	0	0	0	1	0
Task3	0	0	0	1	1	1	0	0	0	0	0	1	0	1	0	0	0	0	0	0
Task4	0	0	0	1	0	0	0	1	0	0	1	1	0	1	0	1	1	1	0	1
Task5	0	1	0	0	0	0	0	0	1	0	0	0	1	0	1	0	1	1	0	1
Task6	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0	0	0
Task7	0	0	1	0	0	0	0	0	1	1	0	1	0	0	0	0	0	0	1	0
Task8	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0
Task9	0	1	0	0	1	0	1	1	0	0	1	0	0	0	0	0	0	0	0	1
Costs	45	30	15	39	30	12	12	24	26	20	36	60	15	42	26	39	45	28	23	45

Step 1: re-ordering the array according to the tasks in increasing order. Considering that if there is more than one bid that has solved the same task set, take the lowest cost and remove the others, since these bids will never generate the optimal solution.

Step 2: re-ordering the array according to the bids (i.e., the bid that solves the most tasks is the first). Considering that the bids that solved the most frequent task to be left side bids (LSB), the other bids will be right side bids (RSB). After the first two steps, the array will be as shown in Table 2.

Table 2. Re-ordered array

Bids / Tasks	B20	B12	B11	B17	B4	B16	B8	B18	B5	B1	B2	B9	B15	B10	B19	B7	B13	B3	B6
Task4	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0
Task9	1	0	1	0	0	0	1	0	1	0	1	0	0	0	0	1	0	0	0
Task5	1	0	0	1	0	0	0	1	0	0	1	1	1	0	0	0	1	0	0
Task2	0	0	0	0	1	0	0	0	1	1	0	0	0	1	0	0	0	0	0
Task1	1	0	1	0	0	1	0	0	0	1	0	0	0	0	1	0	0	0	0
Task7	0	1	0	0	0	0	0	0	0	0	0	1	0	1	1	0	0	1	0
Task3	0	1	0	0	1	0	0	0	1	0	0	0	0	0	0	0	0	0	1
Task8	0	0	0	0	0	1	0	0	0	1	0	0	1	0	0	0	0	0	0
Task6	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Costs	45	60	36	45	39	39	24	28	30	45	30	26	26	20	28	12	15	15	12
	Left side bids (LSB)								Right side bids (RSB)										

Step 3: based on the prepared array, there are two cases to find the winner bidders:

Step 3.1 (Case 1): if the bids that solved the last task are located in *LSB* in the repeated array, these bids will be a part of the solution. Let $Bls = \{Bls_1, Bls_2 \dots Bls_n\}$ be the set bids that solved the least task located in *LSB*. Let $Brc = \{Brc_1, Brc_2 \dots Brc_n\}$ be the complement bids of *Bls* in *RSB*.

The following sup-steps are to solve the winning bidders for **Case 1**:

Step 3.1.1: Set solution (*Sn*) = {}

Step 3.1.2: For each *Bls*, find *Brc* (i.e., the complement bids from *RSB*).

Step 3.1.3: If $Bls_1 \cup Brc_1$ found all tasks, Then $Bls_1 \cup Brc_1$ will be a solution to be saved in *Sn*.

Step 3.1.4: In case, $Bls_1 \cup Brc_1$ found some tasks, take the combination of Bls_1 and Brc_1 then find the complement with the remaining bids in *Brc*.

Step 3.1.5: Repeat the previews steps for the next *Bls*.

Step 3.2 (Case 2): if the bids that solved the last task are located in RSB in the repeated array, these bids will be also a part of the solution. Let $Brs = \{Brs_1, Brs_2 \dots Brs_n\}$ be the set bids that solved the least task located in *RSB*. Let $Blc = \{Blc_1, Blc_2 \dots Blc_n\}$ be the complement bids of *Brs* in *LSB*.

The following sup-steps are to solve the winning bidders for **Case 2**:

Step 3.2.1: Set solution (Sn) = {}

Step 3.2.2: For each Brs, find Blc (i.e., the complement bids from *LSB*)

Step 3.2.3: Find the complement of $\{Brs1 \cup Blc1\}$ from $\{RSB - Brs1\}$.

Step 3.2.4: If $\{Brs_1 \cup Blc_1\} \cup \{RSB - Brs_1\}$ found all tasks, Then $\{Brs_1 \cup Blc_1\} \cup \{RSB - Brs_1\}$ will be a solution to be saved in *Sn*.

Step 3.2.5: In case, $\{Brs_1 \cup Blc_1\} \cup \{RSB - Brs_1\}$ found some tasks, take the combination of $\{Brs_1 \cup Blc_1\} \cup \{RSB - Brs_1\}$ then find the complement with the remaining bids in Brs_n.

Step 3.2.6: Repeat the previews steps for the next *Brs*.

