Decision Analytics Journal 6 (2023) 100175

Contents lists available at ScienceDirect

Decision Analytics Journal

journal homepage: www.elsevier.com/locate/dajour

A new approach for solving the minimum vertex cover problem using artificial bee colony algorithm

Anan Banharnsakun

Computational Intelligence Research Laboratory (CIRLab), Computer Engineering Department, Faculty of Engineering at Sriracha, Kasetsart University Sriracha Campus, Chonburi 20230, Thailand

A R T I C L E I N F O

Keywords:
Minimum vertex cover problem
Artificial bee colony
Optimization
Swarm intelligence
Wireless sensor networks

A B S T R A C T

The minimum vertex cover problem is a well-known problem in classical graph theory. It is known as one of the NP-Hard optimization problems, meaning that the best solution cannot be found within an acceptable time. Therefore, to handle this optimization problem effectively, various alternative approaches based on either approximation or metaheuristic techniques have been considered and proposed in the previous literature. However, developing more effective methods to deal with the problem of finding the minimum vertex cover is still challenging. This study proposes a strategy for tackling the minimum vertex cover problem based on the artificial bee colony (ABC) algorithm. The main novelty and contribution of this research are to show that the ABC algorithm can be used as another useful and efficient way to solve the minimum vertex cover problem and also to demonstrate the application of the minimum vertex cover solutions to real-world wireless sensor network problems. The proposed method is evaluated by assessing the algorithm's performance based on an obtained set of optimal vertices and the amount of computation time in terms of the number of iterations. The empirical results show that the proposed approach can yield satisfactory results and outperform other existing algorithms in finding a set of optimal vertices.

1. Introduction

The minimum vertex cover problem is a basic combinatorial optimization problem, in which the goal is to find a subset of the vertices that cover all edges where the number of vertices in this subset is the smallest. The problem of searching for the minimum vertex cover plays an important role in many real-world applications, such as network dismantling [1], wireless sensor networks [2,3], information retrieval [4], public transportation networks [5], and bioinformatics [6]. However, Karp [7] showed that the finding of a minimal number of vertices is one of the NP-Hard problems. Therefore, to solve this optimization problem in polynomial time, alternative approaches based on either approximation or metaheuristic techniques are thus required.

Many methods have been introduced over the past several decades to solve the vertex cover problem. To approximate the minimal number of vertices, Chen et al. [8] employed a Dijkstra algorithm to construct a vertex cover in polynomial time. To improve the quality of local optima, an idea for finding and constructing a vertex cover by using the partial vertex cover based on the local search strategy was introduced by Cai et al. [9]. A simulated annealing algorithm was presented by Xu and Ma [10] to help with increasing the probability of finding a near-optimal solution during the construction of the vertex cover. Mousavian et al. [11] proposed an algorithm for constructing a vertex cover by using a combination of cellular automata and learning automata. This synergy makes their proposed method able to yield a near-optimal solution. An algorithm known as an isolation algorithm was introduced by Ugurlu [12]. This algorithm starts with the vertex that has a minimal number of neighbors of vertex and then adds all vertices which are adjacent to this vertex to the vertex cover in order to reach the optimal solution. A hybrid genetic algorithm combined with a new repair technique was introduced by Çı naroğlu and Bodur [13] to improve the solution in the minimum vertex cover problem [14–22].

Recently, there have been a remarkable number of new algorithms that have been developed to solve the minimum vertex problem. A hybrid algorithm based on modified chemical reaction optimization and best-first search algorithm was presented by Khattab et al. [14] to generate initial populations with a better quality of initial solutions than is usually generated using random approaches. To achieve higher solution efficiency and faster computation, Qiu et al. [15] introduced a population-based game-theoretic optimizer to gain the capability of yielding an improved vertex cover solution using a Nash equilibrium. The quantum approximate optimization algorithm was proposed by Zhang et al. [16] to obtain the problem solution with high probability in polynomial time. A new membrane evolutionary algorithm framework, which was inspired by the structure and functioning of living cells, was introduced by Guo et al. [17] for solving the minimum vertex cover problem. An effective deep learning approach for

E-mail address: ananb@ieee.org.

https://doi.org/10.1016/j.dajour.2023.100175
Received 24 October 2022; Received in revised form 28 December 2022; Accepted 24 January 2023
Available online 27 January 2023

minimum vertex cover was proposed by Abu-Khzam et al. [18] to improve vertex cover approximation. A localized distributed algorithm that finds the minimum vertex cover using the 2-hop local subgraph of nodes was presented by Akram and Ugurlu [19] in order to reduce time complexity. The mathematical structure of a discernibility matrix based on the connection between graphs and information systems was employed by Zhuang and Chen [20] to deal with the minimum vertex cover problem in terms of approaches to a discernibility matrix in rough sets. Based on the fact that the minimum vertex problem for hypergraphs and minimal attribute reducts for rough sets decision tables are mutually related and both employ a local search paradigm, a new heuristic algorithm using rough sets-based methods for solving the minimum vertex cover problem for hypergraphs was provided and tested by Zhou et al. [21]. The branch-and-bound search scheme was presented by Wang et al. [22] to help prune the search space in the minimum vertex cover problem.

