

Approximating Maximum Weighted Independent Set Using Vertex Support

S. Balaji, V. Swaminathan and K. Kannan

Abstract—The Maximum Weighted Independent Set (MWIS) problem is a classic graph optimization NP-hard problem. Given an undirected graph $G = (V, E)$ and weighting function defined on the vertex set, the MWIS problem is to find a vertex set $S \subseteq V$ whose total weight is maximum subject to no two vertices in S are adjacent. This paper presents a novel approach to approximate the MWIS of a graph using minimum weighted vertex cover of the graph. Computational experiments are designed and conducted to study the performance of our proposed algorithm. Extensive simulation results show that the proposed algorithm can yield better solutions than other existing algorithms found in the literature for solving the MWIS.

Keywords—weighted independent set, vertex cover, vertex support, heuristic, NP - hard problem.

I. INTRODUCTION

IN graph theory an independent set of a graph is a subset of vertices in which no two vertices are adjacent (i.e., connected by an edge) and the maximum independent set problem (MIS) calls for finding the independent set of maximum cardinality. The MIS is a classic one in computer science and in graph theory and it has many important applications, including combinatorial auctions[7], graph coloring[13], coding theory[9], geometric tiling, fault diagnosis, pattern recognition, molecular biology[11], and more recently its application in bioinformatics[16] have become important. The Maximum Weighted Independent Set (MWIS) is a generalization of MIS problem, in which vertices have positive weight and one has to find an independent set of maximum weight and both MIS and MWIS problems are known to be NP-hard[12]. Hence simple algorithms which yield acceptable solutions sufficiently fast are quite important for such related practical problems.

Pardalos and Xue[15] recently published a review with 260 references. This problem is computationally intractable even to approximate with certain absolute performance bounds[6, 10]. Very few numbers of algorithms has been proposed for the MWIS problem. Babel[1] proposed an efficient branch and bound procedure. Ostergard[14] also developed a fast branch and bound based exact method. Bomze et al.[4, 5] proposed a method based on replicator dynamics. It uses the continuous formulation of maximum weight clique problem. Recently, Pullan[18] proposed a local search procedure for

S. Balaji is with the Department of Mathematics, SASTRA University, Thanjavur, India. e-mail: balaji_maths@yahoo.com.

V. Swaminathan is with the Ramanujan Research centre, Saraswathi Narayanan College, Madurai, India. e-mail: sulanesri@yahoo.com.

K. Kannan is with Department of Mathematics, SASTRA University, Thanjavur, India. e-mail: kkannan@maths.sastr.edu.

maximum independent set problem, which was supported by computational experiments.

In this paper for efficiently solving MWIS problems, an effective algorithm called Support Ratio Algorithm (SRA) is proposed. This algorithm effectively finds MWIS of a graph G by finding the Minimum Weighted Vertex Cover (MWVC) of the graph G . The proposed algorithm designed with the term called support of vertices, which involves the sum of the degrees of adjacency vertices, ratio between weight and product of support and degree of vertices to get a near smallest weighted vertex cover of the graph. It's effectiveness for finding the MWVC of the graph shown in[2]. We compared our algorithm with the other existing algorithm namely[1, 5, 14, 18]. The experimental result shows that our algorithm is very fast and yields better solutions than the compared algorithms for many random graphs and DIMACS benchmark graphs.

The paper is organized as follows. Section II briefly describes the maximum weighted independent set (MWIS) problem and the minimum weighted vertex cover problem (MWVC) and its theoretical background. Section III outlines the SRA. In Section IV graph model used in the experiments is briefly described. Section V provides experiments done and their results. Section VI summarizes and concludes the paper.

II. MAXIMUM WEIGHTED INDEPENDENT SET AND MINIMUM WEIGHTED VERTEX COVER

Let $G = (V, E)$ be an arbitrary undirected graph, where $V = \{1, 2, \dots, n\}$ is the set of vertices and $E \subseteq V \times V$ (not in ordered pairs) is the set of edges. Two distinct vertices u and v are called adjacent if they are connected by an edge; an independent set S of G is a subset of V whose elements are pairwise non-adjacent. The maximum independent set (MIS) problem seeks to find an independent set with large number of vertices. The size of the maximum independent set of G is the stability number of G and is denoted by α . MWIS problem is a generalization of MIS in which vertices have positive weight i.e., a weight function $\omega : V \rightarrow \mathbb{R}$ associated with each vertex of V and one has to find the independent set with maximum weight.