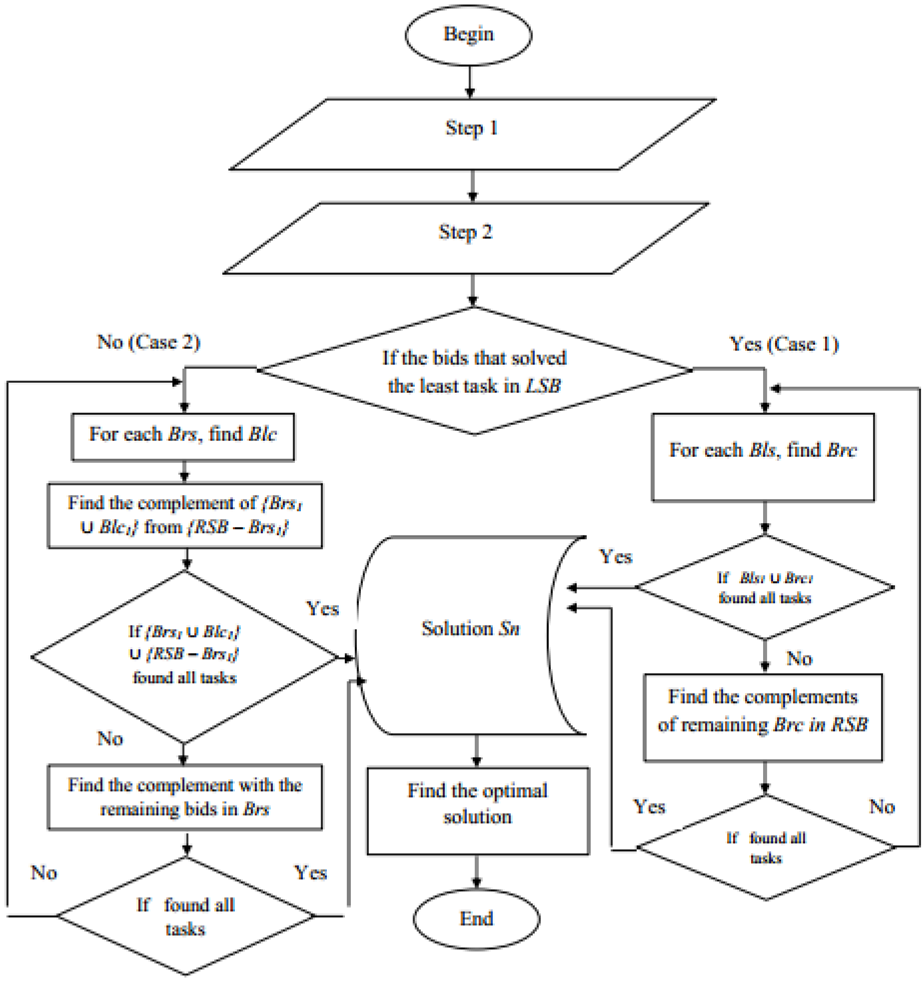

Figure 2. Flowchart of the proposed approach

2.2. Method

In this subsection, we will apply the method of the proposed approach to the actual example, suppose we have five tasks with the following identification: Task1, Task2, Task3, Task4 and Task5, assume that the auctioneer has received ten bids for the mentioned tasks as shown in Figure 3. So, the transformed 2D array of this example is shown in Table 3. Now, we need to apply the steps of the proposed approach to the prepared array.

Step1: re-ordering the array according to the tasks in increasing order. Taking into account that B3 has the same tasks set as B5 and the cost of B5 is the highest cost of B3. So, B5 will be erased from the bids set, as B5 will never generate the optimal solution. While Task 3 will be in the first order and Task 5 will the last one as shown in Table 4.

Step 2: re-ordering the array according to the bid that solves the most tasks will be the first, taking into account that the bids that solved the most frequent task (i.e., Task3) will be *LSB*, and the other bids will be *RSB* as shown in Table 5.

Step 3: depending on the prepared array in Table 5, if there are bids that solve the most frequent task (i.e., Task3) and Least frequent (i.e., Task5) at the same time, Start from these bids, which will certainly be a part of the solutions. Therefore, B9 from *LSB* or B7 from *RSB* will be a part of the solutions.

Step 3.1 (Case 1): for B9 that solved the last task (i.e., Task 5) which is located in *LSB*.

Step 3.1.1: Set solution (Sn) = {}

Step 3.1.2: find the compliments Bids of B9 from *RSB* only, which are B1, B2, and B6.

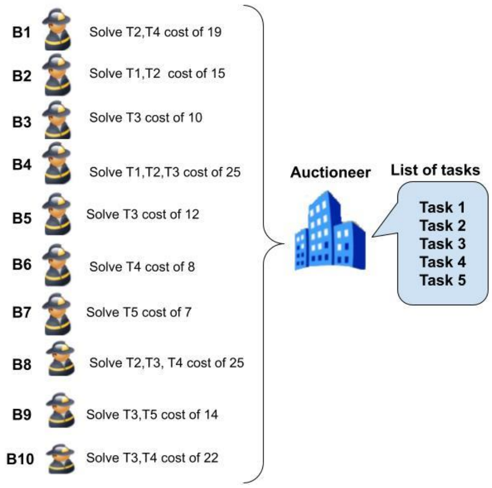

Figure 3. A working example

Table 3. A formulated array of the working example

Bids / Tasks	B1	B2	B3	B4	B5	B6	B7	B8	B9	B10
Task1	0	1	0	1	0	0	0	0	0	0
Task2	1	1	0	1	0	0	0	1	0	0
Task3	0	0	1	1	1	0	0	1	1	1
Task4	1	0	0	0	0	1	0	1	0	1
Task5	0	0	0	0	0	0	1	0	1	0
Costs	19	15	10	25	12	8	7	25	14	22

