ASM Sc. J., 17, 2022

https://doi.org/10.32802/asmscj.2022.1322

The Efficiency of Hybridised Genetic Algorithm and Ant Colony Optimisation (HGA-ACO) in a Restaurant Recommendation System

M.A. Zanizam, M.D.M. Kamal, M.F.I.A. Rahim, M.F. Norulhaizy and A.M. Kassim*

School of Computer Sciences, Universiti Sains Malaysia, Penang, Malaysia

A recommendation system (RS) is used to provide recommendations to users by filtering items based on given inputs. Metaheuristic algorithms such as Genetic Algorithm (GA) and Ant Colony Optimisation (ACO) are known to be used in many RS to provide optimal and good recommendations. Both algorithms are designed based on nature-inspired events, where GA is designed based on the natural evolution process while ACO is based on ants' behaviour in their natural habit. In this paper, both GA and ACO algorithms were implemented in a restaurant RS and evaluated by using the restaurant's attributes, which was then followed by a list of recommended restaurants as the output. With the highest score of 99.64% of accuracy, GA overtakes ACO in terms of recommendation accuracy while ACO computed 67.12% lesser runtime than GA. Considering the results acquired, a new hybrid framework known as the HGA-ACO algorithm was proposed. The proposed HGA-ACO has a recommendation accuracy of 99.57% and achieved a 31.37% runtime reduction from GA. Thus, the proposed framework was observed to have improved the output accuracy of ACO and improved the processing time in GA, thus, improving the overall efficiency of the RS.

Keywords: Recommendation System; Metaheuristic Algorithms; Genetic Algorithm; Ant Colony Optimisation; HGA-ACO algorithm

I. INTRODUCTION

Recommendation systems (RS) are aimed to provide a suggestion of relevant items to users by filtering items based on the users' preferences or ratings. A recommendation system can play a huge role for users to make better decisions. A restaurant's RS is an example of recommendation systems that are widely used these days. As the number of restaurants grows each year, searching for restaurants that match our preferences becomes challenging, especially in places where we are not familiar (Varatharajan *et al.,* 2020). Therefore, with the help of a restaurant RS, searching for restaurants can be made easier and simplified. To achieve and create a good restaurant RS, the efficiency and accuracy of the recommendation algorithm in generating recommendations should be taken into consideration. Many research studies had been conducted to improve the efficiency and accuracy of a RS (Rajabpour *et. al.,* 2018; Tang & Wang, 2018; Roy *et. al.,* 2018; Mahadi *et al.,* 2018). Metaheuristic algorithm was explored in this paper to be implemented in the restaurant RS.

A metaheuristic is a high-level algorithmic framework that provides a set of strategies to develop an algorithm for problem-independent optimisation. In other words, a metaheuristic algorithm is a higher-level procedure designed to generate, find and select any partial search algorithm (heuristic) that can provide a sufficiently good solution to an optimisation problem, particularly a problem with imperfect or incomplete information as well as limited computational capacity. There are two common types of metaheuristic algorithms that are commonly used and implemented in the recommendation system, which are Genetic Algorithm (GA) and Swarm Intelligence (SI), due to their functionality and applicability. Ant Colony Optimisation (ACO) is one of the

*Corresponding author's e-mail: azleena.mk@usm.my

example of SI algorithm. Parekh *et al.* (2018) applied a hybrid model using the basis of GA in a book recommender system to integrate the outputs produced by every recommender to increase the systems' capabilities. Sultana *et al.* (2015) utilised GA and a content-based filtering technique in a RS for music files to overcome the weakness of existing recommendation techniques. Salehi *et al.* (2013) presented an Ant Colony metaheuristic-based recommender system for improvements in electronic commerce. Sobecki *et al.* (2010) used the ACO for a student's course RS to improve the recommendation solution. Sriphaew *et al.* (2015) proposed a modified ACO to be used in food tour recommendations to improve the accuracy of recommended itineraries.

There are still limitations for each metaheuristic algorithm and these limitations should be explored and understood, so that improvements can be made and determined. Thus, this study was conducted to compare the efficiency of metaheuristic algorithms between GA and ACO and design a framework that used the strengths and limitations of both metaheuristic algorithms to increase the overall efficiency of the RS. Therefore, the objectives of this research are 1) to study and find the metaheuristic algorithms that are used in the RS; 2) to compare the efficiency of GA and ACO in restaurant's RS, and 3) to design a framework that increases the efficiency of the metaheuristic algorithms in a restaurant's RS.

II. RELATED WORK

Janjarassuk *et al.* (2019) proposed a product RS using a Genetic Algorithm (GA) that allowed combinations of multiple products with combined features based on recommended customer's preferences. As a result, GA successfully recommended multiple products based on customer's preferences better than what the expert recommends in comparison. However, the algorithm lacked a stopping criterion and has a weak power unit recommendation. Implementation of GA with the improvement of the multi-purpose travel route recommendation system was presented by Yuan and Uehara (2019), by adding a memory module functions as a gene bank to store the genetic code of repeating individuals and the corresponding fitness value. As a result, adaptive memory GA managed to shorten the calculation time compared with the normal GA used in the same RS. Although the calculation time was reduced, the time required to perform the algorithm was still considered high.

