*Sindh Univ. Res. Jour. (Sci. Ser.) Vol. 48 **(4D)** 175-182 (2016)*

SINDH UNIVERSITY RESEARCH JOURNAL (SCIENCE SERIES)

An Optimal Approximation Algorithm for Optimization of Un-Weighted Minimum Vertex Cover Problem

M. FAYAZ, S. ARSHAD*, A. S. SHAH, A. SHAH

University of Malakand, Khyber Pakhtunkhwa, Pakistan

Received 28rd August 2016 and Revised 17th October 2016

Abstract: Mean of Neighbors of Minimum Degree Algorithm (MNMA) is proposed in this paper. The MNMA produces optimal or near optimal vertex cover for any known undirected, un-weighted graph. The MNMA adds a vertex cover at each step among those vertices which are neighbors of minimum degree vertices having degree equal to the mean value to construct vertex cover. The performance of MNMA is compared with other algorithms on small benchmark instances as well as on large benchmark instances such as BHOLIB and DIMACS. The MNMA is an efficient and fast algorithm and outperformed all the algorithms.

Keywords: Approximation Algorithm, Modified Vertex Support Algorithm (MVSA), Maximum Independent Set (MIS), Minimum Vertex Cover (MVC), Vertex Cover (VC), Vertex Support Algorithm (VSA)

1. INTRODUCTION

The graph is a collection of vertices connected with each other through edges (West, 2001). Mathematically a graph can be defined as G= (V, E), where V represents edges and E represents vertices. Vertex Cover (VC) can be defined as Vc⊆ V such that Vc cover all the edges of the graph and minimum vertex cover (MVC), Vc should be as minimum as possible.

Minimum Vertex Cover (MVC) has attracted researchers due to its applications in Electrical, Civil Engineering, Information Retrieval, Experimental Design, Signal Transmission, Computer Vision, Bioinformatics, aligning DNA and protein, multiple alignments of Biochemistry, Sensor Networks, Scheduling, VLSI design and Map Labeling (Richter *et al.,* 2007). The MVC problem can be demonstrated by placement of the minimum number of guards to cover each corridor of the museum, where corridors in the museum correspond the edges (Weigt and Hartmann, 2000).

Those problems which cannot be solved in polynomial time are called NP-Problems and if a problem is NP and NP-hard then known as NP-complete. In 1979, Richard Karp proved that Minimum Vertex Cover is an NP-complete problem (Karp, 1972). The NP-complete problem can be verified in polynomial time, but it cannot be solved in polynomial time. The NP-complete problem can be solved by a complete algorithm.

The complete algorithms are better for small instances, the large problem exponentially increases the time to solve it, if the size of the problem is mandatory, it can take billion or trillion of years with current computation power (Garey and Johnson, 1979).

The approximation algorithms are the best choice for researchers to solve NP-complete problems (Woeginger, 2003), these algorithms provide an approximate solution of NP problems in polynomial time (Hochbaum, 1996).

The performance of an approximation algorithm can be measured by approximation ratio and same can be mathematically defined by (1).

$$\rho i = A(I) / OPT(I) \geq 1 \quad (1)$$

Where 'I' is an instance, 'A' is near optimal solution, of the problem and OPT is the actual optimal solution of the instance, ρi is the approximation ratio for instance, I. If the ρi is equal to 1, the solution is optimal, the deviation from 1 shows a poor solution (Karakostas, 2005).

The MVC problem is further divided into weighted and un-weighted (Balas and Xue, 1996). The MVC problem has two versions i.e. optimization and decision. The vertex cover of minimum size is found in the optimized version, the decision version finds the existence or not existing of an MVC of size k, where k is the set of vertices (Balaji *et al.,* 2010). The proposed MNMA algorithm is for the un-weighted MVC problem and optimization version respectively.

The structure of the paper is organized as in section 2, the literature review of some well-known algorithm is carried out, the proposed algorithm is provided in section 3, section 4 describes experimental results, and finally conclusion and future work is provided.

2. RELATED WORK

The well-known approximation algorithms are reviewed here for comparison of simplicity, low run time complexity, and optimality.