Table 4. The re-ordered array according to tasks appearing

Bids / Tasks	B1	B2	B3	B4	B5	B6	B7	B8	B9	B10
Task3	0	0	1	1	1	0	0	1	1	1
Task2	1	1	0	1	0	0	0	1	0	0
Task4	1	0	0	0	0	1	0	1	0	1
Task1	0	1	0	1	0	0	0	0	0	0
Task5	0	0	0	0	0	0	1	0	1	0
Costs	19	15	10	25	12	8	7	25	14	22

Table 5. The re-ordered array according to bids

Bids / Tasks	B8	B4	B10	B9	B3	B1	B2	B6	B7
Task3	1	1	1	1	1	0	0	0	0
Task2	1	1	0	0	0	1	1	0	0
Task4	1	0	1	0	0	1	0	1	0
Task1	0	1	0	0	0	0	1	0	0
Task5	0	0	0	1	0	0	0	0	1
Costs	25	25	22	14	10	19	15	8	7
			LSB				*RSB*		

Step 3.1.3 to Step 3.1.5 are:

Find {B9 ∪ B1}, if the combination solved all tasks then stop, and save the solution in *Sn*. If not then find ((B9 ∪ B1) ∪ B2) if the combination solved all tasks, then stop and save the solution in *Sn*. If not, then find (((B9 ∪ B1) ∪ B2) ∪ B6) and so on for the remaining bids (i.e., B2 and B6) as explained:

- For {B9 ∪ B1} as shown in Table 6, we can conclude that there is no conflict, but the combination did not solve all tasks. Therefore, we need to find the compliment from the remaining bids (i.e., B2 and B6). For ((B9 ∪ B1) ∪ B2) as shown in Table 7, as we can see there is a conflict in Task2 (i.e., there are two bids solved for the same task at the same time). In this case, skip B2 and find the combination with the next bid, which is B6. For ((B9 ∪ B1) ∪ B6) as shown in Table 8, as we can see there is a conflict in Task4. As long as we finished the remaining bids without finding a solution, then we need to find the combination of B9 with the next complement bids in *RSB*, which is B2. Now, find {B9 ∪ B2}, if the combination solved all tasks then stop and save the solution in *Sn*. if not then find ((B9 ∪ B2) ∪ B6) if the combination solved all tasks then stop and save the solution in *Sn*. as explained:

- For {B9 ∪ B2} as shown in Table 9, we can conclude that there is no conflict, but the combination did not solve all tasks, still, Task 4 is not solved. Therefore, we need to find the compliment from the remaining bids, which is B6. For ((B9 ∪ B2) ∪ B6) as shown in Table 10, we can see this combination has solved all the tasks. Therefore, the combination of ((B9 ∪ B2) ∪ B6) is a solution that will save in *Sn*. Finally, for (B9 ∪ B6), as long as the combination of (B9 ∪ B6) did not solve all tasks then the proposed approach will stop. Repeat the previous step if there are bids in LSB solved the least frequent (i.e., task 5).

Table 6. The combination of B9 ∪ B1

B9	B1	B9 ∪ B1
1	0	1
0	1	1
0	1	1
0	0	0
1	0	1
14	19	33

Table 7. The combination of ((B9 ∪ B1) ∪ B2)

B9 ∪ B1	B2	(B9 ∪ B1) ∪ B2
1	0	1
1	1	conflict
1	0	1
0	1	0
1	0	1
33	15	48

Table 8. The combination of ((B9 ∪ B1) ∪ B6)

B9 ∪ B1	B6	(B9 ∪ B1) ∪ B6
1	0	1
1	0	1
1	1	conflict
0	0	0
1	0	1
33	8	41

Table 9. The combination of (B9 ∪ B2)

B9	B2	B9 ∪ B2
1	0	1
0	1	1
0	0	0
0	1	1
1	0	1
14	15	29

Table 10. The combination of ((B9 ∪ B2) ∪ B6)

B9 ∪ B2	B6	(B9 ∪ B2) ∪ B6
1	0	1
1	0	1
0	1	1
1	0	1
1	0	1
29	8	37

Step 3.2 (Case 2): for B7 that solved the last task (i.e., Task 5) which is located in *RSB.*
Step 3.2.1: Set solution (*Sn*) = {}
Step 3.2.2: find the complement bids of B7 from *LSB*, which are B8, B4, B10 and B3.
Step 3.2.3: find the complement of (B7 ∪ each {B8, B4, B10, B3}) from *RSB* – B7.
Step 3.2.4 to Step 3.2.6 are explained:

