The Journal of Supercomputing (2023) 79:19746–19769
https://doi.org/10.1007/s11227-023-05397-8

A new robust approach to solve minimum vertex cover problem: Malatya vertex-cover algorithm

Selman Yakut[1] · Furkan Öztemiz[1] · Ali Karci[1]

Accepted: 13 May 2023 / Published online: 3 June 2023
© The Author(s), under exclusive licence to Springer Science+Business Media, LLC, part of Springer Nature 2023

Abstract

The minimum vertex-cover problem (MVCP) is an NP-complete optimization problem widely used in areas such as graph theory, social network, security and transportation, etc. Different approaches and algorithms have been proposed in the literature to solve this problem, since MVCP is an optimization problem, the solutions developed for this problem could be more intuitive and give results under certain constraints. In addition, the proposed solution methods for this problem could be more effective, and the determined solution sets change in each iteration. The algorithms/methods developed for solving MVCP are mostly based on heuristic or greedy approaches. This study presents the Malatya vertex-cover algorithm, which provides an efficient solution and a robust approach based on the Malatya centrality value algorithm for MVCP. Although MVCP is an NP-complete problem that cannot be solved in polynomial time, the proposed method offers a polynomial solution to this problem, and the obtained solutions are optimum or near-optimum (optimal solution). This algorithm consists of two basic steps. In the first step, the Malatya centrality values of the nodes in the graph are calculated using the Malatya centrality algorithm. The Malatya centrality value of the nodes in any graph is the summation of the ratio of the node's degree to the adjacent nodes' degrees for each node. In the second step, nodes are selected for the MVCP solution based on the node with the maximum Malatya centrality value (Ψ) in the graph is selected and added to the solution set. Then this node and the edges incident on this node are removed from the graph. For the graph consisting of the remaining nodes, Malatya centrality values are calculated again, and the selection process is continued. The process is terminated when all edges in the graph are covered. The proposed algorithm has been tested on artificial, actual graphs and large-scale random graphs produced with the Erdos–Renyi model. When the results are examined, the proposed algorithm yields a robust solution set in polynomial time and polynomial space independent of constraints. In addition, the successful test results in the sample graphs and the analysis of the proposed approach show the effectiveness/superiority of the Malatya centrality algorithm and the proposed method.

Extended author information available on the last page of the article

Keywords Minimum vertex-cover · Graph theory · Malatya centrality algorithm · Malatya vertex-cover algorithm · Centrality algorithm

1 Introduction

A graph is a basic data model consisting of node and edges [1]. This model is widely used in real-life problems and computer science [2]. The solutions and methods defined on the graph enable to development of solutions and approaches in these areas. However, the graph contains many NP problems [3]. One of these problems is the vertex-cover problem. The node cover problem determines which nodes to choose to cover all the edges in any graph [1]. The vertex-cover problem is one of the NP-complete problems in graph theory. Covering all edges in any graph with a minimum number of nodes is called the minimum vertex-cover problem (MVCP) [4].

Assume that any undirected graph G consists of nodes ($V=\{v_1, v_2, \dots v_n\}$) and edges ($E=\{e_1, e_2, \dots e_m\}$), with $G=(V, E)$. Vertex-Cover C covers all edges in the graph using subsets (C') of nodes in any graph such that it is a subset of G ($C \subseteq V$). If (v_i, v_j) is an edge of G, then for the edges connecting these nodes, there is at least one node including each edge, such as $v_i \in C$ or $v_j \in C$; $v_i \in C$ and $v_j \in C$. For any node v_i, $\{v_i \in V | (v_i, v_j) \in E\}$. Here the degree of v_i is the number of its neighbours. Determining the $C \subseteq V$ clusters where $|C|$ is minimum is called MVCP and is mathematically represented as [5].

$$\min \sum_{i=1}^{|V|} x_i \tag{1}$$

$$\text{subject to} : x_i + x_j \geq 1, \left(v_i, v_j\right) \in E \tag{2}$$

$$\text{where } x_i, x_j \in \{0, 1\} \text{ and } i = 1, 2, \dots |V| \tag{3}$$

where x_i corresponds to node v_i and if v_i is in solution then $x_i=1$, otherwise $x_i=0$.

MVCP is widely used in computer science for real-world modelling problems [6, 7]. Thus, the algorithms developed for MVCP and the solutions found can be used for these problems [8]. For example, the determination of sentry nodes to protect patient data [6], production of stable genetic sequences without repetitions [9], management of city traffic [10], practical use and placement of sensors [8], and addressing security applications in wireless networks with vertex-cover problem [11, 12], routing and monitoring in wireless mobile networks [13], connection monitoring and formation in ad hoc networks [14], presenting a method based on MVCP solution by using fuzzy graphs in case of disaster [15], and choosing facility locations are among the leading application area among these [16], etc.

MVCP is an NP-Complete optimization problem that cannot be solved in reasonable time (such as polynomial time) [17]. For this reason, it does not have an optimal solution that can be applied to all graphs and consists of specific steps

[5]. The proposed complete solutions are also offered for small sizes or some unique graphs. Therefore, different methods and approaches, especially heuristics and metaheuristics, have been proposed to solve MVCP. In general, these methods and approaches do not offer an optimal solution for MVCP. Only after a certain number of iterations and execution steps possible solution sets close to the optimum solution are determined. However, changing initial conditions or method-specific features hinder the determination of the solution set for MVCP. In addition, the complexity and size of real-life applications make solving MVCP more difficult.

In this study, a robust and efficient Malatya vertex-cover algorithm (MVCA) is proposed to solve MVCP based on Malatya centrality value algorithm proposed by same authors group [18], which is an NP-complete problem, effectively. The proposed algorithm consists of predictable processing steps and provides a polynomial time and polynomial space solution. First, the Malatya centrality value [18] is calculated for each node in the graph using the Malatya algorithm, which is a pragmatic algorithm. While calculating the Malatya centrality value of the nodes, the adjacent node degrees are used together with the node's degree. Then, the nodes selected for MVCP are determined using Malatya centrality values (Ψ). First, the node with the maximum value is selected and added to the solution set. This node and its incident edges are then extracted from the graph. Finally, the Malatya vertex-cover algorithm is applied for the new formed graph, and the new Malatya centrality values are calculated. The operations in the reformed graph are continued until the vertex-cover solution set is found.

Our contributions to the paper study:

1. An essential contribution of this study is that it provides a robust and effective algorithm for MVCP with particular steps. This algorithm provides a solution for MVCP without any restrictions, independent of the graph structure, initial conditions, and algorithm-related features such as the number of iterations or the graph itself.
2. The proposed algorithm determines a robust and effective solution for MVCP. For the same graph, the same solution set is selected without any constraints, such as the initial population and the number of iterations. The analyses made on the commonly used graph models and the determined solution sets show the method's effectiveness.
3. Malatya centrality algorithm was used to determine the nodes constituting the optimum or near-optimum (optimal) solution for MVCP. With the pragmatic approach of the Malatya centrality method for nodes, the solution set was calculated effectively. The proposed method has been shown to offer a general solution.
4. Experimental examples demonstrate the determination of an effective solution set for an NP-complete optimization problem. The method's effectiveness has been shown by experimental studies on graphs that are widely used and accepted in graph theory.
5. The computational steps, along with the time and computational complexity, can be made for any graph. While time and cost cannot be predicted for approximate

methods, these calculations are made explicitly for the graph given with the proposed method. In addition, the proposed method provides a solution for MVCP in polynomial time and polynomial space.

6. An approach that provides a solution for MVCP using the centrality values is proposed. Thus, starting from the node where the Malatya centrality value is predominant, it is fast to determine the solution set. These can be seen with the given sample application and analysis.
7. It has been shown that the proposed algorithm produces effective solution sets for graphs that the Integer Linear programming approach cannot solve due to reasons such as memory insufficiency.

The rest of the article is as follows. First of all, the literature on MVCP was included (Sect. 2). Then, Sect. 3 mentions the motivation observed in the proposed study. Then, the proposed algorithm is discussed in Sect. 4. The Malatya centrality value is calculated, and the solution set for MVCP is determined. The fifth section evaluates the proposed algorithm and applications on sample graphs. Finally, the results related to the proposed algorithm are presented in the conclusion.

1.1 Motivation of proposed approach

The primary motivation in this study is to develop an effective and pragmatic solution algorithm for MVCP, which is an NP-complete optimization problem. This problem is used in modelling many real-world problems and developing possible solutions to these problems. Since MVCP is an NP-complete problem, it is one of the most challenging problems to solve in finite step and polynomial time [5]. Therefore, solutions developed for MVCP mostly rely on heuristics, metaheuristics, and optimization algorithms. On the other hand, these approaches and algorithms produce possible solution sets close to the optimum solution under certain constraints rather than finding a complete and clear solution. These constraints are generally in the form of initial conditions, the number of iterations, and constraints specific to the algorithm used. Therefore, it is seen that there is a need for an approach that is independent of all these constraints for MVCP and can reach a result in a certain step depending on the problem being solved and determine an effective solution set.