[3]SZABIST, Islamabad, Pakistan, email: shahsalamss@gmail.com

[4]International Islamic University Malaysia (IIUM), Kuala Lumpur, Malaysia,

The Maximum Degree Greedy (MDG) is based on greedy approach, it is the modified version of the greedy approach for a set cover problem developed in (Chvatal, 1979). In MDG, the vertex having a maximum degree in G is selected for MVC. The run-time complexity of the MDG in the worst case is O (E2) (Clarkson, 1983). The MDG fails on small benchmark instances to provide the optimal result as shown in **(Fig- 2).**

Balaji *et al.,* 2010 produced a new data structure, vertex support, which is the sum of degrees of all adjacent vertices of the minimum degree vertex. The vertex having maximum support is selected for MVC. The run time complexity of Vertex Support Algorithm (VSA) is O (EV2). The VSA is based on greedy approach and like MDG it also fails on small benchmark instances to provide optimal results.

The Modified Vertex Support Algorithm (MVSA) is the modified version of VSA, it calculates the support of each vertex. The vertex having minimum support is selected and all adjacent vertices of that vertex are calculated. Then the minimum support vertex in all adjacent vertices of the minimum vertex is selected for MVC. There is no extra complexity involved in the computation, and the decision to select a vertex for MVC is very simple but complex as compared to MDG and VSA. The running time of MVSA is O (EV2log v) (Imran and Hasham, 2013).

3. PROPOSED APPROACH

After the detailed literature review, it has been observed that simple algorithms are fast, but they do not give good results, so there is a need for a simple and fast algorithm that can also produce good results. This was the motivation behind the proposed MNMA algorithm.

The neighborhood of the minimum degree vertex is considered in the proposed approach. First, the degree of each vertex of a graph G is calculated. A vertex having a minimum degree is identified and then calculated all its adjacent vertices. The arithmetic mean can be calculated by adding, degrees of all adjacent vertices of the minimum degree vertex and divided by a number of vertices adjacent to the minimum degree vertex. The fraction partial of the mean has been removed by floor function (Weigt and Hartmann, 2000). If any degree of adjacent vertices of the minimum degree vertex is equal to the mean value, then that vertex is added to MVC. If the mean is not equal to any degree of adjacent vertices of minimum degree vertex, then 1 is added to the mean value and the same process repeats. This process continues till finding of a mean having a value equal to any degree of adjacent vertices of minimum degree vertex. If mean value becomes equal without any optimization effort, the complexity of MNMA is O (V2 log V); with V = n, the complexity becomes O (n2 log n) which is polynomial.

3.1. Terminologies

Let G= (V, E) be a graph where V: Vertex set, E: Edge set, C: set of vertices cover by the proposed algorithm, the neighborhood of a vertex: , N(v) = { u ∈ V /u is adjacent to v}, d(v): the degree of a vertex ,$\overline{x}$ = floor($\sum_{i=1}^{n} \frac{adj_deg(i)}{n}$) , where floor (x) = $\lfloor x \rfloor$ is the largest integer not greater than x (Graham, 1994), C: vertices cover by an algorithm, S: sum of degrees of adjacent verities of the minimum degree vertex. Maximum degree of a vertex: it is denoted by Δ (G) and δ (G) represent maximum degree vertex in G respectively.

The data structure used in the proposed approach is edge list, in an edge list L every edge e is stored as a tuple (vi, vj) in a sorted list. The edge list of the figure 2 (a) is L = {(1, 2), (1, 3), (1, 4), (1, 5), (2, 6), (3, 7), (4, 8), (5, 9), (6, 10)}. The edge list is a space efficient way to store the graph in computer memory (Cormen, 2009).

Algorithm: MNMA (G = (V, E))
Input: G= (V, E)
Output: C
Begin

1. C ← ∅
//initially no vertex is in the vertex cover set
2. S ← ∅
3. While E ≠∅ do {
4. n ← |G(V,E)|
5. for i ← 1 to n {
6. Calculate d (v_i)
//Calculate the degree of each vertex
7. } u← δ (G)
8. Calculate N (u)
//Calculate the neighbors vertices of minimum degree vertex
9. m ← |N(u)|
10. for each j←1 to m {
11. S ← d(v_J) + S,
12. } Return S
13. . $\overline{x}$ ← ∟ S/m ⌋
14. for each k←1 to m {
15. If (N_d(vk) = . $\overline{x}$){
16. p←N_d(v_k)
17. else
18. $\overline{x}$ ← $\overline{x}$ +1
19. }} V ← V − {p}
20. C← C ∪ {p}
21. } Return C

Exit.

3.2. Pseudo Code of Proposed Algorithm

3.3. Diagrammatic Representation of MNMA

The proposed MNMA algorithm is diagrammatically represented into the **(Fig. 1),** the working of the algorithm and its flow is represented for clear understanding.