Although a number of different techniques have been proposed for solving the minimum vertex cover problem, the solving of this important class of problems by employing biologically-inspired strategies has rarely been performed. Thus, the solving of the minimum vertex cover problem by these strategies still remains a challenge. Over the past two decades, previous research has shown the great potential of biologically-inspired strategies to effectively solve a wide range of problems in science and engineering [23–28]. Even if the use of these biologically inspired algorithms can provide acceptable results, the solution quality obtained from these approaches still requires improvement and the ways that the convergence speed of these methods can be reduced also needs to be considered.

The artificial bee colony (ABC) method proposed by Karaboga [29–31] is one of the most widely recognized and effective methods for solving a wide range of real-world problems [32–34]. The ABC algorithm is a mathematical model for solving optimization problems inspired by observing and mimicking natural bee foraging behavior patterns. Several studies in the previous literature [35–38] have shown that the ABC algorithm can lead to success in finding an optimal solution and is a more effective method compared with other algorithms in many optimization problems.

In this research, the ABC algorithm, a simple and efficient technique, is employed to solve the minimum vertex cover problem in an optimal way. The main novelty and contribution of this research are to show that the ABC algorithm can be used as another useful and efficient way to solve the minimum vertex cover problem and also to demonstrate the application of this minimum vertex cover solution to real-world wireless sensor network problems.

The rest topic of the paper is composed as follows. Section 2 briefly describes the background and knowledge, including minimum vertex cover problem and the artificial bee colony algorithm. Section 3 proposes the solution to minimum vertex cover problem by using ABC. Section 4 describes the experimental settings. Section 5 discusses the results. Finally, Section 6 presents the conclusion of this paper.

2. Background and knowledge

2.1. Minimum vertex cover problem

Let $G = (V, E)$, where G is an undirected graph, $V = \{1,2,3, \ldots ,n\}$ is the set of vertices, $E \subseteq V\times V$ is the set of edges, and $(u,(v) \in E$ is an edge between vertices u and v. The minimum vertex cover problem consists of searching for and constructing the smallest subset $V' \subseteq V$ such that $\forall(u,\ (v) \in E : u \in V'$ or $v \in V'$. In other words, the minimum vertex cover problem involves determining the minimum number of vertices in a set so that every edge in the graph has at least one vertex in that set [39]. The example of constructing the vertex cover in a graph is shown in Fig. 1. Note that each of the gray vertices in the graph makes up the graph's vertex cover.

Fig. 1(a) is an invalid vertex cover because the edge v_2-v_5 is not covered by any vertex in the set (both endpoints are not in the set), while Fig. 1(b) and Fig. 1(c) are examples of valid vertex covers, which can be considered as possible solutions. The size of the minimum vertex cover is 3, as shown in Fig. 1(c), in which there cannot be less than 3 vertices in this graph.

The link monitoring in wireless sensor networks (WSN) [40–42] is one of the real-world applications, in which the vertex cover can be applied and used to describe this application. Since monitor nodes in WSN can be costly in terms of the numerous parameters, such as deployment time and extra hardware costs, minimizing the number of these monitor nodes in WSN is thus an important task [43–45].

2.2. Artificial bee colony algorithm

The Artificial bee colony (ABC) algorithm is a mathematical model for solving optimization problems inspired by observing and mimicking natural bee foraging behavior patterns. The honey bee agents of the ABC in the colony are classified into three types, including the employed bees, the onlookers, and the scouts. The bees in the ABC algorithm are divided into two groups that have the same number of bees. Half are referred to as employed bees and the other half as onlooker bees. The location of the bee's food source is treated as a solution containing parameters that require optimization in the ABC. The quality of the food source is related to the objective function of a problem, which can be represented as the fitness value of the solution. In other words, the foraging process by the bees to obtain a good food source is comparable to the process of finding the optimal solution.

The details of the ABC algorithm are as follows. The initial solutions are randomly generated and treated as the food source positions for the bee agents. After initialization, the bee agents are subjected to repeated cycles of three major steps: updating the feasible solutions, selecting feasible solutions, and avoiding suboptimal solutions.

In order to update the feasible solutions, all employed bees select a new candidate food source position. Their choice is based on the neighborhood of the previously selected food source. The position of the new food source is calculated using Eq. (1)

$$v_{ij}=x_{ij}+\phi_{ij}(x_{ij}-x_{kj}), \tag{1}$$

where v_{ij} is a new feasible solution that is modified from its previous solution value (x_{ij}) based on a comparison with a randomly selected position from its neighboring solution (x_{kj}), ϕ_{ij} is a random number between [−1,1] that is used to randomly adjust the old solution to become a new solution in the next iteration, and $k \in \{1,2,3,..,SN\}$ and $k \neq i$, $j \in \{1,2,3,..,D\}$ are randomly chosen indexes. The difference between x_{ij} and x_{kj} is a difference of position in a particular dimension.

