University of Waterloo

Faculty of Engineering

Department of Electrical and Computer Engineering

ECE 650

Analyzing Performance of Different Approaches for Vertex Cover Problem

Instructor

Prof. Arie Gurfinkel

Prepared By

Chanuka Abeysinghe - 20834096
watiam: kmcrbabe

Sagar Purohit - 20811329
watiam: s2purohi

Dec 2019

1 ABSTRACT

A vertex cover of a graph is a subset of vertices which can represent all the edges of a particular graph. Minimum vertex cover is the smallest set of vertices which satisfy the vertex cover properties.Finding a minimum vertex cover is a classical problem in computer science which requires optimization algorithms. This problem is considered as a NP-hard problem and requires approximation algorithm. In this project we have used 3 approaches to find minimum cover of a graph and analyse the performance of each approach by the means of time and approximation ratio.

2 INTRODUCTION

In this report we have considered three methods to generate minimum vertex cover for a given graph. We have used multi-threaded approach to find these approximations of vertex cover in our program, where one thread is for I/O and one each for the different approaches to solve the minimum vertex cover.

Following approximation algorithms are used to generate vertex cover

- **CNF-SAT solver (MiniSAT):** Boolean satisfiability problem (SAT) is the problem of deciding whether a formula in Boolean logic is satisfiable. A SAT solver can determine if it is possible to find assignments to boolean variables that would make a given expression true and if it does it can also show a set of assignments that make the expression true. This particular SAT solver uses conjunctive normal form (CNF) as the input.
- **APPROX-VC-1:** An algorithm called APPROX-VC-1 is introduced. In that, vertex of the highest degree(most incident edges) is selected and to the vertex cover, then all edges on that vertex is removed. Then select the highest degree vertex from the remaining vertices. Repeat this till no edges remain.
- **APPROX-VC-2:** Pick any edge, and add two vertices associated with that edge to vertex cover. Remove all edges attached to those vertices. Repeat till no edges remain.

Using these three methods, vertex covers generated for different graphs and efficiency is measured for each graph. "Efficiency" is characterized in one of two ways:

- **Running time:** Time taken by the thread to compute the vertex cover. We have used pthread_getcpuclockid() to measure running time.

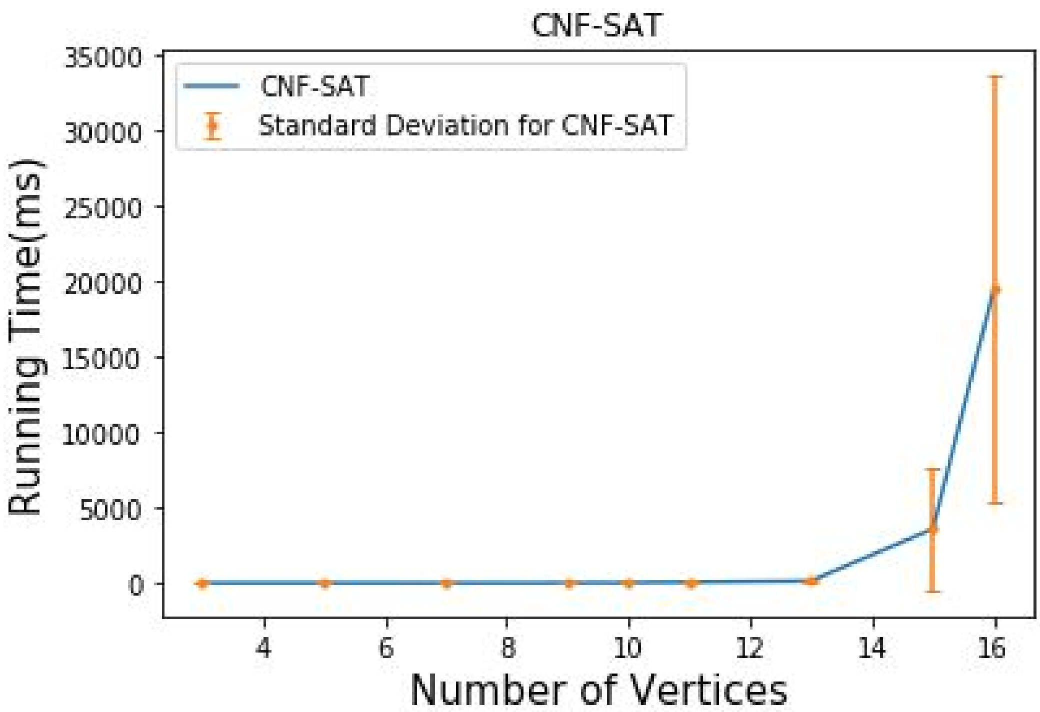
- **Approximation ratio** The ratio of the size of the computed vertex cover to the size of an optimal (minimum-sized) vertex cover. For measuring the approximation ratio, we have assumed CNF-SAT solver (MiniSAT) as the optimal vertex cover.

