A hybrid technique for single-source shortest path-based on A* algorithm and ant colony optimization

Article · June 2020

DOI: 10.11591/ijai.v9.i2.pp256-263

4 authors, including:

Sameer Alani
Universiti Kebangsaan Malaysia
49 PUBLICATIONS **548** CITATIONS
SEE PROFILE

Mustafa Maad Hamdi
Al-Maarif University College
84 PUBLICATIONS **609** CITATIONS
SEE PROFILE

Sami Abduljabbar Rashid
Al-Maarif University
34 PUBLICATIONS **365** CITATIONS
SEE PROFILE

A hybrid technique for single-source shortest path-based on A* algorithm and ant colony optimization

Sameer Alani[1], Atheer Baseel[2], Mustafa Maad Hamdi[3], Sami Abduljabbar Rashid[4]
[1]Department of Computer Technical Engineering, Al-Kitab University College, Kirkuk, Iraq
[2]Information technology center, University of Anbar, Anbar, Iraq,
[3,4]Faculty of Electrical and Electronic Engineering, Universiti Tun Hussein Onn Malaysia 86400 Parit Raja, Batu Pahat, Johor, Malaysia
[3]Department of Computer Engineering Technology, Al-Maarif university college, Al-Anbar, Iraq

Article Info

Article history:

Received Jan 10, 2020
Revised Apr 13, 2020
Accepted Apr 28, 2020

Keywords:

A* algorithm
Ant colony optimization
Parking system
Single-source shortest path

ABSTRACT

In the single-source shortest path (SSSP) problem, the shortest paths from a source vertex v to all other vertices in a graph should be executed in the best way. A common algorithm to solve the (SSSP) is the A* and Ant colony optimization (ACO). However, the traditional A* is fast but not accurate because it does not calculate all node's distance of the graph. Moreover, it is slow in path computation. In this paper, we propose a new technique that consists of a hybridizing of A* algorithm and ant colony optimization (ACO). This solution depends on applying the optimization on the best path. For justification, the proposed algorithm has been applied to the parking system as a case study to validate the proposed algorithm performance. First, A*algorithm generates the shortest path in faster time complexity. ACO will optimize this path and output the best path. The result showed that the proposed solution provides an average decreasing time complexity e is 13.5%.

Corresponding Author:

Sameer Alani,
Department of Computer Technical Engineering,
Al-Kitab University College,
Kirkuk, Iraq.
Email: sameer@siswa.ukm.edu.my

1. INTRODUCTION

This paper presents a hybrid technique for single-source shortest path-based on a* algorithm and ant colony optimization. Recently, time and cost consumption are among the effective factors in human life. This in turn, triggers the attention to reconsider the shortest path issue in different transportation engineering industries and applications. This reconsideration is clearly seen in the contemporary advances in the field of Intelligent Transportation Systems (ITS) [1-2]. To be more specific, these advances are illustrated in the productions of the real-time Automated Vehicle Dispatching System (AVDS) and the vehicle Route Guidance System (RGS) in which these systems require the shortest path guidance with rapid time and precise means [3-4]. In cases of a heavy traffic jam, the RGS commands a built-in device to calculate the ideal paths. The suggested paths are generated in a very short time. Similarly, the AVDS generates the ideal paths and timetables in a practical period after the service is requested by a customer. The ideal paths and timetables depend on the travel time which is vital to the city traffic settings. During the optimization process, a minimum path algorithm is covertly required to be frequently applied [5]. Nowadays, the shortest path algorithm is largely utilized. It constructs the foundations of several issues including the tree issue and the network flow issue, among others. In this regard, several algorithms are utilized in generating the shortest path. Among these algorithms are the Ant colony optimization algorithm, Floyd algorithm, Genetic algorithm, A* algorithm and Dijkstra [6-7]. The employment of such algorithms in solving the shortest path issue can be in several areas including VANET, MANET, and networking [8-9]. To validate the proposed method, the proposed method has been applied in the parking system layout. The remaining parts of this paper are structured as follows. Section one is introductory. Section two discusses the literature relating to the traditional algorithms utilized in generating the shortest path including the Genetic Algorithm, Ant Colony Optimization, Dijkstra's algorithm, and A*algorithm. Section three is the research methodology that describes the integration of ant colony optimization and A* as a model for generating the shortest and ideal path in a very short time. Section five presents and debates on the research finding. Section six summarizes and concludes this paper with suggested points for future studies.

