A study on Bee Algorithm and A* Algorithm for Pathfinding in Games

Aimi Najwa Sabri , Nor Haizan Mohamed Radzi, Azurah A Samah
Faculty of Computing
Universiti Teknologi Malaysia,
Skudai, Johor.
anajwa7@live.utm.my, haizan@utm.my, azurah@utm.my

Abstract—**Pathfinding is essential and necessary for agent movement used in computer games and many other applications. A primary concern of pathfinding is to find the shortest feasible path. The past decade has seen the rapid development of pathfinding in many games industry. Several attempts and solutions have been made to solve the problem. Heuristic A* algorithm has proven given an optimal for single agent and small size of search in current game industry. However, pathfinding solution is often requiring huge amount of resources especially in complex game environment. Hence, generation a feasible and optimal path more computationally burdensome or intractable. More recently, literature has come out that offers findings about metaheuristic in pathfinding. Metaheuristic is usually used in pathfinding problem. Bee algorithm is a metaheuristic algorithm which is naturally behaves as multi-agent approach and proven efficient in solving the path planning problem. The current research on Bee algorithm mostly focused on improving optimization for agent. The aim of this paper is to describe and compare traditional algorithm, A* algorithm and optimization search, Bee algorithm in pathfinding.**

Keywords—Pathfinding, A* algorithm, Bee algorithm, Optimization.

I. INTRODUCTION

Games have existed since the dawn of humanity and have become the integral part of human as they provide challenge and entertainment. Some games even help improve human's logical and creative thinking skill. The advent of technologies has caused revolutionaries in today's life, and that includes gaming. As far as a historian could recall, William Higginbotham and Steve Russell were the first two men who could be attributed as the key people of the video game industry. With advanced technologies, it is noticeable that there has been a positive mixture of academics and industrial in the gaming field. Games are a growing part of entertainment industry and offer a large variety of fundamental AI research problem.

Pathfinding is an indispensable part in many domains including robot path planning, global positioning system, Artificial Intelligence (AI) and games [1]. Almost every game requires pathfinding to make the game more human-like. M. Cowley [2] believes that the more human a robot looks and the extent to which it able to move similarly to humans determines how close it can get to "human intelligence". Pathfinding is the most basic requirement for game agent in games is to be able to navigate the path in game environment. Pathfinding is an important part of many applications, including commercial games and robot navigation [3]. It is important to use an optimized pathfinding algorithm since there are many real-time games are developed nowadays. However, generally, games will become more complex when in larger environment.

The next section discusses pathfinding. The section is divide into three parts which are pathfinding used in games, A* algorithm and map representation in pathfinding. Section 3 reviews metaheuristic and Bee algorithm. Section 4 presents experimental setup, results and discussions.

II. PATHFINDING

A. Pathfinding in Games

Pathfinding is referring to the plotting nodes, in finding the shortest path between two points. The last two decades have seen a growing trend towards applied Artificial Intelligence in games. Pathfinding is an indispensable part in many domains including robot path planning, global positioning system, Artificial Intelligence and games. Today, games are one of the popular form of entertainment. It can be found practically on digital platform such as pc and mobile phones. As games technology have grown, there has been an increasing interest in pathfinding in games. In games, pathfinding is necessary for the agent to navigate path from origin point to goal point. Games can be an excellent experiment to pathfinding research. Moreover, pathfinding is one of the most common applications of game research among Artificial Intelligence techniques [4].

Games can be a lot of fun and engage especially when the agents in game are smart enough to take the shortest path. The most common problem in pathfinding is agent movement [23]. This issue has been attracted researchers to put effort to come out with various solutions such as improving and enhancing the algorithm, improving the map environment and introduce new method to create more realistic representation of game worlds. Generally, a larger and complex environment is difficult because of huge field of view might take some time and consume too much memory to find the path. Eventhough small environments are easier to control and not consume too much memory, they can become a huge task if the number of desired visual is large.