A study to determine the effectiveness of various GA approaches was presented to improve multi-criteria recommendation systems (MCRSs) in movie recommendations (Hassan & Hamada, 2018). The various GAs used included standard GA (SGA) which used trial and error to set its parameter, adaptive GA (AGA) which used population fitness value to change the learning parameters and multi-heuristic GA (MGA) which used the concept of a simulated annealing algorithm to cool down the learning parameters. The results showed significant performance and accuracy compared to other algorithms tested. However, the RS could still be improved using a GA hybrid with other algorithms to train the network's knowledge and to deliver further impressive results.

Sivapalan (2015) contained three approaches in GA to overcome the cold problem caused by the lack of data in a recommender system. The first approach generated recommendations of items rated by users who had rated an item similar to an active user. The second approach was the same as the first approach but done repeatedly as a recall function to obtain a more precise result. The last approach generated recommendations of items rated by users who had a similar preference as the active user. By encapsulating all the approaches in the GA, the system can learn more about the user's preferences that allow the system to provide more accurate results although with minimal information. However, one of the limitations of this system was that it was not built to handle half-star ratings.

Samah *et al.* (2019) described the optimisation of the house purchase recommendation system (HPRS) with GA as the main algorithm. GA was implemented to face the problem of comparing house property websites according to the factors during the house survey. Due to the nature of GA mechanisms, HPRS successfully helped homebuyers in searching and buying a house based on their house preferences and budget in no time. However, the GA used in the system was not compared to other algorithms to determine its performance.

Ant Colony Optimisation (ACO) algorithm is categorised under Swarm Intelligence (SI) algorithm and has become the

main focus of this paper. Nonetheless, there are also other types of SI such as Whale Optimisation Algorithm (WOA), Bat Algorithm (BA) and Artificial Bee Colony (ABC) applied in RS. Sharma *et al.* (2019) and Tripathi *et al.* (2020) used the WOA in a RS. It was shown that WOA had high recall and precision, thus, could be used to obtain optimal clusters and generate relevant recommendations. Tripathi *et al.* (2020) used a new variant of WOA known as map-reduce-based tournament selection empowered WOA (MR-TWOA), giving a fair chance to the bad solutions to overcome the local optima during exploitation and to overcome the large-scale data set. Although MR-TWOA was able to handle extremely large datasets, it sometimes failed in the selection of the best solutions since it was still not unfolded to other real-world problems pertaining to big datasets. Yadav *et al.* (2018) employed the BA to improve a personalised recommendation, but the BA convergence rate slows down as it converged very quickly at the early stage. BA also needs a high number of evaluations to make it more accurate in finding the best recommendation for users.

Trust-based Ant Recommender System (TARS) using the ACO algorithm was presented by Bedi and Sharma (2012). This algorithm produced good recommendations by selecting a small and best neighbourhood based on the biological metaphor of ant colonies and incorporated a notion of dynamic trust between users. This algorithm also considered items and the number of neighbours involved in predicting ratings to enhance better decision-making for active users. Additional information was used along with the predicted rating to enhance recommendations in the system, which did not only improve user satisfaction, as the additional information act to boost the confidence of the predicted rating but also helped users make a better decision. Samia *et al.* (2018) proposed a recommendation for collaborative E-learning using ACO on top of Linked Open Data (LOD). The objective of this approach was to overcome the problem of novelty and diversity in the recommendation. The primary findings demonstrated the utility of exploring the LOD graph in ensuring diversity, while ACO was used in the system to search the LOD graph for relevant pathways and picked the best neighbour of an active learner to give a better and improved recommendation.

Leonardo *et al.* (2018) developed a modified Pareto ACO (MPACO) to handle the multiple-component redundant systems by using spare part list recommendations. Marginal analysis is used to create the initial population in the proposed MPACO. Moreover, it also offered a new method for performing the pheromone update step in the current solution based on the distances between subsequent locations. Numerical tests were carried out to show how the proposed MPACO could be used in three distinct multiple components redundant systems with varying levels of complexity. As a result, the proposed MPACO performed better compared to other algorithms.

Shi *et al.* (2018) proposed a restaurant RS based on the user's multiple features using an improved collaborative filtering method (ICFM). The ICFM considered the similarity of user's preferences, the user's influence and the interaction that follows. A user-based collaborative filtering approach was employed by Fakhri *et al.* (2019) for a restaurant recommendation system. The system worked by recommending a restaurant personally based on ratings given by other users. This was done by implementing two types of similarities to find the proximities between users which are user's attribute similarity and user's rating similarity on top of the collaborative filtering approach. As a result, the algorithm managed to produce accurate results of recommendation to the users as the accuracy was evaluated using the Mean Absolute Error (MAE) value. Wibowo and Handayani (2018) applied the GA for a restaurant RS by generating a travel itinerary, as an experimental study to produce a high-quality itinerary consisting of an efficient route to visit the recommended restaurants at a proper time. It was claimed that the algorithm could effectively solve the problem by tweaking some parts of the algorithm using a factorial design approach. As a result, the study successfully recommended restaurants based on certain constraints. However, the parameter used in the algorithm could be tweaked further to increase the performance of the genetic algorithm to produce better results.

