FINAL COURSE PROJECT REPROT – ECE 650

Md Ismail Alam Mokul

Introduction

A vertex cover of an undirected graph $G = < V, E >$ is a subset $V' \in V$ such that if (u, v) is an edge of G, then either $u \in V'$ or $u \in V'$ $(or\ both)$. Vertex cover set covers all the edges of the graph. The size of the vertex cover is the number of vertices in it [1]. The vertex cover problem is to find a vertex cover of a minimum size that covers all vertices in a given graph. Vertex cover decision problem is $NP - Complete$.

In this project, three different algorithms were tested to find out the efficiency of the algorithm in terms of running-time and approximation ratio. The three algorithms are:

a) REDUCTION-TO-CNF-SAT. Reduction to CNF SAT and solving with SAT Solver
b) APPROX-VC-1. Picking up a vertex in the order of highest degree and adding that to vertex cover and removing away all edges incident on that. Repeat till no edges remain.
c) APPROX-VC-2. Picking up an edge $< u, v >$ and adding the pair vertices in the vertex cover subset. Removing all edges attached to them and repeat till no edges remain.

This report analyzes the running time and approximation ratio in solving the vertex cover problem using the above three algorithms. In the subsequent paragraphs, the project outline and methods, analysis of the results and their evaluation are described.

Project Outline and Methods

According to the project instruction, graphGen on eceubuntu was used to generate graphs. But, due to the long run time of REDUCTION-TO-CNF-SAT and paucity of time, a total of 20 graphs for REDUCTION-TO-CNF-SAT were generated with 4 variations of vertices number, $|V| \in \{5, 20\}$ in increments of 5. Computation time was recorded for all three algorithms: REDUCTION-TO-CNF-SAT, APPROX-VC1, and APPROXVC2. Microsoft Excel is used for results compilation and chart generation.

Three vectors, vector<unsigned>cnfsat, vector<unsigned> approxVC1, vector<unsigned> approxVC2, were used for data point for algorithms running time. For the calculation of the approximation ratio, REDUCTION-TO-CNF-SAT is taken as the optimal solution as it is guaranteed to yield the optimal solution. Following two formulas were used for calculation of approximation ratios:

$$Approx\ ratio\ 1 = \frac{Size\ of\ vertex\ cover\ of approxvc1}{Size\ of\ vertex\ cover\ of\ reductioncnfsat}$$
$$Approx\ ratio\ 2 = \frac{Size\ of\ vertex\ cover\ of\ approxvc2}{Size\ of\ vertex\ cover\ of\ reductioncnfsat}$$

Results and Analysis

Run time for REDUCTION-TO-CNF-SAT is given in Figure 1. The runtime for this algorithm increased exponentially with the size of the input, i.e. the number of vertices. As the running-time for CNF satisfiability problem increases exponentially with the increase of the number of clauses, the run time found in the experiment increased exponentially.