The old food source position in an employed bee's memory will be replaced by a new candidate food source position if the new position has a better fitness value. Employed bees will return to their hive and share the fitness value of their new food sources with the onlooker bees. In the next step, each onlooker bee selects one of the proposed food sources depending on the fitness value obtained from the employed bees. The probability that a proposed food source will be selected can be obtained from Eq. (2):

$$P_i = \frac{fit_i}{\sum_{i=1}^{SN} fit_i} \tag{2}$$

where fit_i is the fitness value of the food source i, and SN is the number of feasible food sources.

The probability of a proposed food source being selected by the onlooker bees increases as the fitness value of the food source increases. After the food source is selected, the onlooker bees will go to the selected food source and select a new candidate food source position in the neighborhood of the previously selected food source. The new candidate food source can be calculated and expressed by Eq. (1).

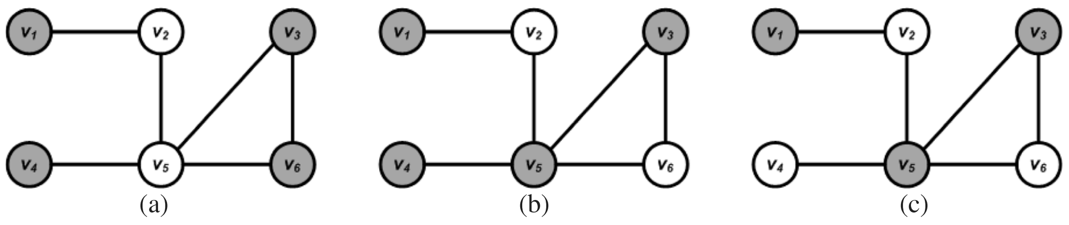
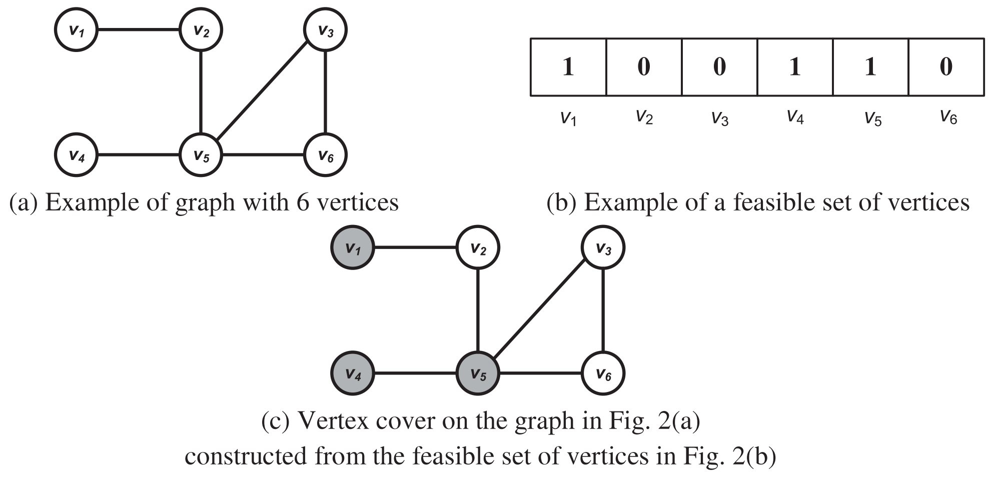


Fig. 1. Examples of minimum vertex covers.



(a) Example of graph with 6 vertices (b) Example of a feasible set of vertices

(c) Vertex cover on the graph in Fig. 2(a) constructed from the feasible set of vertices in Fig. 2(b)

Fig. 2. Example of a feasible solution representation.

In the third step, any food source position that does not have an improved fitness value will be abandoned and replaced by a new position that is randomly determined by a scout bee. This helps avoid suboptimal solutions. The new random position chosen by the scout bee is calculated by Eq. (3):

$$x_{ij} = x_j^{\min} + rand[0,1](x_j^{\max} - x_j^{\min}), \tag{3}$$

where $x_j^{\min}$ and $x_j^{\max}$ are the lower bound and the upper bound of the food source position in dimension *j,* respectively.

The number of function evaluations is used as a termination criterion. The process of the three major steps described above will be repeated until the number of function evaluations equals the determined value.

3. Solving the minimum vertex cover problem using ABC

The mechanism to update the solution in the ordinary ABC algorithm is designed for solving the optimization problems in a continuous domain. However, the minimum vertex cover problem is considered as an optimization problem in a discrete domain. Thus, in this work, the ordinary ABC algorithm is modified. Each set of vertices is considered as a candidate food source or feasible solution in the ABC algorithm. Each feasible set of vertices is represented in a binary string form. For example, the food source which represents a feasible set of vertices can be represented as S = $v1v2v3v4\ldots\ v_d$; in which v is the vertex and d is the total number of vertices. The value of each vertex v can be either 0 or 1. The vertex value 1 indicates a selected vertex and the vertex value 0 indicates a non-selected vertex. An example of feasible solution representation for a graph with six vertices is illustrated in Fig. 2.