Existing methods in the literature find possible solution sets for MVCP solution by mainly scanning with brute force perspective. Moreover, as the graph's structure grows and its complexity increases, it is much more challenging to get results with brute force methods. To solve this problem with optimization methods, the number of iterations is kept at a certain number. This restricts the search space and prevents better results. In addition, the uncertainty and unpredictability in brute force-like approaches can be eliminated with full and clear steps.

These methods offer solutions independently of them without considering the structure of the graph and the characteristics of the nodes. However, each graph structure and node has characteristics, such as the number of neighbours and edges. In an example graph, all nodes do not have the same weight and neighbourhood.

Therefore, developing an approach that considers all these parameters provides more effective solutions.

Another motivation for this study is that it is essential to solving real-life problems modelled using MVCP. The proposed approach can solve many real-world problems, such as transportation, security, and network structures modelled using MVCP. Thus, it is possible to solve these problems, consisting of large and complex structures, in polynomial time and polynomial space within specific and predictable steps.

2 Related work

Since MVCP is an NP-complete problem, the optimum solution set for any graph has not yet been determined precisely in polynomial time and polynomial space. Therefore, it is aimed to find node clusters close to the optimum solution to solve MVCP. Different approaches have been presented for this purpose [19].

The first of these approaches is the use of statistical methods for MVCP. First, the selection of nodes that make up the solution with a small number of queries is to be determined using statistical methods [20]. Although only some queries are made in this approach, they cannot be solved in polynomial time. Next, a new approach is presented for vertex-cover problem in dynamic networks by using probability methods and adaptive convolution algorithms together [21]. In this method, the efficiency of the adaptive development algorithm for vertex cover has been demonstrated, and it is sufficient to use fewer node.

Since MVCP is an NP-complete optimization problem, it has been widely solved by approximate methods in the literature. At the forefront of these are the heuristic and metaheuristic methods. These methods determine the minimum possible solution sets close to the optimum solution. A new method is proposed to solve MVCP using chemical reaction optimization and the best solution algorithm [4]. The proposed method in this study gave better results than the classical optimization methods. Although a solution in linear time cannot be developed using the quantum approximate optimization algorithm, a probabilistic approach that can be performed on quantum computers is given [19]. Another optimization algorithm proposed to solve the minimum vertex-cover problem is the most valuable player algorithm [22]. It has been shown that the proposed algorithm with sample applications and analyses produces effective solution sets. MVCP has been solved with the new evolutionary algorithm developed based on the membrane evolutionary algorithm [23]. It has been shown that this method provides effective solutions for optimization methods using its own parameters. Another study proposed to solve MVCP using an advanced heuristic algorithm by transferring the graph structure to the decision table [24]. To increase the effectiveness of the proposed method, a quantum-behaved particle swarm optimization with an immune mechanism is presented. Thus, bad initial situations were avoided, and global research capability was increased [24]. A heuristic approach has been developed to solve MVCP based on its genetic algorithm [25]. He proposed an efficient memetic algorithm to solve the minimal partial overhead cover problem [26]. This proposed method includes operations similar to the genetic algorithm. In another proposed approach, a solution for MVCP is presented with a developmental search and iterative neighbourhood search-based approach [27]. An evolutionary game algorithm is prosed to solve minimum weighted vertex-cover problem [28]. Chemical reaction optimization is used to solve generalized vertex-cover problem [29].

It is common in the literature to search for possible solutions in local examples using heuristic methods. Different optimization approaches using this local data have been proposed to solve MVCP. Li et al. proposed a new local search algorithm based on taboo strategy and permutation mechanism for the generalized vertex-cover problem [30]. They proposed an effective method for solution quality and computation without getting stuck with local solutions. For MVCP in Jovanovic et al. weighted graphs, the method algorithm is prevented from getting stuck in the local minimum in total values, and learning algorithms are added in problem-solving [31]. MVCP was solved using a three-step method using the local search algorithm with the NuMWVC algorithm [32]. The problem with large graphs is solved with the presented approach for the local search algorithm to calculate MVCP on large graphs [33]. There are also suggested approaches for solving MVCP in large graphs using the local search approach [34]. An effective approach not stuck in local solutions is proposed in the local search approach given on weighted graphs [35]. Generalized vertex-cover problem is solved by an improved local search algorithm [36]. Two dynamic strategies are proposed to enhance a state-of-the-art local search by modifying the behaviour of the algorithm while it is running the search [37]. A brand-new, highly parametric local search framework for MVCP is described. It includes a variety of efficient local search methods [38].

There are approaches in the literature that use specific heuristics to solve MVCP. One of them is to know the value of the two highest-order nodes and to determine the solution set over these values [39]. A new method based on Shapley–Shubik index estimation was proposed by Gusev for MVCP by using graph theory and game theory together [10]. Better results were obtained in the approach developed for MVCP in weighted graphs with game-theoretic learning [40]. In another approach, MVCP is presented in an approach that solves the used graph by separating it into subgraphs [41]. MVCP is solved in the graph divided into layers with the multistage vertex-cover approach [42]. In the approach, the solution is reasonable under some constraints unless there are too many (too different) mismatched nodes in both layers. A potential game for the vertex-cover problem is present for minimum vertex covering states of a general complex network [43]. This paper suggests a general algorithm framework to complete that finding tiny vertex covers in enormous sparse graphs is necessary for complex network analysis [44].

Some of the existing methods developed to solve MVCP are greedy-based approaches. Three metaheuristic methods have been used to monitor energy efficiency in new wireless mobile networks using a combination of genetic algorithms and greedy approaches [8]. With the hybrid use of developmental optimization methods, new methods have been proposed. The vertex-cover problem has been solved using population-based iterative greedy developmental algorithms, which are effective against the graph's theoretical problems [14]. The effectiveness of the proposed method is given in sample graphs and benchmarks. A metaheuristic method based on the bi-objective greedy approach is presented to solve MVCP [31]. A multi-objective approach has been developed that combines the bi-objective method and the neighbourhood search algorithm for weighted graphs [45]. Effective results have been obtained in the approaches developed for MVCP with the population-based iterative greedy approach [46].

One of the proposed solution approaches for MVCP is exact algorithms [47]. These algorithms, unlike the optimization methods, give the sample solution set. A similar algorithm is presented for the independent set.

2.1 Literature gaps

When the literature is examined, it is seen that MVCP is mainly considered an optimization method, and solution sets are determined using heuristic methods. It is seen that these possible solutions are not clear and the algorithm used depends on parameters such as the number of iterations and the initial population, and the obtained solution is not stable for each execution. Therefore, these methods do not give exact solutions independent of the conditions. On the other hand, the special algorithms used are based on some restrictions and assumptions in the solved problem. Therefore, it is seen in the literature that these methods cannot provide general and complete solutions. However, the approaches that give the full solution set offer solutions in some special graphs and under constraints. No exact solution algorithm can provide an effective solution, especially for large graph structures and real-life problems.

A new robust and effective approach is presented in the literature to select vertex-cover nodes by using the relationship of node and edges that make up the graph structure. In this approach, the Malatya centrality algorithm was presented to establish the node selection priority, and vertex-cover members were selected according to this centrality value. Malatya centrality values in the Malatya algorithm, are determined by using the node's degree and neighbouring nodes' degrees together. The amount of processing and computational complexity required to determine the solution set can be predicted in advance with this approach. It is based on the greedy approach since the maximum of Malatya values calculated for nodes is used. Therefore, a new approach has been brought to the literature with the proposed approach.

In Table 1, the approaches of Malatya vertex-cover algorithm and other minimum vertex-cover algorithms in the literature are compared. Algorithms were evaluated in various aspects of the method such as robust, deterministic, and time complexity. When Table 1 is examined, Malatya vertex-cover algorithm offers deterministic and robust solutions in polynomial time. In addition, the proposed method gives the optimal result in any graph, while it gives the optimum result in regular graphs.

3 Proposed robust the Malatya vertex-cover algorithm

In this study, an effective new robust algorithm was proposed for detecting the members of the solution set of MVCP. The proposed algorithm consists of two parts. These are the calculation of Malatya centrality values and node selection for MVCP. The proposed algorithm is shown with sample applications and graphs; in which

Table 1 Comparisons of algorithms for minimum vertex-cover problem

Approach	Robust	Deterministic	Time complexity	Space complexity	Optimum	Same result for all executions
Statistical	No	No	N.A	Polynomial/exponential	Optimal, not stable	No
Heuristic	No	No	N.A	Polynomial	Optimal, not stable	No
Greedy	Yes	Yes	Polynomial	Polynomial	Optimal	Yes
Exact	Yes	Yes	Exponential	Exponential in recursion case	Optimum	Yes
Malatya VC	Yes	Yes	Polynomial	Polynomial	Optimum for regular graphs	Yes

it detects the minimum vertex-cover values. However, the solution set claimed by this algorithm is the vertex-cover solution set, which gives near-optimal results. The general outlines of the proposed algorithm are shown in Fig. 1. The unweighted and undirected graph has been brought into a suitable form for the application of the algorithm to Stage 1. The centrality values of the nodes were calculated by applying the Malatya algorithm to the graph given as input in Stage 2. The node with the highest centrality value is selected as the first vertex-cover node and proceeds to Stage 3. The selected vertex-cover node and edges incident on are removed from the graph, the graph is updated, and Stage 2' is returned to calculate the Malatya centrality value again. In each iteration, one node is selected for the vertex-cover set, and each iteration's Malatya centrality value is recalculated. The vertex-cover node selection continues until all edges in the graph are deleted. Selected vertex-cover nodes are presented as output in Stage 4 at the final stage.