2. LITERATURE REVIEW

Several types of research have been conducted to tackle the shortest path issue. However, only a few types of research have successfully reached this end offering algorithms that generate the shortest path in practical time and in a precise way. Dijkstra's algorithm and A*algorithm are among these few algorithms and the most popular ones. Dijkstra's algorithm generates travel time between two nodes. It produces the shortest path tree by explaining the issue of the one source for the shortest path in a graph search [10-11]. Several applications use Dijkstra's algorithm in calculating the shortest path. In this respect, a Hybrid Genetic algorithm has been integrated with Dijkstra's algorithm in order to offer a solution for the shortest path issue of the Mobile Robot [12]. This integration resulted in a model that requires less time and generates less repetition. This model is largely applicable to great dimensional issues. In 2018, the applicability of the Bellman-Ford Algorithm and Dijkstra's Algorithm had been tested for solving the shortest path issue [13-14]. The findings of this test showed that Dijkstra's algorithm functions more efficiently for many nodes and that the Bellman-Ford Algorithm functions more efficiently for the small number of nodes. A* algorithm is the result of Dijkstra's algorithm with an empirical search, leading A* algorithm to path planning quicker than Dijkstra's algorithm. To determine the shortest path for an Indoor Positioning System, a study has tested the algorithms of A* and Dijkstra's [15-16]. The study found that due to Dijkstra's frequent tries to improve an initial approximation (cost) of each node, A* algorithm is the fastest. In other words, Dijkstra's algorithm requires more time to arrive at the target node. Another study modified the path planning of A* algorithm for mobile robots [17]. The study followed the norms of the Jump Point Search (JPS) in generating its A* algorithm. The findings of this study showed that this modification made the path longer than that of A* algorithm. Nevertheless, the emerging algorithm is applicable for rapidly finding a path. Moreover, Dijkstra's algorithm offers an ideal solution to the shortest path issue, but it is slow compared to other algorithms. A* algorithm offers a quick solution to the shortest path issue by evaluating its search direction [18-19]. Therefore, A* algorithm is employed for path optimization for both computer games and parking lot designs [20]. The literature review suggests that Dijkstra's algorithm is precise but slow because it analyses all graph nodes. It also suggests that A* algorithm requires less time since it only analyses the estimated shortest path. Consequently, both algorithms are to be optimized by a provisional optimization algorithm such as a graphetically search algorithm, ant colony algorithm or a genetic algorithm. Ant colony optimization (ACO) is an algorithm-generated following the norms of ants in real life in their attempts to reach the ideal path from colony to food [21-22]. To find the ideal food path, ants use trail pheromone along the way from colony to food. Thus, other ants follow the trail. In the case of short and long paths, ants tend to heavily follow the shortest path leading to a strong trail pheromone. Eventually, more ants follow the shortest path since it has a heavy pheromone that appeals ants more [23]. Ultimately, all ants find the ideal shortest path. Recently, ACO has triggered the researchers' interest in reconsidering complicated transportation issues including control issues, traffic engineering, public transit, and vehicle routing and scheduling [24]. The Meta-Heuristics in Short Scale Construction contains Genetic Algorithm and Ant Colony Optimization. A model has integrated ACO to reach certain information based on some preferences rather than an extensive engine's search. Therefore, this integration allows users to spend less time waiting for search results. Another model has suggested the integration of the High-order Graph Matching Based and the Ant Colony Optimization [25-26]. This model presented a method of a problem-specific pheromone initialization. It also redefined the domestic and international pheromone rules based on the latest updates. The findings of this model argue that the emerging algorithm reached a higher efficiency than that of the three state-of-the-art methods.