978-1-5386-3527-8/18/$31.00 ©2018 IEEE

Authorized licensed use limited to: M S RAMAIAH INSTITUTE OF TECHNOLOGY. Downloaded on April 23,2024 at 10:01:02 UTC from IEEE Xplore. Restrictions apply.

In recent years, development of pathfinding in dynamic games environment has gained researchers attention, particularly in the dynamic game environment. Based on study by Graham [23], pathfinding leads to drain on CPU resource and taking too much time in searching path when the size of map environment is larger. This has been supported by Bulitko [5], claimed that the challenge is increasing due to complexity of game environment. Dynamic games usually contain open landscape with huge view and large area with more obstacles. Fayyazi [24] also stated that pathfinding is expensive when involve in large size of map environment that uses up CPU and memory. Figure 1 illustrates pathfinding in grid-based environment with obstacles. There are two option paths to find the target from starting point. The shortest path can be determined by calculating the number of steps.

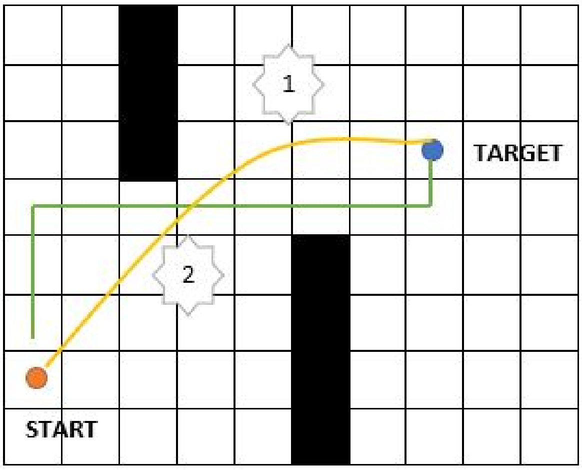

Fig. 1. Pathfinding in grid-based environment

B. Heuristic

Searching path is the significant process in pathfinding, to search and produce the solution path that has connection between initial position start and goal position. Map representation is result of transforming pathfinding problem into graph searching. Various search algorithms were created to solve pathfinding problem such as Dijkstra's algorithm, Depth First search algorithm until the emergence of A* algorithm [26]. A* algorithm can find feasible path between two points in a short time. Due to the success of A* algorithm in pathfinding problem, many researchers have focused on variants of A* algorithm. Dozens of revised pathfinding algorithms have been introduced successfully.

A algorithm*

A* algorithm was proposed by Hart, Nillsson and Raphael in 1967 [28] and widely used in problem solving, such as engineering and pathfinding. A* algorithm is an extensive of Dijkstra's algorithm, by adding a heuristic value that estimate the distance from start node to goal node. A* works by maintaining open and close list. Open list initializes with start node and contains all the new nodes that can be visit in the next step, while close list starts with empty node and contains all the nodes that were already visited. A* algorithm gives faster result than Dijkstra [32]. It is because Dijkstra's is taking more time to reach the goal by repeatedly attempts to visit each node. A* algorithm searches minimum optimal solution in searching tree to provide the shortest path and the least cost path. Unfortunately, there is limitation when searching in larger map size. A* is expensive in terms of execution times when the number of nodes in the map increases [31]. In larger map size, A* algorithm required memory allocation which rapidly change to the environment, thus it may cause excessive memory before produce the solution [6]. Due to its limitation, variants of A* algorithm have been introduced to solve the pathfinding problem.

C. Metaheuristic

Based on the previous studies in pathfinding, A* has become hot research topic in many fields. It is because A* offered more convenient capabilities compared to others. However, there are still have insufficient in A* algorithm that make many improvements are made such as excessive memory and time. Therefore, metaheuristic is introduced in solving pathfinding problem.