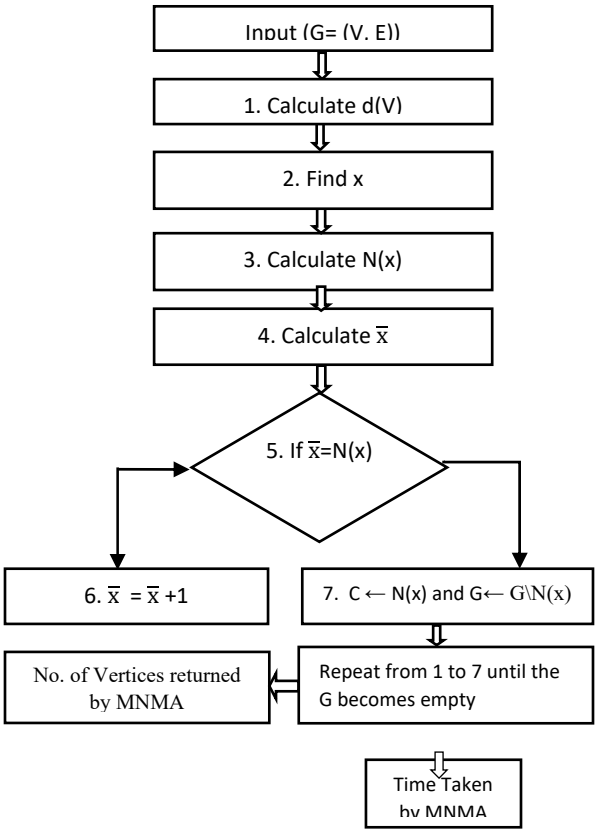


Fig 1: Flow diagram of MNMA

3.4. Implementation

The implementations of the algorithm have been carried out using MATLAB R2010a version 7.10.0.499 with an Intel core i5 system having windows 7 operating system. The proposed MNMA algorithm has been compared with other well-known algorithms like MDG, VSA, and MVSA on the basis of runtime complexity and optimality. The empirical results obtained from different benchmark instances are summarized in experimental results.

4. EXPERIMENTAL RESULTS

The performance comparison of MNMA with small and large benchmark instances is as under:

4.1. Performance Comparison on Small Graphs

The proposed algorithm has been compared with MDG, VSA, and MVSA on small graphs. In figure 2 (a) the output graph by MNMA and MVSA covered four vertices provides the optimal result. The algorithm in **Fig. 2 (b)** with the same graph with MDG, and VSA covered five vertices, do not provide optimal result on the benchmark instances. In **Fig. 2 (c)** the given graph covered four vertices by MNMA, MDG and VSA and provided optimal result, four, hence, these algorithms provide optimal results on the given graph. In **(Fig. 2 (d)** MVSA takes five vertices to cover the graph which is not optimal. The MNMA provides optimal results on these graphs, while the others fail to provide optimal results.

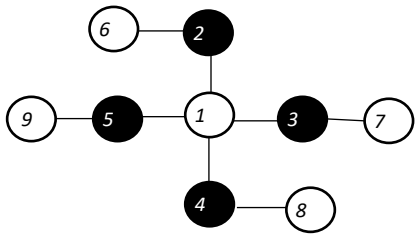


Fig 2: (a) ρ = 4/4=1 returned by MNMA and MVSA

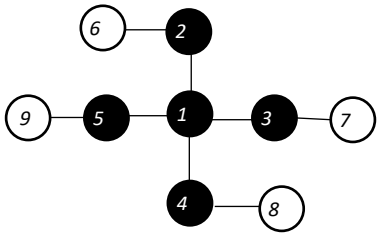


Fig 2: (b) ρ = 5/4=1.25 returned by MDG and VSA

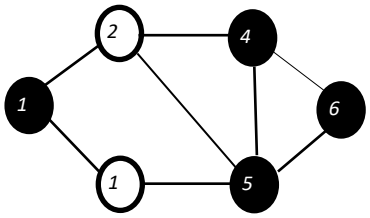


Fig 2: (c) ρ = 4/4=1 returned by MNMA, MDG and VSA

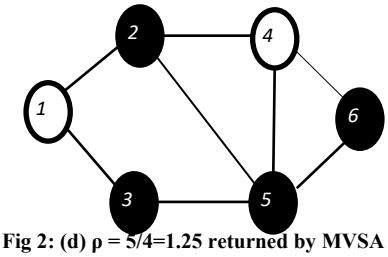


Fig 2: (d) ρ = 5/4=1.25 returned by MVSA

4.2. Comparison on Large Benchmark Instances

To test the performance of the proposed algorithm with other well-known algorithms with respect to the number of vertices cover by the algorithms, approximation ratio comparison and time taken by algorithms on the benchmark graphs with known results are used, all the benchmark instances are taken from DIMACS (Johnson and Trick, 1996)and BHOSLIB (Xu, 2012) challenging suites.