3.1 Malatya centrality values calculation

The centrality is used in many fields, especially graph theory and network analysis. The centrality is weighting the nodes and assigning values to them depending on their position in the graph relatively to their neighbours. In many applications, it is aimed at finding the most efficient node in the graph or network. Numerous algorithms have been proposed to find the central node or nodes [48]. However, the connections on the node are at the beginning of the determining parameters to measure the graph's centrality. This approach, called degree centrality, is included in the structure of many algorithms, such as PageRank, which is widely used [49].

The most crucial step of the proposed approach for MVCP solution is the calculation of Malatya centrality $((v_i))$ values. Malatya centrality algorithm, which offers a pragmatic and effective approach, was used to calculate these values. Malatya centrality values are calculated separately for each node in the graph using the Malatya

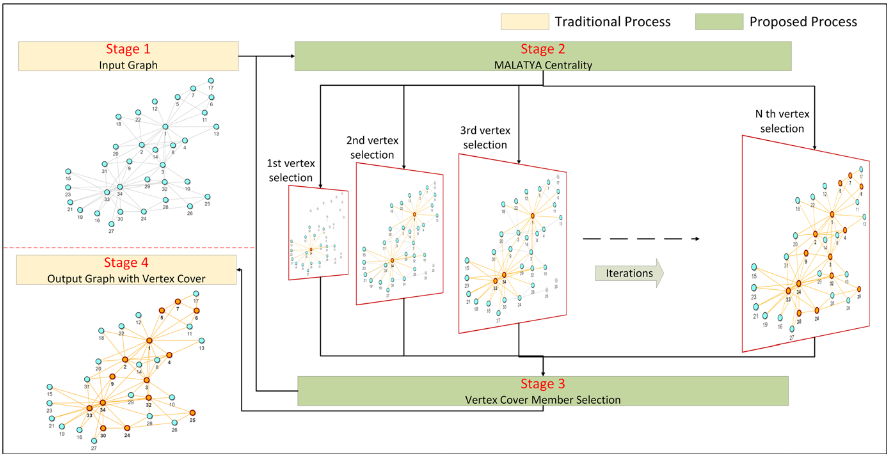


Fig. 1 General structure of the proposed algorithm

centrality algorithm. In calculating this value, each node's degree and its adjacent nodes' degrees are used for each node. The nodes selected for vertex cover are determined by using these values. Thus, by presenting a robust approach that solves MVCP polynomial time and polynomial space, a problem that cannot be solved in polynomial time, the solution is ensured to be in finite steps and in a predictable time.

The algorithm used to calculate the Malatya centrality algorithm node $\Psi\left(v_i\right)$ values is given in Eq. (1). In this equation, n represents the number of nodes in the graph. For a node v_i, the set of neighbour nodes is defined as $N(v_i)$. If two nodes are connected by the same edge, these nodes are called neighbours. The degree of a node is the sum of the edges connected to it. The Malatya centrality value of each node consists of the sum of the values obtained by dividing the node's degree by the degree of each neighbour node. Therefore, in determining the Malatya centrality value of the node, the number and degree of adjacent nodes are also effective, as is the node itself.

$$\Psi\left(v_i\right) = \sum_{\forall v_j \in N\left(v_i\right)} \frac{d\left(v_i\right)}{d\left(v_j\right)} \tag{4}$$

Theorem 1 *Malatya centrality algorithm has effective and polynomial time complexity for MVCP.*

Proof Assume that $G_0 = (V, E)$ is a simple graph.$\delta\left(G_0\right)$ is minimum node degree and $\Delta\left(G_0\right)$ is maximum node degree.

$V_c = \phi$ // V_c is a vertex-cover set. $v_j \in V$ and

$$v_j = \arg\max\left\{ v_i | \Psi\left(v_i\right) = \sum_{\forall v_j \in N\left(v_i\right)} \frac{d\left(v_i\right)}{d\left(v_j\right)}, \quad 1 \le i \le |V| \text{ and } 1 \le i \le |V|, \quad i \ne j \right\}$$

$$\mathrm{V}_c = V_c\left\{v_j\right\}$$

$$G_1 = \left\{V_1, E_1\right\} \text{ and } V_1 = V - v_j, E_1 = E - \left\{\forall (v_r, v_j) \in E, v_r \in N\left(v_j\right)\right\}$$

This step requires maximum $|V|.\, \Delta(G)$ aritmetrik operations.

If $E_1 \ne \phi$, the same process will be applied to G_1.

Assume that $v_j \in V_1$ and

$$v_j = \arg\max\left\{ \Psi\left(v_i\right) = \sum_{\forall v_j \in N\left(v_i\right)} \frac{d\left(v_i\right)}{d\left(v_j\right)}, \quad 1 \le i, \quad j \le \left|V_1\right|, \quad i \ne j \right\}$$

$V_C = V_C \cup \{v_j\}$ and

$$G_2 = \{V_2, E_2\} \text{ and } V_2 = V_1 - v_j \text{ and } E_2 = E_1 - \{\forall\,(v_r, v_j) \in E,\ v_r \in N(v_j)\}$$

The number of arithmetic operations is less then $|V|.\ \Delta(G)$.

If $E_1 \neq \phi$, the same process will be applied to G_2, otherwise, algorithm is terminated.

Assume that the obtained graph series is.

$G_0, G_1, G_2, \dots G_m$.

The total number of arithmetic operations is

$$\sum_{i=0}^{m-1} |V_i|.\Delta(G) < \sum_{i=0}^{m-1} |V|.\Delta(G) = m|V|.\Delta(G). \tag{5}$$

Finally,

$$T(n) = O\left(\sum_{i=0}^{m-1} |V|.\Delta(G)\right)$$

and $T(n)$ is the time complexity of algorithm.

Corollary *Malatya centrality algorithm has polynomial space complexity.*

Proof Malatya centrality algorithm uses at most two adjacency matrices. So space complexity is $O(|V|^2)$.

3.2 Malatya vertex-cover algorithm

The most basic and vital parameter to solve MVCP is to determine the nodes to be selected. Malatya centrality algorithm is used to determine the nodes to be selected in the proposed approach. This algorithm calculates a centrality value for each node in the graph with a pragmatic approach. This value, called the Malatya centrality value, enables it to produce an effective solution set for MVCP. This proposed approach determines the solution set for MVCP, unlike the optimization or heuristic methods.

After calculating the Malatya centrality values, the node with the maximum Malatya centrality value is selected. Then, the selected node and its incident edges are removed from the graph. Malatya centrality values are recalculated for the new formed graph, and selection processes are continued. This calculation and subtraction are continued until all edges are covered. When all edges are covered, the node selection process is terminated. The resulting output will be the minimum edge coverage set or near-optimal edge cover set. The extended pseudocode of the proposed approach for MVCP is given in algorithm 1. In the given codes, the operations of the proposed algorithm are listed in order. Explanations about these codes are shown next to the relevant piece of code.

Algorithm 1. Proposed Algorithm Pseudo Code

Malatya Vertex-Cover Algorithm

```
G:(V, E)                                          // G graph
vector = c()                                      // Create an empty array named vector.
MalatyaCentrality <- function(g)                  // Function that calculates Malatya centrality value
  VertexList = c(V(g))                            // Assign vertex list to variable
  for (i in VertexList)                           // Work up to the number of vertex
    Vertexdegree = degree(g,v = V(g)[i])          // Calculate the adjacent node degrees of the given node
    NeighborsDegree = degree(g,v = neighbors(g,v = V(g)[i])) // Calculate the node degree of the neighbors of the relevant node
    Value = Vertexdegree /NeighborsDegree          // Degree of related node / degree of neighboring node
    vector = c(vector, sum(Value))                // Calculate the centrality value of the nodes
  return(vector)
FindMaximum <- function(graph)                    // Returns the node with the maximum Malatya centrality value
   data = data.frame(MalatyaCentrality(graph))
   return (order(data$MalatyaCentrality.graph.,decreasing = TRUE)[1])
FindVertexCover <- function()                     // Identifies Vertex Cover members
VertexCover = c()                                 // Create empty array
while(ecount(g) > 0 )                             // Work as long as the edge exists
    maxvertex = FindMaximum(g)[1]
    VertexCover = append(VertexCover,V(g)[maxvertex]) // Add the vertex cover node you find in each iteration to the array
    g = delete_vertices(g,maxvertex)              // Remove the selected node and its connections from the graph
Print(VertexCover)                                // Print detected Vertex Cover members
```

In algorithm 2, the mathematical expressions of the proposed MVCA operations are given. The proposed algorithm consists of calculating Malatya centrality values and vertex-cover operations. This algorithm uses $\Psi(v_i)$ Malatya centrality value, V_c solution set, $d(v_i)$ the degree of the node v_i, |*V*| indicates the number of nodes in the graph.