A vertex cover for G is a subset V_c of V such that for each edge $(u, v) \in E$, at least one of u or v or both belongs to V_c . The minimum vertex cover (MVC) problem consists of identifying the vertex cover V_c of G which has minimum cardinality and the size of the minimum vertex cover of G is denoted by β . The MIS and MVC problems are related in that the maximum independent set S of G contains all those vertices that are

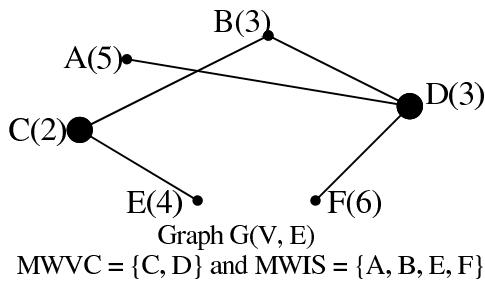


Fig. 1. Illustration of MWIS and MWVC Problem

not in the minimum vertex cover of G . i.e. $S = V - V_c$ and $\alpha + \beta = n$. The relationship identified above for MIS and MVC problems also hold for MWIS and MWVC instances and, in addition, if W_T is the total weight of the vertices in G and W_C , W_S denotes the total weights in minimum weighted vertex cover, the maximum weighted independent set in G then $W_T = W_C + W_S$. Fig.1 briefly explains the above criteria. Graph of MWIS and MWVC instances shown in the Fig. 1, where the index of the vertices are denoted by alphabets and the number in the brackets is the weight of the associated vertices. The optimal solution for MWIS is $S = \{A, B, E, F\}$ with a total weight of 18 and MWVC is $V_c = \{C, D\}$ with total weight of 5. However there is no vertex set in G with pairwise non adjacent vertices and total weight greater than 18 and no vertex set in G covering all the edges with total weight less than 5. Moreover we can see clearly that $W_T = W_C + W_S$.

There are two versions of the vertex cover problem: the decision and optimization versions. In the decision version, the task is to verify for a given graph G , a weight function $\omega : V \rightarrow R^+$ and k^+ , weighted vertex cover asks for a vertex cover of total weight at most k but in the optimization version the task is to find a vertex cover of minimum total weight. In this paper we consider the optimization version of the minimum weighted vertex cover with the goal of obtaining optimum solution. Now the minimum weighted vertex cover problem can be formulated as an integer programming problem by using the following conditions:

Binary variables a_{ij} ($i = 1, 2, 3, \dots, n; j = 1, 2, 3, \dots, n$) form the adjacency matrix of the graph G . Each variable has only two values (1 or 0) according as an edge exists or not. In other words, if an edge (v_i, v_j) is in E , then a_{ij} is 1 else a_{ij} is 0. For example, for the graph of Fig.1 has the following adjacency matrix

$$A = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \end{bmatrix}$$

The output of the program expresses the vertex v_i is in the MWIS or not. $v_i=1$ if it is in the MWIS otherwise $v_i=0$. Thus the total weight in the MWIS can be expressed by $Z = \sum w_i v_i$, $1 \leq i \leq n$. Any one or none of the vertex of the edge (v_i, v_j) is included in the independent set, so we

have the constrained condition of the minimum vertex cover can be written as $v_i + v_j \leq 1$. Thus the problem can be mathematically transformed into the following optimization problem as

$$\begin{aligned} \text{Max } Z &= \sum w_i v_i \\ \text{Subject to} \\ v_i + v_j &\leq 1 \quad \forall (v_i, v_j) \in E \\ v_i &\in \{0, 1\} \quad \forall v_i \in V \end{aligned}$$

Thus we find the MWIS of the graph G by finding MWVC of the graph G . i.e., if the set V_c contains the vertices of MWVC of the graph then the set S of all vertices of MWIS of the graph G can be extracted from MWVC by $S = V - V_c$.

III. TERMINOLOGIES AND PROPOSED ALGORITHM

Neighborhood of a vertex: Let $G = (V, E)$, V is a vertex set and E is an edge set, be an undirected graph and let $|V| = n$ and $|E| = m$. Then for each $v \in V$, the neighborhood of v is defined by $N(v) = \{u \in V / u \text{ is adjacent to } v\}$ and $N[v] = v \cup N(v)$.

Degree of a vertex: The degree of a vertex $v \in V$, denoted by $d(v)$ and is defined by the number of neighbors of v .

Support of a vertex: The support of a vertex $v \in V$ is defined by the sum of the degree of the vertices which are adjacent to v , i.e., $\text{support}(v) = s(v) = \sum_{u \in N(v)} d_G(u)$.