3. PROPOSED METHOD

The proposed model of hybridizing Ant colony optimization and A* algorithm is designed to find the ideal path in a short time. The first step is the initial distance estimation generated by A* algorithm between the source node represented in the parking lot gate and the destination represented in the building entrance. A* algorithm is significant here because it grants a value for each node of the space configuration as follows:

$$f(v) = h(v) + g(h) \tag{1}$$

In this equation, n is the following node on the trail, g(n) represents the value of the trail between the starting node to N, and h(n) is an empirical estimation of the value of the most effective path from n i.e. the building entrance. A* algorithm determines the most efficient path it may consider reaching the target destination. When the empirical test of the ideal path is successful, the estimated value is proven correct. The layout design of the parking allows A* algorithm to initialize *h(x)*, the estimated value of a straight line between the starting point i.e. the gate to the ending point i.e. the entrance of the building. The priority queue is performed at this stage to test the frequent option for the nodes of the minimum value to expand. This priority queue is called the open set. At all steps, nodes with a value less than *f(x)* are unselected from the queue leaving the values *f* and *g* as well as their neighboring nodes to be selected and calculated accordingly.

As discussed, the neighboring nodes are included in the priority queue. Consequently, the search algorithm resumes its search until it reaches a node with the lowest value in the queue less than *f*. The value of the shortest path is the value of *f* considering *h* at a zero value for the building entrance within an acceptable empirical test. Furthermore, ACO optimizes the resulting path to generate the ideal path from the gate as a starting point to the entrance as a final destination. At the early time, ants randomly search for food to return with them to their colony. Using pheromone, ants mark their path to food leading other ants to follow this path. The path that has the highest value of pheromone is the one leads to food. Thus, the value is determined by the pheromone-based on the usage of such by ants. In this proposed model, food is given the value of (0, 1). In the case of an empty parking bay, the status of the parking is zero (0) which refers to food for ants. In the case of a full parking bays, the status of the parking is one (1) which refers to no food for ants. On another perspective, ants use pheromone for the path they followed for food. However, this pheromone fades over time when the path is not used. This case suggests that ants follow another path of food. This process updates ants with the latest ideal path for food. Therefore, in this model, ants are referred to as a number and pheromone are referred to as a value that changes based on ants' usage of paths. Finally, when ants find an empty parking bay within the parking suggested by A* algorithm, it begins its path optimization according to certain measures. In the beginning, the ACO will initialize its parameters which are the nodes and ants' number, the initial path is generated by A* algorithm and the pheromone. Then ACO will set the distances of the nods and the matrices. Once all the parameters are ready, the ACO will start by the first ant which is K assigned as 1 and locate it on the initial node which is the gate used by the car. Then the ant will start from the first node and will define a random number from zero to one. The random number is defined as 0.5. If this random number is smaller than the initial path which is q0, that means there is a path and ANT should follow this path which means perform (1). Otherwise ANT will explore new paths then perform (2).

$$S = \begin{cases} argmax\, u \in\ jk\ (r) \left\{[\tau(r,u)]^{\alpha}, [\eta\ (r,u)]^{\beta}\right\}) \\ s_{,}\ otherwise\ (biased\ exploration) \end{cases} \tag{2}$$

If the random number is bigger than the initial path which is q0, equation 2 will be applied.

$$P_{k(r,s)} \begin{cases} \frac{[\tau(r,s)]^{\alpha}, [\eta\ (r,s)]^{\beta}}{\sum_{u \in jk(r)} [\tau(r,u)]^{\alpha}, [\eta\ (r,u)]^{\beta}},\ if\ s \in jk(r) \\ 0_{,}\ otherwise \end{cases}$$