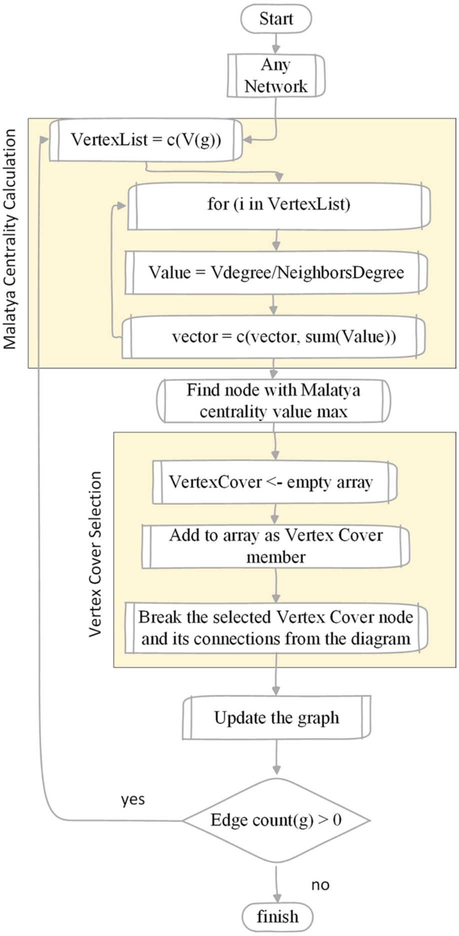
Algorithm 2. Mathematical representation of the Malatya Algorithm

Minimum Vertex-Cover Algorithm

```
Input: Adjacency matrix of G is A and G = (V, E)                    // G graph
Output: $V_c \subseteq V$, $V_c$ is a set of nodes and it is a solution for vertex-cover problem
    $V_c \leftarrow \varnothing$
    While $E \neq \varnothing$ do
        $i \leftarrow 1, \ldots, |V|$
            $\Psi(v_i) = \sum_{\forall v_j \in N(v_i)} \frac{d(v_i)}{d(v_j)}$
        $V_c = V_c \cup \{\max(\Psi(v_i))\}$
        $V = V - \{v_i\}$, and $E = E - \forall (v_i, v_j) \in E$
    Output=$V_c$
```

The flow chart of the proposed MVCA is given in Fig. 2. In this diagram, the operations of the proposed algorithm are shown in detail.

Fig. 2 Malatya vertex-cover algorithm's flow chart


Start
Any Network
Malatya Centrality Calculation
VertexList = c(V(g))
for (i in VertexList)
Value = Vdegree/NeighborsDegree
vector = c(vector, sum(Value))
Find node with Malatya centrality value max
Vertex Cover Selection
VertexCover <- empty array
Add to array as Vertex Cover member
Break the selected Vertex Cover node and its connections from the diagram
Update the graph
yes
Edge count(g) > 0
no
finish

Theorem 2 *Malatya vertex-cover algorithm finds optimum solutions for regular graphs.*

Proof In order to prove the theorem, we start with path graph, cycle graph, hypercube, and r-regular graph.

Case 1 In case of path graph, assume that $P_n = (V,E)$ is a path graph where $|V| = n$, $|E| = n - 1$. The Malatya centrality values are: $\Psi(v_1) = 1/2$, $\Psi(v_2) = 3$, $\Psi(v_3) = \Psi(v_4) = \Psi(v_5) = \ldots = \Psi(v_{n-2}) = 2$, $\Psi(v_{n-1}) = 3$, and $\Psi(v_n) = 1/2$. The solution set is $V_{\mathrm{mvc}} = \{\}$.

Without losing generality, $V_{\mathrm{mvc}} = V_{\mathrm{mvc}} \cup \{v_2\}$. Assume that v_2 is added to solution. v_1, v_2 nodes and (v_1, v_2), (v_2, v_3) edges are removed from P_n. $P_{n-2} = P_n - \{v_1, v_2\}$. $\Psi(v_3) = 1/2$ and $\Psi(v_4) = 3$ and Malatya centrality values are unchanged for the remaining nodes.

$V_{\mathrm{mvc}} = V_{\mathrm{mvc}} \cup \{v_4\}$ and $P_{n-4} = P_{n-2} - \{v_3, v_4\}$.

$\Psi\left(v_5\right) = 1/2$ and $\Psi\left(v_6\right) = 3$

$$V_{\mathrm{mvc}} = V_{\mathrm{mvc}} \cup \left\{v_6\right\} \text{ and } P_{n-6} = P_{n-4} - \left\{v_5, v_6\right\}$$

Finally, if n is even, $V_{\mathrm{mvc}} = V_{\mathrm{mvc}} \cup \{v_n\}$, otherwise, $V_{\mathrm{mvc}} = V_{\mathrm{mvc}} \cup \{v_{n-1}\}$. This solution is an optimum solution for path graph.

Case 2 Assume that $C_n = (V,E)$ is a cycle graph. The Malatya centrality values are $\Psi\left(v_1\right) = \Psi\left(v_2\right) = \Psi\left(v_3\right) = \ldots = \Psi\left(v_{n-1}\right) = \Psi\left(v_n\right) = 2$. Any node can be selected to V_{mvc} arbitrarily, since all nodes have same Malatya centrality values. This node and incident edges on this node are removed from graph C_n. The obtained graph is $P_{n-1} = C_n - \{$arbitrarily selected node$\}$, and it known that P_{n-1} is a path graph. Malatya vertex-cover algorithm is optimum for path graph. Consequently, Malatya vertex-cover algorithm is optimum for cycle graph.

Case 3 Assume that $H_n = (V,E)$, and $|V| = 2^n$ and $|E| = n2^n$. Without losing generality, the node label is $b_n b_{n-1} b_{n-2} \ldots b_2 b_1$. The neighbours of node $b_n b_{n-1} b_{n-2} \ldots b_2 b_1$ are as follow based on Hamming distance:

$N\left(b_n b_{n-1} \ldots b_2 b_1\right) = \left\{b_n b_{n-1} \ldots b_2 \overline{b_1}, b_n b_{n-1} \ldots \overline{b_2} b_1, b_n \overline{b_{n-1}} \ldots b_2 b_1, \overline{b_n} b_{n-1} \ldots b_2 b_1\right\}$.

The Malatya centrality values for all nodes are equal and

$$\begin{aligned}\Psi\left(b_n b_{n-1} \ldots b_2 b_1\right) &= \Psi\left(b_n b_{n-1} \ldots b_2 \overline{b_1}\right) = \Psi\left(b_n b_{n-1} \ldots \overline{b_2} b_1\right) \\ &= \ldots = \Psi\left(\overline{b_n b_{n-1}} \ldots \overline{b_2} b_1\right) = \Psi\left(\overline{b_n b_{n-1}} \ldots \overline{b_2 b_1}\right) = n.\end{aligned}$$

Assume that node $V_{\mathrm{mvc}} = \{b_n b_{n-1} b_{n-2} \ldots b_2 b_1\}$, and removing node $b_n b_{n-1} b_{n-2} \ldots b_2 b_1$ and edges incident on this node from H_n.

The Malatya centrality values for reformed graph are as follows:

$\Psi\left(b_n b_{n-1} \ldots b_2 \overline{b_1}\right) = \Psi\left(b_n b_{n-1} \ldots \overline{b_2} b_1\right) = \ldots = \Psi\left(\overline{b_n b_{n-1}} \ldots \overline{b_2} b_1\right) = \Psi\left(\overline{b_n b_{n-1}} \ldots \overline{b_2 b_1}\right) = \frac{(n-1)(n-1)}{n} < n$ and other nodes in reformed hypercube have Malatya centrality values as n except neighbours of $\left\{b_n b_{n-1} \ldots b_2 \overline{b_1}, b_n b_{n-1} \ldots \overline{b_2} b_1, b_n \overline{b_{n-1}} \ldots b_2 b_1, \overline{b_n} b_{n-1} \ldots b_2 b_1\right\}$.

$$\begin{aligned}\Psi\left(N(b_n b_{n-1} \ldots b_2 \overline{b_1})\right) &= \Psi\left(N\left(b_n b_{n-1} \ldots \overline{b_2} b_1\right)\right) = \ldots = \Psi\left(N\left(\overline{b_n b_{n-1}} \ldots \overline{b_2} b_1\right)\right) \\ &= \Psi\left(N\left(\overline{b_n b_{n-1}} \ldots \overline{b_2 b_1}\right)\right) = \frac{(n-2)(n-1) + 2n}{n-1} > n.\end{aligned}$$

One of the node which is element of set $N(b_n b_{n-1} \ldots b_2 \overline{b_1}) \cup N\left(b_n b_{n-1} \ldots \overline{b_2} b_1\right) \cup \ldots \cup N\left(\overline{b_n b_{n-1}} \ldots \overline{b_2} b_1\right) \cup N\left(\overline{b_n b_{n-1}} \ldots \overline{b_2 b_1}\right)$, is selected for V_{mvc}, since they have maximum Malatya centrality values. This process carries on in this way until all edges are covered. Malatya vertex-cover algorithm is optimum for hypercube.

