Analysis of the Phase Transition in the Complexity of Solving Vertex Cover

Jose Torres Postigo

2023-12-06

1 Objectives

The objectives of this lab exercise is the implementation and evaluation of a backtracking algorithm for solving the Vertex Cover problem, and use it to find the critical complexity point.

This algorithm aim to resolve the question *"Does the Graph G have a vertex cover of size, at most, of positive integer k?"*. This problem is an NP-complete problem, so there is no general polynomial solution known.

For understanding better this algorithm, let $G(V,\ E)$ be an undirected graph. Consider:

- For any vertex $v \in V,\ sucG(v) = \{w \mid (v,\ w)\ \in E\}$, i.e., the vertices connected to v.
- For any partial solution $S \subseteq V$ and vertex $v \in V$,

$$unc_G\ (v,\ S) = \begin{cases} \emptyset & v\ \in\ S \\ suc_G\ (v)\ \setminus\ S & v\ \notin\ S \end{cases}$$

are the uncovered neighbors of v given S, i.e., for any $w \in unc_G\ (v,\ S),\ (v,\ w) \in E$ is an edge uncovered by S.

- For any partial solution $S \subseteq V$, S is a vertex cover if, and only if, $v \in V\ :\ unc_G(v,\ S) = \emptyset$.

The backtracking approach of this algorithm will follow some steps. Lets consider each search state is a pair $(S,\ i)$, i.e., the current partial solution and the index of the next vertex to be considered.

If $|S|\ >\ k$, the state is unfeasible and can be discarded. Otherwise, check if S is a vertex cover:

- If yes, a solution has been found.
- If not, check if $|S| < k$ and $i \leq |V|$.
  - If not, no further vertex can be added and the state can be discarded.
  - If yes, check if $unc_G(vi,\ S) = \emptyset$:
    * If yes, v_i is useless given S. Explore state $(S,\ i+1)$.
    * If not, explore two states: $(S \cup \{v_i\},\ i+1)$ and $(S \cup unc_G(v_i,\ S),\ i+1)$.

2 Experimental Setup

For these experiments, the algorithm will be run through a class given by the lecturer, which is going to run tests for it using graphs with increasing number of vertices in steps of 5, with vertex covers from 1 to the total number of vertices, generating 1,000 tests for each size. The initial value for the number of vertices is 20, and goes up to 40.

In all cases, graphs are generated following the Erdős-Renyi model with edge probability $\frac{2}{n}$, where n is the number of vertices.

Table 1: Computational environment considered.

CPU	Intel® Core™ i5 11600KF, 16GB
OS	Windows 11 Home 22H2
Java	openjdk 17.0.7 2023-04-18

3 Empirical Results

A summary of the experimental results is provided in Table 2 in the Appendix.

The graph of Figure 1 shows the number of nodes in the search tree for increasing the size of the vertex cover, expressed relative to the number of nodes, and for differents numbers of vertices. The x-axis represents the vertex cover size, being this our control parameter γ, which is the fraction of the vertices that are included in the vertex cover. The y-axis represents the number of nodes in the search tree. The vertical axis uses a logarithmic scale to make the exponential growth more visible.

The graph shows that the number of nodes in the search tree increases exponentially with the size of the vertex cover. This is because the backtracking algorithm must explore all possible combinations of vertices to find a vertex cover. As the size of the vertex cover increases, the number of possible combinations increases exponentially.

This really happens until a vertex cover size of 0.4, where we can find the critical complexity point. This means that the peak of difficulty reaches the maximum value of our control parameter around $\gamma = 0.4$. From this point, the number of nodes expanded decreases until a vertex cover of 0.65 approximately, and it stabilizes then.

We can see in the Figure 1 there are two well defined regions: the first one when $\gamma \leq 0.4$, and the other $\gamma > 0.4$.

The plot of Figure 2 shows the fraction of instances of the vertex cover problem that are solvable for increasing values of the vertex cover size. The vertex cover size is expressed relative to the number of nodes in the graph. The x-axis of the graph shows the vertex cover size, from 0 to 1.0, where 1.0 represents a vertex cover of the entire graph. The y-axis shows the fraction of instances that are solvable for each vertex cover size.