Then the ant will deposit pheromone on the edges as in (3).

$$\tau(r,s) \leftarrow (1-p).\tau(r,s) + p.\Delta\tau_0 \tag{3}$$

$$\Delta\tau_0({}^{1}/_{l_{mn}}.n) \tag{4}$$

where $0 < \rho < 1$ is a parameter. The term $\Delta\tau ij$ may be defined as (4).
This step will be repeated until all the ants are initialized. If all the ants are initialized, there will be 10 ants in this case. Then ACO will determine the best route of the iteration by depositing additional pheromone on each visited edge using (5).

$$\tau(r,s) \leftarrow (1-p).\tau(r,s) + p.\Delta\tau(r,s) \tag{5}$$

The pheromone level update is performed at the end of an iteration using the formula (6).

$$\Delta\tau(r,s) = \begin{cases} \frac{1}{L_{gb}}, & \text{if } (r,s) \in \text{globalbest tour} \\ 0, & \text{otherwise} \end{cases} \tag{6}$$

where ρ is the pheromone decay parameter ($0 < \rho < 1$), and L best is either LGB (the length of the globally best tour since the start of algorithm execution) or Lib (the length of the best tour found during the current iteration of the algorithm). Then the best path will be determined and showed to the user. The flowchart of the proposed solution is shown in Figure 1.

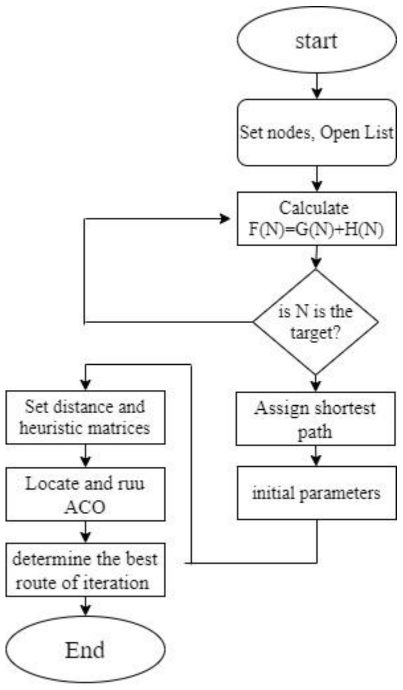


Figure 1. The proposed solution

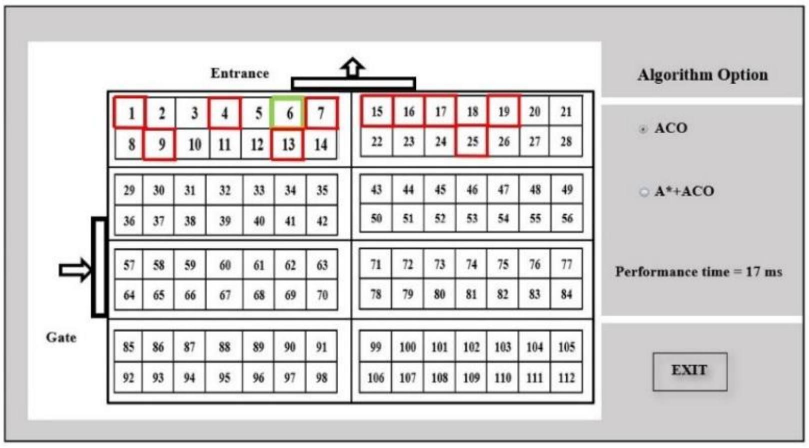
4. SIMULATION ENVIRONMENT

The simulation has been done using two algorithms in Java software. The paths and the parking lot in Java are represented as nodes. The simulation has been done on a workstation computer with high specifications. The CPU was Core i7 and 16GB for RAM. Exit button, algorithm option and the computing time are presented in the GUI as well. Two scenarios and three cases have been selected to show and prove the differences in the time complexity between the traditional ACO and the proposed algorithm (ACO+A*).