Case 4 Assume that $G=(V,E)$ is a regular graph with node degree as r. Malatya centrality values for all nodes at the first step are equal to r. So, any node can be selected for minimum vertex-cover set, since all Malatya centrality values are equal. The neighbours of selected node have node degrees as $r-1$, since selected node and edges incident on it are removed from graph and new version of graph is reformed.

Assume that selected node is v_k. The Malatya centrality values for nodes of reformed graph are as follows:

$\Psi\left(v_j\right) = \frac{(r-1)(r-1)}{r} < r, v_j \in N\left(v_k\right).$

The neighbours of v_k at distance 2 have Malatya centrality values as $(r-1)(r-1) + r/r - 1 > r$ and all other nodes in reformed graph have Malatya centrality values r. The maximum Malatya centrality value is $(r-1)(r-1) + r/r - 1$. By this way, it can be seen easily thet Malatya vertex-cover algorithm is optimum for regular graph.

In Fig. 3, the solution steps of the proposed algorithm on a sample graph are given. In Stage 1, the visual structure of the sample graph to be used for the analysis process was presented. Malatya centrality values should be calculated for selecting vertex-cover members of this diagram. Table 2 contains Malatya centrality values calculated for all stages. When Stage 2 was examined, the Malatya centrality value of the graph given in Stage 1 was calculated. When the centrality values in Table 2 are examined, it is seen that the '*e*' node has the value of 9.50 and the maximum Malatya centrality value for the 1st iteration. In Stage 2, the node *e* is selected as the first member of the vertex cover and deleted from the graph along with the edge connections. Then, the graph is updated, and Stage 3 is started for selecting the second vertex-cover member. According to the 2nd iteration centrality data in Table 2, the node with the highest Malatya centrality value of 6 is the '*f*' node. The '*f*' node is deleted from the graph along with its connections by selecting the vertex-cover

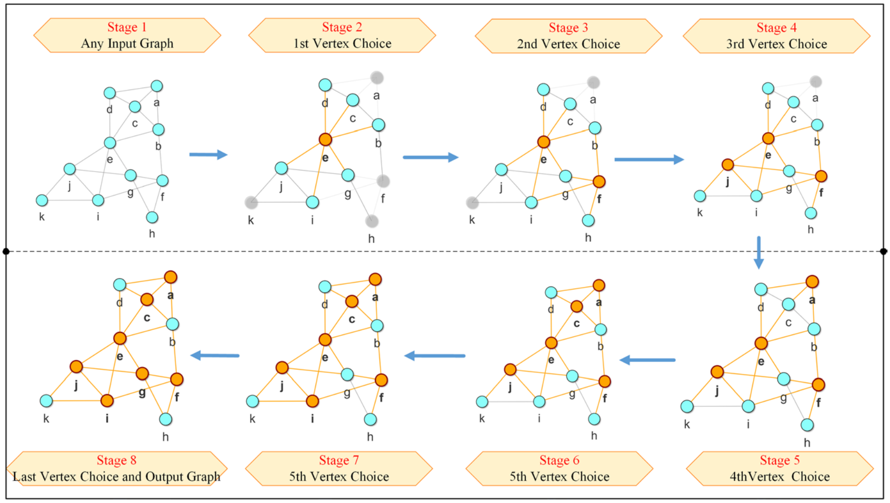


Fig. 3 Selection stages of vertex-cover members

Table 2 Malatya centrality values of the sample graph

Nodes\ Iterations	1	2	3	4	5	6	7	8
a	2.50	3.50	4.00	**4.00**				
b	4.00	2.75	1.33	1.33	0.50	0.00	0.00	0.00
c	4.33	3.50	4.00	4.00	**4.00**			
d	2.25	1.33	1.33	1.33	0.50	0.00	0.00	0.00
e	**9.50**							
f	5.00	**6.00**						
i	4.67	3.25	1.67	1.00	1.00	**1.00**		
g	4.67	3.25	2.67	1.00	1.00	1.00	**1.00**	
h	1.00	1.17	0.50	1.00	1.00	1.00	1.00	0.00
j	4.67	3.50	**4.50**					
k	1.00	1.33	1.67	1.00	1.00	1.00	0.00	0.00

node. Node ‘*f*’ is selected as the 2nd vertex-cover member, and the proposed algorithm continues to run until there are no edge remaining in the graph. Nodes selected after deletion of the last edge correlation represent the vertex-cover members of the graph in Stage 1, as highlighted in orange in Stage 8. The orange-coloured nodes in each stage in the images represent the vertex-cover nodes up to that iteration, and the orange edges represent the deleted edges until that iteration. All edges being orange means that all selected nodes cover all edges. Thus, nodes that make up the solution set for the sample graph structure are determined.

Table 2 shows Malatya centrality values in the current graph for each Stage. The values indicated in orange indicate the node with the maximum centrality value in the relevant iteration.

4 Analysis and evaluation of results

The concept of centrality on the graph is expressed in various ways [50]. The degrees of a node are widely used in calculating centrality values. However, the degree of the node itself is not sufficient to determine the centrality value. However, with the degree of the node, the degree of the adjacent nodes is one of the important parameters determining the centrality. Nodes with weak neighbours will be relatively stronger than their neighbours. In determining the Malatya centrality values produced by the Malatya algorithm, the degree of the adjacent nodes is also effective along with the node’s degree. For each node, the degree of the node is obtained by dividing the degree of its neighbour nodes. Therefore, for a node to be considered central, its degree must be higher than its adjacent node degrees. In addition, the degree of neighbour nodes is not taken exactly, but proportionally, it contributes to determining the centrality value in proportion to its rate. The nodes with more neighbourhoods are provided with a high centrality value. There are no methods for MVCP where the centrality values are decisive in the solution. In the proposed approach, the position and properties of node and edges in the graph structure are taken into account. The successful results obtained on the sample graphs show the effectiveness of the method.

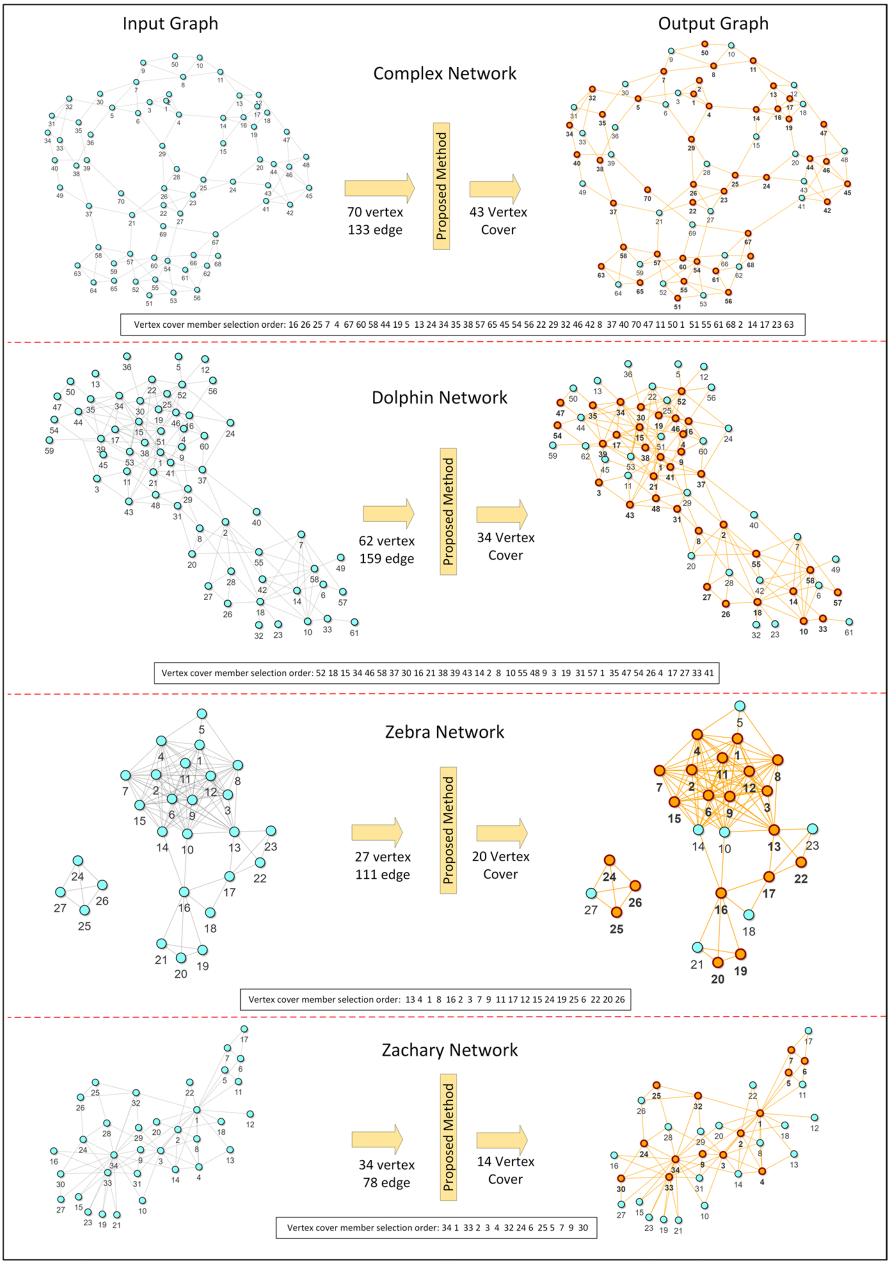
Approaches based on brute force perspective scan the entire possible space. In addition, the amount of computation and resources required for these approaches is costly. Therefore, it is costly in terms of space and time, especially in large and complex graphs. Moreover, in such networks, the problem cannot be solved within a reasonable time, and optimum solution sets cannot be determined. The proposed algorithm provides a robust solution for MVCP consisting of specific steps. Thus, there is no need to scan the entire space, and the computational cost is reduced. Moreover, for an NP-complete problem MVCP, effective solution sets are generated in a reasonable time. The proposed algorithm overcomes these problems, and effective solution sets are produced.