- For B8 fined the combination of (B7 ∪ B8). Then find the compliments of (B7 ∪ B8) from the bids that are located in *RSB* - B7, which are B1, B2 and B6. For ((B7 ∪ B8) ∪ B1) as shown in Table 11, we can see there is a conflict between Task 2 and Task 4. In this case, skip B1 and find the combination with the next bid, which is B2. For ((B7 ∪ B8) ∪ B2) as shown in Table 12, we can see there is a conflict in Task 2. In this case, skip B2 and find the combination with the next bid, which is B6. For ((B7 ∪ B8) ∪ B6) as shown in Table 13, we can see there is a conflict in Task 4. Therefore, there is no solution for the compliments of (B7 ∪ B8) from *LSB*. Now, check the complement of B7 with the next bid in step 3.2.3, which is B4. For B4 fined the combination of (B7 ∪ B4). Then find the compliments of (B7 ∪ B4) from the bids that are located in *RSB* - B7, which is B6 only. For ((B7 ∪ B4) ∪ B6) as shown in Table 14, we can see this combination has solved all the tasks. Therefore, the combination of ((B7 ∪ B4) ∪ B6) is a solution that will save in *Sn*. Next, check the complement of B7 with the next bid in step 3.2.3, which is B10. For B10 fined the combination of (B7 ∪ B10). Then find the compliments of (B7 ∪ B10) from the bids that are located in *RSB* - B7, which is B2 only. For ((B7 ∪ B10) ∪ B2) as shown in Table 15, we can see this combination has solved all the tasks. Therefore, the combination of ((B7 ∪ B10) ∪ B2) is a solution that will save in *Sn*. Next, check the complement of B7 with the next bid in step 3.2.3, which is B3. For B3 fined the combination of (B7 ∪ B3).

Then find the compliments of (B7 ∪ B3) from the bids that are located in *RSB* - B7, which are B1, B2 and B6. For B1, the combination of ((B7 ∪ B3) ∪ B1) there is no conflict but this combination didn't solve all tasks, in this case, take the new combination and find the compliments from the remaining bids (i.e., B2 and B6). The combination of (((B7 ∪ B3) ∪ B1) ∪ B2) there is a conflict in Task 2. Check for the last bid which is B6, The combination of (((B7 ∪ B3) ∪ B1) ∪ B6) there is a conflict in Task 4. We conclude that there is no solution for B1. For B2, the combination of ((B7 ∪ B3) ∪ B2) there is no conflict, but this combination didn't solve all tasks, still Task4 not solved, in this case, take the new combination to find the compliments from the remaining bids (i.e., B6). As we can see from Table 16, the combination of (((B7 ∪B3) ∪ B2) ∪ B6) has solved all tasks. Therefore, this solution will save in *Sn.*

Finally, for B6, if the combination of ((B7 ∪ B3) ∪ B6) has not solved all tasks then the proposed approach will stop. Repeat the previous step if there are bids in *RSB* solved the least frequent (i.e., task 5). As long as the bids that solved the least task have been finished in this example, get the optimal solution from *Sn*, which is the combination of ((B9 ∪ B2) ∪ B6) with a total cost of 37.

Table 11. The combination of ((B7 ∪ B8) ∪ B1)

B7 ∪ B8	B1	(B7 ∪ B8) ∪ B1
1	0	1
1	1	conflict
1	1	conflict
0	0	0
1	0	1
32	19	51

Table 12. The combination of ((B7 ∪ B8) ∪ B2)

B7 ∪ B8	B2	(B7 ∪ B8) ∪B2
1	0	1
1	1	conflict
1	0	0
0	1	1
1	0	1
32	15	47

Table 13. The combination of ((B7 ∪ B8) ∪ B6)

B7 ∪ B8	B6	(B7∪B8) ∪ B6
1	0	1
1	0	1
1	1	conflict
0	0	0
1	0	1
32	8	40

Table 14. The combination of ((B7 ∪ B4) ∪ B6)

B7 ∪ B4	B6	(B7 ∪ B4) ∪ B6
1	0	1
1	0	1
0	1	1
1	0	1
1	0	1
32	8	40

Table 15. The combination of ((B7 ∪ B10) ∪ B2)

B7 ∪ B10	B2	(B7 ∪ B10) ∪ B2
1	0	1
1	1	1
0	0	1
1	1	1
1	0	1
29	15	44

Table 16. The combination of (((B7 ∪ B3) ∪ B2) ∪ B6)

(B7 ∪ B3) ∪ B2	B6	((B7 ∪ B3) ∪B2) ∪B6
1	0	1
1	0	1
0	1	1
1	0	1
1	0	1
32	8	40

3. RESULTS AND DISCUSSION

To justify the performance and effectiveness of the proposed approach, this section was divided into two main sections. Firstly, three experiments were conducted on three specific cases to show the proposed approach performances. Secondly, to justify the effectiveness of the proposed approach, a comparative experiment was conducted against the GA approach proposed by [12]. The proposed approach was implemented in Python, and all experiments have been executed on a Dell laptop equipped with an Intel(R) Core(TM) i7 CPU 1.73 GHz, 1.73 GHz with 8.00 GB of RAM operated with 64-bit Windows 10 Professional operating system.

