Journal of Engineering and Applied Sciences

MEDWELL PUBLICATIONS SINCE 2000

A Novel Methodology to Measure the Approximation of Near Optimal Algorithms for Minimum Vertex Cover Problem

Zahid Ullah and Su-Hyun Lee
Department of Computer Engineering, Changwon National University, Changwon, South Korea

Key words: Graph instances, minimum vertex cover, optimization problem, NP-complete problem, Maximum Independent Set (MIS), approximation algorithm

Abstract: Many algorithms have been proposed for the solution of the Minimum Vertex Cover (MVC) problem but the researchers are unable to find the optimality of an approximation algorithm. In this study, we have proposed a method to evaluate that either the result returned by an approximation algorithm for the minimum vertex cover problem is optimal or not. The proposed method is tested on three algorithms, i.e., Maximum Degree Greedy (MDG) algorithm, Modified Vertex Support Algorithm (MVSA) and Clever Steady Strategy Algorithm (CSSA). The proposed method provides an opportunity to test the optimality of an approximation algorithm for MVC problem with low computation complexity. The proposed method has performed well during experimentation and its results brighten the path of successful implementation of the method for the evaluation of approximation algorithms for the Minimum Vertex Cover (MVC) problem. The testing of the proposed method was carried out on small graph instances. The proposed method has resolved the problem to test the optimality of the approximation algorithm for the minimum vertex cover problem. This technique has digitized the process of finding out the accuracy of the optimal solution returned by approximation algorithms for MVC.

Corresponding Author:
Zahid Ullah
Department of Computer Engineering, Changwon National University, Changwon, South Korea

Page No.: 3391-3398
Volume: 15, Issue 19, 2020
ISSN: 1816-949x
Journal of Engineering and Applied Sciences
Copy Right: Medwell Publications

INTRODUCTION

The minimum vertex cover is a very popular NP-complete problem like other famous NP-hard problems including Maximum Independent Set (MIS) and Maximum Clique (MC) and so forth. The Maximum Independent Set (MIS) contains those vertices in G = (V, E) that are not in MVC such as $\text{MIS} = \text{G}(\text{V}, \text{E})\text{-MVC}\left(\hat{\text{G}} = \left(\text{V}, \hat{\text{E}}\right)\right)$ and vice versa. Similarly, the maximum clique contains those vertices in G = (V, E) that are in MVC of complement graph such as MC = G = (V, E)-MVC where a complement graph is an undirected graph $\hat{\text{E}} = \left\{(\text{vi}, \text{vj}) \mid \text{vi}, \text{vj} \in \text{V}, \text{i} \neq \text{j}, (\text{vi}, \text{vj}) \notin \text{E}\right\}$, hence, solving the MVC will definitely solve all NP-complete problems[1].

A polynomial-time algorithm that has the capability to solve the MVC problem has not been devolved to date. It is also not possible for an algorithm to prove a super polynomial-time bound for MVC. We can solve the MVC problem in two ways, either by using the approximation algorithms or exact algorithms. The approximation algorithm can solve the MVC problem in polynomial time but no guarantee of optimality. The exact algorithms provide the exact solution of the problem but as time goes on they require increases exponentially with the size of the problem. Hence, the approximation algorithms are the better choice where the size of the problem is large and the solution requires does not to be optimal. The exact algorithm is the best choice where the size of the problem is small with no time constraint. However, the issue arises where the size of the problem is large and the optimal solution is required in a short time. Hence, we need a technique to measure the optimality of approximation algorithms[2].

The Minimum Vertex Cover (MVC) is normally used in different real-world applications. The approximation algorithms have been successfully used for the solution of the MVC problem. The approximation algorithms provide a near-optimal solution within a reasonable time[3]. The placement of guards in museums is the real-time application of the MVC. Furthermore, the MVC problem can also be used for the construction of hospitals, schools and so forth, so that, they can cover the maximum area and the maximum number of people can easily approach the hospitals and schools[2]. Due to these applications, the MVC problem has grasped the attention of many researchers in the last decades and a lot of methods have been proposed to solve this problem.