There are two types of stochastic algorithm, which are heuristic and metaheuristic. Heuristic algorithms are used to find optimal solution; however, it is not working all the time. Further development of heuristic algorithm is metaheuristic algorithm. Meta means beyond or higher level. Generally, they perform better than heuristic. Since there are many limitations in heuristic pathfinding, a huge number of metaheuristics have been proposed to overcome the pathfinding problem. Yagiura and Ibaraki [33] mentioned metaheuristic is the combination of heuristic tools in framework to generate new solution. The combination performed a stochastically based on some natural phenomenon.

Several metaheuristic algorithms that have been developed to solve optimization problem such as evolutionary algorithm was originally proposed by Fogel [34], followed by Jong [35] and Koza [36], genetic algorithm was proposed by Holland [37] and Goldberg [38]. Evolutionary algorithm is among the most popular algorithm. This algorithm uses biology-inspired such as mutation, crossover, natural selection and survival of the fittest. The algorithm performed consistently well in many kinds of problems.

Over the last decade, almost all new research studies in metaheuristic algorithms are referred to nature-inspired metaheuristic algorithm. These nature-inspired metaheuristic algorithms usually based on swarm intelligence, biological systems, physical and chemical systems that can solve optimization problems [39]. Swarm intelligence based is one of the optimization methods. The goal of SI is designing multi agent by taking inspiration from insect behaviour such as ants, bats, bees and others. Good examples swarm intelligence is particle swarm optimization (PSO), ant colony optimization (ACO), bee algorithm (BA), bat algorithm, firefly algorithm and cuckoo search.

Authorized licensed use limited to: M S RAMAIAH INSTITUTE OF TECHNOLOGY. Downloaded on April 23,2024 at 10:01:02 UTC from IEEE Xplore. Restrictions apply.

Bee Algorithm

Honey bee algorithm was proposed by S. Nakrani and C. Tovey in 2004 [40], which followed by the development of bee algorithm by D.T. Pham et al in 2005 [41]. Bee algorithm performs a neighborhood search combined with random search. Generally, bee algorithm is used to solve in complex optimization problem.

In bee's nature, scout bees are assigned to search randomly for food from one flower patch to another and evaluate every patch. Then, bees communicate with each other through waggle dance in hive which contain information of direction and quality of the flower patch. Regarding the quality of flower patch, patches can either be visited by more bees or maybe abandoned. Each solution represents a food source and the population is representation of bees that is used to search for food. The objective of this algorithm is to find and explore good sites in search space. The algorithm required many parameters such as number of scout bee, number of sites selected, number of best sites, number of bees recruited for best sites, number of bees recruited for other selected site and initial size of patches. Pham [41] proved that BA was success in term of speed and accuracy compared to other methods. Besides, it could reach optimum 10 times faster than genetic algorithm. However, the number of tunable parameter is one of the disadvantages of this algorithm.

In 2007, Pham [17] also proposed bee algorithm is used in data clustering to overcome the tendency of clustering problem to converge to local optima. Due to the ability of BA in performing local and global search simultaneously, the proposed method for different data sets results better performances than other of the k-means algorithm. Nonetheless, the tunable parameters are still become the disadvantage of BA and need to be solve in future work. Shatnawi [42] enhanced basic BA by adding local memory at scout bees and follower in complex multi model optimization.

D. Map representation in Pathfinding

Maps in games help AI works better and make better decisions by providing information about the environment. There is a correlation between map representation and pathfinding, which can make a huge difference in performance and path quality in games. There are many efforts have been done by researchers in solving pathfinding problem, improve and optimize technique used in game environment is one of the them. The most common techniques used in pathfinding are waypoint graph, navigation mesh and grids. Table 1 represents the advantages and disadvantages of these three techniques.

TABLE I. COMPARISON OF MAP REPRESENTATIONS IN PATHFINDING

Map representation	Advantages	Shortcomings
Waypoint	• Suitable used in small search space and static map environment • Traditional approach used to navigate path	• Not suitable used in dynamic map environment (obstacles exist) • Expensive time consuming when in large map environment
Navigation Mesh	• Better than waypoint graph technique • Guarantee to find optimal path • Reducing the number of nodes used in map environment • Suitable in static and dynamic map environment	• Have limitation when in large map environment and dynamic environment. • Computational cost increase when creating and updating navigation mesh graph. • It may cost time and memory consuming when the location of obstacle changes.
Grid	• Simple and easy to understand in game environment • Cheap in updating graph • Very suitable in static and dynamic environment • Do not required complicated node for rebuilding process	• Required a grid representation of each map • Inaccuracy in minimum distance.