III. METHODS ON RESTAURANT RECOMMENDATION

In this paper, the metaheuristic algorithms that were explored for the restaurant RS were the GA and ACO. The reason is that GA is known to have an easy encoding for the solution of the problems in a few arrays. While the ACO, it is determined to be one of the metaheuristic algorithms that can generate optimal solutions besides the GA (Bedi & Sharma, 2012). Both algorithms are well known to be able to find the best solution for a problem given to the system. Despite the advantages, such as giving the best solution to a given problem, some algorithms may have major drawbacks such as the computational cost. This depends on the different implementations of algorithms with different strategies of engagement. Hence, the differences between algorithms make it crucial to compare and analyse which algorithm is better in terms of computational cost, and memory usage and gives the best solution to the given problem. A hybridised approach of GA and ACO (HGA-ACO) was proposed in this paper to explore the possibility of improvement to the results achieved in both GA and ACO separately.

A. Datasets and Initial Settings

In the experiment setting, all the tested algorithms were developed using Java programming with the same dataset. The dataset contained features of the restaurant from 1,000 different restaurants in different locations. Table 1 shows the depiction of the dataset and the variables (features and location distance). In this dataset, there were three different price ranges of a restaurant: cheap, moderate and expensive. The restaurants were also differentiated based on the types of food that the restaurants mainly served. The restaurant category types were Malay, Chinese, Indian and Western. The maximum distance was set to 999 kilometres (km), whereas restaurants with a distance of less than 1 km were set as 0.

To standardise the experiments and ensure results consistency, the same input for features was used in the experiments and from the same location dataset to ensure the distance to the restaurant was the same for all the different experiments. Since the dataset was standardised, the comparison could be made between the algorithms, as they were designed to find the best possible list of restaurants to be recommended. The desired outcome is to have a list of 10 restaurants to be shown as the final recommendations to the users.

Table 1. The restaurant dataset and the variables used

Restaurant's ID	**Price Range**	**Restaurant Type**	**Distance**
sequential number	Cheap, Moderate, Expensive	Malay, Chinese, Indian, Western	0-999 kilometres (km)

The algorithms would be evaluated based on the computational cost and the value of solutions from the accuracy of the fitness value produced. The runtime as shown in Equation (1) is the total time taken by the algorithm to complete and calculated by taking the final time (algorithm stops) subtracted with the initial time (algorithm starts) in seconds. Meanwhile, accuracy (Equation (2)) was calculated by dividing the fitness value of the result by an algorithm with the maximum fitness value that could be gained (which is 30).

$$Runtime(s) = \left(time_{final} - time_{initial}\right) \quad (1)$$

$$Accuracy(\%) = \frac{fitness_{value_{obtained}}}{fitness_{value_{maximum}}} \times 100\% \quad (2)$$

B. Fitness Value

The fitness function uses the candidate's solution to the problem as input and produces as an output how good and fit the solution is, concerning a problem. In GA, the fitness function calculates a score on how good the chromosome is. Figure 1 shows an example of how the fitness value for a chromosome was calculated. In this example, assuming a customer's preference of a restaurant for the menu price range is "Cheap" and type of restaurant is "Chinese". The algorithm is set to compare each restaurant's attributes to the stated customer's preference. If any of the attributes is the same as the stated customer's preference, the fitness value for that chromosome is incremented by value 1. Note that the distance between the user and the restaurant is also taken into consideration. As for the distance, the formula as shown in Equation (3) was used to calculate the fitness value of the distance, d, where $RD_{current}$ is the current distance of the restaurant from the search location, and the RD_{max} is the maximum distance allowed in the search which was set to 999km.

$$fitness\_value_d = 1 - (\frac{RD_{current}}{RD_{max}}) \quad (3)$$

Customer Preference										
Price	Cheap (CP)									
Type	Chinese (CHN)									
Current Chromosome:										
Gene	1	2	3	4	5	6	7	8	9	10
ID	R51	R26	R17	R99	R65	R62	R49	R23	R32	R45
Price	EX	MD	CP	MD	EX	EX	CP	CP	CP	CP
Type	MLY	MLY	CHN	IND	CHN	WST	MLY	WST	IND	WST
Distance	169	358	705	827	995	436	902	382	718	726
Fitness Calculation of Current Chromosome										
Price	0	0	1	0	0	0	1	1	1	1
Type	0	0	1	0	1	0	0	0	0	0
Distance: *1-(distance / max distance)*	0.83	0.64	0.29	0.17	0.01	0.56	0.1	0.61	0.28	0.27
Gene fitness	0.83	0.64	2.29	0.17	1.01	0.56	1.1	1.61	1.28	1.27
Chromosome fitness	0.83 + 0.64 + 2.29 + 0.17 + 1.01 + 0.56 + 1.1 + 1.61 + 1.28 + 1.27 = **10.76**									
Legend:										
Abbreviation for Price					***Abbreviation for Type***					
EX	*Expensive*				**MLY**	*Malay*				
MD	*Moderate*				**CHN**	*Chinese*				
CP	*Cheap*				**IND**	*Indian*				
					WST	*Western*				