A. Support Ratio Algorithm (SRA - Proposed)

The following algorithm is designed to find the general maximum weighted independent set of a graph G . Adjacency matrix (a_{ij}) of the given graph G of n vertices and m edges and the weights of the each vertexes are given as the input of the program. The degree $d(v)$ and support $s(v)$ of the each vertex $v \in V$ are calculated. Support of the vertex calculated by the relation $\sum_{u \in N(v)} d_G(u)$. Moreover the ratio $r(v)$ of each $v \in V$ calculated by the relation $r(v) = \frac{s(v)d(v)}{w(v)}$. Add the vertex which has the maximum value of the ratio $r(v)$ into the vertex cover V_c . If one or more vertices have equal maximum value of the support, in this case if $(s(v_i) \geq s(v_j))$, add the vertex v_i into the vertex cover V_c otherwise add v_j into V_c . Update the adjacency matrix of G by putting zero in to the row and column entries of the corresponding vertex $v \in V_c$. Proceed the above process until the edge set E has no edges. i.e., up to $a_{ij} \neq 0 \quad \forall i, j$. The pseudo-code of the proposed algorithm is given below.

```

Input:  $G (V, E)$ 
Output: Max. Weighted Independent Set  $S(G) = V - V_c$ 
and  $Z = \sum_{v_i \in S} w_i v_i$ 
while  $E \neq \emptyset$  do
step 1:
for  $i \leftarrow 1$  to  $n$ 
for  $j \leftarrow 1$  to  $n$ 
 $d(v_i) = \sum_j a_{ij}$ 
step 2:
for  $i \leftarrow 1$  to  $n$ 
for  $j \leftarrow 1$  to  $n$ 

```

```

 $s(v_i) = \sum_{v_j \in N(v_i)} d_G(v_j)$ 
step 3:
for i←1 to n
 $r(v_i) = \frac{s(v_i)d(v_i)}{w(v_i)}$ 
step 4:
max = r(v1);
k = 1;
select the vertex which has the maximum value
of r(v) in to Vc
for i←2 to n
if(max < r(vi))
max = r(vi);
t = i;
Vc ← Vc ∪ vi
end if
if multiple vertices have equal maximum value of s(v)
then follow step 4a
step 4a:
if((max = r(vi)&(s(vi-k) <= s(vi)))
max = r(vi);
t = i;
Vc ← Vc ∪ vi
end if
if((max = r(vi)&(s(vi-k) > s(vi)))
max = r(vi-k);
t = i-k;
Vc ← Vc ∪ vi-k
end if
k = k+1;
end for
step 5:
for i←1 to n
(ati) = 0;
(ait) = 0;
end for
end while.
for i←1 to n
if(vi ∈ Vc)
vi = 1;
else
vi = 0;
end for
end

```

IV. GRAPH MODELS

This section outlines the graph models used to assess the effectiveness of the proposed algorithm in previous section. The graph models used are (i) G(n, p) graphs[3] and (ii)G(n, m) graphs[3][19]. The models are standard random graph models from the graph theory and all the graphs are undirected.

1) G(n, p) Model: The G(n, p) model is also called Erdos Renyi random graph model[3], consists of graphs of n vertices for which the probability of an edge between any pair of nodes is given by a constant $p > 0$. To ensure that graphs are almost always connected, p is chosen so that $p > \frac{\log(n)}{n}$. To generate a G(n, p) graph we start with an empty graph. Then we iterate through all pairs of nodes and connect each of these pairs with probability p.

2) Algorithm to generate (G, n, p)graphs: The pseudo code for generating G (n, p) graphs as follows

```

initialize graph G(V, E)
for i ← 1 to n
for j← i+1 to n
add edge (i, j) to E with probability p
return (G).

```

The expected number of edges of G(n, p) graph is $pn(n - 1)/2$ and expected degree is np. Graphs are generated for different p and n values.

3) G(n, m) Model: The G(n, m) model consists of all graphs with n vertices and m edges. The number of vertices n and the number of edges m are related by $m = nc$, where $c > 0$ is constant. To generate a random G(n, m) graph, we start with a graph with no edges. Then, cn edges are generated randomly using uniform distribution over all possible graphs with cn edges. Each node is thus expected to connect to 2c other nodes on average. The pseudo-code for the random graph generation is shown in the following algorithm.