The **(Table 1)** contains comparison of the proposed algorithm with MDG, VSA and MVSA by the number of vertices cover by an algorithm on DIMACS, and BHOSLIB instances.

Table 1. Vertices cover by MNMA, MDG, VSA and MVSA

Sr. No	Benchmarks	V	C*	MNMA |C|	MDG |C|	VSA |C|	MVSA |C|
1	graph50_6	50.0	38.0	38.0	38.0	44.0	38.0
2	graph50_10	50.0	35.0	35.0	35.0	41.0	35.0
3	graph100_1	100.0	60.0	60.0.	60.0	95.0	60.0
4	graph100_10	100.0	70.0	70.0	70.0	96.0	70.0
5	graph200_5	200.0	150.0	150.0	150.0	184.0	150.0
6	Graph500_1	500.0	350.0	350.0	350.0	485.0	350.0
7	graph500_2	500.0	400.0	400.0	400.0	484.0	400.0
8	graph500_5	500.0	290.0	290.0	290.0	454.0	290.0
9	phat300-1	300.0	292.0	293.0	293.0	292.0	294.0
10	phat300-2	300.0	275.0	276.0	278.0	275.0	279.0
11	phat300-3	300.0	264.0	267.0	269.0	264.0	272.0
12	phat700-1	700.0	689.0	692.0	693.0	689.0	692.0
13	phat700-2	700.0	656.0	658.0	660.0	656.0	660.0
14	phat700-3	700.0	638.0	640.0	642.0	638.0	649.0
15	johnson8-2-4	28.0	24.0	24.0	24.0	24.0	24.0
16	johnson8-4-4	70.0	56.0	56.0	62.0	56.0	56.0
17	johnson16-2-4	120.0	112.0	112.0	112.0	112.0	112.0
18	johnson32-2-4	496.0	480.0	480.0	480.0	480.0	480.0
19	sanr200_0.7	200.0	182.0	185.0	184.0	182.0	186.0
20	sanr200_0.9	200.0	158.0	163.0	164.0	158.0	163.0
21	sanr400_0.5	400.0	387.0	389.0	392.0	387.0	389.0
22	sanr400_0.7	400.0	379.0	382.0	384.0	379.0	381.0
23	fbr35-17-2	595.0	560.0	565.0	570.0	573.0	424.0
24	fbr_30_15_5	450.0	420.0	426.0	429.0	429.0	565.0
25	c125	125.0	91.0	92.0	93.0	91.0	95.0
25	C250.9	250.0	206.0	208.0	211.0	206.0	211.0
27	C500.9	500.0	443.0	448.0	453.0	443.0	449.0
28	C2000.9	2000.0	1922.0	1930.0	1944.0	1923.0	-.
29	brock200_1	200.0	188.0	190.0	190.0	188.0	191.0
30	brock200_4	200.0	183.0	192.0	192.0	183.0	193.0
31	gen200_p0.9_44	200.0	156.0	163.0	165.0	156.0	166.0
32	hamming6-2	64.0	32.0	32.0	32.0	32.0	32.0
33	hamming6-4	64.0	60.0	60.0	60.0	60.0	60.0
34	hamming8-2	256.0	128.0	128.0	128.0	128.0	128.0
35	hamming8-4	256.0	240.0	240.0	240.0	240.0	240.0
36	hamming10-2	1024.0	512.0	512.0	512.0	512.0	512.0
37	dsjc-500	500.0	487.0	489.0	491.0	487.0	489.0
38	killer4	171.0	160.0	160.0	164.0	160.0	160.0
39	killer5	776.0	749.0	753.0	764.0	749.0	754.0
40	c-fat200-1	200.0	188.0	188.0	188.0	188.0	188.0
41	c-fat200-2	200.0	176.0	176.0	176.0	176.0	176.0
42	c-fat200-5	200.0	142.0	142.0	142.0	144.0	142.0
43	c-fat500-1	500.0	486.0	486.0	486.0	486.0	486.0
44	c-fat500-2	500.0	474.0	474.0	474.0	474.0	474.0
45	c-fat500-5	500.0	436.0	436.0	436.0	436.0	436.0
46	c-fat500-10	500.0	374.0	374.0	374.0	374.0	374.0
47	MANN_a27.clq.b	378.0	252.0	253.0	261.0	253.0	253.0

