International Journal of Innovative Technology and Exploring Engineering (IJITEE)
ISSN: 2278-3075 (Online), Volume-3 Issue-11, April 2014

Analysis of Various Algorithms to Solve Vertex Cover Problem

Sangeeta Bansal, Ajay Rana

Abstract*— The vertex cover (VC) problem belongs to the class of Non Deterministic Polynomial time complete (NPC) graph theoretical problems, which plays a central role in theoretical computer science and it has a numerous real life applications. Since the problem is Non Deterministic Polynomial time complete (NPC) it is unlikely to find a polynomial-time algorithm for solving vertex-cover problem exactly. This paper analyses the various algorithms to find minimum vertex cover for standard classes of random graph. The performance of all algorithms is compared with the complexity and the output solution that of the approximation algorithm, clever greedy algorithm, branch-and-bound algorithm, and simple genetic algorithm (GA).

***Index Terms*— Minimum vertex cover, Branch and bound, greedy algorithm, genetic algorithm, crossover, mutation.**

I. INTRODUCTION

Vertex cover problem is a classical Non Deterministic Polynomial time complete (NPC) problem in computational complexity theory and is one of the Karp's 21 Non Deterministic Polynomial time complete (NPC) problems. The minimum number of vertex that covers all the edges of an undirected graph is an optimization problem generally referred to as Minimum Vertex Cover. It has been an area of interest for most of the researchers and practitioners because of the Non Deterministic Polynomial time completeness and because many difficult real-world problems can be expressed as instances of the minimum vertex cover. Areas where the minimum vertex cover can be applied are Engineering, Research, Mathematics, and Science.

II. VERTEX COVER

A **vertex cover** (V') of an undirected graph G = (V, E) where V and E are respectively vertex and edges is a subset V' ⊆ V in such way that if (u, v) is an edge of G, Then either u ∈ V' or v ∈ V' or both. Minimum vertex cover represents the minimum number of vertex required to cover all the edges of an undirected graph.

Vertex cover problem is Non Deterministic Polynomial time (NP) Complete

In order to show that the problem is Non Deterministic Polynomial time complete (NPC) we have the following steps.

1. Show that the problem is in Non Deterministic Polynomial time (NP),
2. Reduce a Non Deterministic Polynomial time complete (NPC) problem to it, and
3. Show that the reduction is a polynomial time function.

CIRCUIT-SAT is the first Non Deterministic Polynomial time complete (NPC) problem. All proofs follow by reduction from the Non Deterministic Polynomial time completeness of CIRCUIT-SAT

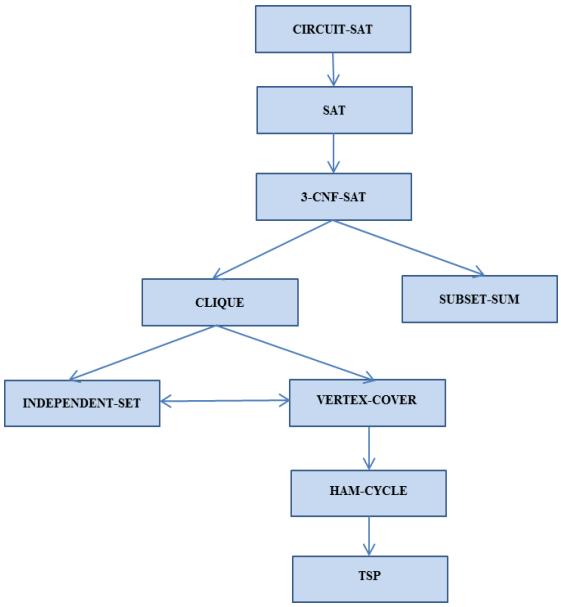

Figure 1: Structure of Non Deterministic Polynomial time complete (NPC)

VERTEX COVER is in Non Deterministic Polynomial time (NP) because we can verify any solution in polynomial time with a simple n^2 examination of all the edges for endpoint inclusion in the given vertex covers.

In order to prove that Vertex cover problem (X) is Non Deterministic Polynomial time complete (NPC)

1. We select a problem Y (CLIQUE/ INDEPENDENT-SET) that is already known to be in Non Deterministic Polynomial time complete (NPC).
2. We do polynomial time reduction from Y to X.
3. We prove that given an instance of Y, Y has a solution in case X has a solution.