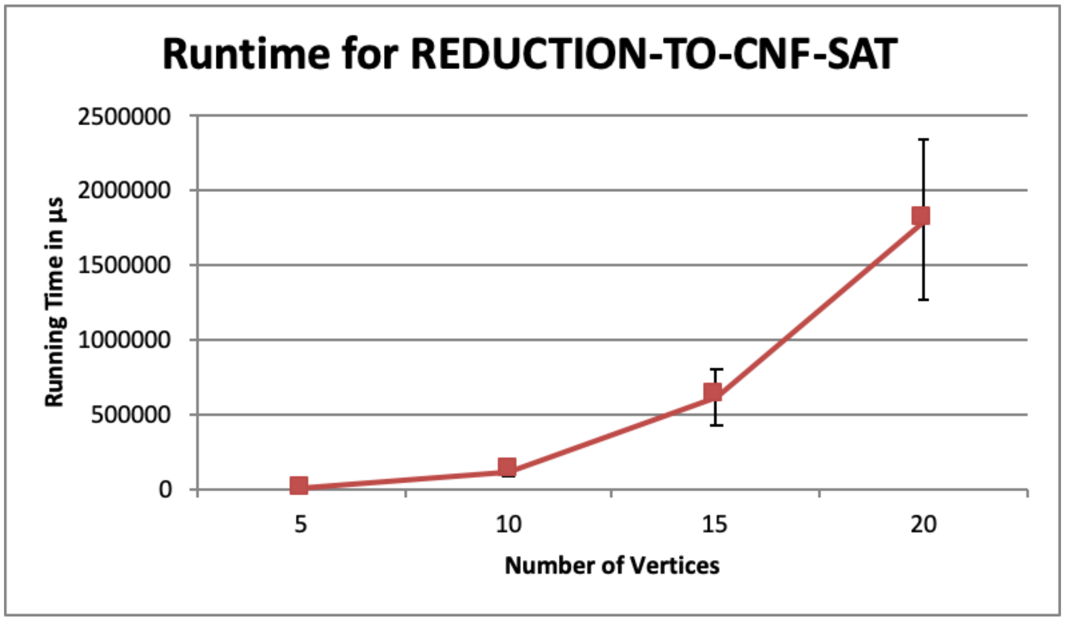

Figure 1: Run time of REDUCTION-TO-CNF-SAT for 5, 10,15 and 20 vertices.

The comparison of running time for APPROX-VC1 and APPROXVC2 is given in Figure 2. For both the algorithms, the runtime grows linearly with the increase of the input size. But, it is observed that APPROX-VC1 took less run time than APPROXVC2. This is because of the way the algorithm solves the vertex cover problem. In the case of APPROX-VC1, it picks up the vertices with the highest degree and throws away all edges incident on that vertex. On the other hand, APPROXVC2 chooses an edge randomly with 'dev/random/' function and throw away all edges attached to the vertices. Thus, APPROX-VC1 takes much less time for a small increment of the size of input vertices.

Figure 2 also shows the variation of standard deviation for APPROX-VC1 and APPROXVC2 algorithms. Due to the randomness of the APPROXVC2 algorithm, the run time varies significantly. But, for APPROX-VC1, the picking up of the highest degree vertex is deterministic. Therefore, the standard deviation of the running-time for APPROXVC2 is significantly larger than that of APPROX-VC1.

.

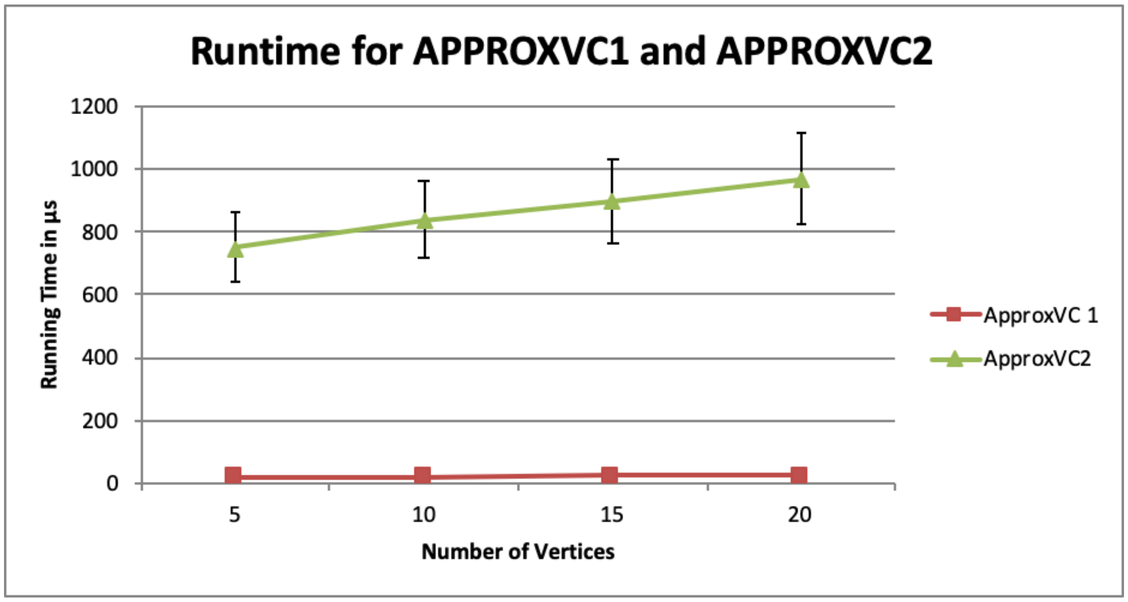

Figure 2: Comparison of run time of APPROX-VC1 and APPROXVC2 for 5, 10,15 and 20 vertices.