All possible solutions have been used with optimization methods for the MVCP solution since the space scanning approach did not yield results in a reasonable time, and an effective solution could not be produced. However, the optimization methods and heuristic approaches cannot produce a precise solution for MVCP, and it is assumed that there are solutions close to the optimum solution. The solutions produced depend on various parameters and constraints, such as the algorithm used, the number of iterations, and initial conditions. In addition, the proposed greedy-based optimization approaches to solve MVCP run the risk of staying in local solutions. The proposed algorithm provides a solution for MVCP consisting of a finite number of steps in polynomial time, independent of the graph structure and constraints. In addition, this approach produces results independent of various constraints, such as the number of iterations and initial conditions. Since the graph structure is considered a whole, generating local solutions is no problem.

Some of the solutions proposed for MVCP in the literature are exact approaches. These approaches deal with the related problem theoretically or offer solutions in some particular graphs. These approaches cannot provide an effective solution for MVCP due to the limitations mentioned. The proposed solution for the MVCP solution is a robust algorithm and offers an exact solution. Therefore, examples of graphs that can be applied to graphs in general and that produce effective solutions are given on graphs. In addition, it is shown on sample graphs in the next section that the proposed approach generally produces effective solutions for many different graph structures.

It also produces effective solution sets in graphs where the proposed algorithm's integer linear programming (ILP) solution does not yield any results. Although this approach is widely used in NP problems, it contains various limitations and problems. Especially in recursive-like approaches, resources such as memory and processor are insufficient due to the nature of the problem. The proposed MVCA also offers solutions for graphs that cannot yield effective results due to these limitations. Since MVCA is an iterative approach, these restrictions do not prevent generating a solution set for the graph.

There are real-life and computer problems modelled based on MVCP. Existing problems in areas such as transportation, social networks, wireless networks and security, etc. are based on the same model by nature. Therefore, developing an effective solution for MVCP allows for solving problems based on similar structures and



Vertex cover member selection order: 16 26 25 7 4 67 60 58 44 19 5 13 24 34 35 38 57 65 45 54 56 22 29 32 46 42 8 37 40 70 47 11 50 1 51 55 61 68 2 14 17 23 63

Vertex cover member selection order: 52 18 15 34 46 58 37 30 16 21 38 39 43 14 2 8 10 55 48 9 3 19 31 57 1 35 47 54 26 4 17 27 33 41

Vertex cover member selection order: 13 4 1 8 16 2 3 7 9 11 17 12 15 24 19 25 6 22 20 26

Vertex cover member selection order: 34 1 33 2 3 4 32 24 6 25 5 7 9 30

Fig. 4 Implementation of the proposed method in special graphs

models. The successful results obtained in the sample graphs show that the proposed algorithm is effectively applicable to these problems.

4.1 Experimental results of the proposed approach

The proposed algorithm efficiency is shown on sample graphs widely used in the literature. Complex [51], Dolphin [52], Zebra [53], and Zachary [54] networks of unweighted and undirected special graphs were used for analysis. The complex network is a network created by combining artificial graphs such as circle, star, line, etc. [51] While Dolphin, Zebra, and Zachary networks are special networks determined from real-world social relations [55]. In Fig. 4, the results obtained by applying the method proposed in this study to special graphs are given. When the results were examined, the number of nodes in vertex-cover set for the complex network consisting of 70 nodes and 133 edges was determined as 43. The detected vertex-cover nodes are highlighted in orange in the figure. In addition, the results of the selection order of the vertex-cover members were shared in the figure. Selection ranking values, in other words, show which node is selected as vertex-cover member in each iteration. When the other networks are examined, 34 node members were selected for Dolphin network with 62 nodes and 159 edges, and 20 node members for Zebra network with 111 edges, and 14 vertex-cover members for Zachary network with 78 edges. When the results are examined, it is seen in these graphs that the solution set for MVCP consists of a minimum number of elements. These show the proposed algorithm's effectiveness in these networks known in the literature.

The proposed vertex-cover method was tested on large graphs created with the Erdos–Renyi model [56] generator. Figure 5 shows the results of 15 test procedures performed according to different parameter values. The Erdos–Renyi model has two important parameters in which the number of nodes in the graph (n) and the probability of forming an edge between two nodes (p) are given [57]. When the test results in Fig. 5 are examined, a graph with 620 edges was produced for Erdos–Renyi model $n = 250$ and $p = 1/50$, and 141 node selection was made to reach all edges in this graph. In other words, the algorithm we presented selected 141 vertex-cover members for this random graph with 250 nodes. In another example, Erdos–Renyi model graph with 1,250,893 random edge correlation was generated for $n = 5000$ and $p = 1/10$ value. The vertex-cover value of the algorithm we proposed for this graph was determined as 4937. When the test data are considered, it was determined that the number of vertex-cover members increased depending on the graph densities. In addition, it is observed that increasing the number of nodes in the graph and the graph densities increased the number of vertex-cover members. In addition, the random graph created on the figure and the vertex-cover values determined by the algorithm were given.

5 Conclusion

Malatya vertex-cover algorithm, a robust approach to solving MVCP, has been developed in this study. This approach consists of two steps. First, the Malatya centrality value is calculated for each node using the Malatya centrality algorithm. Malatya centrality value is a centrality value calculated by associating the degree of each node with the degree of adjacent nodes. In the second step, the node of