III. ALGORITHMS TO FIND MINIMUM VERTEX COVER

Below are some algorithms to find Minimum Vertex Cover:

1. Approximation Algorithm for Vertex Cover[1]

Consider a graph G = (V, E), where V is the number of vertices and E is the number of edges. This algorithm finds a minimum subset S ⊆ V, such that S covers all edges in E, i.e., every edge E is incident to at least one vertex in S.

Manuscript published on 30 April 2014.
*Correspondence Author(s)
Sangeeta Bansal, Student M.Tech., Deptt. of Computer Science, Amity University, Noida (U.P.), India.
Dr. Ajay Rana, Director, Deptt. of Computer Science, Amity University, Noida (U.P.), India

APPROX-VERTEX-COVER

1. S← φ
2. While E ≠ φ
3. Pick any {u, v} ϵ E
4. S← S ∪ {u, v}
5. Delete all edges that incident on either of u or v.
6. Return S

In short, approximation algorithm picks any edge and adds the corresponding vertices to S. It also removes all the edges that are incident to either of the added vertex.

APPROX-VERTEX-COVER is a Polynomial time-2 approximation algorithm. It runs in O (V + E) time and loops until all edges have been removed returning a vertex cover that is twice the optimal cover. This is the major drawback of this algorithm.

2. Greedy Technique [1, 3]

A greedy algorithm recurrently performs the procedure that tries to maximize the return on the basis of examining local conditions, with the assumption that the result will be the desired result for the global problem. In some cases such a strategy is provides optimal solutions while in some other cases it just provides a compromise by providing acceptable approximations.

Clever greedy algorithm [6]

It is a type of Greedy Algorithm where the vertex having the maximum degree (that covers maximum number of edges) is selected and is added to the set of vertex cover.

1. C ← Ø
2. While E ≠ Ø
3. Pick a vertex v ∈V of maximum degree in the *current* graph
4. C ← C U {v}
5. E ← E \ {e ∈ E : v ∈ e}
6. Return C

The Clever greedy algorithm always achieves the ratio O (log n) for vertex cover.

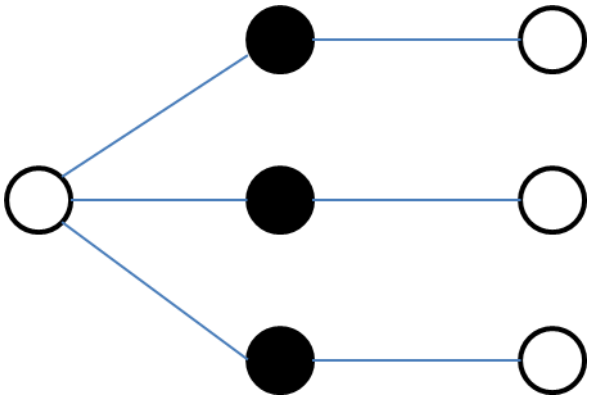

Figure 2: Optimal solution is the black vertices, but greedy would pick the four white vertices.

The drawback of this algorithm is that in some cases it will provide the optimal vertex cover while in some other cases it may not provide the optimal vertex cover.

3. Branch and Bound Algorithm[5]

Branch and Bound is an exact algorithm that incorporates the first heuristic to gain high efficiency. Without the heuristic, the algorithm would still remain exact, but it would run very slow.

The two basic strategies of a general Branch and Bound method are:

- **Branching**: splitting the problem into sub-problems
- **Bounding**: calculating lower and / or upper bounds for the objective function value of the sub-problem

The branching is performed in the algorithm by separating the current subspace into two parts using the integrity requirement. Using the bounds, unpromising sub-problems can be eliminated.

The basic idea of the method is to find vertex cover of minimum size. As each vertex becomes either covered or uncovered, there are 2N possible configurations which can be arranged as leaves of a binary configuration tree.

At each node, the two sub trees represent the sub-problems where the corresponding vertex is either covered (left sub-tree) or uncovered (right sub-tree). Vertices which have not been touched at a certain level of the tree are said to be free. The algorithm does not have to descend further into the tree when a cover has been found, i.e. when all edges are covered. After this, the search continues in higher levels of the tree for a cover which has possible smaller vertex cover, or the backtracking occurs.

The disadvantage of this algorithm is that in a large undirected graph, it runs slower.

4. Genetic Algorithm

Genetic Algorithms (GAs) are adaptive heuristic search algorithm based on the evolutionary ideas of natural selection and genetics inspired by Darwin's theory of evolution - "survival of the fittest". They represent an intelligent exploitation of a random search used to solve optimization problems.