Cook in 1971 has divided all problems into classes named polynomial-time problems (P-Problems), non-deterministic polynomial-time problems (NP-Problems), NP-hard problems and NP-complete problems[1]. A problem that can be sorted out in a specific time is called polynomial-time problems or P-Problems. Examples of p-problems are the complement of a graph, subtraction of a set of vertices from a graph. As opposed to polynomial time, the NP problem cannot be solved in a specific but rather require exponential time to be solved. By using the current computation resources, it is impossible to solve a large NP problem at a specific time. There are a lot of NP-problems such as traveling salesman problem, graph coloring problem and MVC problem and so forth. According to Cook, the NP-complete problem retains two properties which are the problem should be NP and reducible in polynomial time to the NP-complete problem[1]. The fundamental problem of the NP-complete set is Boolean Satisfactory (SAT). The other problems that can be converted in SAT Problem in polynomial time are clique problem, MIS and so forth. MVC can also be converted to MIS in polynomial time, hence included in NP-complete problem set [4].

A vertex cover (V, C) in an undirected graph G = (V, E) is a subset of vertices which covers all the edges in a graph but as in MVC that VC should be minimum[2]. Figure 1 the filled vertices represent the vertices in vertex cover and the empty vertices represent that the vertices do not belong to vertex cover in a graph.

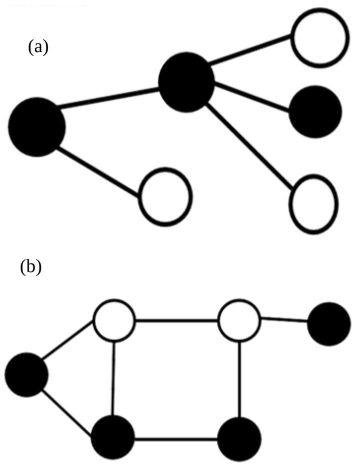

Fig. 1(a, b): (a) Graph with three vertices in vertex cover and (b) Graph with four vertex cover

The least vertex cover in a graph is recognized as the MVC. The vertex cover will be represented by λ that denotes MVC size, i.e., $\lambda = |C|$[3]. The MVC solutions have two main categories, namely approximation algorithms and exact algorithms. Approximate algorithms solve the problem in a reasonable time with guaranteed optimality. Approximation algorithms provide a near-optimal solution in a reasonable time. Approximation algorithms are useful when the size of the problem is large and the solution is required at a reasonable time. The exact algorithms provide a guaranteed optimal solution but time-explosion occurs as the dimension of the problem grows. For solving a moderate size problem, it can take billion or trillion years with currently available computation power[5, 6]. The MVC problem can be efficiently tackled by using approximation algorithms while for the clique problem; complete algorithms are the best choice. The approximation algorithms can be used to solve a problem where the near-optimal solution is required within no time. The problems in which optimality has a critical concern and cannot be compromised even in a long time, to solve these problems complete algorithms are the best choice.

It is difficult to design a better complete algorithm for the MVC problem due to its complex nature. Most of the developed algorithms for the MVC problem are approximation algorithms. In applications where optimality is required within no time we cannot apply approximation algorithms because of optimality constraint and also the exact algorithm cannot be deployed due to its low computational complexity.

The problem of MVC is an earlier NP problem which was proven NP-complete[7]. The algorithms have limitations so the NP-complete problems cannot be solved with polynomial-time algorithms. Also, the algorithms do not have the capability to prove a super polynomial-time bound for any of these problems.

The MVC problem can be divided into weighted and un-weighted[8]. In addition, the MVC problem can be divided into two versions, i.e., optimization and decision. In the optimization version, the algorithm has to find the best solution from the available feasible solutions. In the decision version, the algorithm has to find the existence of the desired size of 'k' where k is the MVC[9]. This study deals with the un-weighted MVC problem. The important preliminaries are presented in the subsequent section for a better understanding of the algorithms.