Figure 1. Example of fitness calculation

As shown in Figure 1, the chromosome had a fitness value of seven (7) since only seven restaurant's attributes across the genes in its chromosome fit the customer's preference. This value was then added with the total value calculated from the formula for the distance between the restaurant and the user which was 3.76. Therefore, the final fitness value for the current chromosome was equal to 10.76. The highest fitness value for a chromosome was capped at 30 since there were only two attributes inside each gene in each chromosome being compared with the customer's preference attributes as there were only 10 genes in the chromosomes [10 x 2 = 20]. Another 10 came from the final calculated value from the distance between the restaurant and the user. For ACO, this fitness value was used to calculate the solution accuracy from the list of restaurants created as the solution with the same calculation procedures.

C. Genetic Algorithm (GA) in the Restaurant Recommendation System

The process begins with an initial population that could be randomly generated or produced using other heuristics. The initial population for this GA consisted of 100 chromosomes containing 10 genes of different restaurants. For this problem, random initialisation was used to initialise the population.

Next, a fitness function was used as the indicator value, as explained in Section III(B). After that, the parent was selected from this population for mating. The chromosomes were evaluated whether to proceed with the mating process of crossover. Thus, the probability of the selected pair of chromosomes to perform mating was based on the crossover and mutation rates, respectively. In this study, the crossover rate of 0.95 was adopted, as it was reported to have good results on the population size range of 50 to 100, as reported by Asadzadeh (2015) and Capa and Ulusoy (2015). The mutation rate of 0.5 was reported to have a better quality of solution in their experiment. Thus, this mutation rate was adapted (Liu *et al.,* 2016). Lastly, existing individuals in the population were replaced by these newly generated off-springs and the process repeats until termination criteria are reached (Hassan & Hamada, 2018; Sivapalan, 2015). The selection phase ensured the next generation was better than the previous generation. To retain the number of chromosomes to be the same as the number of chromosomes in the initial population, tournament selection is carried out, whereby chromosomes will be sorted or ranked based on their fitness value in descending order, and the best 100 chromosomes are retained in the next generation of population. In this experiment, the GA ended when one of the following termination conditions were met: 1) When an absolute number of generations, 5,000 is obtained; 2) When the fitness value of the chromosome reaches 30.

D. Ant Colony Optimisation (ACO) in the Restaurant Recommendation System

Ant Colony Optimisation Algorithm (ACO) uses the concept of ants which favour community sustainability rather than as individual species (Menezes *et al.,* 2019). Ants communicate between them using touch, sound and pheromone. ACO algorithm uses pheromones to observe the movement of the ants from their nests to search for food in the shortest possible path. Ants are known to mostly live on the ground, thus the surface of the ground will preserve the ants' pheromone trails as they move around, and these trails are then followed by other ants. ACO follows this concept by monitoring the movement of the ants from their nests to find the food from the path with the most pheromone. At first, ants move randomly around their nest in search of food, then

more routes will emerge from the nest to the food source. Then, the ants will carry some portions of the food while leaving pheromone trails on its return path. Based on these pheromone trails, the probability of selection of the other following ants for that specific path would be a guiding factor to the food source. This probability is based on the concentration and the rate of evaporation of pheromone (Bedi & Sharma, 2012).

In this study, the initial population for the ACO algorithm was created with random initialisation, which consisted of a list of 100 different restaurants. Pheromone function for ACO algorithm would check the attribute of the menu's price range, types of cuisine and the distance for each restaurant. It would check whether each of the restaurants satisfies the user's condition by choosing the highest value of the pheromone function. Hence, the highest pheromone value and the highest probability of the list would be selected. The pheromone function would then calculate a score based on each attribute of the restaurant in each list. If the attribute of the restaurant in a current list satisfies the user's preference, the pheromone value for that list was incremented by 1.

Given an example where the input for the restaurant price range was 'Cheap' and the category was 'Chinese', each list (from list 1 to list 100) was compared to the input and the pheromone value of each list was calculated. Figure 2 shows the example of how the pheromone value was calculated for three lists.

The first list (list [1]) as shown in Figure 2 had the fitness value of 10.76. Next, for the list to be accepted, the fitness value was set to be greater than a benchmark value of 10. This is because, in the ACO experiment, it was found that the fitness value under 10 was unreliable, which decreased the result reliability. When this condition was fulfilled by the list, then the algorithm would increment the pheromone value of each restaurant. The initial pheromone value was 0 for each restaurant. The pheromone value of restaurants in list [1] was updated. As shown in Figure 2, the fitness value for list [1] was 10.76 and the list was accepted for pheromone incrementation.