A genetic algorithm usually starts with a randomly produced population of individuals. These individuals have possible solutions of problem that is being studied. Three genetic operators used are:

- *Selection:* Selection technique is required to choose the individual as per to their values, the selected individuals reproduce the next generation.
- *Crossover:* It means choosing a random position in the string and exchanging the segments either to the right or to the left of this point with another string partitioned similarly to produce two new off spring.
- *Mutation:* Mutation operator injects variations in the chromosomes by complementing every single bit of individual with a particular possibility of mutation.

Let G be an undirected graph represented as G = (V, E) where V is the set of vertices and E specifies the edges. VT contains vertices (which specify genes in the chromosomes) and is known as vertex table and ET is edge table. P1 and P2 are 2 parent chromosomes chosen for crossover.

P1 and P2 are generated at random by choosing vertices gradually such that all the edges are exposed. *V'* represents the vertex cover.

Algorithm for Heuristic Vertex Crossover (HVX) [8]

Begin

V' = { }

Create table VT and ET

VT = (F(v), N(v)), where F(v) is the frequency of the vertex v in P1 and P2, N(v) is the degree of vertex v in G, for ∀ v ∈P1 and ∀ v ∈ P2

Published By:
Blue Eyes Intelligence Engineering
and Sciences Publication (BEIESP)

ET = E (x, y) for ∀ E∈ G
While ET ≠ { } do
Select v1 ∈ VT such that N (v1) > N (v) for ∀ v ∈VT. If more than one vertex has same number of degree then select that vertex, whose frequency (F (v1)) is high. If still more than one vertex is candidate for selection then select any vertex randomly. Say v1
ET = ET-{E (x, y) : x = v1 or y = v1)}
$V' = V' \cup \{v1\}$
End while
Return V'
End
The heuristic vertex crossover (HVX) for minimum vertex cover problem works well and converges fast to optimal solution.
The drawback of this algorithm is that it is slower than local step of branch-and-bound. It also fails to obtain consistent results for specific type of regular graphs.

IV. CONCLUSION

Based on the study and analysis of various algorithms to solve the Vertex Cover problem, the below table provides a comparative study of the best algorithm to choose given a desired problem situation.

TABLE I
COMPARATIVE STUDY OF ALGORITHMS TO FIND MINIMUM VERTEX COVER

Algorithm type	Complexity	Remarks
Approximation	O (V + E)	This is a Polynomial time-2 approximation algorithm. It runs in O (V + E) time and loops until all edges have been removed returning a vertex cover that is twice the optimal cover.
Clever Greedy	O (logV)	In some cases it will provide the optimal vertex cover while in some other cases it may not provide the optimal vertex cover.
Branch and Bound	It grows exponentially fast with problem size for all values of c.	It is an exact algorithm to find the minimum vertex cover. However, in a large undirected graph, it runs slower.
Genetic Algorithm	Time Complexity measured by the overall number of candidate solutions examined until the optimum is found.	It fails to obtain consistent results for specific type of regular graphs. It gives better results when it is combined with local optimization technique.

REFERENCES

1. T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein. Introduction to Algorithms. McGraw-Hill, New York, 2nd edition, 2001.
2. Dorit S.Hochbaum. Approximation Algorithms for NP-hard problems. 2002.
3. X. Huang, J. Lai, and S. F. Jennings. Maximum common subgraph: Some upper bound and lower bound results. BMC Bioinformatics, 7(Suppl 4):S6, 2006, pp.80-94.
4. M.Pelikan. Hierarchical Bayesian optimization algorithm: Toward a new generation of evolutionary algorithms. Springer-Verlag, 2005, pp.102-160.
5. Alexandra K Hartmann and Martin Weigt. Statistical mechanics of the vertex-cover problem, j.phys. A; Math. Gen. 36(2003) 11069-11093.
6. K.Clarkson. A modification to the greedy algorithm for the vertex cover problem, IPL, vol 16:23-25,(1983).
7. R. Arakaki, and L. Lorena. A Constructive Genetic Algorithm for the Maximal Covering Location Problem, in Proceedings of Metaheuristics International Conference, 2001, pp 13-17.
8. Ketan Kotecha and Nilesh Gambhava. A hybrid genetic algorithm for Minimum Vertex-cover Problem, vol 2: pp 16-20.

Published By:
Blue Eyes Intelligence Engineering
and Sciences Publication (BEIESP)