Preliminaries:

- We are concerned with an undirected graph G = (V, E) throughout this study where V represents the number of vertices and E indicates the number of edges
- For a vertex v∈V, let deg (v) is the set of edge occurrence to it, the minimum degree is represented by δ in G = (V, E)
- A graph is indicate complemented graphs such that Ĝ = (V, Ê) where Ê = {(νi, νj∈V, i≠j, (νi, νj)∉E}

Where Ap is an approximate algorithm for the MVC, symbols←, Δ and δ are used for assignment, maximum degree and the minimum degree, respectively. In this study, we have proposed a method named as Optimality Measurement of Approximation Algorithms (OMAA). The purpose of the problem algorithm is to measure the optimality of approximation algorithms for the minimum vertex cover problem. The time requires to solve a problem by using the complete algorithm increase exponentially as the size of the problem increase. Hence, we need a solution to find the optimal solution in a reasonable time. Hence, in the proposed approach we have proposed a technique to check the optimality of an approximation algorithm. Hence, the proposed approach is a better choice in a situation where optimal results are required for large graph instances in a reasonable time. This method provides the opportunity to check whether the result returns by an approximation algorithm for the minimum vertex cover problem is optimal or not.

Literature review: In this study, the literature review of some well-known state of the art algorithm has been carried out for the MVC problem; all the selected algorithms are approximate algorithms.

The simplest algorithm that was initially developed for MVC is MDG[10]. The MDG is the alternative form of set cover algorithm suggested by Chavatal *et al*.[11]. In this approach initially, each node degree is calculated in the graph and that node is selected as the MVC node having the highest degree.

The Maximum Degree Greedy (MDG) approach calculates the degree continuously until a maximum degree node has been found out. The network bench model is used for the calculation of the degree in MDG which takes O (E) steps. The technique is so simple and works on the simple steps but the best solution is not guaranteed in greedy approaches, the same is the issue with MDG. When MDG selects a node for the calculation of maximum degree for vertex cover, there is no surety that the selected vertex is part of maximum vertex cover or not. There is a possibility of selection of extra vertexes that can cause poor results. The computation complexity is reasonable, therefore preferred over other approximation algorithms.

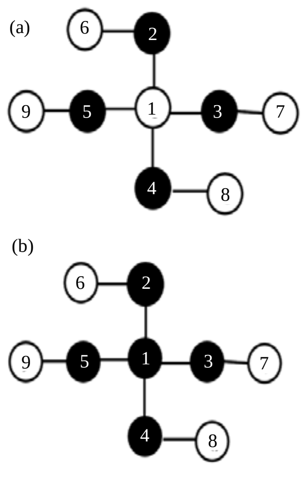

Fig. 2(a, b): (a) Optimal vertex cover and (b) Vertex cover returned by MDG

The Maximum Degree Greedy (MDG) algorithm fails to provide the optimal result on some benchmark graphs as shown in Fig. 2. The shape • represents a vertex in vertex cover and vertex not in the vertex cover is represented by the shape □.

The worst runtime complexity of the Maximum Degree Greedy (MDG) is O (E2) which indicates the computation efficiency of the MDG algorithm. Another simple heuristic algorithm is the Modified Vertex Support Algorithm (MVSA)[12] which is achieved by modifying the vertex support algorithm[13]. The same data structure has been introduced in MVSA as in VSA. First, the calculation of each node degree is carried out then the least degree node is chosen and the resultant node is considered as MVC node having minimum support values in the neighbor is selected as MVC node.

The MVSA is computationally efficient; the decisions are made straightforward. The runtime complexity of the Modified Vertex Support Algorithm (MVSA) is O (EV2log v). The MVSA also fails to deal with small instances as in Fig. 3.