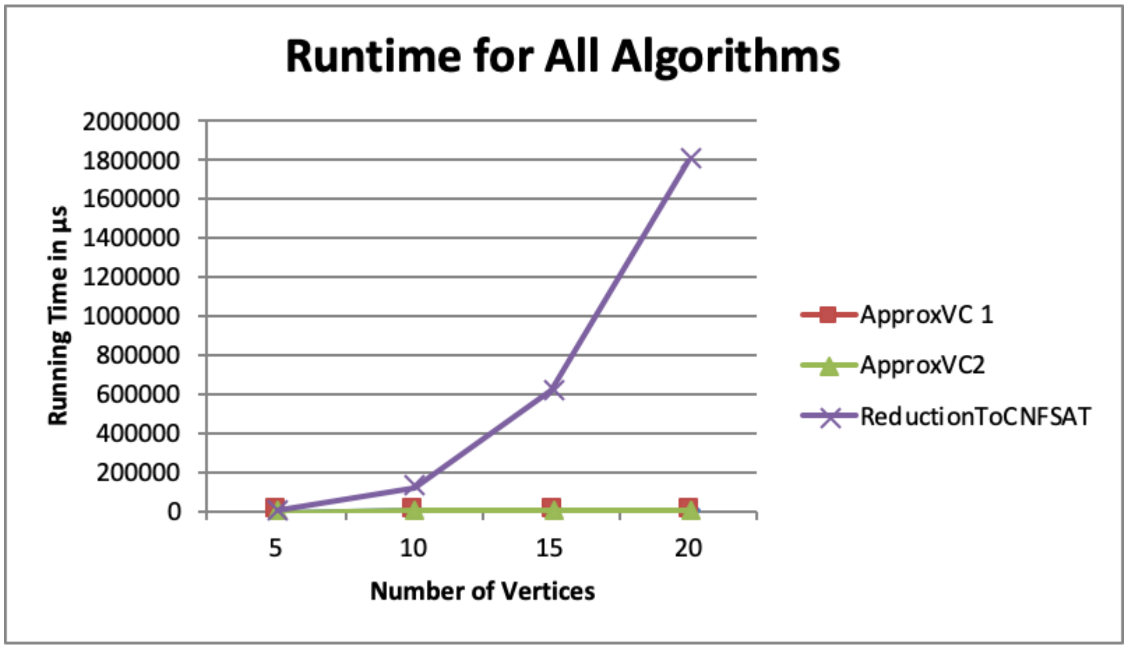

Figure 3: Comparison of run time of REDUCTION-TO-CNF-SAT , APPROX-VC1 and APPROXVC2 for 5, 10,15 and 20 vertices.

A comparison of all three algorithms running-time is given in Figure 3. As the figure shows, the run time for REDUCTION-TO-CNF-SAT for a small number of vertices such as 5 is closer to the running-time of other algorithms. But, as the size of input i.e. the number of vertices was increased, there was an exponential growth for the REDUCTION-TO-CNF-SAT algorithm.