5. RESULTS AND DISCUSSION

In this section, we evaluate the proposed algorithm based on different cases. To evaluate and compare the searching performance based on the proposed algorithm. Finally, it shows that the proposed algorithm found the best solution in almost all cases.

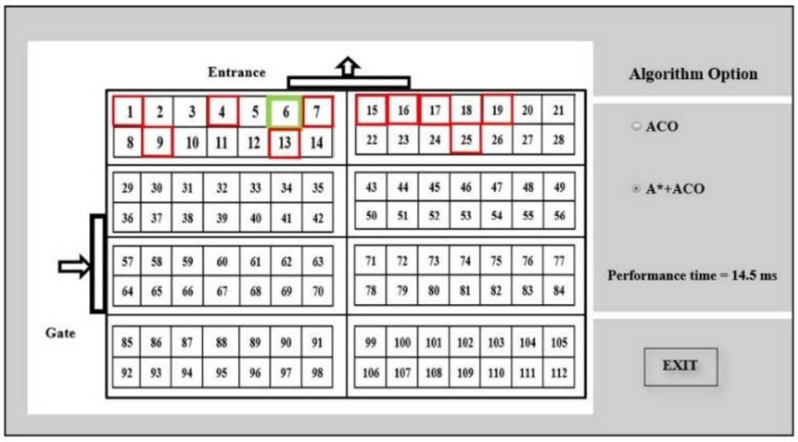
CASE1:
In the first case, as shown below. ACO has been applied to find the nearest parking lot. The parking lot number 6 has been assigned as the nearest parking to the entrance. Moreover, the performance algorithm time was 17 ms, as shown in Figure 2.
While the proposed algorithm was applied to find the nearest parking lot, the results show parking number 6 has been assigned as the nearest parking to the entrance, which is the same solution by the traditional ACO. However, the proposed solution has been achieved it in a faster time, which is 14.5 ms, as shown in Figure 3, which can prove the impact of the new method.



[✓ ACO] A*+ACO

Performance time = 17 ms

Figure 2. Case 1 the nearest parking to the entrance by ACO

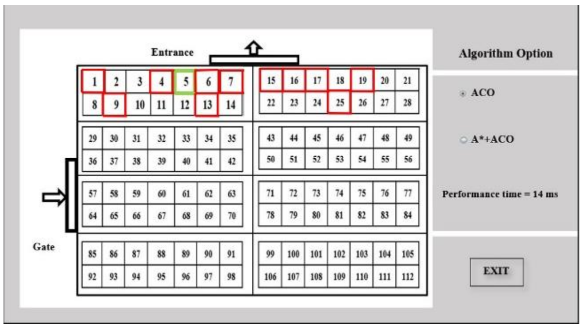


ACO [✓ A*+ACO]

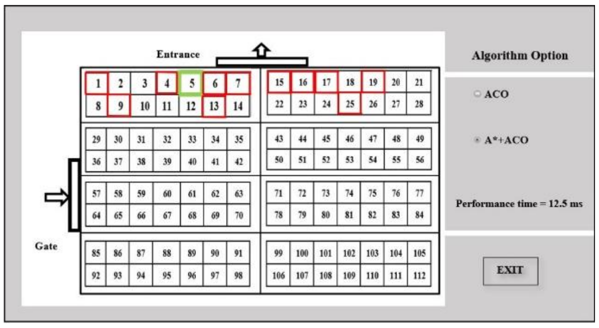
Performance time = 14.5 ms

Figure 3. Case 1 the nearest parking lot to the entrance by the proposed algorithms

CASE 2:
In this case, when the ACO has been selected, the parking number 5 has been chosen as the nearest empty parking to the entrance within 14 ms, as illustrated in Figure 4(a). While the proposed algorithm has been applied the same parking, a lot has been assigned but in a shorter time which is 12.5 ms as shown in Figure 4(b).