The Clever Steady Strategy Algorithm (CSSA) is a simple approximate algorithm for solving the MVC

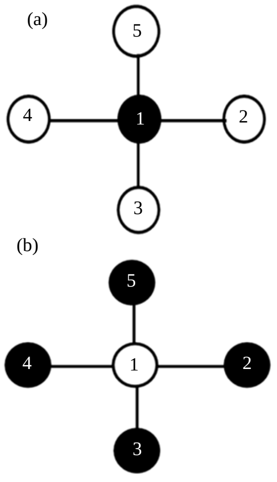

Fig. 3(a, b): (a) Optimal vertex cover and (b) Vertex cover returned by MVSA

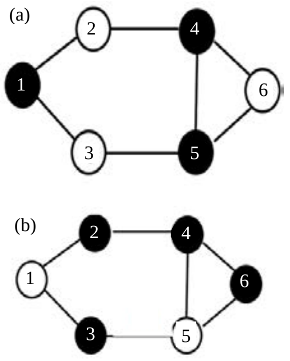

Fig. 4(a, b): (a) Optimal vertex cover and (b) Vertex cover returned by CSSA

problem[4]. The algorithm first locates the minimum degree vertex and then its entire neighbor vertices are searched out, the minimum degree vertex in all neighbors of the vertex having the least degree will be selected for adding to the vertex cover.

The Clever Steady Strategy Algorithm (CSSA) does not provide an optimal solution on some benchmark instances as shown in Fig. 4. The other most prominent near-optimal algorithms proposed for the MVC problem, are; Vertex Support Algorithm (VSA)[13], Advanced Vertex Support Algorithm (AVSA)[14], Degree Contribution Algorithm (DCA)[15], Max Degree Around (MDA) algorithm[2], Mean of Neighbors of Minimum degree Algorithm (MNMA)[16] and the Maximum Adjacent Minimum degree Algorithm (MAMA)[17].

MATERIALS AND METHODS

Proposed method: An algorithm time that has the potential to provide the optimal solution of the MVC problem has not been devolved to date. It is also not possible for an algorithm to prove a super polynomial-time bound for MVC. We can solve the MVC problem in two ways, either by using the approximation algorithms or exact algorithms. The approximation algorithm can solve the MVC problem in polynomial time but there is no guarantee of optimality. The exact algorithms provide the exact solution of the problem. Hence, the approximation algorithms are the better choice where the size of the problem is large and the require solution is not optimal. The exact algorithm is the best choice where the size of the problem is small with no time constraint. However, the issue arises where the size of the problem is large and the optimal solution is required in a short time. Hence, we need a technique to measure the optimality of approximation algorithms in this section, a novel technique has been suggested for optimality of an approximation algorithm and if the approximation for a certain application is optimal when there is no need to deploy multiple approximate algorithms for it (Fig. 5).

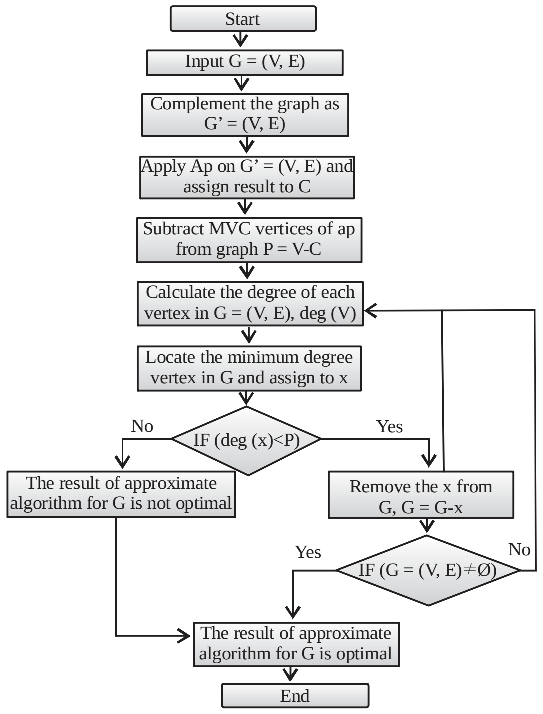
Start
Input G = (V, E)
Complement the graph as G' = (V, E)
Apply Ap on G' = (V, E) and assign result to C
Subtract MVC vertices of ap from graph P = V-C
Calculate the degree of each vertex in G = (V, E), deg (V)
Locate the minimum degree vertex in G and assign to x
No
IF (deg (x)<P)
Yes
The result of approximate algorithm for G is not optimal
Remove the x from G, G = G-x
Yes
IF (G = (V, E)≠Ø)
No
The result of approximate algorithm for G is optimal
End