The set of vertices is optimized and discovered by using the ABC algorithm. In other words, the ABC algorithm is employed to minimize the number of vertices in the set so that every edge in the graph has at least one vertex in that set. The proposed algorithm is shown in Fig. 3.

As seen in Fig. 3, the initial food source positions represented as the solutions (feasible set of vertices shown as an example in Fig. 2(b)) are randomly generated and assigned to the employed bees in the ABC algorithm. Thereafter, the bee agents in the ABC algorithm search for an optimal solution through the three processes of seeking the feasible solutions, choosing the optimal solutions, and avoiding the suboptimal solution processes.

In the process of seeking the feasible solutions, all employed bees search for new feasible solutions based on updating their old solutions by using the information of their neighbors' solutions. Since the solutions are represented in binary number form, the bit wise AND operation ($\wedge$) is thus employed to perform the solution updates in this process, as shown in Eq. (4):

$$v_{ij} = \begin{cases} x_{ij} \wedge x_{kj} & \text{if } (x_{ij} = x_{kj}) \\ 0 & \text{if } (x_{ij} \neq x_{kj} \text{ and } \phi \leq 0.5) \\ 1 & \text{if } (x_{ij} \neq x_{kj} \text{ and } \phi > 0.5) \end{cases} \tag{4}$$

where
v_{ij} = the new candidate solution i vertex j
x_{ij} = the current solution i vertex j
x_{kj} = the neighboring solution k vertex j
ϕ = a real number randomly generated between 0 to 1
$\wedge$ = the bit wise AND operation

The old solution in an employed bee's memory will be replaced with a new feasible solution if the new solution has a better fitness value. The fitness value of the solution can be calculated from the number of vertices in the set as shown in Eq. (5). The smaller the number of vertices in the set is, the better the fitness value.

$$fit_i = \frac{1}{1 + NVS_i} \tag{5}$$

where fit_i is the fitness value of the solution i and NVS_i is the number of vertices in the set of solution *i.*

After that, the employed bees' information will be shared with the onlooker bees waiting at the nest.

In choosing the optimal solutions, onlooker bees probabilistically choose their feasible solutions based on the information provided by the

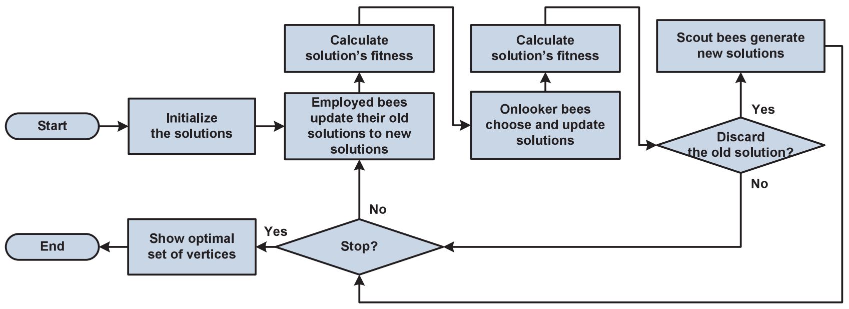

Start
Initialize the solutions
Employed bees update their old solutions to new solutions
Calculate solution's fitness
Onlooker bees choose and update solutions
Calculate solution's fitness
Scout bees generate new solutions
Discard the old solution?
Yes
No
Stop?
Yes
No
Show optimal set of vertices
End

Fig. 3. Flowchart of the ABC method for searching for the optimal set of vertices.

employed bees. The probability (P_i) that a solution shared by employed bees will be chosen by onlooker bees can be obtained from Eq. (6):

$$P_i = \frac{fit_i}{\sum_{i=1}^{SN} fit_i} \tag{6}$$

where fit_i is the fitness value of the solution i and SN is the number of feasible solutions.

The chance of a shared solution being chosen by the onlooker bees increases as the fitness value of the solution increases. After the onlooker bees have chosen solutions shared by employed bees, the onlooker bees will search for new feasible solutions based on updating their chosen solutions by using the information of their neighbors' solutions through the bit wise AND operation, the same step in seeking the feasible solutions process expressed by Eq. (4).

To avoid the suboptimal solution processes, any solution that has been updated but its quality is not improved will be discarded and a new solution that is initially determined by a scout bee will be regenerated in the same way that was used in the initial feasible solutions step. This keeps the algorithm from becoming stuck in a suboptimal solution.

The three processes described above will be looped continuously until the number of iterations reaches the predefined maximum number of iterations.