Thus, the pheromone value of all the restaurants in the list would be incremented by 1 from their previous pheromone value. The second list (list [2]) had a fitness value of 13.69, thus, the pheromone value in each restaurant in list [2] would be updated with increment as well. There were some restaurants from list [1] that appeared again in list [2], thus when list [2] was accepted for pheromone value increment, these repeating restaurants had a higher value of pheromone as they had been incremented twice. For the third list (list [3]), the fitness value was 4.85, and thus, the pheromone value for the restaurants in this list would not be incremented. Towards the end of the ACO process, to select the best restaurant's population, each of the restaurants was ranked based on their pheromone value, sorted in descending order. Then, only the first 10 of the restaurant's population with higher pheromone values were selected as top 10 recommendations, while the rest were discarded. The fitness value of the last list was calculated and will be the final fitness value achieved by ACO.

Fitness Calculation of List [1]										
Restaurant ID	*51*	*26*	*17*	*99*	*65*	*62*	*49*	*23*	*32*	*45*
Price	0	0	1	0	0	0	1	1	1	1
Type	0	0	1	0	1	0	0	0	0	0
Distance	0.83	0.64	0.29	0.17	0.01	0.56	0.1	0.61	0.28	0.27
Total fitness	0.83 + 0.64 + 2.29 + 0.17 + 1.01 + 0.56 + 1.1 + 1.61 + 1.28 + 1.27 = **10.76**									
Fitness Calculation of List [2]										
Restaurant ID	*45*	*23*	*99*	*11*	*5*	*122*	*259*	*59*	*17*	*705*
Price	1	1	0	1	0	0	1	1	1	0
Type	0	0	0	0	1	0	0	1	1	0
Distance	0.27	0.61	0.17	0.88	0.71	0.06	0.1	0.65	0.29	0.95
Total fitness	1.27+1.61+0.17+1.88+1.71+0.06+1.1+2.65+2.29+0.95 = **13.69**									
Fitness Calculation of List [3]										
Restaurant ID	*45*	*122*	*65*	*259*	*77*	*705*	*9*	*68*	*99*	*40*
Price	1	0	0	1	0	0	0	0	0	0
Type	0	0	1	0	0	0	0	1	0	0
Distance	0.27	0.06	0.01	0.1	0.03	0.95	0.16	0.05	0.17	0.05
Total fitness	1.27+0.06+1.01+1.1+0.31+0.95+0.16+1.05+0.17+0.05 = **4.85**									

Calculation of pheromone values:

Initial PV	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Restaurant ID	5	9	11	17	23	26	32	40	45	49	51	59	62	65	68	77	99	122	259	705
After evaluation of list [1] (fitness 10.59)																				
Updated PV	0	0	0	1	1	1	1	0	1	1	1	0	1	1	0	0	1	0	0	0
After evaluation of list [2] (fitness 13.69)																				
Updated PV	1	0	1	2	2	0	0	0	2	0	0	1	0	0	0	0	2	1	1	1
After evaluation of list [3] (fitness 4.85)																				
Updated PV	1	0	1	2	2	0	0	0	2	0	0	1	0	0	0	0	2	1	1	1

*PV= Pheromone Value

Figure 2. Example of calculation and update of pheromone value

IV. HYBRIDISING GA AND ACO ALGORITHM IN THE RESTAURANT RECOMMENDATION SYSTEM

From the results obtained from ACO and GA, which is presented in the next section of this paper, both algorithms showed their strengths and limitations. ACO took lesser time to compute with less accurate results, while GA took more time to compute with more accurate results. Therefore, to overcome the limitations for each of the algorithms, a hybrid framework was proposed, whereby both algorithms were combined, known as the HGA-ACO for the restaurant recommendation system.

In this framework, the lists resulted from ACO was employed as the initial population of GA. The population

obtained from ACO would then undergo the steps in GA. Furthermore, the GA computing time was expected to be improved as the GA algorithm used the population from ACO as its initial population instead of the random initial population. Therefore, this would take less time for the GA to satisfy its termination condition. With the proposed framework, both accuracy and computation time was expected to have some improvements. Figure 3 shows the flowchart of the proposed hybrid HGA-ACO algorithm for the restaurant RS. The framework begins with the ACO Algorithm, where the randomly generated initial population undergo the processes in ACO. The processes in ACO comprise population initialisation (randomly generated population), pheromone function and selection. Then, the population generated from ACO will be passed to the GA process. The processes in GA comprised of the calculation of the fitness function, crossover, mutation, selection and termination.