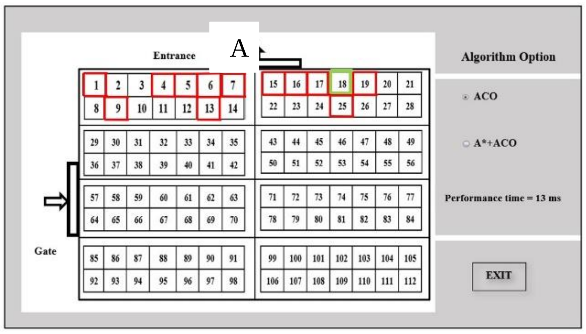
(a)



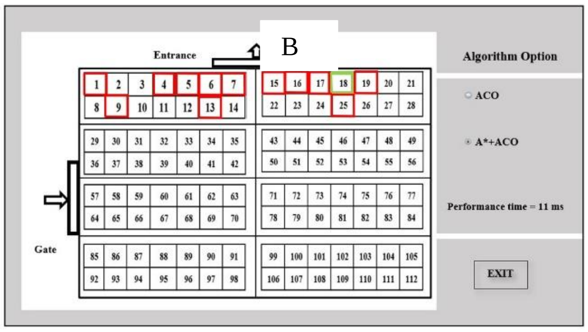
(b)

Figure 4. (a) Case 2 the nearest parking to the entrance by ACO and (b) Case2, the nearest parking lot to the entrance by the proposed algorithm.

In CASE 3, it shows that the traditional ACO has been assigned the parking lot number 18 as the nearest lot to the entrance within 13ms, as shown in Figure 5(a). However, the proposed algorithm assigned the same parking lot within a shorter time, which is 11Msec shown in Figure 5(b).



(a)



(b)

Figure 5. (a) Case 3 the nearest parking lot to the entrance by ACO, and (b) Case3 the nearest parking lot to the entrance by the proposed algorithm

Finally, Figure 6 illustrates the evaluation performance of the proposed algorithm when compare it with ACO alone, then the proposed algorithm A*ACO the best result based on the computation time. From the Figure, we can see that our proposed algorithm dramatically reduces the distance of the planned path from the location. The results demonstrate that the proposed method is very effective for general ACO. Finally, the proposed method shows better results compared with the previous work, which has been done by the ACO algorithm in terms of time complexity, as shown below.

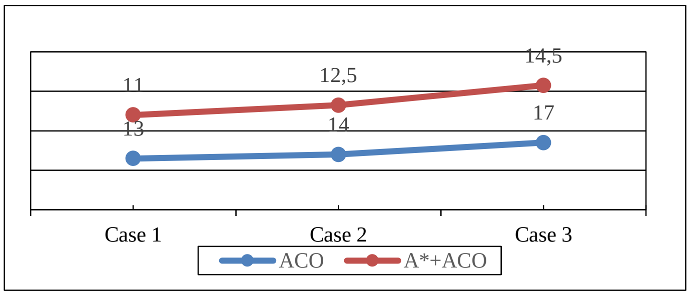


Figure 6. Time complexity

6. CONCLUSION

In this study, the (SSSP) has been presented. Several algorithms have been explained to solve this problem in different ways. This study introduced a new technique to find the best path and prove it by applying the new method on the parking system. The simulation has been examined in different cases. Three scenarios were implemented for each algorithm. Each scenario has been examined for 3 cases in the different parking lot to validate and to evaluate the improvement of the new method. The results of each scenario prove the power of the hybridization of the new method on solving the shortest path problem in contrast with ACO in term of the time computing. A* plays an important role, especially when proved the effective searching performance compared with ACO. Future work includes the experiments using huge data and improve the ACO algorithm such as hybrid it with local search algorithm or formulate a new ACO for multi-objective optimization.