Recently, grid-based graph is widely used in many fields especially pathfinding. In games, grid-based graph is simple and easy to understand. There is type of grid used in games map environment, which are tile, octile and hexagonal. The horizontal and vertical movement of one cell has cost 1, while diagonal movement has cost 1.4. Hexagonal grid is mathematically proven better than tile grid for IDA* search [29]. Besides that, searching path using hexagonal grid will results faster. Anderson [30] presented a new additive heuristic on four connected square grid. The combination heuristic and grid are effectively providing optimal path. Grid-based graph is primarily used in dynamic environment due to its simplicity. However, A* is not always optimal in dynamic grid maps. Grid-based graph works well in games because the agent can move freely along both x- axis and y-axis.

Authorized licensed use limited to: M S RAMAIAH INSTITUTE OF TECHNOLOGY. Downloaded on April 23,2024 at 10:01:02 UTC from IEEE Xplore. Restrictions apply.

III. THE COMPARISON OF BEE COLONY AND A* ALGORITHM

A. Experimental Setup

In this paper, we conducted experiments with and without the presence of obstacles for both algorithm in different size of map. The application is a two-dimensional representation of all pathfinding model performed by A* algorithm and Bee algorithm. The map range in size map from 200x200 and 500x500. The initial node, goal node and obstacles are in fixed position. The timings were performed on a 2.2 GHz Intel Core i7-CPU 3632QM with 4GB of memory. This experiment using MATLAB R2013a (see Figure 2), represented as a grid-based environment. The comparison results are discussed in next section.

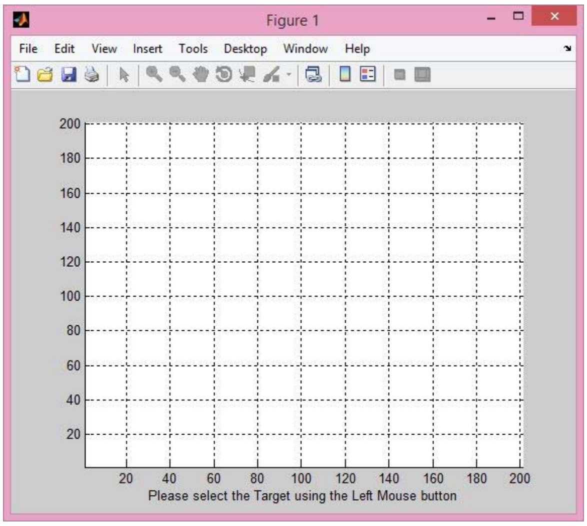

Fig. 2. The grid-based environment in 2D games

B. Results

The experiment compares the time and memory of both algorithms to reach the goal node and time with different size map on grid environment and time reach the goal point in obstacle environment are calculated using MATLAB R2013a. We conducted the experiment for 10 iterations (see Table 2 and 3) to reduce the amount of random errors. In this experiment, starting with a fixed start point and goal point in map, randomly placed the number of obstacles. Computational time and memory usage are evaluated in this study. Measuring unit for computing time is second (s) and measuring unit for memory is kilobyte (Kb).