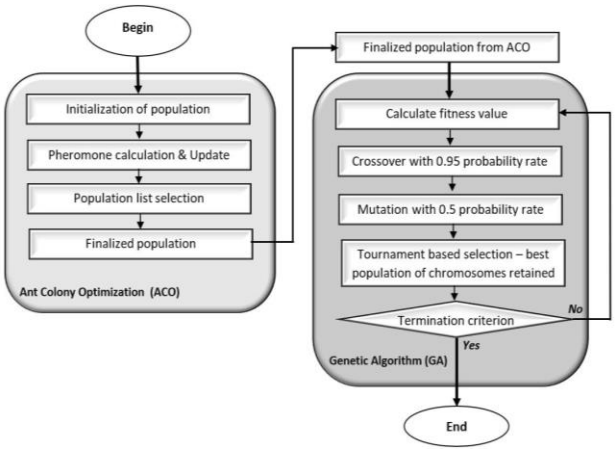

Figure 3. Flowchart of the proposed HGA-ACO framework

V. RESULTS AND DISCUSSION

In this section, the results from experiments carried out on the restaurant dataset are presented. Section 5.1 presents the results from the GA and ACO algorithms, respectively. As mentioned in Section 4, these results became the motivation to develop a new hybrid algorithm from GA and ACO for the restaurant recommendation system. Thus, in Section 5.2, the results from the proposed HGA-ACO for the restaurant recommendation system are presented and discussed.

A. Results of GA and ACO in the Restaurant Recommendation Systems

The experiments were conducted in ten runs with the same users' preferences (input) where four different combinations of users' preferences were fixed for the menu price range and restaurant food type, as shown in Table 2. This was to moderate and standardise the input for the experiments.

The distance of the restaurant would then be computed based on the restaurant selected as the population in the GA's chromosomes or ACO's lists, to calculate the fitness value by comparing the restaurant's variables with the users' preferences. In each run, the runtime of the algorithm and the accuracy of the recommendation was recorded. The runtime of the algorithm was calculated in the second (s) unit. The accuracy of the recommendation from the fitness value indicated how many recommended restaurants fit the preferred choice of the price range, food type and distance. The runtime and accuracy were calculated using Equations (1) and (2), respectively.

Table 2. Restaurant dataset and the variables used

Input	Parameter of Preferences		
	Price range	**Food type**	**Combination code**
1	Cheap	Chinese	CC
2	Expensive	Indian	EI
3	Moderate	Western	MW
4	Cheap	Malay	CM

Since a solution of recommended restaurants consisted of ten restaurants, the best and maximum fitness value was 30. Thus, the evaluation criteria could be categorised into three: the fitness value, the accuracy (in %) and the time taken (in seconds). The average value, μ, of the 10 runs of each evaluation criteria was computed. The results obtained from experiments done on the GA and ACO were presented in Table 3 where it shows the average value, μ, for each of the different inputs in each of the evaluation criteria.

Table 3. Compiled results from GA and ACO

Criteria	Algorithm / Input type	GA: Average of each input, μ	GA: Final average, μ	ACO: Average of each input, μ	ACO: Final average, μ
Fitness	CC	29.91	**29.89**	19.03	19.60
	EI	29.88		20.58	
	MW	29.94		20.31	
	CM	29.84		18.49	
Accuracy %	CC	99.70	**99.64**	63.43	65.34
	EI	99.60		68.60	
	MW	99.80		67.70	
	CM	99.47		61.63	
Time (s)	CC	5.98	5.93	2.00	**1.95**
	EI	6.16		1.89	
	MW	5.88		2.00	
	CM	5.71		1.90	

To show the differences based on each evaluation criterion between these two algorithms, the final average values of all users' preferences combination of each criterion were calculated. The best value of the final average for each criterion is highlighted in bold in the table. Based on the results, it was found that GA had the highest average value of fitness (29.89) compared to ACO (19.60). It also showed an average of 99.64% accuracy, meanwhile, ACO only provided 65.34% of accuracy. However, the runtime for GA provided an output of 67.12% more time than ACO which was calculated as shown in Equation (4). Therefore, this study can conclude that each algorithm had its strength and limitation. In terms of accuracy, GA was better than ACO while in terms of runtime, ACO provided better output than GA.

$$runtime_{increase} = \left(\frac{GA_{runtime} - ACO_{runtime}}{GA_{runtime}}\right) \times 100\% \quad (4)$$

B. Results of HGA-ACO in the Restaurant Recommendation Systems

To test the efficiency of the proposed algorithm (HGA-ACO), this algorithm was tested using the same setting as the experiments for GA and ACO. The experiment was carried out in ten runs on the same 1,000 restaurant datasets and tested with input preferences combinations, as shown in Table 2. There were 3 evaluation criteria, but the main criteria to be compared were the accuracy of the algorithm and the time taken as these were the two issues found in GA and ACO.

The proposed HGA-ACO was expected to benefit from both the strength and overcome each other's weaknesses. The results of the final average from each criterion obtained from the HGA-ACO experiment is shown in Table 4, together with the results achieved from GA and ACO.