REFERENCES

[1] N. Allali, Z. Chaouch, and M. Tamali, "Dashboard of intelligent transportation system (ITS) using mobile agents strategy on notification authentication process," *Int. J. Electr. Comput. Eng.*, vol. 9, no. 1, p. 621, 2019.

[2] R. Arya, R. Yadav, R. Agarwal, and O. V. Gnana Swathika, "Dijkstra's algorithm for shortest path identification in reconfigurable microgrid," *Journal of Engineering and Applied Sciences*, vol. 13, no. 3. pp. 717–720, 2018.

[3] Bar-Even, "Patent Application Publication (10) Pub . No . : US 2019 / 0086110 A1," vol. 1, 2019.

[4] L. Yu, H. Jiang, and L. Hua, "Anti-congestion route planning scheme based on Dijkstra algorithm for automatic valet parking system," *Appl. Sci.*, vol. 9, no. 23, 2019.

[5] H. Bast *et al.*, "Route planning in transportation networks," *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 9220 LNCS, pp. 19–80, 2016.

[6] K. I. M. Ata, A. C. Soh, A. J. Ishak, H. Jaafar, and N. A. Khairuddin, "Smart Indoor Parking System Based on Dijkstra ' s Algorithm," *Int. J. Integr. Eng.*, vol. 2, no. 1, pp. 13–20, 2019.

[7] M. Gendreau and J.-Y. Potvin, "Variable Nneighborhood search (chapter)," *Handb. Metaheuristics*, vol. 146, no. September, p. 648, 2010.

[8] S. A. Hussein and D. P. Dahnil, "A New Hybrid Technique to Improve the Path Selection in Reducing Energy Consumption in Mobile AD-HOC Networks," *Int. J. Appl. Eng. Res.*, vol. 12, no. 3, pp. 277–282, 2017.

[9] S. Alani, Z. Zakaria, and H. Lago, "A new energy consumption technique for mobile Ad-Hoc networks," *Int. J. Electr. Comput. Eng.*, vol. 9, no. 5, pp. 4147–4153, 2019.

[10] L. Parungao, F. Hein, and W. Lim, "Dijkstra algorithm based intelligent path planning with topological map and wireless communication," *ARPN J. Eng. Appl. Sci.*, vol. 13, no. 8, pp. 2753–2763, 2018.

[11] H. Jaafar, M. H. Zabidi, A. C. Soh, T. P. Hoong, S. Shafie, and S. A. Ahmad, "Intelligent guidance parking system using modified dijkstra's algorithm," *J. Eng. Sci. Technol.*, vol. 9, no. Spec. Issue on Applied Engineering and Sciences (SAES2013), October 2014, pp. 132–141, 2014.

[12] S. Beyhan and E. Boğar, "A Hybrid Genetic Algorithm for Mobile Robot Shortest Path Problem," *Int. J. Intell. Syst. Appl. Eng.*, vol. 4, no. Special Issue-1, pp. 264–267, 2016.

[13] X. Z. Wang, "The Comparison of Three Algorithms in Shortest Path Issue," *J. Phys. Conf. Ser.*, vol. 1087, no. 2, 2018.

[14] J. B. Singh and R. C. Tripathi, "Investigation of Bellman-Ford Algorithm, Dijkstra's Algorithm for suitability of SPP," *Int. J. Eng. Dev. Res. 1 Dean Res.*, vol. 6, no. 1, pp. 2321–9939, 2018.

[15] R. Mavrevski, M. Traykov, and I. Trenchev, "Finding the shortest path in a graph and its visualization using C# and WPF," *Int. J. Electr. Comput. Eng.*, vol. 10, no. 2, pp. 2054–2059, 2020.

[16] G. Wu and X. Sun, "Research on path planning of locally added path factor dijkstra algorithm for multiple AGV systems," *IOP Conf. Ser. Mater. Sci. Eng.*, vol. 711, no. 1, 2020.