For the performances of the proposed approach, three cases were applied to determine the appropriate set of winning bids with minimum cost, there was a small case, a medium case and a large case. Where a 20 2D array generated in each case with two parameters, *n* parameter represents the number of bidders and *m* parameter represents the number of tasks, and for credibility, we deliberately entered bidders that solved a few tasks in the arrays. In the first experiment, we conduct a small random dataset with specific parameters of *n*=5 and *m*=5. Then, we extracted the execution time (in seconds) for each ran array as shown in Figure 4. For the second experiment, we conduct a medium random dataset with specific parameters of *n*=20 and *m*=10. Then, we extracted the execution time for each ran array as shown in Figure 5. Finally, we conduct a large random dataset with specific parameters of *n*=100 and *m*=30. Then, we extracted the execution time for each ran array as shown in Figure 6.

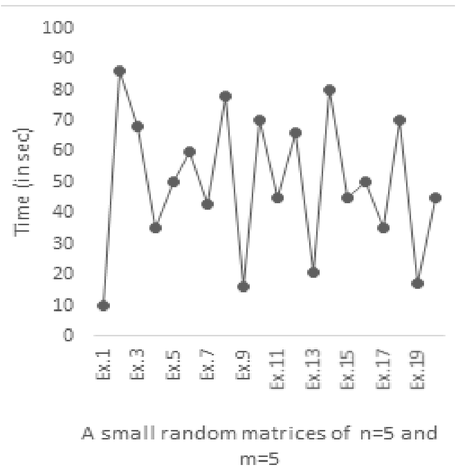

Figure 4. The global best solution for *n*=5 and *m*=5

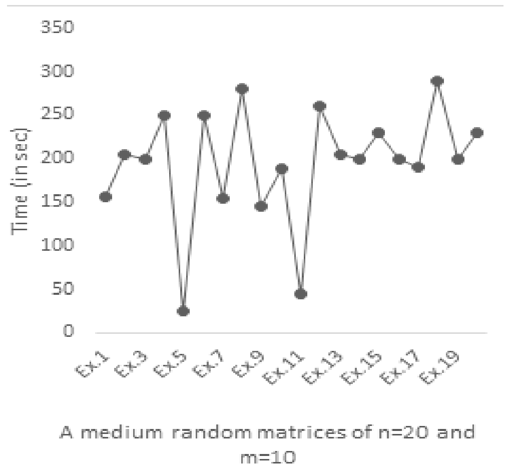

Figure 5. The global best solution for n=20 and m=10

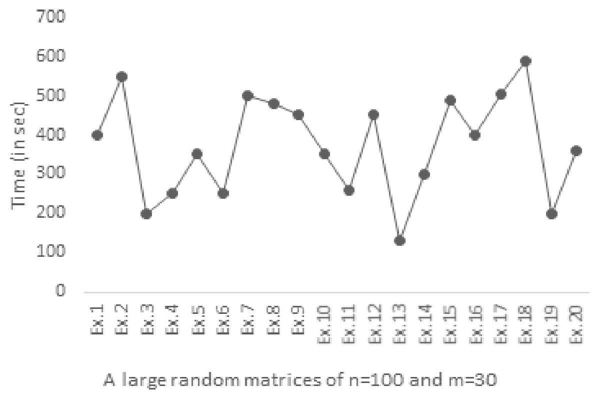

Figure 6. The global best solution for *n*=100 and *m*=30

Previous results, shows that as the number of tasks and bidders increases, the time taken to solve the problem increases because of the increase in computations. On the other hand, the proposed approach can efficiently determine winners with more numbers of products and the bidders in a reasonable time. In addition, it can also be seen that the proposed approach can suddenly solve the problem faster in some cases, As we can see in the first experiment, Ex.1, Ex.9, Ex.13 and Ex.19 have solved the problem in a relatively short time compared to other cases. As well as Ex.5 and Ex.11 in the second experiment and Ex.13 in the third experiment, this is because the entered array had a task that was solved by only one bidder, where the search space for the proposed approach was reduced.

For the effectiveness of the proposed approach, where reverse combinatorial auction finds more and more applications in diverse domains but determining the winners in the reverse combinatorial auction is a complex problem [25], this paper compared the proposed approach against GA approach proposed by [12], where a similar public data was re-run for both approaches. In this section, two experiments were conducted to determine the winner bidders in reverse combinatorial auctions, in the first experiment, a similar dataset was run for a fixed number of tasks, and then we extracted the execution time of the proposed approach compared to the GA approach. Figures 7-9 present the comparison of execution time (in seconds) versus the number of bidders. As we can see, the test results show that the proposed approach takes far less time than the GA approach. In addition, as the number of bidders increases, the execution times of the GA approach increase exponentially, while the proposed approach increases steadily.

Another experiment was run for a fixed number of bidders, the comparison of execution time between the proposed approach and the GA approach versus several tasks has been presented in Figures 10-12. As we can see, the test results show that the execution time of the proposed approach and GA approach was very close in task No.6 and task No.7 in most cases. While the number of tasks increases, the execution times of the GA approach increase suddenly compared to the proposed approach.