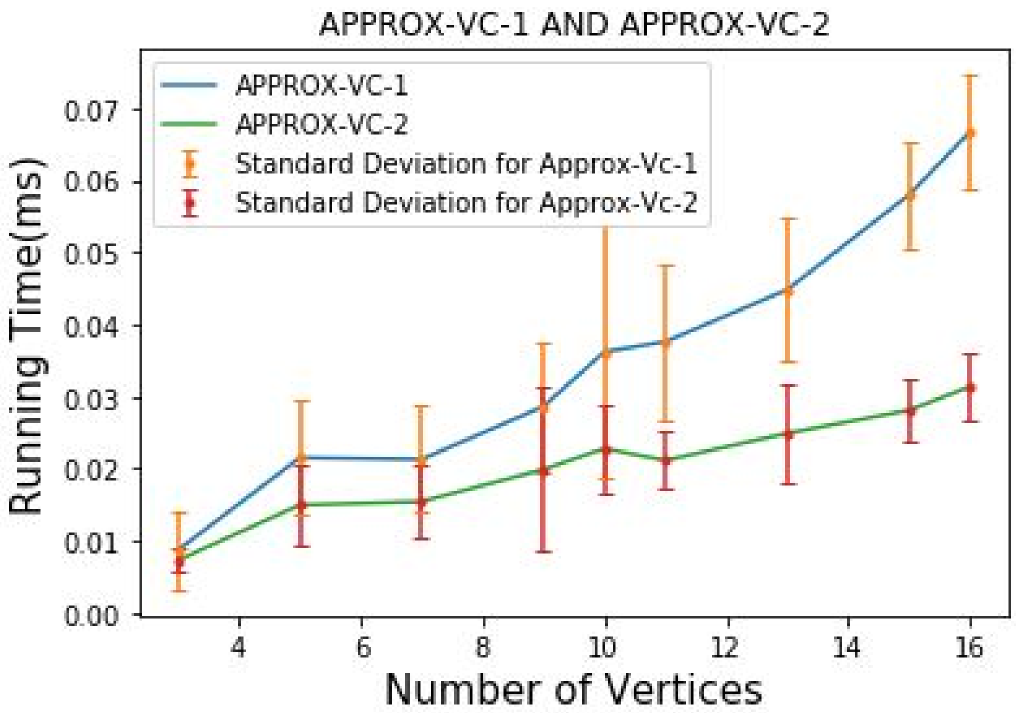
3 ANALYSIS

We have measured running time and approximation ratio for graphs with different number of vertices. We have started with 5 vertices and continue till 50 with increments of 5.
For a particular number of vertices (i.e 5) we have generated 10 graphs and measured running time and approximation ratio for each graph. Then we calculated mean and standard deviation of running time and approximation ratio. Then we plotted these properties against number of vertices.
Since there is a considerable difference between the running times of CNF-SAT and the other two algorithms, we have separated these into two graphs for the sake of illustration .