4) Algorithm to generate (G, n, c)graphs: The pseudo code for generating G (n, m) graphs as follows

```

initialize graph G(V, E)
m ← n * c
for i ← 1 to m
repeat
e ← random edge
until e not present in E
E ← E ∪ {e}
return (G).

```

V. EXPERIMENTAL RESULTS AND ANALYSIS

All the procedures of SRA have been coded in C++ language. The experiments were carried out on an Intel Pentium Core2 Duo 1.6 GHz CPU and 1 GB of RAM. The effectiveness of the SRA heuristic was evaluated using 117 instances. These instances are divided into 3 sets as shown in the TABLE I. Simulations are carried out on three types of graphs: the randomly generated small size, moderate and large scale graphs for the maximum weighted independent set problem.

TABLE I
 MWIS INSTANCES

| Problem set | No. of Instances | Range of Weights | Graph Model | Optimal Solution |
|-------------|------------------|------------------|-------------|------------------|
| 1 | 44 | [1,40] | G(n, p) | Known |
| 2 | 53 | [1, 10] | DIMACS | Unknown |
| 3 | 20 | [1, 100] | G(n, m) | Unknown |

A. Experiment 1

We first tested the SRA on 20 random graphs generated based on the concept explained in Section IV(1). The weight w(i) on vertex i was randomly selected in the range [1 - 40], $1 \leq i \leq n$. The result we recorded for each test graph and their information are shown in the TABLE II, W_S is the total weight of the independent set found by corresponding algorithm and $\alpha_v(G)$ represents the cardinality of the maximum weighted

independent set. The results are compared with Ostergard[14] method. From the TABLE II, we can see that the SRA approach delivers the optimal solutions to the most of the MWIS test instances and the quality of the solution of the proposed algorithm is much better than that of Ostergard method. We are interested to test the proposed algorithm on high density and large size (number of vertices) random graphs. For these large size random graphs we have chosen the same range of weights and the result we recorded are shown in the TABLE III. The time taken (in seconds) to find the MWIS for each of the instances are reported, in which the maximum time taken of 28 sec. (1000; 0.9) is an encouraging one and it is comparatively very less time for the graph of high density and large number of vertices.

TABLE II
 SIMULATION RESULTS FOR 1ST SET OF INSTANCES

| Graph | | Ostergard | | SRA | | Optimum | |
|-------|------|---------------|-------|---------------|-------|---------------|-------|
| V | p | $\alpha_v(G)$ | W_S | $\alpha_v(G)$ | W_S | $\alpha_v(G)$ | W_S |
| 20 | 0.85 | 5 | 166.7 | 5 | 166.7 | 5 | 166.7 |
| 25 | 0.85 | 8 | 294.4 | 8 | 294.4 | 8 | 294.4 |
| 30 | 0.85 | 7 | 228.3 | 7 | 228.3 | 7 | 228.3 |
| 40 | 0.85 | 9 | 278.9 | 9 | 278.9 | 9 | 278.9 |
| 45 | 0.9 | 9 | 236.3 | 9 | 236.3 | 9 | 236.3 |
| 45 | 0.85 | 6 | 159.2 | 6 | 159.2 | 6 | 159.2 |
| 50 | 0.9 | 6 | 143.2 | 6 | 143.2 | 6 | 143.2 |
| 50 | 0.85 | 4 | 100.1 | 4 | 100.1 | 4 | 100.1 |
| 55 | 0.9 | 10 | 244.5 | 10 | 244.5 | 10 | 244.5 |
| 55 | 0.85 | 7 | 168.5 | 7 | 168.5 | 7 | 168.5 |
| 60 | 0.9 | 11 | 269.4 | 11 | 269.4 | 11 | 269.4 |
| 60 | 0.85 | 8 | 190.2 | 8 | 193.9 | 8 | 193.9 |
| 65 | 0.9 | 7 | 157 | 7 | 157 | 7 | 157 |
| 65 | 0.85 | 5 | 113.8 | 5 | 113.8 | 5 | 113.8 |
| 70 | 0.9 | 7 | 147.9 | 7 | 147.9 | 7 | 147.9 |
| 70 | 0.85 | 5 | 107.7 | 4 | 97.3 | 5 | 107.7 |
| 75 | 0.9 | 5 | 116.4 | 7 | 146.8 | 7 | 146.8 |
| 75 | 0.85 | 6 | 122.4 | 6 | 130.4 | 6 | 130.4 |
| 80 | 0.9 | 10 | 193.4 | 10 | 217.2 | 10 | 217.2 |
| 80 | 0.85 | 7 | 157.3 | 8 | 172.8 | 8 | 177.8 |