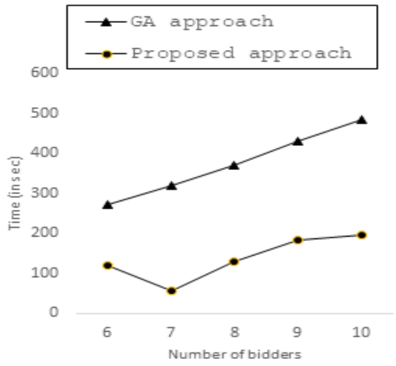

Figure 7. Time vs. Number of bidders (for fixed no. of tasks =10)

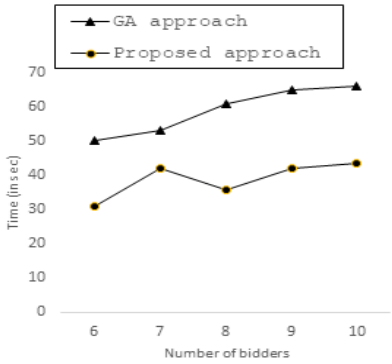

Figure 8. Time vs. Number of bidders (for fixed no. of tasks =8)

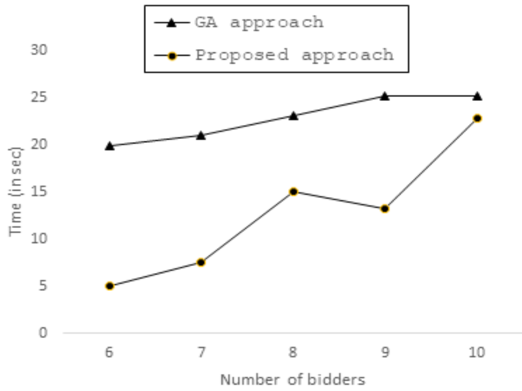

Figure 9. Time vs. Number of bidders (for fixed no. of tasks =6)

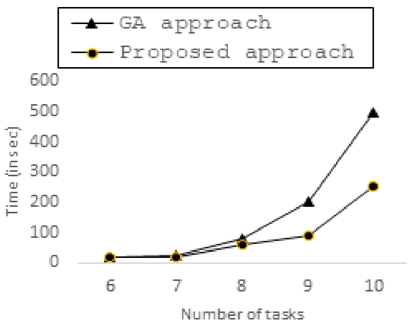

Figure 10. Time vs. Number of tasks (for fixed no. of bidders =10)

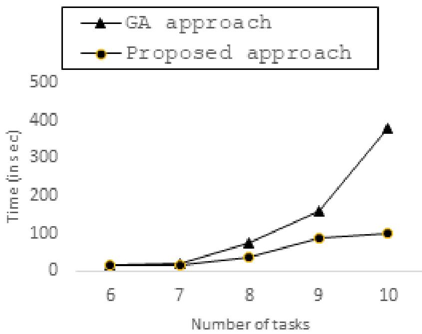

Figure 11. Time vs. Number of tasks (for fixed no. of bidders =8)

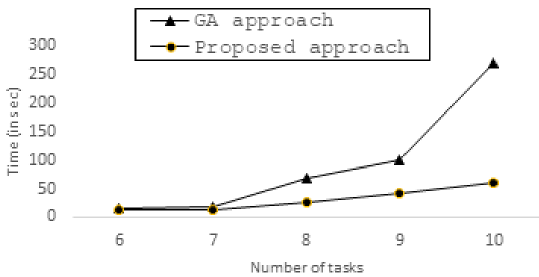

Figure 12. Time vs. Number of tasks (for fixed no. of bidders =6)

4. CONCLUSION

Reverse combinatorial auction finds more and more applications in diverse domains but determining the winners in the reverse combinatorial auction is a complex problem. In this paper, an enhanced approach was investigated for WDP in the reverse combinatorial auctions. Where, the optimal WDP in the reverse combinatorial auction problem was transformed into a two-dimensional array with *m* rows and *n* columns, in which *m* represents the tasks and *n* represents the bids, Then the proposed approach has been applied to the transformed array to solve WDP in the reverse combinatorial auction. The performance of the proposed approach has been justified through three different cases, there were a small, medium and a large number of tasks and bidders, and the results showed that the proposed approach has been solved the problem in a reasonable time. On the other hand, the proposed approach can efficiently determine the optimal solution with more numbers of tasks and bidders, where a very competitive result has been achieved compared to the GA approach proposed. The proposed approach was very effective in case of more than two bidders have to solve a large number of tasks, where the proposed approach reduced the search space while finding the winners for more than two combinations of bidders, exactly if there is a task that was solved by only one bidder. Therefore, the proposed approach is not a competitor, It constitutes an exciting alternative approach in special cases to finding the winners in the reverse combinatorial auction.

ACKNOWLEDGEMENTS

The authors would like to thank the Centre for Robotics and Industrial Automation (CeRIA), Center for Advanced Computing Technology (C-ACT), and Faculty of Information and Communication Technology at Universiti Teknikal Malaysia Melaka (UTeM) for the facilities provided and for supporting this work.

REFERENCES

[1] A. Dorri, S. S. Kanhere, and R. Jurdak, "Multi-agent systems: a survey," *IEEE Access*, vol. 6, no. April, pp. 28573–28593, 2018, doi: 10.1109/ACCESS.2018.2831228.