4.3. Performance Comparison by the Time

MNMA |C|, VSA |C|, MVSA |C| and MDG |C|: Covers returned by MNMA, MDG, VSA, MVSA where C* and 'V' represents the optimal solution and number of vertices respectively in the graph The **(Table 2)** contains performance comparison among MNMA, MDG, VSA and MVSA on DIMACS and BHOSHLIB benchmarks with respect to Approximation ratio. The table 3 contains comparison of the MNMA with MDG, and VSA with respect to time complexity (sec).

Table 2. Approximation ratio comparison among MNMA, VSA, MDG and MVSA

Sr. No	Benchmark	V	C*	MNMA ρi	MDG ρi	VSA ρi	MVSA ρi
1	graph50_6	50.0	38.0	1	1	1.1580	1
2	graph50_10	50.0	35.0	1	1	1.1710	1
3	graph100_1	100.0	60.0	1	1	1.5830	1
4	graph100_10	100.0	70.0	1	1	1.3710	1
5	graph200_5	200.0	150.0	1	1	1.2270	1
6	graph500_1	500.0	350.0	1	1	1.3860	1
7	graph500_2	500.0	400.0	1	1	1.2100	1
8	graph500_5	500.0	290.0	1	1	1.5650	1
9	phat300_1	300.0	292.0	1.0030	1.0030	1	1.0060
10	phat300_2	300.0	275.0	1.0030	1.0100	1	1.0140
11	phat300_3	300.0	264.0	1.0110	1.0180	1	1.0300
12	phat700_1	700.0	689.0	1.0040	1.0050	1	1.0040
13	phat700_2	700.0	656.0	1.0030	1.0060	1	1.0060
14	phat700_3	700.0	638.0	1.0030	1.0060	1	1.0170
15	jhonson8_2_4	28.0	24.0	1	1	1	1
16	jhonson8_4_4	70.0	56.0	1	1.1070	1	1
17	jhonson16_2_4	120.0	112.0	1	1	1	1
18	jhonson32_2_4	496.0	480.0	1	1	1	1
19	sanr200-0.7	200.0	182.0	1.0160	1.010	1	1.0210
20	sanr200-0.9	200.0	158.0	1.0310	1.0370	1	1.0310
21	sanr400_0.5	400.0	387.0	1.0020	1.0120	1	1.0050
22	sanr400_0.7	400.0	379.0	1.0070	1.0130	1	1.0050
23	fbr_30_15_5	450.0	420.0	1.0140	1.0210	1.0210	1.0090
24	fbr_35_17_2	595.0	560.0	1.0090	1.0170	1.0230	1.0080
25	c 125	125.0	91.0	1.0110	1.0210	1	1.0430
26	c 250	250.0	206.0	1.0090	1.0240	1	1.0240
27	c500.9	500.0	≤443.0	1.0110	1.0220	1	1.0130
28	C2000.9	2000.0	1922.0	1.0040	1.0110	1.0010	----
29	broc200_2	200.0	188.0	1.0100	1.0210	1	1.0150
30	broc200_4	200.0	183.0	1.0490	1.0270	1	1.0540
31	gen200_p0.9_44	200.0	156.0	1.0490	1.0570	1	1.0460
32	Hamming6_2	64.0	32.0	1	1	1	1
33	Hamming6_4	64.0	60.0	1	1	1	1
34	Hamming8_2	256.0	128.0	1	1	1	1
35	Hamming8_4	256.0	240.0	1	1.0330	1	1
36	Hamming10_2	10240	512.0	1	1	1	1
37	dsjc500	5000	487.0	1.0040	1.0080	1	1.0040
38	keller4	171.0	160.0	1	1.0250	1	1
39	keller5	776.0	749.0	1.0050	1.0200	1	1.0060
40	cfat200_1	200.0	188.0	1	1	1	1
41	cfat200_2	200.0	176.0	1	1	1	1
42	cfat200_5	200.0	142.0	1	1	1.0140	1
43	cfat500_1	500.0	486.0	1	1	1	1
44	cfat500_2	500.0	474.0	1	1	1	1
45	cfat500_5	500.0	436.0	1	1	1	1
46	c-fat500-10	500.0	374.0	1	1	1	1
47	mann_a27	378.0	252.0	1.0040	1.0350	1.0040	1. 004