TABLE II. SIZE MAP FREE OBSTACLE

Bil	Time Taken (s)			
	200x200		500x 500	
	A* algorithm	Bee algorithm	A* algorithm	Bee algorithm
1.	0.000307	4.84153	1.18648E-12	18.61115
2.	0.00037	3.944554	7.89798E-13	10.47825
3.	0.000397	2.220254	6.27772E-13	7.63938
4.	0.000421	1.439501	4.95838E-13	4.74402
5.	0.000443	0.98205	4.25127E-13	3.24286
6.	0.000466	0.705321	2.40454E-13	2.367489
7.	0.000487	0.568587	2.11852E-13	1.590747
8.	0.000509	0.379121	1.45503E-13	1.007779
9.	0.000873	0.2504772	8.226E-14	0.773887
10.	0.000907	0.1630031	8.05764E-14	0.610227

TABLE III. SIZE MAP WITH OBSTACLE

Bil	Time Taken (s)			
	200x200		500x 500	
	A* algorithm	Bee algorithm	A* algorithm	Bee algorithm
1.	18.61115	4.0118	25.51933	22.44363
2.	10.47825	3.555361	10.43387	15.15678
3.	7.63938	2.936951	6.60746	9.98978
4.	4.74402	1.776337	5.234421	7.05697
5.	3.24286	1.550681	4.650751	4.413542
6.	2.367489	0.825025	4.000641	3.054762
7.	1.590747	0.5859131	2.79043	2.143172
8.	1.007779	0.3779152	2.164828	1.457337
9.	0.773887	0.2646752	1.642566	0.813103
10.	0.610227	0.1510567	1.543349	0.61429

From Table 2, in both size map 200x200 and 500x500 without obstacle, A* algorithm shows promising solution compared to Bee algorithm. In last iteration, A* algorithm get better result in both size map 200x200 and 500x500 which are 0.000907s and 8.05764E-14s. It is because A* algorithm can travel searching in direct path without changing path. However, in Table 3, Bee significantly faster than A* especially in size map 500x500. Thus, the result shows that Bee algorithm is faster than A* algorithm when in larger size map environment and with the existence of obstacles.

Memory and time in this study are calculated by using Matlab R2013a. Memory allocated, and time execution are displayed in the profiler. Table 4 and 5 show result of memory usage. Both experiments with different size map show Bee algorithm is better in optimizing memory than A* algorithm. Nevertheless, high memory is used when the obstacles are exist. The result shows that A* is taking too much memory. It

Authorized licensed use limited to: M S RAMAIAH INSTITUTE OF TECHNOLOGY. Downloaded on April 23,2024 at 10:01:02 UTC from IEEE Xplore. Restrictions apply.

is because redundancy occurs while searching the path in A* algorithm. Thus, we can conclude that Bee algorithm is efficient in optimizing time and memory in pathfinding.

TABLE IV. SIZE MAP FREE OBTACLE

Bil	Memory (Kb)			
	200x200		*500x 500*	
	A algorithm*	*Bee algorithm*	*A* algorithm*	*Bee algorithm*
1.	16652	7408	7592	20748
2.	68980	10780	12388	5828
3.	5360	6960	6176	5796
4.	5040	6624	15424	7228
5.	34552	7292	5644	6744
6.	12314	5632	7524	5800
7.	32636	6028	5884	5576
8.	28836	6304	5520	4532
9.	17708	7820	8828	5240
10.	7004	7940	6412	4912

TABLE V. SIZE MAP WITH OBSTACLE

Bil	Memory (Kb)			
	200x200		*500x 500*	
	A algorithm*	*Bee algorithm*	*A* algorithm*	*Bee algorithm*
1.	66884	183124	15684	13660
2.	13664	15444	14616	13384
3.	14536	14416	13148	12752
4.	15404	14932	13896	12832
5.	11056	14160	14100	7228
6.	14524	12220	16884	13244
7.	13520	14860	35500	13320
8.	63372	15472	160532	12868
9.	83316	15304	14396	13136
10.	202160	15312	15112	12760

C. Discussion

Despite the importance and amount of work done in pathfinding, there are not many publications about Bee algorithm for pathfinding in game environment. The most important highlight that we can conclude in this paper is A* algorithm searches 10 times faster in free obstacle game environment compared to Bee algorithm. However, when in complex game environment, which is obstacles exist, Bee algorithm outperforms A* algorithm. Bee algorithm has advantages since it has local and global search ability that can search quickly compared to A* algorithm.