Fig. 5: Flow diagram of the proposed algorithm

The proposed approach will also evaluate the optimality of approximate algorithms for the MVC problem. In the proposed approach, first the complement of the graph will be carried out then approximate algorithm for MVC will be deployed on it and the output result will be saved in C. Afterward the nodes in the VC will be subtracted from the vertices in G = (V, E) and the result will be saved in P.

Furthermore, the minimum degree vertex in G = (V, E) will be calculated, if the degree of minimum degree vertex becomes less than P, then it will be removed from G = (V, E) and the graph will be updated. The process continues until the graph becomes empty or the degree of minimum degree vertex becomes equal or greater than P. If the graph becomes empty, it means that the solution of the deployed approximation algorithm is optimal. In the second case if after deleting the vertex when a situation arises where the degree of a minimum degree vertex becomes greater or equal to P, then it means that the solution provided by the approximation is not optimal. The detailed flow diagram of the suggested technique is provided in Fig. 5.

Algorithm 1; Pseudocode for Optimality Measurement of Approximation (OMA) method:

```
Input: G = (V, E)
Output: Optimality
Begin:
    1. C←Ap (G'= (V, E'))
           // apply approximate algorithm on the completed graph
    2. P←V-C
           // subtract the minimum vertex cover from G
    3. FOR i←1 to V { i. deg(vi) }
           // compute each node degree
    4. x←δ (G = (V, E))
           // compute the minimum degree node in G = (V, E)
    5. IF (deg(x)< P) {i. G = (V, E)←G = (V, E)-x}
           //Remove the least degree vertex from G = (V, E), G = G-x
    6. IF((G = (V,E)) = Ø) {result of Ap is optimal go to End }
           // If the graph become empty
    7. IF (deg(x) =>P) { result of AP is not optimal go to End}
           // If the degree of the minimum degree vertex becomes greater or equal to P
    8. Else Go to step 3
End
```

RESULTS AND DISCUSSION

In this study, we have applied some well-known approximation algorithms on the small benchmark graphs to elaborate on the working mechanism of the proposed approach better. There are numerous approximation algorithms for solving the MVC problem. We have selected the Maximum Degree Greedy (MDG) algorithm, Modified Vertex Cover Algorithm (MVSA) and Clever Steady Strategy Algorithm (CSSA), the detailed description of these methods is provided in the literature review section. The selected methods are approximate having different working mechanisms.

The proposed method starts with complementing the graph given in Fig. 6a. The graph complementation is the essential step of the proposed method before applying any processing. After the graph complementation, we applied the Maximum Degree Greedy (MDG) approximation algorithm given in Fig. 6b. The vertex cover by any algorithm depends upon the strategy of the selection of the first vertex of the graph. The Maximum Degree Greedy (MDG) algorithm covers the complemented graph in 3 vertices (if the vertex 1 is selected first) as shown in Fig. 6c, hence, according to the proposed approach it will be saved in C (C←3), then P will be calculated, i.e., P←5-3 = 2. In step 3, each node degree of the graph in Fig. 6a will be calculated such as 1(2), 2(2), 3(3), 4(2) and 5(3). Afterward, the minimum degree vertex will be located in the given graph. In case of the vertices 1, 2 and 4 having the same minimum degree, one of them will be selected, (we selected the vertex 1) and the degree of the same will be compared to P. The p-value is equal to the minimum degree that proves the solution provided by Maximum Degree Greedy (MDG) algorithm is not optimal.