Table 4. The comparison between the algorithms

Criteria	Final average, μ: GA	ACO	HGA-ACO	Improvement Rate, IP of HGA-ACO: From [a]$Algo_r$	IP %
Fitness	29.89	19.60	29.87	GA	-0.07
				ACO	52.34
Accuracy %	99.64	65.34	99.57	GA	-0.07
				ACO	52.38
Time (s)	5.93	1.95	4.07	GA	31.37
				ACO	-108.72

a. $Algo_r$ are the original algorithms, whether its GA or ACO compared to the hybrid HGA-ACO

The final compilation of all these tested algorithms (GA, ACO and HGA-ACO) focused on the final average values of all users' preferences combination. From the results, it was proven that the accuracy of ACO results improved in HGA-ACO after its generated population underwent the processes in GA, which increased its accuracy from an average of 65.34% to 99.57% recommendation accuracy. The results from HGA-ACO also proved that GA computing time was reduced from 5.93 seconds to 4.07 seconds.

To calculate the improvement rate, the formula is shown in Equation (5), where the difference Δ of the average value of the respective evaluation criteria, μ of the initial algorithm (GA or ACO), $Algo_r$ with the hybrid algorithm (HGA-ACO), $Algo_h$ is divided with the average value of $Algo_r$.

$$IP = \left(\frac{\Delta\ \mu Algo_r\ \ \mu Algo_h}{\mu Algo_r}\right) x\ 100 \quad (5)$$

In terms of the quality of the solution, the HGA-ACO showed improvement from the original ACO, with the improvement rate of 52.34% for the fitness value and 52.38% for the accuracy value. Although the HGA-ACO had slight declination from the original GA, but the values are almost near to 0, with -0.07% for both fitness value and accuracy. Thus, it can be concluded that in terms of the quality of solution, the improvement rate in the HGA-ACO had shown a good result.

In terms of performance based on time, the results from HGA-ACO also proved that GA computing time was improved (reduced) with the improvement rate of 31.37% runtime reduction from the original GA. This was because GA by itself with a random initial population caused it to take a longer time to reach the final result. With HGA-ACO, the population initialisation was done by ACO, and thus, helped the overall process time taken from traditional GA to be reduced. However, if HGA-ACO is compared to performance time in ACO, the processing time had increased, where the improvement rate declined to -108.72%.

The findings from these experiments showed that newly proposed hybrid framework known as HCA-ACO improved the ACO to find a close to accurate result and helped improve the original GA to take a shorter time to reach the final desired result.

VI. CONCLUSION

In this paper, a RS using GA and ACO was evaluated using a list of restaurants and its attributes as data and output lists of recommended restaurants. This was the first objective of this paper. The second objective was to compare the efficiency of GA and ACO in a restaurant RS and from the experiments. It was found that limitations were determined based on the results obtained by each of the algorithms, which had higher GA runtime and less accuracy in the ACO results. Therefore, to fulfil the third objective, a framework was proposed by combining both algorithms, which created a hybrid HGA-ACO algorithm.

The proposed framework overcame the limitations of each algorithm (GA and ACO, respectively), where it improved the accuracy of ACO by 52.38% and reduced GA runtime by 31.37%. For future research, implementation of GA and ACO in a restaurant RS can perhaps be implemented using more attributes and carry-on to further improve the accuracy of the result. Besides that, further improvement can be carried out by hybridising other metaheuristic algorithms, for instance, the use of Simulated annealing with the GA operators in order to improve the global search of the solution. This can be followed by an extended comparison to other existing works that have applied similar hybridised approaches in their solutions.

VII. ACKNOWLEDGEMENT

Acknowledgement to "Ministry of Higher Education Malaysia for Fundamental Research Grant Scheme with Project Code: FRGS/1/2020/ICT02/USM/02/4.

VIII. REFERENCES

Asadzadeh, L 2015, 'A local search genetic algorithm for the job shop scheduling problem with intelligent agents', Computers and Industrial Engineering, vol. 85, pp. 376–383.

Bedi, P & Sharma, R 2012, 'Trust based recommender system using ant colony for trust computation', Expert Syst. Appl., vol. 39, pp. 1183–1190.

Beldjoudi S, Seridi-Bouchelaghem, H & Karabadji, NE 2018, 'Recommendation in collaborative E-Learning by using linked open data and ant colony optimization', Intelligent Tutoring Systems, pp. 23–32.

Çapa, C & Ulusoy, G 2014, 'Proactive project scheduling with a bi-objective genetic algorithm in an R&D department', in the Proceedings of the 2015 International Conference on Industrial Engineering and Operations Management, pp. 966–971.

Fakhri, AA, Baizal, A & Setiawan, EB 2019, 'Restaurant recommender system using user-based collaborative filtering approach: A case study at Bandung Raya region', Journal of Physics Conference Series, 1192:012023.

Hassan, M & Hamada, M 2018, 'Genetic algorithm approaches for improving prediction accuracy of multi-criteria recommender systems', Int. J. Comput. Intell. Syst., vol. 11, pp. 146–162.