[2] J. Abusalama, A. R. Alkharabsheh, L. Momani, and S. Razali, "MultiAgents system for early disaster detection, evacuation and rescuing*," in 2020 Advances in Science and Engineering Technology International Conferences (ASET)*, Dubai, United Arab Emirates, 2020, pp. 1–6, doi: 10.1109/ASET48392.2020.9118322.

[3] S. Andrea, "Dynamic task allocation and coordination in cooperative multi-agent environments," *PhD. Dissertation, College of Computer Science*, University of Girona, Spain, 2010.

[4] A. Koubaa, O. Cheikhrouhou, H. Bennaceur, M. F. Sriti, Y. Javed, and A. Ammar, "Move and improve: a market-based mechanism for the multiple depot multiple travelling salesmen problem," *J. Intell. Robot. Syst. Theory Appl*., vol. 85, no. 2, pp. 307–330, 2017, doi: 10.1007/s10846-016-0400.

[5] P. Mishra, A. Moustafa, and T. Ito, "Fairness based multi-preference resource allocation in decentralised open markets," *in International Workshop on Optimization and Learning in Multiagent Systems, May 2020, Auckland, New Zealand*, 2021, pp. 1–5, doi: 10.48550/arXiv.2109.00207.

[6] E. Schneider, M. Poulton, A. Drake, L. Smith, G. Roussos, S. Parsons, and E. I. Sklar, "The application of market-based multi-robot task allocation to ambulance dispatch," *in eprint arXiv: 2003.05550,* 2020, pp. 1–19, doi: 10.48550/arXiv.2003.05550.

[7] M. B. Dowlatshahi and V. Derhami, "Winner determination in combinatorial auctions using hybrid ant colony optimization and multi neighborhood local search," *J. AI Data Min*., vol. 5, no. 2, pp. 169–181, 2017, doi: 10.22044/JADM.2017.880.

[8] M. Takalloo, A. Bogyrbayeva, H. Charkhgard, and C. Kwon, "Solving the winner determination problem in combinatorial auctions for fractional ownership of autonomous vehicles," *Int. Trans. Oper. Res.,* vol. 28, no. 4, pp. 1658–1680, 2021, doi: 10.1111/itor.12868.

[9] R. Li, Y. Wang, S. Hu, J. Jiang, D. Ouyang, and M. Yin, "Solving the set packing problem via a maximum weighted independent set heuristic," *Math. Probl. Eng*., vol. 2020, no. 1, December 2020, Art. no. 3050714, doi:10.1155/2020/3050714.

[10] C. C. Lin, Y. F. Chang, C. C. Chang, and Y. Z. Zheng, "A fair and secure reverse auction for government procurement," *Sustainability*., vol. 12, no. 20, pp. 1–12, Oct. 2020, doi:10.3390/su12208567.

[11] S. Al Shaqsi, "Combinatorial reverse auctions in construction procurement," M.S. thesis, Massachusetts Institute of Technology, Oman, 2018.

[12] P. Patodi, A. K. Ray, and M. Jenamani, "GA based winner determination in combinatorial reverse auction," *Proc. - 2nd Int. Conf. Emerg. Appl. Inf. Technol*. EAIT 2011, no. 2, pp. 361–364, 2011, doi: 10.1109/EAIT.2011.80.

[13] S. K. Shil, M. Mouhoub, and S. Sadaoui, "Winner determination in combinatorial reverse auctions*," in Contemporary Challenges and Solutions in Applied Artificial Intelligence*, pp. 35–40, 2014, doi: 10.1007/978-3-319-00651-2_5.

[14] B. H. Zaidi, I. Ullah, M. Alam, B. Adebisi, A. Azad, A. R. Ansari, and R. Nawaz, "Incentive based load shedding management in a microgrid using combinatorial auction with IoT infrastructure," *Sensors*, vol. 21, no. 6, pp. 1–16, 2021, doi: 10.3390/s21061935.

[15] S. K. Shil and M. Mouhoub, "Considering multiple instances of items in combinatorial reverse auctions," *Lect. Notes Comput. Sci*. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics), vol. 8482 LNAI, no. PART 2, pp. 487–496, 2015, doi: 10.1007/978-3-319-07467-2_51.

[16] Q. Ning and W. Ding-wei, "An improved ant colony algorithm for winner determination in multi-attribute combinatorial reverse auction," *in Proceedings of the 2014 IEEE Congress on Evolutionary Computation, CEC 2014, 2014*, vol. 12, no. April, pp. 33-37, doi: 10.1109/CEC.2014.6900493.

[17] X. Qian, M. Huang, J. Tu, and X. Wang, "An improved particle swarm optimization algorithm for winner determination in multi-attribute combinatorial reverse auction*," in Proceedings of the World Congress on Intelligent Control and Automation (WCICA),* vol. 2015-March, no. March, pp. 605–609, 2015, doi: 10.1109/WCICA.2014.7052783.