3.1 Running times

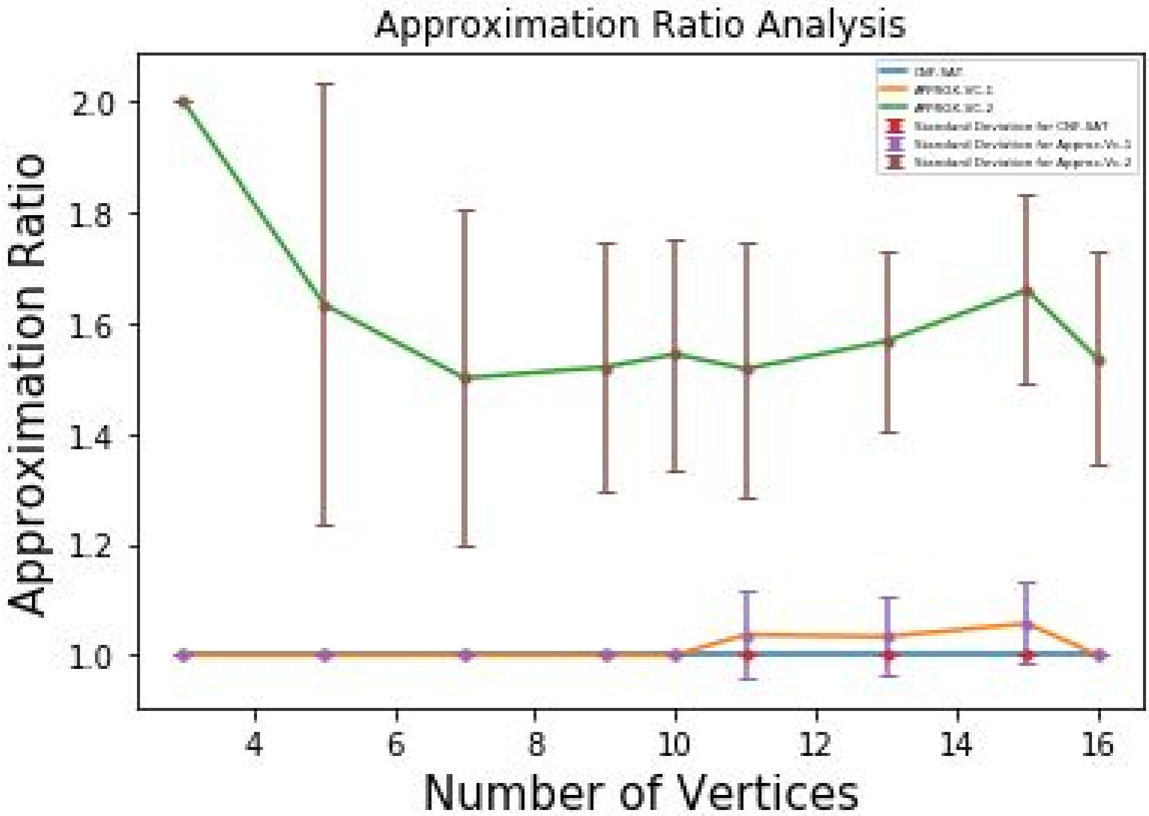
It is observed that average running time of all three algorithms increases with number of vertices.
As illustrated in below graphs SAT Solver has taken the maximum time to run among all the three approaches. If the number of vertices are low SAT solver tend to perform very well. However, when number of vertices was increasing running time increased dramatically. This is expected because with the number of vertices number of clauses are also increased. If there are v vertices then there will be k*v*(v-1)/2 clauses where k is the size of the vertex cover.More clauses means it has to satisfy large number of combinations and it degrades the performance with respect to running time.
Between Approximation-1 and Approximation-2, Approximation-1 always takes more time because in Approximation-1 it search for highest degree vector.


CNF-SAT
CNF-SAT
Standard Deviation for CNF-SAT
Running Time(ms)
35000
30000
25000
20000
15000
10000
5000
0
4
6
8
10
12
14
16
Number of Vertices


APPROX-VC-1 AND APPROX-VC-2
APPROX-VC-1
APPROX-VC-2
Standard Deviation for Approx-Vc-1
Standard Deviation for Approx-Vc-2
Running Time(ms)
0.07
0.06
0.05
0.04
0.03
0.02
0.01
0.00
4
6
8
10
12
14
16
Number of Vertices

3.2 Approximation ratio

In this part, analysis is done based on approximation ratio for the three algorithms. This criteria analyse the quality of output given by each method. Since these methods are approximation algorithms an error is associated with each method. Based on the this error portion quality of the output or how well each method can approximate the vertex cover will be decided.



According to above graph it is evident that although APPROX-VC-2 took less time it performed very poorly in case of Approximation Ratio. This is because it is choosing vertices without considering the degree. This introduces an error and most of the time it costs the minimum vertex cover. So, it has more chances to calculate bigger vertex cover than APPROX-VC-1. The result of CNF-SAT, however, can be regarded as the minimum sized vertex cover. Most of the time, the ratio of APPROX-VC-1 is also 1, meaning APPROX-VC-1 is also produced a minimum vertex cover most of the time. However, approximation ratio of APPROX-VC-2 is much higher than APPROX-VC-1. So, the probability of getting a correct minimum vertex cover for APPROX-VC-1 is much higher than APPROX-VC-2.

4 CONCLUSION

In this report we have studied three different approaches to get the minimum vertex cover of a graph. We have measured the efficiency of each approach by the means of running time and approximation ratio. It can be concluded from this analysis that the running time increases with the number of vertices irrespective of the algorithm used to calculate vertex cover.

If only running time is considered, then it can be concluded that total running time of CNF-SAT is considerably larger than that of the other two approaches. APPROX-VC-2 is the fastest of all three methods and it holds more significantly for bigger graphs. Thus APPROXVC-2 is the most effective way to compute the minimum vertex cover problem according to the running time analysis.

However, if considering the analysis with respect to approximation ratio, it can be safely concluded that APPROX-VC-2 performed poorly in generating minimum vertex cover. Hence, although the approximation algorithms are efficient in terms of time and space complexity, their results of approximations are not always guaranteed optimal, whereas CNF-SAT always gives the optimal result.

After analysing above mentioned algorithms we can conclude that time complexity is inversely related to finding optimal solution for studied approaches.