Table 3. Time taken by MNMA, VSA, MDG

Sr.No	Benchmarks	|V|	MNMA time	MDG time	VSA time
1	brock200_1	200	3.67	2.030	3.090
2	brock200_4	200	4.3	2.590	3.931
3	brock800_4	800	804.13	832.670	1097.5
4	c125	125	0.83	0.1710	0.300
5	C250.9	250	3.300	1.10	2.310
6	C500.9	500	27.700	13.81	21.902
7	C2000.9	2000	1925.7	1350.4	1709.5
8	c-fat200-1	200	9.160	3.560	5.430
9	c-fat200-2	200	5.444	2.932	4.790
10	c-fat200-5	200	4.450	1.700	2.745
11	c-fat500-1	500	460.30	328.21	454.51
12	c-fat500-2	500	499.90	318.18	433.5
13	c-fat500-5	500	292.50	228.040	285.7
14	c-fat500-10	500	146.55	67.7040	105.369
15	dsjc-500	500	98.050	71.280	105.525
16	gen200_p0.9_44	200	2.6360	0.682	5.6161
17	johnson8-2-4	28	0.10	0.0540	0.0634
18	johnson8-4-4	70	0.30	0.080	0.3745
19	johnson16-2-4	120	0.951	0.250	0.4215
20	johnson32-2-4	496	29.905	17.140	25.999
21	san200_0.7_1	200	4.087	1.3940	2.0123
22	sanr200_0.9	200	1.669	0.5140	0.9514
23	sanr400_0.5	400	48.875	29.190	44.617
24	sanr400_0.7	400	27.175	17.850	27.159
25	phat300-1	300	24.164	14.300	21.575
26	phat300-2	300	16.114	9.297	14.455
27	phat300-3	300	9.048	4.620	7.161
28	phat700-1	700	1198.4	862.86	1228.1
29	phat700-2	700	744.062	640.583	858.842
30	phat700-3	700	172.459	128.95	194.059
31	hamming6-2	64	0.078	0.0620	0.0934
32	hamming6-4	64	0.312	0.1240	0.1405
33	hamming8-2	256	1.248	6.7540	1.0923
34	hamming8-4	256	8.985	3.810	0.1244
35	hamming10-2	1024	32.276	15.26	39.964
37	MANN_a27.clq.b	378	3.822	0.860	1.5132
38	fbr35-17-2	595	56.035	45.29	68.1411
39	fbr_30_15_5	450	24.273	16.801	25.5372
40	killer4	171	3.244	1.420	2.1993
41	killer5	776	237.168	1293.4	183353
42	graph50_6	50	0.202	0.1240	0.124
43	graph50_10	50	0.234	0.1240	0.0783
44	graph100_1	100	0.530	0.2490	0.4683
45	graph100_10	100	0.655	0.220	0.4362
46	graph200_5	200	4.555	1.830	3.1823
47	graph500_1	500	58.765	50.7	106.299
48	graph500_2	500	228.869	225.64	310.769
49	graph500_5	500	161.835	154.300	279.725

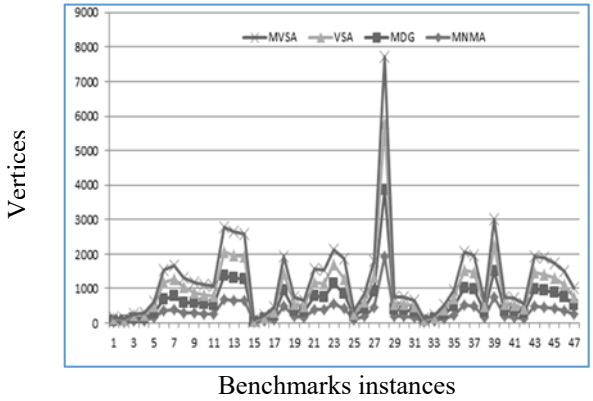


Fig. 3. Graphical Representation of the Table 1

Fig. 3 is the graphical representation of table 1 in order to better exhibit the results.

The **(Table 3 and Fig. 4)** exhibit the time complexity of proposed algorithm along with the counterpart's algorithms. The time comparison of the proposed is carried out with MDG and VSA, these algorithms have been selected for comparison due to its simplicity and low computation time. The result illustrated in the table 3 and figure 6 on DIMACS and BHOSLIB benchmark graphs, illustrate that the MNMA takes as less time to compute as VSA and MDG.