B. Experiment 2

As there are no resulted benchmark set for the maximum weighted clique, to test the performance of SRA approach, further we have tested the proposed algorithm on benchmark graphs of maximum clique, which are made available by DIMACS [8], and this suite structured from the perspective of finding maximum cliques, so we considered the benchmark graphs as \overline{G} . These DIMACS instances were converted into weighted benchmark instances by allocating $w(i)$, for vertex i ($1 \leq i \leq n$), uniformly in the interval [1 - 10]. The result we recorded are shown in TABLE IV, in which the first two columns reports the name and size of the graphs; For the MWIS obtained by SRA, the third column W_C reports the total weight; the fourth column $\alpha_v(G)$ reports the cardinality; the fifth column A(C) gives the average vertex weight and sixth column delivers the the time (in sec.) taken to find the MWIS.

TABLE III
 SRA PERFORMANCE ON LARGE SIZE GRAPH OF 1 SET

| n | p | Proposed SRA | | |
|-----|-----|---------------|-------|---------|
| | | $\alpha_v(G)$ | W_S | Time(s) |
| 100 | 0.7 | 15 | 375 | <1 |
| | 0.8 | 20 | 476 | <1 |
| | 0.9 | 31 | 688 | <1 |
| | 150 | 0.8 | 23 | 580 |
| | | 0.9 | 37 | 858 |
| | | 0.95 | 54 | 1177 |
| | | 0.7 | 19 | 475 |
| | | 0.8 | 26 | 603 |
| | | 0.9 | 43 | 912 |
| | 200 | 0.7 | 21 | 529 |
| | | 0.8 | 29 | 754 |
| | | 0.9 | 51 | 1122 |
| | | 0.7 | 23 | 602 |
| | | 0.8 | 31 | 744 |
| 400 | 0.9 | 53 | 1219 | 4 |
| | 500 | 0.7 | 32 | 812 |
| | | 0.8 | 41 | 951 |
| | | 0.9 | 59 | 1298 |
| | | 0.7 | 46 | 1150 |
| | 700 | 0.8 | 52 | 1237 |
| | | 0.9 | 72 | 1620 |
| | | 0.7 | 83 | 2058 |
| | | 0.8 | 107 | 2504 |
| | | 0.9 | 112 | 2576 |
| | | 28 | | |

To check whether the SRA reaches the optimum (best solution) for these maximum weighted clique instances, we took the experimental results reported in Bomze et al 2000, Babel 1994 and Pullan 2008 and we analyzed the percentage of deviation of these heuristics with the proposed algorithm. i.e., for some of these instances of same condition, we compared the performance of SRA with other heuristics Babel[1], RD[5] and PLS[17, 18]. These results are reported in TABLE V and from the last three columns of TABLE V, we can see that positive values tells us that SRA reaches the best optimum solution and negative values represents the SRA fails to reach the optimum solution than the other heuristics compared. If the values are exactly equal to zero then SRA and compared heuristics reaches the same optimum. Out of 23 compared instances, for the instances c-fat500-1 and p_hat1500-3, SRA fails to reach the optimum than PLS and SRA reaches the same optimum with PLS in brock800.1, brock800.2 and p_hat1500-1 instances and with Babel in c-fat500-1 instance and for the remaining instances the SRA reaches the best solution.

In order to assess the amount of deviation of other heuristics from SRA, further we obtained the statistical quantities of last three columns values of TABLE V and obtained results are shown in TABLE VI. It shows that Babel heuristic deviated highly and PLS gets low deviation from SRA. From these results shown in TABLES V and VI, we can see that the quality of the solution delivered by SRA is much better than the other heuristics, involved in this experiment.