4. Experimental settings

Experiments were conducted to compare the performance of the proposed method with other metaheuristic based methods, including the Greedy algorithm [46], the Genetic Algorithm (GA) [47], the Particle Swarm Optimization (PSO) [48], and the Grey Wolf Optimizer (GWO) [49]. The objective was to compare the solutions obtained from the proposed approach with the other methods in terms of the minimum number of vertices. C++ was used to program all methods in this experiment, and all experiments were performed on a PC with an Intel Core i7 CPU, 3.6 GHz and 24 GB of memory. Experiments were conducted with both the vertex cover benchmark instances and the vertex cover in the real-world wireless sensor network problems.

For each method, the number of agents (bees in ABC, chromosomes in GA, particles in PSO, and wolves in GWO) was set to 200 and the number of iterations in each algorithm was set to 100. For the parameter settings of the GA method, the crossover probability (CR), the chance that two chromosomes exchange some of their parts, was set to 0.3, and the mutation rate (MR), the rate that determines how many chromosomes should be mutated in one generation, was set to 0.15. The parameters used in the PSO were defined as: $c_1 = c_2 = 2$, $\omega = 0.7$. For the parameter settings of the GWO method, the coefficient a was set to linearly decrease from 2 to 0 along iterations. For an accurate comparison, the number of function evaluations in each algorithm is defined as the same value, which is equal to 400 per iteration.

Note that these parameters were chosen by the experimenter in the preliminary study of this work. The preliminary results show that these specific values are suitable for use with each mentioned algorithm with the dataset in this experiment.

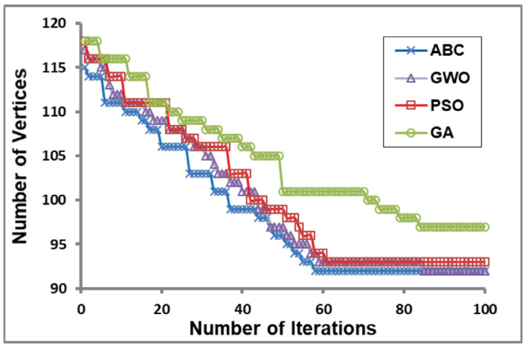


Fig. 4. Convergence speeds of proposed methods on the C125.9 benchmark instance.

5. Results and analysis

Firstly, the seven vertex cover benchmark instances of the DIMACS data set [50] were used to investigate the performance of the proposed method. Table 1 shows the comparative average number of vertices results of 25 runs with different initial solutions of the vertices for each benchmark instance for all of the methods. The smaller value of the average number of vertices means that the solutions are optimized closer to the optimal MVC (minimum vertex cover), or vice versa. The optimal MVC column displays the best known solution for the instance reported by Nguyen and Bui (2019).

It is obvious that the proposed method can achieve satisfactory performance for all benchmark instances. The average number of vertices obtained from the proposed method is smaller than the Greedy, the GA, and the PSO methods. This indicates that the proposed method can achieve the best quantitative results compared with the aforementioned methods. The improvements of the average number of vertices with our method when compared to the Greedy, the GA, and the PSO methods were an increase of 11%, 22%, and 10%, respectively.

Moreover, in terms of the convergence speed, the proposed technique also yields better results compared with the GA and the PSO techniques. The results of the convergence speed in terms of the number of iterations for these methods on the C125.9 benchmark instance are compared in Fig. 4. It shows that the proposed technique provides the convergence speed to optimal solution faster than the other compared techniques.

Next, to show the significance of the minimum vertex cover problem and to demonstrate that the proposed method is an effective alternative technique for solving the vertex cover problem, the identification of the minimum number of monitor nodes in a WSN was thus conducted in this experiment. The nodes in vertex cover set were assigned as monitor nodes. The objective of this experiment was to find the set of optimal monitor nodes in such a way that each link should be incident to at least one monitor node from this set.

Next, in order to demonstrate the performance of the proposed algorithm on the real-world problem, the experiment was conducted

Table 1
Average number of vertices results of 25 runs for each benchmark instance.

Benchmarks	Vertices	Edges	Optimal MVC	Greedy	GA	PSO	GWO	ABC
C125.9	125	787	91	93	98.16	92.56	**92**	**92**
hamming6_2	64	192	32	**32**	39.50	**32**	**32**	**32**
hamming8_2	256	1024	128	184	209.90	181.84	175.67	**163.72**
johnson8_2_4	28	168	24	**24**	**24**	**24**	**24**	**24**
johnson16_2_4	120	1680	112	113	112.50	112.32	112.24	**112**
keller4	171	5100	160	163	162.80	162.52	**162**	**162**
keller5	776	74710	749	760	762.20	761.64	760	**759.36**

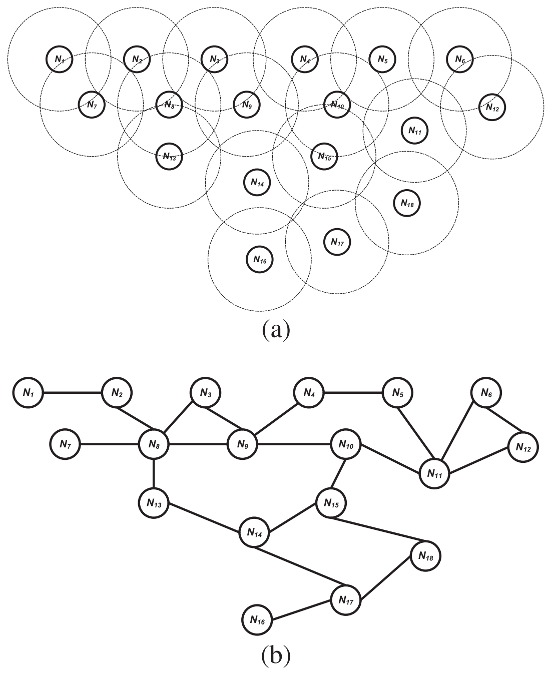


Fig. 5. An example of a wireless sensor network.