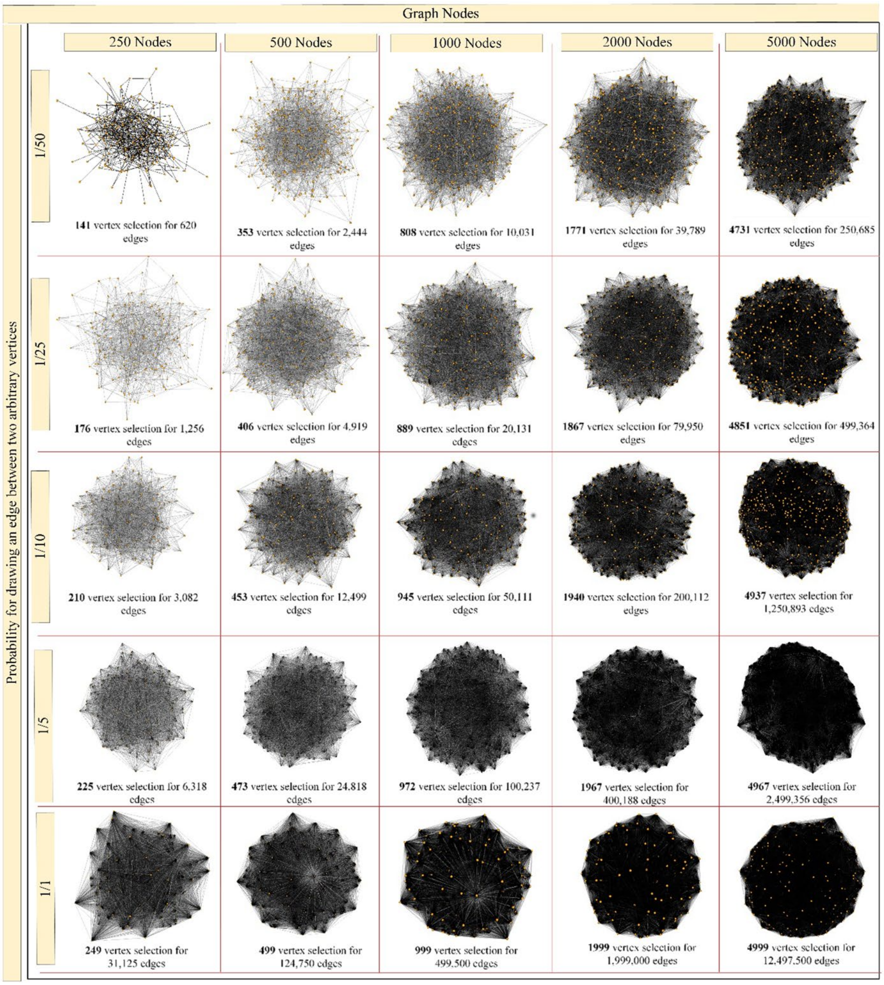

Graph Nodes
250 Nodes
500 Nodes
1000 Nodes
2000 Nodes
5000 Nodes
Probability for drawing an edge between two arbitrary vertices
1/50
1/25
1/10
1/5
1/1
141 vertex selection for 620 edges
353 vertex selection for 2,444 edges
808 vertex selection for 10,031 edges
1771 vertex selection for 39,789 edges
4731 vertex selection for 250,685 edges
176 vertex selection for 1,256 edges
406 vertex selection for 4,919 edges
889 vertex selection for 20,131 edges
1867 vertex selection for 79,950 edges
4851 vertex selection for 499,364 edges
210 vertex selection for 3,082 edges
453 vertex selection for 12,499 edges
945 vertex selection for 50,111 edges
1940 vertex selection for 200,112 edges
4937 vertex selection for 1,250,893 edges
225 vertex selection for 6,318 edges
473 vertex selection for 24,818 edges
972 vertex selection for 100,237 edges
1967 vertex selection for 400,188 edges
4967 vertex selection for 2,499,356 edges
249 vertex selection for 31,125 edges
499 vertex selection for 124,750 edges
999 vertex selection for 499.500 edges
1999 vertex selection for 1,999,000 edges
4999 vertex selection for 12,497.500 edges

Fig. 5 Erdos–Renyi model

maximum Malatya centrality is selected from the Malatya centrality values calculated for the vertex cover. Then, the related node and the edges associated with this node are removed from the graph, creating a new graph structure. By calculating Malatya centrality values again, appropriate node selections are made for the minimum vertex cover. By selecting all the edges, the robust solution set for the graph is determined.

The proposed MVCA has an approach based on Malatya centrality algorithm and provides a robust solution for MVCP. In addition, unlike the approaches in the literature, it offers a polynomial time solution in a reasonable time. This solution graph is generated independently of its characteristics or constraints, such as initial conditions, number of iterations. The solutions obtained with the proposed algorithm are robust. The solutions produced do not change unless the graph structure changes. The selection of an efficient solution set for an NP-complete optimization problem is shown through experimental examples. Experimental investigations on popular and accepted graphs in graph theory have demonstrated the efficiency of the strategy. Any graph can have its computational steps, time, and computational complexity calculated. Although time and cost for approximate approaches cannot be estimated, these calculations are made explicitly for the graph provided with the suggested method. Starting from the node where the Malatya centrality value is predominant, it is fast to determine the solution set. The method we offer also works on graphs where the ILP solution does not work, thanks to its low processing power and memory management. The ILP solution does not produce a solution for various reasons, such as high memory consumption in some graphs. Proposed algorithm is used to determine the nodes constituting the optimum or near-optimal solution set for MVCP. The algorithm was tested on many different networks, including artificial, real social networks, and randomly generated graphs. In particular, random graphs with different parameters and intensity values are advantageous to evaluate the results in sample graphs. When the test results were examined, it was observed that the proposed algorithm was robust, and the vertex-cover results it detected generally formed an effective set for MVCP. In addition, the parameters show the effectiveness of the proposed algorithm and its success in different graph types when the sample applications are considered.

Thanks to the developable structure of the proposed algorithm, it will contribute to many academic studies. In future, it is aimed to deal with real-world problems modelled based on MVCP with a similar approach and to offer solutions. Furthermore, it aims to identify and solve potential problems, especially in transportation and security.

Author contributions Selman Yakut took part in the writing of the article and literature studies, Furkan Öztemiz in the implementation of the method, and Ali Karcı in the development of the method. Three authors took an active part in all stages of the article.

Funding No funding was received to assist with the preparation of this manuscript.

Data availability The data used are available in the literature. Data are referenced in the article. There are no restrictions on its access.

Declarations

Conflict of interest The interest statement is not applicable to the manuscript. The authors have no affiliation with any organization with a direct or indirect financial interest in the subject matter discussed in the manuscript.

Ethical approval This manuscript has not been submitted to, nor is it under review at, another journal or other publishing venue.

References

1. Thulasiraman K, Swamy MN (2011) Graphs: theory and algorithms. John Wiley & Sons, Montreal
2. Hark C, Karcı A (2022) A new multi-document summarisation approach using saplings growing-up optimisation algorithms : simultaneously optimised coverage and diversity. J Inf Sci. https://doi.org/10.1177/01655515221101841
3. Thulasiraman K, Arumugam S, Brandstädt A, Nishizeki T (2016) Handbook of graph theory, combinatorial optimization, and algorithms. Chapman & Hall/CRC, Boca Raton
4. Khattab H, Mahafzah BA, Sharieh A (2022) A hybrid algorithm based on modified chemical reaction optimization and best-first search algorithm for solving minimum vertex cover problem. Neural Comput Appl 34(18):15513–15541. https://doi.org/10.1007/s00521-022-07262-w
5. Cormen TH, Leiserson CE, Rivest R, Clifford S (2001) Introduction to algorithms. Introducti, London
6. Angel D (2022) Protection of medical information systems against cyber attacks: a graph theoretical approach. Wirel Pers Commun 126(4):3455–3464. https://doi.org/10.1007/s11277-022-09873-x
7. Wang L, Du W, Zhang Z, Zhang X (2017) A PTAS for minimum weighted connected vertex cover P3 problem in 3-dimensional wireless sensor networks. J Comb Optim 33(1):106–122. https://doi.org/10.1007/s10878-015-9937-z
8. Dagdeviren ZA (2021) Weighted connected vertex cover based energy-efficient link monitoring for wireless sensor networks towards secure internet of things. IEEE Access 9:10107–10119. https://doi.org/10.1109/ACCESS.2021.3050930
9. Hossain A et al (2020) Automated design of thousands of nonrepetitive parts for engineering stable genetic systems. Nat Biotechnol 38(12):1466–1475. https://doi.org/10.1038/s41587-020-0584-2
10. Gusev VV (2020) The vertex cover game: application to transport networks. Omega 97:102102. https://doi.org/10.1016/j.omega.2019.08.009
11. Yigit Y, Dagdeviren O, Challenger M (2022) Self-stabilizing capacitated vertex cover algorithms for internet-of-things-enabled wireless sensor networks. Sensors 22(10):3774. https://doi.org/10.3390/s22103774
12. Yigit Y, Dagdeviren ZA, Dagdeviren O, Challenger M (2021) Performance evaluation of capacitated vertex cover algorithms for security applications in wireless sensor networks. In: 7th International Conference on Electrical, Electronics and Information Engineering: Technological Breakthrough for Greater New Life, ICEEIE 2021, no. C. https://doi.org/10.1109/ICEEIE52663.2021.9616719.
13. Yigit Y, Akram VK, Dagdeviren O (2021) Breadth-first search tree integrated vertex cover algorithms for link monitoring and routing in wireless sensor networks. Comput Netw 194:108144. https://doi.org/10.1016/j.comnet.2021.108144
14. Dagdeviren ZA (2022) A Metaheuristic algorithm for vertex cover based link monitoring and backbone formation in wireless Ad hoc Networks. Expert Syst Appl 213:118919. https://doi.org/10.1016/j.eswa.2022.118919
15. Mandal S, Patra N, Pal M (2021) Covering problem on fuzzy graphs and its application in disaster management system. Soft Comput 25(4):2545–2557. https://doi.org/10.1007/s00500-020-05263-2
16. Alamatsaz K, Aghadavoudi Jolfaei A, Iranpoor M (2020) Edge covering with continuous location along the network. Int J Ind Eng Comput 11(4):627–642. https://doi.org/10.5267/j.ijiec.2020.4.002
17. Dinur I, Safra S (2005) On the hardness of approximating vertex cover. Ann Math 162(1):439–485. https://doi.org/10.4007/annals.2005.162.439
18. Karci A, Yakut S, Öztemiz F (2022) A new approach based on centrality value in solving the minimum vertex cover problem: Malatya centrality algorithm. J Comput Sci 7:81–88
19. Zhang YJ et al (2022) Applying the quantum approximate optimization algorithm to the minimum vertex cover problem. Appl Soft Comput 118:108554. https://doi.org/10.1016/j.asoc.2022.108554
20. Derakhshan M, Durvasula N, Haghtalab N Stochastic minimum vertex cover in general graphs: a 3/2-approximation
21. Li X, Chen J, Yuan Q (2022) Adaptive vertex cover of dynamical networks with prospect theoretic perspective. IEEE Trans Netw Sci Eng 9(6):4159–4170. https://doi.org/10.1109/TNSE.2022.3196054
22. Khattab H, Sharieh A, Mahafzah BA (2019) Most valuable player algorithm for solving minimum vertex cover problem. Int J Adv Comput Sci Appl 10(8):159–168