TABLE IV
SIMULATION RESULTS FOR THE WEIGHTED DIMACS INSTANCES

| $\overline{G}(V, \overline{E})$ | V | W_C | $\alpha_v(G)$ | A(C) | Time(s) |
|---------------------------------|------|-------|---------------|------|---------|
| brock200_1 | 200 | 150 | 18 | 8.32 | <1 |
| brock400_1 | 400 | 211 | 23 | 9.17 | <1 |
| brock400_2 | 400 | 210 | 23 | 9.13 | <1 |
| brock400_3 | 400 | 237 | 27 | 8.77 | <1 |
| brock400_4 | 400 | 237 | 28 | 8.46 | <1 |
| brock800_1 | 800 | 163 | 18 | 9.05 | 2 |
| brock800_2 | 800 | 163 | 19 | 8.57 | 2 |
| brock800_3 | 800 | 165 | 19 | 8.68 | 5 |
| brock800_4 | 800 | 172 | 20 | 8.6 | 4 |
| c-fat200-1 | 200 | 86 | 12 | 7.16 | <1 |
| c-fat200-2 | 200 | 156 | 22 | 7.09 | <1 |
| c-fat200-5 | 200 | 384 | 56 | 6.86 | <1 |
| c-fat500-1 | 500 | 96 | 14 | 6.86 | <1 |
| c-fat500-2 | 500 | 186 | 24 | 7.75 | <1 |
| c-fat500-5 | 500 | 423 | 64 | 6.6 | <1 |
| Hamming6-2 | 64 | 211 | 31 | 6.8 | <1 |
| Hamming6-4 | 64 | 29 | 4 | 7.25 | <1 |
| Hamming8-2 | 256 | 718 | 126 | 5.69 | 3 |
| Hamming8-4 | 256 | 102 | 16 | 6.37 | 2 |
| Hamming10-2 | 1024 | 2906 | 512 | 5.67 | 9 |
| Hamming10-4 | 1024 | 316 | 40 | 7.9 | 23 |
| Johnson8-2-4 | 28 | 29 | 4 | 7.25 | <1 |
| Johnson8-4-4 | 70 | 86 | 12 | 7.16 | <1 |
| Johnson16-2-4 | 120 | 51 | 8 | 6.37 | <1 |
| Johnson32-2-4 | 496 | 96 | 15 | 6.4 | 6 |
| MANN_a9 | 45 | 100 | 14 | 7.14 | <1 |
| MANN_a27 | 378 | 786 | 121 | 6.5 | 12 |
| MANN_a45 | 1035 | 2658 | 338 | 7.86 | 33 |
| p_hat300-1 | 300 | 61 | 8 | 7.63 | <1 |
| p_hat300-2 | 300 | 175 | 23 | 7.6 | <1 |
| p_hat300-3 | 300 | 218 | 33 | 6.6 | <1 |
| p_hat500-1 | 500 | 78 | 9 | 8.66 | 2 |
| p_hat500-2 | 500 | 247 | 35 | 7.05 | 5 |
| p_hat500-3 | 500 | 342 | 46 | 7.43 | 3 |
| p_hat700-1 | 700 | 80 | 9 | 8.88 | <1 |
| p_hat700-2 | 700 | 288 | 43 | 6.69 | 15 |
| p_hat700-3 | 700 | 405 | 59 | 6.86 | 18 |
| p_hat1000-1 | 1000 | 87 | 9 | 9.66 | 8 |
| p_hat1000-2 | 1000 | 297 | 43 | 6.9 | 23 |
| p_hat1000-3 | 1000 | 436 | 63 | 6.92 | 30 |
| p_hat1500-1 | 1500 | 91 | 10 | 9.1 | 23 |
| p_hat1500-2 | 1500 | 431 | 58 | 7.43 | 26 |
| p_hat1500-3 | 1500 | 583 | 88 | 6.62 | 24 |
| san200-0.7.1 | 200 | 219 | 27 | 8.11 | <1 |
| san200-0.7.2 | 200 | 125 | 16 | 7.82 | <1 |
| san200-0.9.1 | 200 | 483 | 67 | 7.21 | 5 |
| san400-0.7.1 | 400 | 275 | 38 | 7.23 | <1 |
| san400-0.7.2 | 400 | 207 | 29 | 7.14 | <1 |
| san400-0.9.1 | 400 | 643 | 96 | 6.7 | 8 |
| san1000 | 1000 | 95 | 10 | 9.5 | <1 |
| sanr200-0.7 | 200 | 101 | 18 | 5.61 | <1 |
| sanr200-0.9 | 200 | 257 | 41 | 6.27 | <1 |
| sanr400-0.5 | 400 | 75 | 13 | 5.77 | <1 |
| sanr400-0.7 | 400 | 143 | 20 | 7.15 | <1 |

C. Experiment 3

In this experiment the parameter set opted like small-large scale problems, that is V varied from 50 to 1000. The weight w(i) on vertex i was also randomly drawn from the interval [1 - 100]. Here we used the G(n, m) graph model to generate the random graphs. For most of the test instances the optimal solutions are unknown, we obtained the time (in sec.) taken by the SRA for finding the MWIS of the graph. These results are shown in the Fig. 2 where the major axis represents the size (in terms of number of vertices) of the 20 test instance's and for each test instances the time taken by SRA were plotted as points and for each instances their points are linked by a line. It is clear from the Fig. 2 that the time taken by the SRA to find the optimum value of each of the MWIS instances increases steadily when the size of the problem increases and the maximum time taken is 12.31 sec. With this figure we show that the proposed algorithm took very less time to produce a maximum weighted clique for each of the test instances.