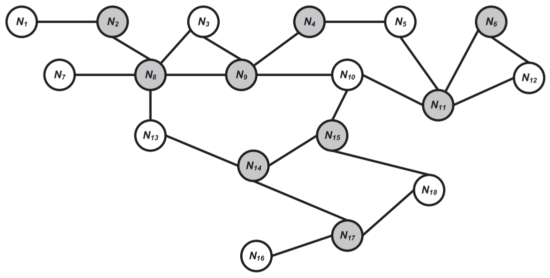


Fig. 6. Example set of optimal monitor nodes obtained using the proposed method.

on the wireless sensor network application. The network model used in this experiment is shown in Fig. 5(a), and its underlying communication graph with 18 nodes (N_1 to N_{18}) and 22 links is depicted in Fig. 5(b). The example set of optimal monitor nodes acquired by the proposed technique on this network model, consisting of nodes N_2, N_4, N_6, N_8, N_9, N_{11}, N_{14}, N_{15}, and N_{17}, is illustrated in Fig. 6. It can be seen that the proposed method can be used to solve the practical problem in the vertex cover domain and offers reasonable results.

To summarize, although all of the presented algorithms utilize a similar search process based on metaheuristic and biologically inspired computational methods, the ABC algorithm is able to provide better results than the Greedy, the GA, the PSO, and the GWO algorithms. Although the Greedy algorithm is usually very fast, the choices it makes may depend on earlier choices but not on the future ones. Consequently, the solutions obtained by the Greedy algorithm always converge to the local optimum. While the GA utilizes crossover operators to produce candidate solutions from the present ones, the ABC algorithm produces the candidate solutions from their parent by a simple operation based on obtaining the difference of the local best parts of the parent found by the onlooker bees and a randomly chosen solution from the population. This process increases the convergence speed of the search into a local optimum. When comparing the PSO, the GWO and the ABC algorithms, if any of the particles in the PSO, or the wolves in the GWO cannot locate a better global best solution after some amount of time, they will eventually converge about the existing one, which may not be the global optima due to a lack of exploration power, whereas the ABC algorithm employs both exploitation and explicit exploration in the search process. The exploitation is handled by the employed bees and onlooker bees, while the exploration is maintained by the scout bees in the ABC algorithm. If any solutions become entrapped at any local optima, the scout bees will randomly search for a new solution again. Consequently, this proposed technique based on the ABC method is able to maintain a level of quality for the obtained set of optimal vertices that is higher than that of the other aforementioned techniques. However, the performance of the proposed method is similar to that of the GWO in some cases because the convergence rate of the proposed method may slow down in later stages.

6. Conclusions

This work has presented a method that is efficient for handling the problem of minimizing the vertex cover. The artificial bee colony method was employed in this work to find the set of optimal vertices. Measurement of the performance of the proposed method was conducted by comparing the results with other metaheuristic techniques, including the GA and PSO methods, based on the perspective of the minimum number of vertices found by each method.

The experimental results showed that the proposed method offers a better set of optimal vertices compared to the other conventional metaheuristics based on biologically inspired techniques. Moreover, the amount of computation time in terms of the number of iterations required by the proposed approach is much lower when compared to the other methods. It can thus be concluded that the proposed technique has the potential for further improvement as a solution for the vertex cover problem.

Although the proposed method provides better results compared to other methods, the limitations of the proposed method are the potential slowing of the convergence rate in later stages and the occasional inability of the accuracy of the optimal value to fulfill the requirements. Thus, there is still room for improvement in the capability of the proposed method. In future work, the sensitivity of the parameter settings on the proposed method will be addressed. Additional novel approaches based on metaheuristics should be studied and applied to the minimum vertex cover problem. Furthermore, it is also planned that, in upcoming research, this proposed method will be applied to other real-world applications related to the minimum vertex cover problem.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

No data was used for the research described in the article.