ACKNOWLEDGMENT

This research was supported by Tier 2 grant (Q.J130000.2628.12J26) at Universiti Teknologi Malaysia. The authors are part of Department of Computer Science, Faculty of Computing, Universiti Teknologi Malaysia.

REFERENCES

[1] Zhu, Weiping, et al. "Waypoint graph based fast pathfinding in dynamic environment." *International Journal of Distributed Sensor Networks* 11.8 (2015): 238727.

[2] Cowley, Ben, et al. "Toward an understanding of flow in video games." *Computers in Entertainment (CIE)* 6.2 (2008): 20.

[3] Cazenave, Tristan. "Optimizations of data structures, heuristics and algorithms for path-finding on maps." *Computational Intelligence and Games, 2006 IEEE Symposium on*. IEEE, 2006.

[4] Botea, Adi, et al. "Pathfinding in games." *Dagstuhl Follow-Ups*. Vol. 6. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2013.

[5] Bulitko, Vadim, Yngvi Björnsson, and Ramon Lawrence. "Case-based subgoaling in real-time heuristic search for video game pathfinding." *Journal of Artificial Intelligence Research*39 (2010): 269-300.

[6] Algfoor, Zeyad Abd, Mohd Shahrizal Sunar, and Hoshang Kolivand. "A comprehensive study on pathfinding techniques for robotics and video games." *International Journal of Computer Games Technology* 2015 (2015):7.

[7] Botea, Adi. "Ultra-Fast Optimal Pathfinding without Runtime Search." *AIIDE*. 2011.

[8] Gendreau, Michel, and Jean-Yves Potvin. "Metaheuristics in combinatorial optimization." *Annals of Operations Research*140.1 (2005): 189-213.

[9] Korf, Richard E. *Artificial intelligence search algorithms*. Chapman & Hall/CRC, 2010.

[10] Schadd, Maarten PD, et al. "Single-player monte-carlo tree search." *International Conference on Computers and Games*. Springer, Berlin, Heidelberg, 2008.

[11] Björnsson, Yngvi, and Kári Halldórsson. "Improved Heuristics for Optimal Path-finding on Game Maps." *AIIDE* 6 (2006): 9-14.

[12] Sturtevant, Nathan R. "Memory-Efficient Abstractions for Pathfinding." *AIIDE* 684 (2007): 31-36.

[13] Fu, Liping, D. Sun, and Laurence R. Rilett. "Heuristic shortest path algorithms for transportation applications: state of the art." *Computers & Operations Research* 33.11 (2006): 3324-3343.

[14] Cui, Xiao, and Hao Shi. "A*-based pathfinding in modern computer games." *International Journal of Computer Science and Network Security* 11.1 (2011): 125-130.

[15] Xi, Chen, Fei Qi, and Li Wei. "A new shortest path algorithm based on heuristic strategy." *Intelligent Control and Automation, 2006. WCICA 2006. The Sixth World Congress on*. Vol. 1. IEEE, 2006.

[16] Sturtevant, Nathan R. "Benchmarks for grid-based pathfinding." *IEEE Transactions on Computational Intelligence and AI in Games* 4.2 (2012): 144-148.

[17] Pham, D. T., et al. "Data clustering using the bees algorithm." *40th CIRP International Manufacturing Systems Seminar*. 2007.

[18] Akay, Bahriye, and Dervis Karaboga. "A modified artificial bee colony algorithm for real-parameter optimization." *Information Sciences* 192 (2012): 120-142.

Authorized licensed use limited to: M S RAMAIAH INSTITUTE OF TECHNOLOGY. Downloaded on April 23,2024 at 10:01:02 UTC from IEEE Xplore. Restrictions apply.

[19] Pham, D. T., and Afshin Ghanbarzadeh. "Multi-objective optimisation using the bees algorithm." *Proceedings of IPROMS 2007 Conference*. 2007.

[20] Zarembo, Imants, and Sergejs Kodors. "pathfinding algorithm efficiency analysis in 2d grid." *Proceedings of the 9th International Scientific and Practical Conference*. Vol. 1. 2013.