TABLE V
% OF DEVIATION OF OTHER HEURISTICS FROM SRA

| Name | RD | Babel | PLS | % of deviation from SRA | | |
|-------------|-------|-------|-------|-------------------------|-------|-------|
| | W_C | W_C | W_C | RD | Babel | PLS |
| brock800_1 | 136 | 156 | 163 | 16.56 | 4.29 | 0 |
| brock800_2 | 142 | 157 | 163 | 12.88 | 3.68 | 0 |
| brock800_3 | 141 | 148 | 162 | 14.54 | 10.30 | 1.82 |
| brock800_4 | 136 | 147 | 161 | 20.93 | 14.53 | 6.40 |
| c-fat500-1 | 93 | 96 | 97 | 3.13 | 0 | -1.04 |
| c-fat500-2 | 181 | 181 | 168 | 2.69 | 2.68 | 9.67 |
| c-fat500-5 | 371 | 395 | 418 | 12.29 | 6.62 | 1.18 |
| hamming10-2 | 2674 | 1923 | 2853 | 7.98 | 33.83 | 1.82 |
| hamming10-4 | 258 | 277 | 306 | 18.35 | 12.34 | 3.16 |
| MANN_a45 | 2643 | 2653 | 2133 | 0.56 | 0.19 | 19.75 |
| san1000 | 80 | 94 | 90 | 15.78 | 1.05 | 5.26 |
| p_hat1000-1 | 77 | 80 | 82 | 11.49 | 8.04 | 5.75 |
| p_hat1000-2 | 285 | 284 | 293 | 4.04 | 4.37 | 1.35 |
| p_hat1000-3 | 435 | 349 | 407 | 0.22 | 19.95 | 6.65 |
| p_hat1500-1 | 74 | 87 | 91 | 18.68 | 4.39 | 0 |
| p_hat1500-2 | 327 | 309 | 427 | 24.12 | 28.30 | 0.93 |
| p_hat1500-3 | 490 | 390 | 586 | 15.95 | 33.10 | -0.51 |
| p_hat500-1 | 56 | 72 | 75 | 28.20 | 7.69 | 3.84 |
| p_hat500-2 | 213 | 228 | 243 | 13.76 | 7.69 | 1.62 |
| p_hat500-3 | 307 | 294 | 338 | 10.23 | 14.03 | 1.17 |
| p_hat700-1 | 54 | 76 | 76 | 32.5 | 5.00 | 5 |
| p_hat700-2 | 275 | 272 | 283 | 4.51 | 5.55 | 1.74 |
| p_hat700-3 | 371 | 316 | 397 | 8.39 | 21.97 | 1.97 |

TABLE VI
STATISTICAL VALUES OF % OF DEVIATION

| Algorithms | Min. | Median | Average | Max. | Std. Dev |
|------------|-------|--------|---------|-------|----------|
| RD | 0.23 | 12.88 | 12.95 | 32.5 | 8.58 |
| Babel | 0 | 7.69 | 10.85 | 33.83 | 10.09 |
| PLS | -1.04 | 1.81 | 3.37 | 19.75 | 4.47 |