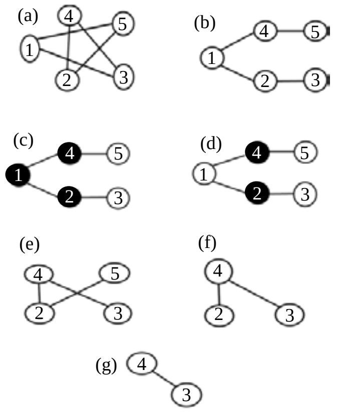

Fig. 6(a-g): (a) Given Graph G = (V, E), (b) Complemented Graph G = (V, E'), (c) Vertex Cover returned by MDG, (d) Vertex Cover returned by MVSA and CSSA, (e) Updated Graph Given by MVSA and CSSA, (f) Reduced Graph Given by MVSA and CSSA and (g) Final Graph Given by MVSA and CSSA

Similarly, the proposed technique was applied to the MVSA on the complemented graph Fig. 6b. The MVSA covers the graph in 2 vertices as shown in Fig. 6d, hence, C←3. The vertices 1, 2 and 4 have the same minimum degree vertices (the degree is 2), so, we have selected vertex 1 randomly. Whereas, the comparison of p-value and the minimum degree value take place where p-value is greater than the minimum degree value. Hence, the vertex 1 will be removed from the graph and the graph will be updated as shown in Fig. 6e. Again the minimum

Table 1: Optimality results of approximation algorithm on large graph instances

Benchmarks	V	I*	CSSA (I, ρ)	MDG (I, ρ)	MVSA (I, ρ)
graph50_6	50	38	(38,1)	(38,1)	(38,1)
graph50_10	50	35	(35,1)	(35,1)	(35,1)
graph100_1	100	60	(60,1)	(60,1)	(60,1)
graph100_10	100	70	(70,1)	(70,1)	(70,1)
graph200_5	200	150	(150,1)	(150,1)	(150,1)
graph500_1	500	350	(350,1)	(350,1)	(350,1)
graph500_2	500	400	(400,1)	(400,1)	(400,1)
graph500_5	500	290	(290,1)	(290,1)	(290,1)
phat300-1	300	292	(293,1.003)	(293,1.003)	(294,1.006)
phat300-2	300	275	(275,1)	(278, 1.010)	(279,1.014)
phat300-3	300	264	(266,1.001)	(269,1.010)	(272,1.030)
phat700-1	700	689	(692,1.004)	(693,1.004)	(692,1.004)
phat700-2	700	656	(658,1.003)	(660,1.003)	(660,1.006)
phat700-3	700	638	(640,1.003)	(642,1.003)	(649,1.017)
johnson8-2-4	28	24	(24,1)	(24,1)	(24,1)
johnson8-4-4	70	56	(56,1)	(62,1)	(56,1)
johnson16-2-4	120	112	(112,1)	(112,1)	(112,1)
johnson32-2-4	496	480	(480,1)	(480,1)	(480,1)
sanr200_0.7	200	182	(183,1.005)	(184,1.005)	(186,1..021)
sanr200_0.9	200	158	(164,1.03)	(164,1.005)	(163,1.031)
sanr400_0.5	400	387	(388,1.002)	(392,1.0026)	(389,1.005)
sanr400_0.7	400	379	(382,11.007)	(384,1.007)	(381,1.005)
fbr35-17-2	595	560	(565,1.008)	(570,1.014)	(565,1.008)
fbr_30_15_5	450	420	(426,1.01)	(429,1.009)	(424,1.009)
c125	125	91	(94,1.03)	(93,1.003)	(95,1.043)
C250.9	250	206	(211,1.02)	(211,1.0145)	(211,1.0145)
brock200_1	200	188	(190,1.01)	(190,1.004)	(191,1.015)
brock200_4	200	183	(185,1.01)	(192,1.01)	(193,1.054)
gen200_p0.9_4	200	156	(164,1.05)	(165,1.05)	(166,1,064)
hamming6-2	64	32	(32,1)	(32,1)	(32,1)
hamming6-4	64	60	(60,1)	(60,1)	(60,1)
hamming8-2	256	128	(128,1)	(128,1)	(128,1)
hamming8-4	256	240	(240,1)	(240,1)	(240,1)
hamming10-2	1024	512	(512,1)	(512,1)	(512,1)
dsjc-500	500	487	(489,1.004)	(491,1.004)	(489,1)
killer4	171	160	(160,1)	(164,1.025)	(160,1)
killer5	776	749	(754,1.006)	(764,1.02)	(754,1)
c-fat200-1	200	188	(188,1)	(188,1)	(188,1)
c-fat200-2	200	176	(176,1)	(176,1)	(176,1)
c-fat200-5	200	142	(142,1)	(142,1)	(142,1)
c-fat500-1	500	486	(486,1)	(486,1)	(486,1)
c-fat500-2	500	474	(474,1)	(474,1)	(474,1)
c-fat500-5	500	436	(436,1)	(436,1)	(436,1)
c-fat500-10	500	374	(374,1)	(374,1)	(374,1)
MANN_a27.clq.b	378	252	(253,1.003)	(261,1.04)	(253,1)