[17] I. Hassani, I. Maalej, and C. Rekik, "Robot Path Planning with Avoiding Obstacles in Known Environment Using Free Segments and Turning Points Algorithm," *Math. Probl. Eng.*, vol. 2018, 2018.

[18] F. Duchon *et al.*, "Path planning with modified A star algorithm for a mobile robot," *Procedia Eng.*, vol. 96, no. January 2015, pp. 59–69, 2014.

[19] Y. Takasaki, "Flow Direction," *SpringerReference*, 2011.

[20] Y. Weng, J. Wu, and Y. Wang, "An improved hierarchical A∗ algorithm in the optimization of parking lots," *AIP Conf. Proc.*, vol. 1864, no. August, pp. 3–6, 2017.

[21] K. Loubna, B. Bachir, and Z. Izeddine, "Ant colony optimization for optimal low-pass state variable filter sizing," *Int. J. Electr. Comput. Eng.*, vol. 8, no. 1, pp. 227–235, 2018.

[22] U. Brehm and S. Buchholz, "Is there a wrong time for a right decision?," *J. Fam. Res.*, vol. 3, no. 82, pp. 126–134, 2014.

[23] H. Parvin, P. Moradi, and S. Esmaeili, "TCFACO : Trust-aware collaborative filtering method based on ant colony optimization," *Expert Syst. Appl.*, vol. 118, pp. 152–168, 2019.

[24] M. Bagherian, S. Massah, and S. Kermanshahi, "A swarm based method for solving transit network design problem," in *Australasian Transport Research Forum, ATRF 2013 - Proceedings*, 2013, no. October, pp. 1–10.

[25] Y. Wu, M. Gong, W. Ma, and S. Wang, "High-order graph matching based on ant colony optimization," *Neurocomputing*, vol. 328, pp. 97–104, 2019.

[26] T. Tsironi *et al.*, “Modeling the effect of active modified atmosphere packaging on the microbial stability and shelf life of gutted sea bass,” *Appl. Sci.*, vol. 9, no. 23, pp. 1–17, 2019.

BIOGRAPHIES OF AUTHORS

Sameer Alani was born in Iraq, in 1989. He received the B.S. degree in computer engineering and the M.Sc. degree in wireless communication and Computer networking technology from The National University of Malaysia (UKM), in 2017. He is currently pursuing the Ph.D. degree in wireless communication and networking. His research interests include antenna applications, wireless communication, artificial intelligent and networking Technology.

Atheer bassel received the B.Sc. degree in computer science from University of Anbar, Iraq, in 2004, the M.Sc. degree in computer science, in 2011 and the Ph.D degree in computer science from Universiti Kabangsaan Malaysia (UKM) in 2018. He is currently an assistant professor in the Department of Computer Science, Faculty of Computer Science and Information Technology, University of Anbar, Iraq. His main research area are metaheuristics, optimization algorithms, Image processing and their applications.

Mustafa Maad Hamdi was born in Al-Anbar, Iraq. He received the B.Eng. degree in Computer Engineering Technology from Al-Maarif University College, Iraq. and the M.Sc. degree in Communication and Computer Engineering from University Kebangsaan Malaysia (UKM), Malaysia. He is currently pursuing the Ph.D. degree in the department of communication engineering, University Tun Hussein Onn Malaysia (UTHM), Malaysia. His research interests include Wireless and Mobile Communications, VANET, MANET, and satellite Communication, and Cryptography.

Sami Abduljabbar Rashid was born in Al-Anbar, Iraq. He received the B.Eng. degree in computer engineering technology from Al-Maarif University College, Iraq. and the M.Sc. degree in communication and computer engineering from University Kebangsaan Malaysia (UKM), Malaysia. He is currently pursuing the Ph.D. degree in the department of communication engineering, University Tun Hussein Onn Malaysia (UTHM), Malaysia. His research interests include wireless mobile communications and VANET.