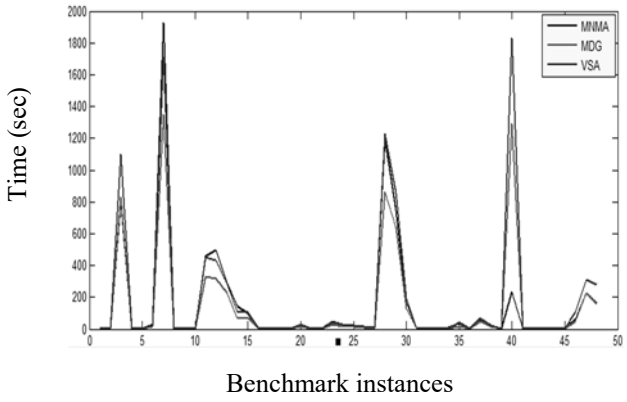


Fig. 4. Time complexity (sec) of MNMA, MDG, VSA

The approximation ratio result of the MNMA on the given benchmark graphs is compared with MDG, VSA and MVSA are depicted in **(Fig 5, 6 and 7)** respectively. The result shows that the approximation ratio of MNMA are far better as compared to the competent approximation algorithm for MVC. The comparison of the proposed algorithm with the other state of the art algorithms for MVC has been carried out in fig. 5, 6 and 7, undoubtedly the MNMA provides improved results as compared to the other approximation algorithms for MVC.

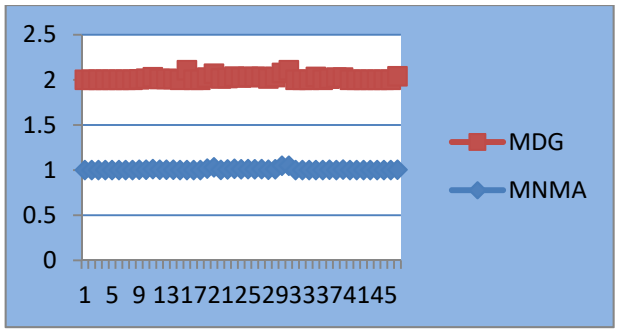


Fig.5. Approximation Ratio Comparison of MDG and MNMA

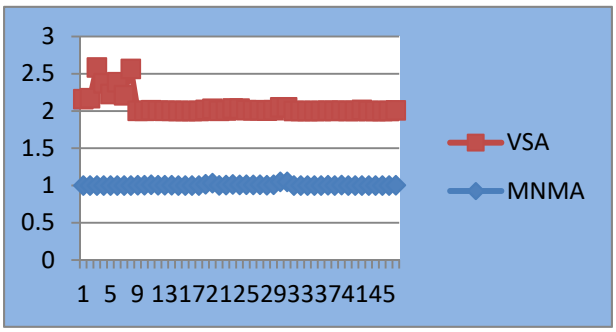


Fig.6. Approximation RatioComparison of MNMA and VSA

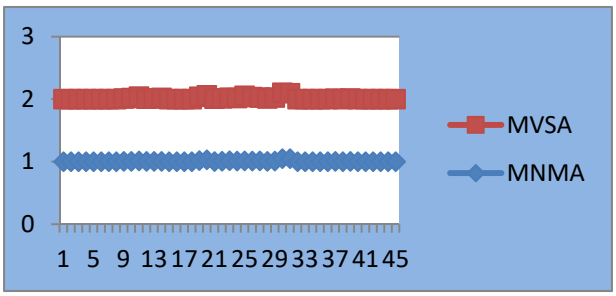


Fig.7. Approximation Ratio Comparison of MNMA and MVSA

4.4. Worst, Average and Standard Deviation Approximation Ratios

(Table 4) is about the worst and average approximation ratios. The worst approximation ratio means more deviation from `1`, more deviation illustrates the poor result. Average approximation is used to measure the overall performance of an algorithm and the standard deviation is measure the deviation for each benchmark instance. The average and standard deviation approximation ratios of an algorithm is more approximate to '1' indicates that the algorithm overall performance is virtuous. MNMA has outstripped MDG, VSA and MVSA on worst, average and standard deviation approximation ratios as presented in the **(Table 4 and Fig.8)** respectively.