References

[1] D. Zhao, S. Yang, X. Han, S. Zhang, Z. Wang, Dismantling and vertex cover of network through message passing, IEEE Trans. Circuits Syst. II 67 (11) (2020) 2732–2736.
[2] Z.A. Dagdeviren, A metaheuristic algorithm for vertex cover based link monitoring and backbone formation in wireless ad hoc networks, Expert Syst. Appl. 213 (2023) 118919.
[3] Y. Yigit, V.K. Akram, O. Dagdeviren, Breadth-first search tree integrated vertex cover algorithms for link monitoring and routing in wireless sensor networks, Comput. Netw. 194 (2021) 108144.
[4] A. Gupta, M. Kaur, Text summarisation using Laplacian centrality-based minimum vertex cover, J. Inf. Knowl. Manag. 18 (04) (2019) 1950050.
[5] N. Lindner, J. Reisch, An analysis of the parameterized complexity of periodic timetabling, J. Sched. 25 (2) (2022) 157–176.
[6] D. Angel, A graph theoretical approach for node covering in tree based architectures and its application to bioinformatics, Netw. Model. Anal. Health Inf. Bioinform. 8 (1) (2019) 12.
[7] R.M. Karp, Reducibility among combinatorial problems, in: Complexity of Computer Computations, Springer, Boston, MA, 1972, pp. 85–103.
[8] J. Chen, L. Kou, X. Cui, An approximation algorithm for the minimum vertex cover problem, Procedia Eng. 137 (2016) 180–185.
[9] S. Cai, K. Su, Q. Chen, EWLS: A new local search for minimum vertex cover, in: Twenty-Fourth AAAI Conference on Artificial Intelligence, 2010, pp. 45–50.
[10] X. Xu, J. Ma, An efficient simulated annealing algorithm for the minimum vertex cover problem, Neurocomputing 69 (7–9) (2006) 913–916.
[11] A. Mousavian, A. Rezvanian, M.R. Meybodi, Solving minimum vertex cover problem using learning automata, 2013, arXiv preprint arXiv:1311.7215.
[12] O. Ugurlu, New heuristic algorithm for unweighted minimum vertex cover, in: 2012 IV International Conference Problems of Cybernetics and Informatics (PCI), 2012, pp. 1–4.
[13] S. Çınaroğlu, S. Bodur, A new hybrid approach based on genetic algorithm for minimum vertex cover, in: 2018 Innovations in Intelligent Systems and Applications (INISTA), 2018, pp. 1–5.
[14] H. Khattab, B.A. Mahafzah, A. Sharieh, A hybrid algorithm based on modified chemical reaction optimization and best-first search algorithm for solving minimum vertex cover problem, Neural Comput. Appl. 34 (2022) 15513–15541.
[15] H. Qiu, C. Sun, X. Wang, W. Sun, Q. Zhou, A population-based game-theoretic optimizer for the minimum weighted vertex cover, Appl. Soft Comput. 116 (2022) 108272.
[16] Y.J. Zhang, X.D. Mu, X.W. Liu, X.Y. Wang, X. Zhang, K. Li, T.Y. Wu, D. Zhao, C. Dong, Applying the quantum approximate optimization algorithm to the minimum vertex cover problem, Appl. Soft Comput. 118 (2022) 108554.
[17] P. Guo, C. Quan, H. Chen, MEAMVC: A membrane evolutionary algorithm for solving minimum vertex cover problem, IEEE Access 7 (2019) 60774–60784.
[18] F.N. Abu-Khzam, A. El-Wahab, M. Mohamed, M. Haidous, N. Yosri, Learning from obstructions: An effective deep learning approach for minimum vertex cover, Ann. Math. Artif. Intell. (2022) 1–12, http://dx.doi.org/10.1007/s10472-022-09813-2.
[19] V.K. Akram, O. Ugurlu, A localized distributed algorithm for vertex cover problem, J. Comput. Sci. 58 (2022) 101518.
[20] S. Zhuang, D. Chen, A novel algorithm for the vertex cover problem based on minimal elements of discernibility matrix, Int. J. Mach. Learn. Cybern. 10 (12) (2019) 3467–3474.
[21] Q. Zhou, X. Xie, H. Dai, W. Meng, A novel rough set-based approach for minimum vertex cover of hypergraphs, Neural Comput. Appl. 34 (24) (2022) 21793–21808.
[22] L. Wang, S. Hu, M. Li, J. Zhou, An exact algorithm for minimum vertex cover problem, Mathematics 7 (7) (2019) 603.
[23] X.S. Yang, Z. Cui, R. Xiao, A.H. Gandomi, M. Karamanoglu (Eds.), Swarm intelligence and bio-inspired computation: theory and applications, Newnes (2013).
[24] A.K. Kar, Bio inspired computing–a review of algorithms and scope of applications, Expert Syst. Appl. 59 (2016) 20–32.
[25] M.H. Nadimi-Shahraki, S. Taghian, S. Mirjalili, An improved grey wolf optimizer for solving engineering problems, Expert Syst. Appl. 166 (2021) 113917.
[26] M.H. Nadimi-Shahraki, S. Taghian, S. Mirjalili, H. Faris, MTDE: An effective multi-trial vector-based differential evolution algorithm and its applications for engineering design problems, Appl. Soft Comput. 97 (2020) 106761.