23. Guo P, Quan C, Chen H (2019) MEAMVC : a membrane evolutionary algorithm for solving minimum vertex cover problem. IEEE Access 7:60774–60784. https://doi.org/10.1109/ACCESS.2019.2915550
24. Xie X, Qin X, Yu C, Xu X (2018) Test-cost-sensitive rough set based approach for minimum weight vertex cover problem. Appl Soft Comput J 64:423–435. https://doi.org/10.1016/j.asoc.2017.12.023
25. Nagy B, Szokol P (2021) A genetic algorithm for the minimum vertex cover problem with interval-valued fitness. Acta Polytech Hungarica 18(4):105–123
26. Qiu C, Wang Y, Fan M, Yin M (2019) An improved memetic algorithm for the partial vertex cover problem. IEEE Access 7:17389–17402. https://doi.org/10.1109/ACCESS.2019.2895738
27. Hu S, Li R, Zhao P, Yin M (2018) A hybrid metaheuristic algorithm for generalized vertex cover problem. Memet Comput 10(2):165–176. https://doi.org/10.1007/s12293-016-0216-z
28. Li Y, Chai Z, Ma H, Zhu S (2023) An evolutionary game algorithm for minimum weighted vertex cover problem. Soft Comput. https://doi.org/10.1007/s00500-023-07982-8
29. Islam MR, Arif IH, Shuvo RH (2019) Generalized vertex cover using chemical reaction optimization. Appl Intell 49(7):2546–2566. https://doi.org/10.1007/s10489-018-1391-z
30. Li R, Hu S, Wang Y, Yin M (2017) A local search algorithm with tabu strategy and perturbation mechanism for generalized vertex cover problem. Neural Comput Appl 28(7):1775–1785. https://doi.org/10.1007/s00521-015-2172-9
31. Jovanovic R, Sanfilippo AP, Voß S (2022) Fixed set search applied to the multi-objective minimum weighted vertex cover problem. J Heuristics 28(4):481–508. https://doi.org/10.1007/s10732-022-09499-z
32. Li R, Hu S, Cai S, Gao J, Wang Y, Yin M (2020) NuMWVC : a novel local search for minimum weighted vertex cover problem. J Oper Res Soc 71(9):1498–1509. https://doi.org/10.1080/01605682.2019.1621218
33. Quan C, Guo P (2021) A local search method based on edge age strategy for minimum vertex cover problem in massive graphs. Expert Syst Appl 182:115185. https://doi.org/10.1016/J.ESWA.2021.115185
34. Cai S, Li Y, Hou W, Wang H (2019) Towards faster local search for minimum weight vertex cover on massive graphs. Inf Sci (Ny) 471:64–79. https://doi.org/10.1016/j.ins.2018.08.052
35. Li R, Hu S, Zhang H, Yin M (2016) An efficient local search framework for the minimum weighted vertex cover problem. Inf Sci (Ny) 372:428–445. https://doi.org/10.1016/j.ins.2016.08.053
36. Tai R, Ouyang D, Li R, Zhang L (2023) ILSGVCP: an improved local search algorithm for generalized vertex cover problem. J Oper Res Soc. https://doi.org/10.1080/01605682.2022.2147458
37. Cai S, Hou W, Lin J, Li Y (2018) Improving local search for minimum weight vertex cover by dynamic strategies. In Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence. pp. 1412–1418. https://doi.org/10.24963/ijcai.2018/196.
38. Luo C, Hoos HH, Cai S, Lin Q, Zhang H, Zhang D (2019) Local search with efficient automatic configuration for minimum vertex cover. In: Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence. 2019: 1297–1304. https://doi.org/10.24963/ijcai.2019/180.
39. Akram VK, Ugurlu O (2021) A localized distributed algorithm for vertex cover problem. J Comput Sci 58:101518. https://doi.org/10.1016/j.jocs.2021.101518
40. Sun C et al (2022) Better approximation for distributed weighted vertex cover via game-theoretic learning. IEEE Trans Syst Man, Cybern Syst 52(8):5308–5319. https://doi.org/10.1109/TSMC.2021.3121695
41. Manan A, Bashir S, Majid A (2022) Vertex cover optimization using a novel graph decomposition approach. Comput Mater Contin 73(1):701–717. https://doi.org/10.32604/cmc.2022.027064
42. Fluschnik T, Niedermeier R, Rohm V, Zschoche P (2022) Multistage vertex cover. Theor Comput Syst 66(2):454–483. https://doi.org/10.1007/s00224-022-10069-w
43. Chen J, Li X (2023) Toward the minimum vertex cover of complex networks using distributed potential games. Sci China Inf Sci 66(1):1–20. https://doi.org/10.1007/s11432-021-3291-3
44. Gu J, Guo P (2021) PEAVC: an improved minimum vertex cover solver for massive sparse graphs. Eng Appl Artif Intell 104:104344. https://doi.org/10.1016/j.engappai.2021.104344
45. Hu S, Wu X, Liu H, Wang Y, Li R, Yin M (2019) Multi-objective neighborhood search algorithm based on decomposition for multi-objective minimum weighted vertex cover problem. Sustainability 11(13):3634. https://doi.org/10.3390/su11133634
46. Bouamama S, Blum C, Boukerram A (2012) A population-based iterated greedy algorithm for the minimum weight vertex cover problem. Appl Soft Comput J 12(6):1632–1639. https://doi.org/10.1016/j.asoc.2012.02.013

47. Wang L, Hu S, Li M, Zhou J (2019) An exact algorithm for minimum vertex cover problem. Mathematics 7(7):603. https://doi.org/10.3390/math7070603
48. Borgatti SP (2005) Centrality and network flow. Soc Networks 27(1):55–71. https://doi.org/10.1016/j.socnet.2004.11.008
49. Kumar G, Duhan N, Sharma AK (2011) Page ranking based on number of visits of links of Web page. In 2011 2nd International Conference on Computer and Communication Technology (ICCCT-2011). pp. 11–14. https://doi.org/10.1109/ICCCT.2011.6075206
50. Öztemiz F, Karcı A (2021) Malatya İli ulaşım ağı kavşak noktalarının merkezlilik analizi. Gazi Üniversitesi Mühendislik-Mimarlık Fakültesi Derg 37(1):511–528. https://doi.org/10.17341/gazimmfd.834255
51. Öztemiz F (2021) Karmaşık ağlarda hakim düğümlerin belirlenmesi için yeni bir yöntem. İnönü University, Malatya
52. Dickinson B, Hu W (2015) The effects of centrality ordering in label propagation for community detection. Soc Netw 04(04):103–111. https://doi.org/10.4236/sn.2015.44012
53. Rajmonda S, Berger-Wolf T, Grossman R (2010) Meaningful selection of temporal resolution for dynamic networks. In: Proceedings of the Eighth Workshop on Mining and Learning with Graphs (MLG'10). pp. 127–136. [Online]. Available: https://doi.org/10.1145/1830252.1830269
54. Chintalapudi SR, Prasad MHMK (2015) A survey on community detection algorithms in large scale real world networks. In: 2015 2nd International Conference on Computing for Sustainable Global Development (INDIACom). pp. 1323–1327. [Online]. Available: https://ieeexplore.ieee.org/document/7100465?arnumber=7100465
55. Networks http://konect.cc/networks/, Accessed time: 30 Apr 2023
56. R igraph manual pages. https://igraph.org/r/doc/erdos.renyi.game.html, Accessed time: 30 Apr 2023
57. Weber K (1985) On the evolution of random graphs in the n-cube. Publ Math Inst Hung Acad Sci 5(1):17–60

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Springer Nature or its licensor (e.g. a society or other partner) holds exclusive rights to this article under a publishing agreement with the author(s) or other rightsholder(s); author self-archiving of the accepted manuscript version of this article is solely governed by the terms of such publishing agreement and applicable law.

Authors and Affiliations

Selman Yakut[1] · Furkan Öztemiz[1] · Ali Karci[1]

✉ Selman Yakut
selman.yakut@inonu.edu.tr

Furkan Öztemiz
furkan.oztemiz@inonu.edu.tr

Ali Karci
ali.karci@inonu.edu.tr

[1] Software Engineering Department, Inonu University, Malatya, Turkey