Table 4. Worst and Average and standard deviation approximation ratios of MNMA, VAS, MDG and MVSA

Algorithms	Worst ρ	Average ρ	Standard deviation(σ) ρ
MNMA	1.049	1.005696E0	.01099
MDG	1.107	1.012613E0	.01925
VSA	1.583	1.059435E0	.14278
MVSA	1.064	1.008	.01345

The **(Fig. 8).** illustrated the worst, average and standard deviation of approximation ratios of MNMA, MDG, VSA and MVSA respectively. The MNMA has flattened these algorithms by providing healthier worst, average and standard deviation approximation ratios as compared to other competitor approximation algorithms.

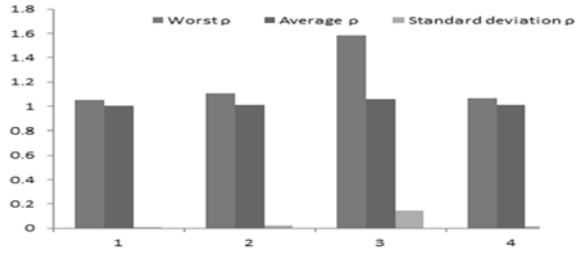


Fig.8. Worst, average and standard deviation of MNMA, VSA and MVSA

5. CONCLUSSION

In this paper an approximation algorithm entitled Mean of Neighbors of the Minimum Degree Algorithm (MNMDA) has been proposed in order to solve the MVC problem in an accurate fashion with low time and less complexity. The assessment of the proposed algorithm has been carried out on both run time complexity and optimality on BHOSLIB and DIMACS benchmark instances. The performance of MNMDA the proposed algorithm is better as compared to other recognized algorithm. The designing of the proposed algorithm is very simple, can be easily implemented and used for the practical applications that demand quality solution for the MVC problem in shorter span of time. The current algorithm works for un-weighted MVC, in the future we would like to apply it on weighted MVC and will also include some optimization tweaks to improve its performance and keep the algorithm as simple as possible.

REFERENCES:

Balas, E. (1996). Weighted and Un-Weighted Maximum Clique Algorithms with Upper Bounds from Fractional Coloring. Algorithmica,15(5), 397-412.

Balaji, S., V. Swaminathan, and K. Kannan,(2010). Optimization of Un-Weighted Minimum Vertex Cover. World Academy of Science, Engineering and Technology, 4(7), 941-946.

Chvatal, V. (1979). A Greedy Heuristic for the Set-Covering Problem. Mathematics of operations research,4(3), 233-235.

Cormen, T. H. (2009). Introduction to Algorithms, MIT press.

Clarkson, K. L. (1983). A Modification of the Greedy Algorithm for Vertex Cover. Information Processing Letters, 16(1), 23-25.

Garey, M. R. and D.S. Johnson, (1979). A Guide to the Theory of NP-Completeness. W.H. Freeman and Company, San Francisco.

Graham, R. L. (1994). Concrete Mathematics: a Foundation for Computer Science. Pearson Education India.

Hochbaum, D. S. (1996). Approximating Covering and Packing Problems. In: Approximation Algorithms for NP-Hard Problems. Ed. By S.D. Hochbaum, PWS Publishing Company, 94-143.

Karakostas, G. (2005). A Better Approximation Ratio for the Vertex Cover Problem. In: Proceedings of the 32nd International Colloquium on Automata, Languages and Programming, ICALP, 1043–1050.

Imran, K. and K. Hasham, (2013). Modified Vertex Support Algorithm: A New Approach for Approximation of Minimum Vertex Cover. Research Journal of Computer and Information Technology Sciences,1(16),1-6.

Johnson, D. S. and M.A. Trick, (1996). Cliques, Coloring, and Satisfiability: Second DIMACS implementation challenge, October 11-13, Vol. 26 American Mathematical Soc.

Richter, S., M. Helmert, and C. Gretton, (2007). A Stochastic Local Search Approach to Vertex Cover. 30th Annual German Conference on AI, KI 2007,Osnabrück, Germany, September 10-13, Proceedings, 412-426.

West, D. B. (2001). Introduction to Graph Theory. Prentice hall Upper Saddle River.China

Weigt, M. and A. K. Hartmann, (2000). Number of Guards Needed by a Museum: A Phase Transition in Vertex Covering of Random Graphs. Physical review letters, 84: 6118.26.

Woeginger, G. J. (2003). Exact Algorithms for NP-Hard Problems: A Survey. Combinatorial Optimization-Eureka, You Shrink. LNCS, 2570, 185-207, Springer.

Xu, K. (2012). Vertex Cover Benchmark Instances (DIMACS & BHOSLIB). IJEA (international journal of Experimental algorithms), 3(1), 1-18.