[21] Graham, Ross, Hugh McCabe, and Stephen Sheridan. "Pathfinding in computer games." *ITB Journal* 8 (2003): 57-81.

[22] Botea, Adi, Bruno Bouzy, Michael Buro, Christian Bauckhage, and Dana S. Nau. "Pathfinding in Games." In *Artificial and Computational Intelligence in Games*, pp. 21-31. 2013.

[23] Graham, Ross. "Realistic agent movement in dynamic game environments." (2005).

[24] Fayyazi, Mehri, Faranak Fotouhi Ghazvini, and Ali Ramzi. "Efficient Grid-Based Path Finding Techniques For Android Games Environments."

[25] Anguelov, Bobby. "Video game pathfinding and improvements to discrete search on grid-based maps." PhD diss., University of Pretoria Pretoria, 2011

[26] Ansari, Ahlam, et al. "An optimized hybrid approach for path finding." *arXiv preprint arXiv:1504.02281* (2015).

[27] Reddy, Harika. "PATH FINDING-Dijkstra's and A* Algorithm's." *International Journal in IT and Engineering* (2013): 1-15.

[28] Mathew, Geethu Elizebeth. "Direction based heuristic for pathfinding in video games." *Procedia Computer Science* 47 (2015): 262-271.

[29] Yap, Peter. "Grid-based path-finding." *Conference of the Canadian Society for Computational Studies of Intelligence*. Springer, Berlin, Heidelberg, 2002.

[30] K. Anderson, "Additive heuristic for four-connected gridworlds," in Proceedings of the 3rd Annual Symposium on Combinatorial Search, 2010.

[31] Mocholi, Jose A., et al. "An emotionally biased ant colony algorithm for pathfinding in games." *Expert Systems with Applications* 37.7 (2010): 4921-4927.

[32] Saleh Alija, Amani. *Analysis of Dijkstra's and A* algorithm to find the shortest path*. Diss. Universiti Tun Hussein Onn Malaysia, 2015.

[33] Yagiura, Mutsunori, and Toshihide Ibaraki. "On metaheuristic algorithms for combinatorial optimization problems." *The Transactions of the Institute of Electronics, Information and Communication Engineers* 83.1 (2000): 3-25.

[34] Fogel LJ, Owens AJ, Walsh MJ. Artificial Intelligence through Simulated Evolution. Wiley, New York, NY, 1966.

[35] Kenneth A. De Jong. An analysis of the behavior of a class of genetic adaptive systems. Ph.D. Dissertation, University of Michigan, U.S.A, 1975.

[36] Koza, John R. "Genetic programming as a means for programming computers by natural selection." *Statistics and computing* 4.2 (1994): 87-112.

[37] Holland. Adaptation in Natural and Artificial Systems. University of Michigan Press, Ann Arbor, 1975.

[38] David Goldberg. Genetic Algorithms in Search, Optimization, and Machine Learning. Addison Wesley Publishing Company Inc, New York, 1989.

[39] Fister Jr, Iztok, et al. "A brief review of nature-inspired algorithms for optimization." *arXiv preprint arXiv:1307.4186*(2013).

[40] Nakrani, Sunil, and Craig Tovey. "On honey bees and dynamic server allocation in internet hosting centers." *Adaptive Behavior* 12.3-4 (2004): 223-240.

[41] Pham, D. T., et al. "The bees algorithm. Technical note." *Manufacturing Engineering Centre, Cardiff University, UK*(2005): 1-57.

[42] Shatnawi, Nahlah, Shahnorbanun Sahran, and Mohammad Faidzul. "A memory-based Bees Algorithm: an enhancement." *J Appl Sci* 13 (2013): 497-502.

Authorized licensed use limited to: M S RAMAIAH INSTITUTE OF TECHNOLOGY. Downloaded on April 23,2024 at 10:01:02 UTC from IEEE Xplore. Restrictions apply.