degree vertex degree will be equal to 2 which is less than p-value, (all have the same degree, we selected vertex 2 randomly), hence, it will be removed from Fig. 6e and again the graph will be updated as shown in Fig. 6f. Now, the minimum degree vertices were 2 and 3 having the degree 1, we selected 1, compared with p-value and removed from Fig. 6f and the graph will be updated as shown in Fig. 6g. Next the minimum degree was equal to 1 of both remaining vertices (4 and 3) of Fig. 6g, we selected vertex 4, its degree was <P, so, removed from Fig. 6g, after this vertex deletion the graph became empty which indicates that the solution returned by Modified Vertex Support Algorithm (MVSA) was optimal. Hence, no need to apply any further approximation algorithm.

If we apply the CSSA algorithm on the graph given in Fig. 6a, it will return the same result as provided by MVSA. The MVSA and Clever Steady Strategy Algorithm (CSSA) both are optimal on Fig. 6a. Eventually, the CSSA and MVSA algorithms are the best choices for the stated Fig. 6a.

We have also applied the CSSA, MVSA and MDG algorithms on large graph instances for proper evaluation of the proposed method. The benchmark instances are taken from two public libraries namely DIMACS and BHOSLIB. Approximation ration formula is given in Eq. 1 in order to check the performance of approximation algorithms (Table 1):

$$\text{pi} = \frac{I_k}{I_k^*} \geq 1 \tag{1}$$

Here pi represents the approximation ratio, Ai represents the approximate solution and OPTi represent the optimal solution of a problem. After putting values in I_k and I_k^* in equation 1, if pi = 1 it indicates that the solution is optimal. More deviation from 1 specify that the solution is poor[4].

We have applied the algorithms on graphs 2-4. The Maximum Degree Greedy (MDG) fails to provide an optimal solution on the graph given in Fig. 2. The MVSA fails to provide the best solution on the graph given in Fig. 3 and Clever Steady Strategy Algorithm (CSSA) fails to provide an optimal solution on the graphs given in Fig. 4. On the graph given in Fig. 2 the Maximum Degree Greedy (MDG) fails to provide an optimal result but on the same graph, Modified Vertex Support Algorithm (MVSA) and Clever Steady Strategy Algorithm (CSSA) return optimal solution. On the graph given in Fig. 3, Modified Vertex Support Algorithm (MVSA) fails but Clever Steady Strategy Algorithm (CSSA) and Maximum Degree Greedy (MDG) provide optimal solutions. Similarly, on the graph in Fig. 4 the Clever Steady Strategy Algorithm (CSSA) and Modified Vertex Support Algorithm (MVSA) both fail but Maximum Degree Greedy (MDG) returns optimal solution.

This strategy was needed to test that either the given approximation algorithm provides an optimal solution for a graph or not. These results reveal the importance of the suggested evaluation method of the approximate algorithms for the MVC problem. In order to support the effectiveness of our proposed method, we have also tested the proposed approach on large graph instances of some standard libraries. The results indicate that the proposed method is equally effective on small graph instances as well as on large graph instances.