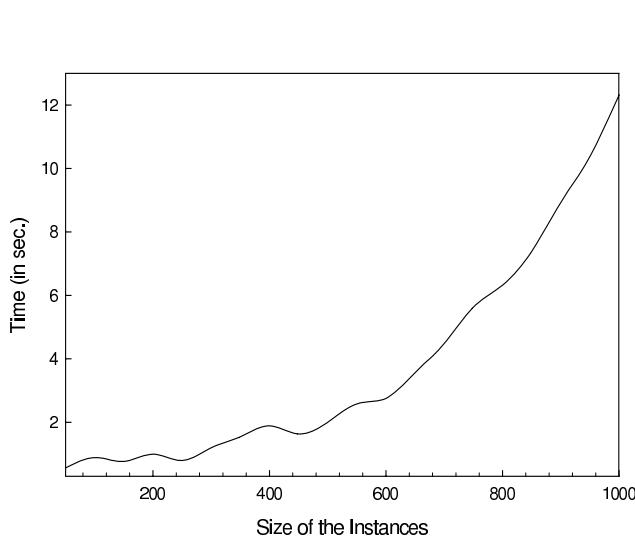


Fig. 2. Time taken (in sec.) by SRA for MWIS instances of $G(n, m)$ graphs

VI. CONCLUSION

A new SRA for MWIS in a graph using MWVC has been proposed and its effectiveness has been shown by simulation experiments. The terminology support of a vertex introduced in the new model, with that, the new model can find the maximum weighted independent set effectively. Experimental result shows that this approach greatly reduce the execution time and in addition, the simulation results show that the new SRA can yield better solutions than Babel, Ostergard, RD and PLS heuristics found in the literature. At the same time, our approach gives best solutions for DIMACS weighted instances and also for random graphs. The proposed algorithm has led to give near optimal solutions for most of the test instances where we know the optimal solutions. Furthermore attractiveness of this heuristic is its outstanding performance for solving MWIS.

REFERENCES

- [1] L. Babel: *A fast algorithm for the maximum weight clique problem*, Computing, (1994), Vol. 52, pp. 31-38.
- [2] S.Balaji, V. Swaminathan & K. Kannan: *An Effective Algorithm for Minimum Weighted Vertex Cover problem*, to be appear in International Journal of Computational and Mathematical Sciences, Vol.4 (1) (2010), pp. 34-38.
- [3] B. Bollobas: *Random graphs* (2nd Ed.). Cambridge, UK: Cambridge University press (2001)
- [4] I. Bomze, M. Budinich, M. Pellioli & Rossi: *Annealed Replication: A new heuristic for the maximum clique problem*, Discrete Applied Mathematics, (2002), Vol. 121, pp. 27-49.
- [5] I. Bomze, M. Pellioli & V. Stix: *Approximating the maximum weight clique using replicator dynamics*, IEEE Transactions Neural Network, (2000), vol. 11.
- [6] P. Crescenzi, C. Fiorini & R. Silvestri: *A note on the approximation of the max. clique problem*, Information Processing Letters, (1991), vol. 40, No.1, pp. 1-5.
- [7] S. De Vries & R. Vohra: *Combinatorial auctions: a survey*, INFORMS Journal on Computing (2003), vol. 15, pp. 284-309.
- [8] *DIMACS clique benchmarks*, Benchmark instances made available by electronic transfer at dimacs.rutgers.edu, Rutgers Univ., Piscataway, NJ. (1993).
- [9] T. Etzion & P. R. J. stergrd: *Greedy and heuristic algorithms for codes and colorings*, IEEE Transactions on Information Theory, (1998), vol. 44, pp. 382-388.
- [10] U. Feige, S. Goldwasser, L. Lovasz, S. Safra & M. Szegedy: *Approximating clique is almost NP-complete*, Proceedings of the 32nd IEEE Annual symposium on Foundations of computer science, San Juan, Puerto Rico, (1991), pp. 2 -12.
- [11] E. J. Gardiner, P. J. Artymink & P. Willett: *Clique detection algorithms for matching three-dimensional structures*, Journal of Molecular Graphics and Modeling, (1998), vol. 15, pp. 245-253.
- [12] M. R. Garey, D. S. Johnson: *Computers and Intractability: A Guide to the theory NP - completeness*, San Francisco: Freeman (1979).
- [13] A. Mehrotra & M. A. Trick: *A Column generation approach for graph coloring*, INFORMS Journal on Computing (1996), vol. 8, pp. 344-354.
- [14] P R J Ostergard: *A New Algorithm for the Maximum Weight Clique problem*, Nordic Journal of Computing, (2001), vol. 8, pp 424-436.
- [15] P. Pardolos & J. Xue: *The maximum clique problem*, Journal of Global optimization, (1994), vol.4, pp. 301-328.
- [16] P. A. Pevzner & S. H. Sze: *Combinatorial approaches to finding subtle signals in DNA sequences*, Proceedings of Eighth International Conference on intelligent systems for Molecular Biology, AAAI Press, (2000), pp.269-278.
- [17] W. Pullan: *Approximating the maximum vertex/edge weighted clique using local search*, Journal of Heuristics, (2008), vol. 14, pp. 117-134.
- [18] W. Pullan: *Optimisation of unweighted/weighted maximum independent sets and minimum vertex covers*, Discrete Optimization, (2009), vol. 6, pp. 214-219.
- [19] M. Weight & A. K. Hartmann: *Minimal vertex covers on finite-connectivity random graphs - A hard-sphere lattice-gas picture*, Phys. Rev. E, 63, 056127.