[27] M.H. Nadimi-Shahraki, S. Taghian, S. Mirjalili, L. Abualigah, M. Abd Elaziz, D. Oliva, EWOA-OPF: effective whale optimization algorithm to solve optimal power flow problem, Electronics 10 (23) (2021) 2975.
[28] M.H. Nadimi-Shahraki, S. Taghian, S. Mirjalili, H. Zamani, A. Bahreininejad, GGWO: Gaze cues learning-based grey wolf optimizer and its applications for solving engineering problems, J. Comput. Sci. 61 (2022) 101636.
[29] D. Karaboga, An idea based on honey bee swarm for numerical optimization, Technical report-tr06, Erciyes university, engineering faculty, computer engineering department, 2005.
[30] D. Karaboga, B. Basturk, A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm, J. Global Optim. 39 (3) (2007) 459–471.
[31] D. Karaboga, B. Basturk, On the performance of artificial bee colony (ABC) algorithm, Appl. Soft Comput. 8 (1) (2008) 687–697.
[32] J.C. Bansal, H. Sharma, S.S. Jadon, Artificial bee colony algorithm: a survey, Int. J. Adv. Intell. Paradigms 5 (1/2) (2013) 123–159.
[33] D. Karaboga, B. Gorkemli, C. Ozturk, N. Karaboga, A comprehensive survey: artificial bee colony (ABC) algorithm and applications, Artif. Intell. Rev. 42 (1) (2014) 21–57.
[34] B. Akay, D. Karaboga, A survey on the applications of artificial bee colony in signal, image, and video processing, Signal Image Video Process. 9 (4) (2015) 967–990.
[35] Q.K. Pan, L. Wang, K. Mao, J.H. Zhao, M. Zhang, An effective artificial bee colony algorithm for a real-world hybrid flowshop problem in steelmaking process, IEEE Trans. Autom. Sci. Eng. 10 (2) (2012) 307–322.
[36] A. Banharnsakun, Artificial bee colony approach for enhancing LSB based image steganography, Multimedia Tools Appl. 77 (20) (2018) 27491–27504.
[37] H. Rao, X. Shi, A.K. Rodrigue, J. Feng, Y. Xia, M. Elhoseny, X. Yuan, L. Gu, Feature selection based on artificial bee colony and gradient boosting decision tree, Appl. Soft Comput. 74 (2019) 634–642.
[38] A. Banharnsakun, Towards improving the convolutional neural networks for deep learning using the distributed artificial bee colony method, Int. J. Mach. Learn. Cybern. 10 (6) (2019) 1301–1311.
[39] T.H. Cormen, C.E. Leiserson, R.L. Rivest, C. Stein, Introduction to Algorithms, MIT Press, 2009.
[40] Z. Hao, N. Qu, X. Dang, J. Hou, Node optimization coverage method under link model in passive monitoring system of three-dimensional wireless sensor network, Int. J. Distrib. Sens. Netw. 15 (8) (2019) 1550147719869877.
[41] R. Elhabyan, W. Shi, M. St-Hilaire, Coverage protocols for wireless sensor networks: Review and future directions, J. Commun. Netw. 21 (1) (2019) 45–60.
[42] J. Amutha, S. Sharma, J. Nagar, WSN strategies based on sensors, deployment, sensing models, coverage and energy efficiency: Review, approaches and open issues, Wirel. Pers. Commun. 111 (2) (2020) 1089–1115.
[43] Z.A. Dagdeviren, Weighted connected vertex cover based energy-efficient link monitoring for wireless sensor networks towards secure internet of things, IEEE Access 9 (2021) 10107–10119.
[44] H. ZainEldin, M. Badawy, M. Elhosseini, H. Arafat, A. Abraham, An improved dynamic deployment technique based-on genetic algorithm (IDDT-GA) for maximizing coverage in wireless sensor networks, J. Ambient Intell. Humaniz. Comput. 11 (10) (2020) 4177–4194.
[45] P. Liu, Z. Zhang, X. Huang, Approximation algorithm for minimum weight connected-k-subgraph cover, Theoret. Comput. Sci. 838 (2020) 160–167.
[46] Q. Xie, Y. Li, S. Hu, Y. Zhu, H. Wang, Two heuristic algorithms for the minimum weighted connected vertex cover problem under greedy strategy, IEEE Access 10 (2022) 116467-116472.
[47] A. Muñoz, F. Rubio, Evaluating genetic algorithms through the approximability hierarchy, J. Comput. Sci. 53 (2021) 101388.
[48] Y. Zhang, S. Wang, G. Ji, A comprehensive survey on particle swarm optimization algorithm and its applications, Math. Probl. Eng. 2015 (2015).
[49] M. Roayaei, On the binarization of Grey Wolf optimizer: a novel binary optimizer algorithm, Soft Comput. 25 (23) (2021) 14715–14728.
[50] T.H. Nguyen, T. Bui, Vertex cover benchmark instances, [Internet]. [cited 2019 Nov 19]. Available from: https://turing.cs.hbg.psu.edu/txn131/vertex_cover.html.