CONCLUSION

The proposed method evaluates the optimality of approximation algorithms for the MVC problem. The basic determination of developing this technique was to save time and test the application compatibility of an approximation algorithm of MVC for a particular problem. The detailed description of some well-known algorithms that were used for the evaluation of the proposed method for MVC has been provided. The proposed method was tested on small benchmark graphs and proved a successful technique. We can apply the proposed technique in a situation where the exact solution is required in a reasonable time. Hence, we can escape from applying the complete algorithms and we can apply the approximation algorithms.

REFERENCES

01. Karp, R.M., 1972. Reducibility among Combinatorial Problems. In: Complexity of Computer Computations, Miller, R.E. and J.W. Thatcher (Eds.). Plenum Press, New York, pp: 85-133.
02. Fayaz, M., S. Arshad, A.S. Shah and A. Shah, 2016. Max degree around (MDA) algorithm: A smart and efficient approximate algorithm for vertex cover and independent set problems. Sindh Univ. Res. J. (Sci. Ser.), 48: 17-26.
03. Li, X.Y. and Y. Wang, 2006. Simple approximation algorithms and PTASs for various problems in wireless ad hoc networks. J. Parallel Distrib. Comput., 66: 515-530.
04. Fayaz, M. and S. Arshad, 2015. Clever steady strategy algorithm: A simple and efficient approximation algorithm for minimum vertex cover problem. Proceedings of the 2015 13th International Conference on Frontiers of Information Technology (FIT), December 14-16, 2015, IEEE, Islamabad, Pakistan, pp: 277-282.
05. Bomze, I.M., M. Budinich, P.M. Pardalos and M. Pelillo, 1999. The Maximum Clique Problem. In: Handbook of Combinatorial Optimization, Du, D.Z. and P.M. Pardalos (Eds.). Springer, Boston, Massachusetts, pp: 1-74.
06. Pardalos, P.M. and J. Xue, 1994. The maximum clique problem. J. Global Optim., 4: 301-328.
07. Baamann, K., 2003. The maximum clique problem-on finding an upper bound with application to protein structure alignment. Ph.D. Thesis, Georgia Institute of Technology, Atlanta, Georgia.
08. Tomita, E., Y. Sutani, T. Higashi and M. Wakatsuki, 2013. A simple and faster branch-and-bound algorithm for finding a maximum clique with computational experiments. IEICE Trans. Inf. Syst., 96: 1286-1298.
09. Pollatos, S., 2008. Solving the maximum clique problems on a class of network graphs, with applications to social networks. Ph.D. Thesis, Naval Postgraduate School, Monterey, California.
10. Clarkson, K.L., 1983. A modification of the greedy algorithm for vertex cover. Inf. Process. Lett., 16: 23-25.
11. Chvatal, V., 1979. A greedy heuristic for the set-covering problem. Math. Operat. Res., 4: 233-235.
12. Imran, K. and K. Hasham, 2013. Modified vertex support algorithm: A new approach for approximation of minimum vertex cover. Res. J. Comput. Inf. Technol. Sci., 1: 1-6.

13. Balaji, S., V. Swaminathan and K. Kannan, 2010. Optimization of unweighted minimum vertex cover. World Acad. Sci. Eng. Technol., 43: 716-729.
14. Ahmad, I. and M. Khan, 2014. AVSA, modified vertex support algorithm for approximation of MVC. Int. J. Adv. Sci. Technol., 67: 71-78.
15. Khan, I. and H. Khan, 2014. Degree contribution algorithm for approximation of MVC. Int. J. Hybrid Inf. Technol., 7: 183-190.
16. Fayaz, M., S. Arshad, A.S. Shah and A. Shah, 2016. An optimal approximation algorithm for optimization of un-weighted minimum vertex cover problem. Sindh Univ. Res. J. (Sci. Ser.), 48: 175-182.
17. Fayaz, M., S. Arshad, U. Zaman and A. Ahmad, 2016. A simple, fast and near optimal approximation algorithm for optimization of un-weighted minimum vertex cover. Proceedings of the 2016 International Conference on Frontiers of Information Technology (FIT), December 19-21, 2016, IEEE, Islamabad, Pakistan, pp: 176-180.