# Vertex Cover ProblemA picture containing text, clock, watch Description automatically generated

The vertex cover problem consists of finding the minimum number of vertices in a graph (G) so that each edge within G has at least one vertex touching it. The vertex cover of the graph (right) is denoted in red, since all edges are connected to at least one of the selected vertices.

## Proofs:

Graphical user interface, text, application

Description automatically generatedPseudocode of decision variant:

This can be run within polynomial time, showing the decision version of the vertex cover problem is within NP.

Graphical user interface, text, application, email

Description automatically generatedFor the optimisation problem, the pseudocode is as follows:

### Recurrence relation:

A picture containing table

Description automatically generated

### Reducing the Clique problem:

To prove that it is NP-complete, the clique problem will be reduced to the vertex cover problem. The clique problem involves finding all subsets of vertices that are adjacent to each other, called cliques of size k. A clique of size k is a subgraph with k vertices.

Consider graph G with V vertices and E edges and graph G’. Graph G’ consists of all edges not in G. All cliques of size k within graph G. If vertex cover is NP-Complete, then finding if a clique of k exists in graph G is the same finding if there is a vertex cover of V– k for G’.

Bubble chart

Description automatically generated with low confidence

Assume there is a clique of size k in G and all the vertices within it assigned Vk - meaning Vk has k elements. An edge in G’ will be assigned (a, b). This shows that at least vertex a or b are in the graph V- Vk. If both were in Vk then the (a, b) would be in Vk and by extension G. However, this is impossible since no edges in G’ are in G and therefore Vk.

Given there is a vertex cover of G’ (V’c), assume it is size V – k. This means that is an edge of G’ is chosen at random it is not possible for both vertices outside V’c. All edges outside V’c would be within graph G and constitute Vk. This gives V’c a size of V – k.

Since the clique problem can be reduced to the vertex cover problem, the problem is within NP-Hard. The problem is within both NP and NP-hard showing it to be NP-complete

## Practicality:

Note there are cases that vertex cover can be solved within polynomial time or within a reasonable amount of time. These conditions include:

* Trees with dynamic programming
* if k ≤ log(n) with the above algorithm
* Bipartite Graphs (application of the maximum flow problem)

Given the current calculated value O(2n), assume that it took a computer 10 seconds to run with the input n = 100. Each incremental increase would result in the time doubling.

|  |  |
| --- | --- |
| n | Time |
| 100 | 10 |
| 101 | 20 |
| 102 | 40 |
| 103 | 80 |
| 104 | 160 |
| 105 | 320 |
| 106 | 640 |

If calculation took a week to compute, it would take two weeks to calculate something with just one more input.

If the estimate was incorrect and rather took 0.001 seconds to compute it would quickly increase to one second after 10 more inputs, 1048.576s after 20 and 1.267 \* 1027s (around 1.5 \* 1022 days) after 100.

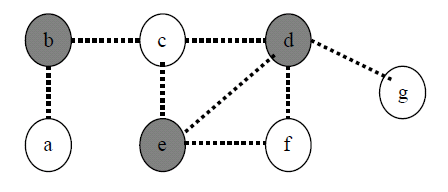
The speed of a computer is proportional to the number of transistors that one can fit in the smallest space possible. However, this has an upper limit. For transistors to act as a switch, they must be large enough to stop an electron. This means with no more decreases in size and increases in speed and the problem continually growing by a factor of two, the time it takes to computing an answer will increase beyond the capability of any traditional computer available.

## Approximate solution:

Graphical user interface, text, application

Description automatically generatedOne such approximate algorithm is Alom’s algorithm. The pseudocode is as follows:

The algorithm has a time complexity of O(E), but in worst case is O(n). Alom’s algorithm selects the vertex with the maximum number of edges. All edges connected to the vertex are discarded. If more than one vertex has the same maximum number of edges, the vertex that has at least one edge that is not covered by other vertices, which has maximum edge. This process is repeated until to cover all the vertices of the graph.

When applied to the graph (right), it gives the following vertex cover (highlighted in grey). It starts at vertex d since it has the most incident edges, continue to b and finish at e.

### Pros:

Produces an optimal solution if the cover has the vertex with the most incident edges. While most other algorithms have a large upper bound on the cost of their solution, Alom’s upper bound is small and can produce an optimal solution.

Time complexity is the same as other simpler algorithms. The most common algorithms involving random edge selection and deletion run also on O(E). However, these often produce solutions at least twice as costly as the optimal one.

Graphical user interface, text, application

Description automatically generatedCan be combined with following pseudocode to find increase the accuracy of the algorithm.

### Cons:

The optimal solution is not guaranteed. Only and exhaustive search can produce the most optimal.

For some larger graphs, an optimal is not found due to its structure and that the Vc of the graph doesn’t contain the vertex with the highest incident edges.

## Bibliography

Approximation Algorithms: Vertex Cover. (2005, February 21). Retrieved July 26, 2021, from Approximation Algorithms: Vertex Cover website: http://tandy.cs.illinois.edu/dartmouth-cs-approx.pdf

BruteForce. (2014, June 7). Retrieved July 23, 2021, from Yale.edu website: https://www.cs.yale.edu/homes/aspnes/pinewiki/BruteForce.html

‌

Dharwadker, A., & Pirzada, S. (2007). Applications of Graph Theory. Retrieved July 26, 2021, from www.dharwadker.org website: http://www.dharwadker.org/pirzada/applications/

‌

Dr. Garg, D. (2009). Choosing the Efficient Algorithm for Vertex Cover Problem (Thesis). Retrieved from https://www.gdeepak.com/thesisme/thesis-Choosing%20the%20Efficient%20Algorithm%20for%20Vertex%20Cover%20problem.pdf

‌

Panyam, K. (2018, August 3). Proof that vertex cover is NP complete. Retrieved July 23, 2021, from GeeksforGeeks website: https://www.geeksforgeeks.org/proof-that-vertex-cover-is-np-complete/

‌

Roughgarden, T. (2021). Smarter Search for Vertex Cover II - Week 2. Retrieved July 27, 2021, from Coursera website: https://www.coursera.org/lecture/algorithms-npcomplete/smarter-search-for-vertex-cover-ii-lPiFO

‌

Studies Studio. (2018, June 9). NP-Completeness of Vertex Cover with Proof || By Studies Studio. Retrieved July 25, 2021, from www.youtube.com website: https://www.youtube.com/watch?v=8exd1Oa4afI

‌

Vertex Cover | Brilliant Math & Science Wiki. (2016). Retrieved July 22, 2021, from brilliant.org website: https://brilliant.org/wiki/vertex-cover/

‌