Janjarassuk, U & Puengrusme, S 2019, 'Product recommendation based on genetic algorithm', in the 2019 5th International Conference on Engineering, Applied Sciences and Technology (ICEAST), pp. 1–4.

Liu, Z, Li, X, Jiang, J & Wang, S 2016, 'A novel improved quantum genetic algorithm for robot coalition problem', in the 2016 IEEE International Conference on Information and Automation (ICIA), pp. 2061–2064.

Mahadi MI, Zainuddin N, Shah NBA, Naziron NA, Rum SFM 2018, 'E-Halal restaurant recommender system using collaborative filtering algorithm', Journal of Advanced Research in Computing and Applications, vol. 12, no. 1, pp. 22-34.

Menezes BAM, Kuchen H, Amorim Neto HA & Lima Neto F Bde 2019, 'Parallelization strategies for GPU-Based ant colony optimization solving the traveling salesman problem', in the 2019 IEEE Congress on Evolutionary Computation (CEC), pp. 3094–3101

Parekh P, Mishra I, Alva A, and Singh V 2018, 'Web based hybrid book recommender system using genetic algorithm', International Research Journal of Engineering and Technology, vol. 5, no. 8, pp. 1536–1539.

Rajabpour, N, Mohammadighavam, A, Naserasadi, A & Estilayee, M 2018, 'TFR: A tourist food recommender system based on collaborative filtering', International Journal of Computer Applications, vol. 181, pp. 30–39.

Rodrigues, LR, & Gomes, J 2018, 'Spare parts list recommendations for multiple-component redundant systems using a modified pareto ant colony optimization approach', IEEE Transactions on Industrial Informatics, vol. 14, pp. 1107–1114.

Roy D, Agarwal, R & Jain, H 2019, 'An efficient ensemble machine learning system for restaurant recommendation', International Research Journal of Engineering and Technology (IRJET), vol. 6, no. 10, pp. 468–475

Salehi, M, Fathi, A & Abdali-Mohammadi, F 2013, 'ANTSREC: A semantic recommender system based on ant colony meta-heuristic in electronic commerce', International Journal of Advanced Science and Technology, vol. 56, pp. 119–130

Samah, K, Badarudin, I, Odzaly, EE, Ismail, K, Nasarudin, N, Tahar, NF & Khairuddin, M 2019, 'Optimization of house purchase recommendation system (HPRS) using genetic algorithm', Indonesian Journal of Electrical Engineering and Computer Science, vol. 16, pp. 1530–1538

Sharma, B, Hashmi, A & Kumar, A 2019, 'Whale optimization-based recommendation system', in the International Journal of Innovative Technology and Exploring Engineering (IJITEE), vol. 8, no. 10, pp. 2640–2644

Shi, Y, Zhao, Q, Wang, Y & Cao, J 2018, 'An improved restaurant recommendation algorithm based on user's multiple features', in the 2018 IEEE 9th International Conference on Software Engineering and Service Science (ICSESS), pp. 191–194.

Singh A & Deep, K 2019, 'Artificial bee colony algorithm with improved search mechanism', Soft Computing, vol. 23, pp. 12437–12460.

Sivapalan, S 2015, 'A genetic algorithm approach to recommender system cold start problem', PhD thesis, Ryerson University, Canada.

Sobecki, J & Tomczak, JM 2010, 'Student courses recommendation using ant colony optimization', ACIIDS, 124–133.

Sriphaew, K & Sombatsricharoen, K 2015, 'Food tour recommendation using modified ant colony algorithm', in the Proceedings of the 5th International Conference on Computing and Informatics (ICOCI)

Sultana, T, Sharma, N, Aher, S, Pate, N & Kinikar, M 2015, 'Recommendation system for music file', International Journal of Computer Science and Information Technologies, vol. 6, pp. 275–271.

Tang, J & Wang, K 2018, 'Ranking distillation: Learning compact ranking models with high performance for recommender system', in the Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, pp. 2289–2298.

Tripathi, AK, Mittal, H, Saxena, P & Gupta, S 2020 'A new recommendation system using map-reduce-based tournament empowered whale optimization algorithm', Complex & Intelligent System, vol. 7, no. 1, pp. 297–309.

Varatharajan N, Guruprasad J & Mathumitha K 2020, 'Restaurant recommendation system using machine learning', International Educational Applied Research Journal (IEARJ), vol. 4, no. 3.

Wibowo, BS & Handayani, M 2018, 'A genetic algorithm for generating travel itinerary recommendation with restaurant selection', in the 2018 IEEE International Conference on Industrial Engineering and Engineering Management (IEEM), pp. 427–431.

Yadav S, Vikesh, S & Nagpal S 2018 'An improved collaborative filtering-based recommender system using bat algorithm', in the International Conference on Computational Intelligence and Data Science (ICCIDS).

Yuan, C & Uehara, M 2019, 'Improvement of multi-purpose travel route recommendation system based on genetic algorithm', in the 2019 Seventh International Symposium on Computing and Networking Workshops (CANDARW), pp. 305–308.