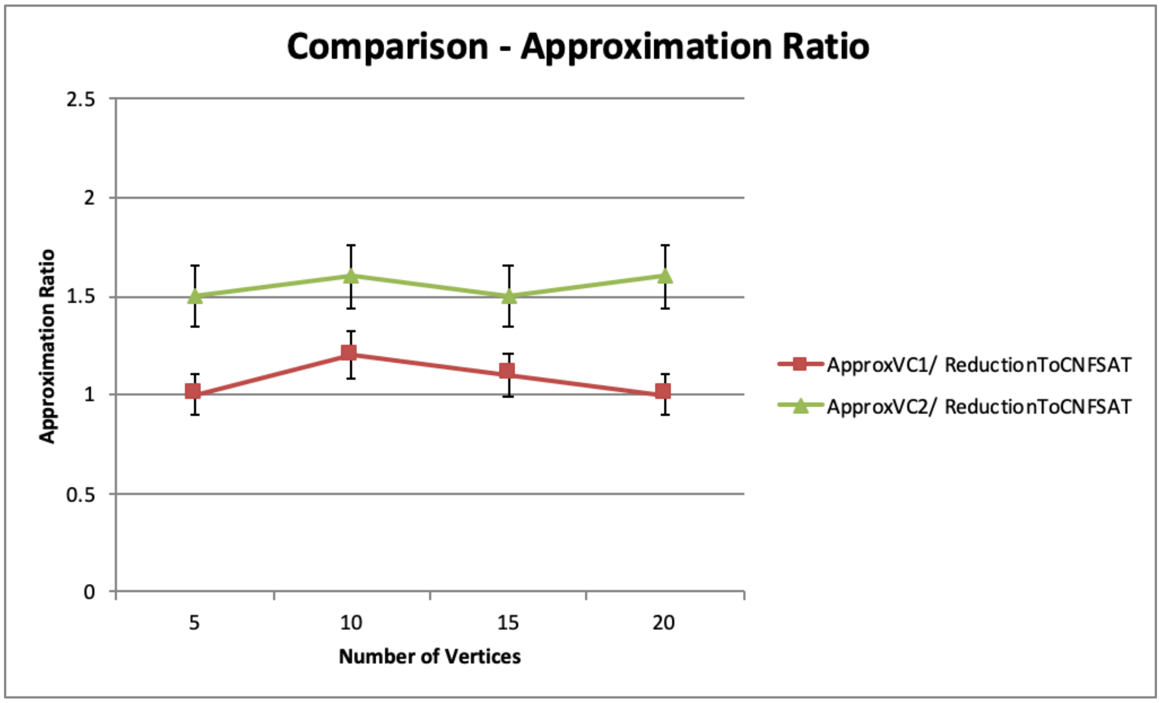

Figure 4: Approximation ratio, keeping REDUCTION-TO-CNF-SAT result to be guaranteed optimal solution, for APPROX-VC1 and APPROXVC2 algorithms for 5, 10,15 and 20 vertices.

The approximation ratio, calculated from the REDUCTION-TO-CNF-SAT algorithm, for both APPROX-VC1 and APPROXVC2 algorithms is given in Figure 4. For a small number of vertices, $|V| \in \{5, 20\}$, it was observed that the approximation ratio for APPROX-VC1 is closer to 1. This implies that picking up the vertex with the highest degree ensures a better chance of getting the optimum solution. On the other hand, due to randomness, the approximation ratio for APPROX-VC2 is near 1.5. The standard deviation for the APPROX-VC2 approximation ratio is higher than the APPROX-VC1 approximation ratio, again, due to the randomness of choosing vertices.

Discussions and Conclusions

Vertex cover problem is an NP-complete problem. With a higher size of the input, the time for finding the optimum solution for the vertex cover problem increases exponentially. Due to the longer time taken by the REDUCTION-TO-CNF-SAT algorithm and paucity of time, the study could not test the algorithms for the higher size of input. From the study, it is evident that for a smaller number of input size alternative algorithms like APPROX-VC1 and APPROXVC2 can be more efficient. However, for the optimal solution, the REDUCTION-TO-CNF-SAT algorithm is guaranteed to yield the optimal result.

A close look at the running-time for all three algorithms shows that while REDUCTION-TO-CNF-SAT yields the guaranteed optimal solution, the running-time increases exponentially with the size of the input. APPROX-VC1 was deterministic to choose the highest degree vertex at all times; thus its running-time with the increase of input size, for a small range in $|V| \in \{5, 20\}$, was linear and lower than the APPROX-VC2. The variation in the result, represented by standard deviation, also shows that APPROX-VC2 has a higher variant due to randomness in picking up the vertices.

Finally, the APPROX-VC1 algorithm is suited when the trade-off is between optimal solution and running-time. For a small number of vertices, it is always better to use the REDUCTION-TO-CNF-SAT algorithm for its guaranteed optimal solution.

Reference:

[1] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to Algorithms*, 3rd Editio. Cambridge, MA: The MIT Press, 2009.