[18] F. S. Hsieh and S. M. Tsai, "Combinatorial reverse auction based on lagrangian relaxation," *in Proceedings of the 3rd IEEE Asia-Pacific Services Computing Conference, APSCC 2008*, 2008, pp. 329–334, doi: 10.1109/APSCC.2008.112.
[19] F. S. Hsieh, "Combinatorial reverse auction based on revelation of Lagrangian multipliers," *Decis. Support Syst.*, vol. 48, no. 2, pp. 323–330, 2019, doi: 10.1016/j.dss.2009.08.009.
[20] B. Mansouri and E. Hassini, "A lagrangian approach to the winner determination problem in iterative combinatorial reverse auctions," *Eur. J. Oper. Res.*, vol. 244, no. 2, pp. 565–575, 2015, doi: 10.1016/j.ejor.2015.01.053.
[21] B. Lopez, S. Suarez, and J. L. De La Rosa, "Task allocation in rescue operations using combinatorial auctions," *Artif. Intell. Res. Dev.*, vol. 100, pp. 233–243, 2003.
[22] J. Abusalama, S. Razali, and Y. Choo, "Tasks allocation approach for rescue agents in reverse combinatorial auction," *in International Conference on Computer and Information Sciences (ICCIS 2020) held in Jouf University, College of Computer and information sciences*, 2021, pp. 301–306.
[23] S. Katoch, S. S. Chauhan, and V. Kumar, "A review on genetic algorithm: past, present, and future," *Multimed. Tools Appl.*, vol. 80, no. 5, pp. 8091–8126, 2021, doi: 10.1007/s11042-020-10139-6.
[24] J. Kiser, "Developing optimization techniques for logistical tendering using reverse combinatorial auctions," M.S. thesis, Department of Mathematics, East Tennessee State University, Johnson City, USA, 2018.
[25] L. S. Granada, F. N. Kazama, and P. B. Correia, "Combinatorial reverse auction to coordinate transmission and generation assets in Brazil: conceptual proposal based on integer programming," in *In: Neufeld J.S., Buscher U., Lasch R., Möst D., Schönberger J. (eds) Operations Research Proceedings 2019. Operations Research Proceedings (GOR (Gesellschaft für Operations Research e.V.))*, 2020, pp. 241–274, doi: 10.1007/978-3-030-48439-2_29.

BIOGRAPHIES OF AUTHORS

Jawad Abusalama received his B.Sc from Arab American University in Palestine 2006 from information technology Department, M.Sc from Middle East University in Jordan 2009 from Faculty of Information and Communication Technology major of Computer Information System. He is working as Lecturer in college of Engineering and Computer Sciences in Mustaqbal University in Saudi Arabia. Currently, he is a PhD candidate in Faculty of Information and Communication Technology at Universiti Teknikal Malaysia Melaka in Malaysia. His research interests include Software engineering, artificial intelligent, string search algorithms, cloud computing, E-learning, multi-agent system and multi robotics coordination problem. He can be contacted at email: jenin2101@gmail.com.

Ts. Dr. Sazalinsyah Razali has a Ph.D. in Computer Science, Masters in Computer Science, BSc. (Hons.) in Information Technology, a Certified Tester (Foundation Level - MSTB/ISTQB, 2013), and a Certified IT Professional (FE Module - MITPE/ITPEC, 2005). He joined Universiti Teknikal Malaysia Melaka in 2004 as a Lecturer at the Faculty of Information and Communication Technology and now a Senior Lecturer since 2014. He started his academic career in 2003 as a Lecturer at Kolej Universiti Kejuruteraan Utara Malaysia, now known as Universiti Malaysia Perlis. Before that, Dr. Sazalinsyah Razali was a Lead Web Developer in a local IT company and was involved in various web-based and e-commerce projects. He has professionally served as Program Committee and Reviewer in several international conferences & journals, and he is a Professional Member of several professional societies, such as IEEE, IEEE Robotics & Automation Society (IEEE-RAS), IEEE Computational Intelligence Society (IEEE-CIS), IEEE Systems, Man & Cybernetics Society (IEEE-SMC), Malaysian Society for Engineering & Technology (MySET), and Malaysia Automatic Control Engineers Society (MACE). He can be contacted at email: sazalinsyah@utem.edu.my.

Yun-Huoy Choo is currently working as an associate professor at the Fakulti Teknologi Maklumat dan Komunikasi, Universiti Teknikal Malaysia Melaka, Malaysia since 2015. She was awarded PhD in System Management and Science from the National University of Malaysia in 2008. Her technical expertise is in the fundamental studies of computational intelligence, specifically in data mining, machine learning, and data analytics. Her research interest includes various data analytics topics such as person authentication using EEG signals, muscle endurance analysis using sEMG signals, machine OEE analysis, KPIs Analytics, Youth Profiling Analytics, Explanatory AI, Consumer Analytics, personalized itinerary, and route planning. She is professionally certified as Dell EMC2 Data Science Associate, RapidMiner Professional Analyst, Microsoft Azure AI and Data Fundamental Associate, Tableau Desktop Specialist, and ISTQB CTFL Instructor. She has been appointed as the data analytics expert at the Institute of Youth Research, the Ministry of Youth & Sports Malaysia, and the Panda Software House Melaka. Besides, she is currently serving on the IEEE SMC Technical Committee, IEEE SMC TC on Humanized Crowd Computing, and the IEEE SMC (Malaysia Chapter) Executive Committee on professional development, mentorship, and personal growth. She can be contacted at email: huoy@utem.edu.my.