The plot shows that the fraction of solvable instances increases as the vertex cover size increases. This is because it becomes easier to find a vertex cover that covers all edges in the graph as the number of vertices that must be included in the vertex cover increases.

I consider for values $\gamma < 0.35$, the instances are over-constrained. For values in between, $0.35 \leq \gamma < 0.45$, we can find the critically-constrained instances. And, for value $\gamma \geq 0.45$, the instances are under-constrained.

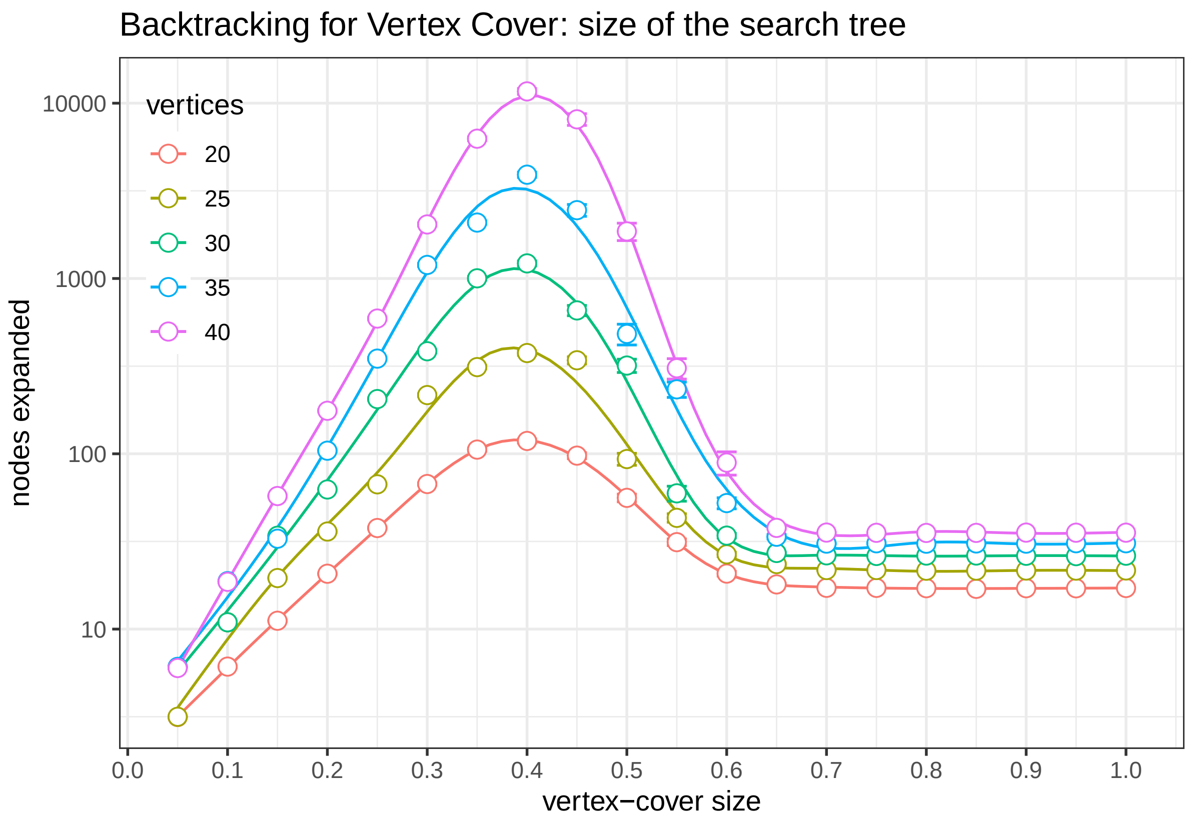

Figure 1: Number of nodes in the search tree for increasing size of the vertex cover (expressed relative to the number of nodes). Note the use of a logarithmic scale in the vertical axis.

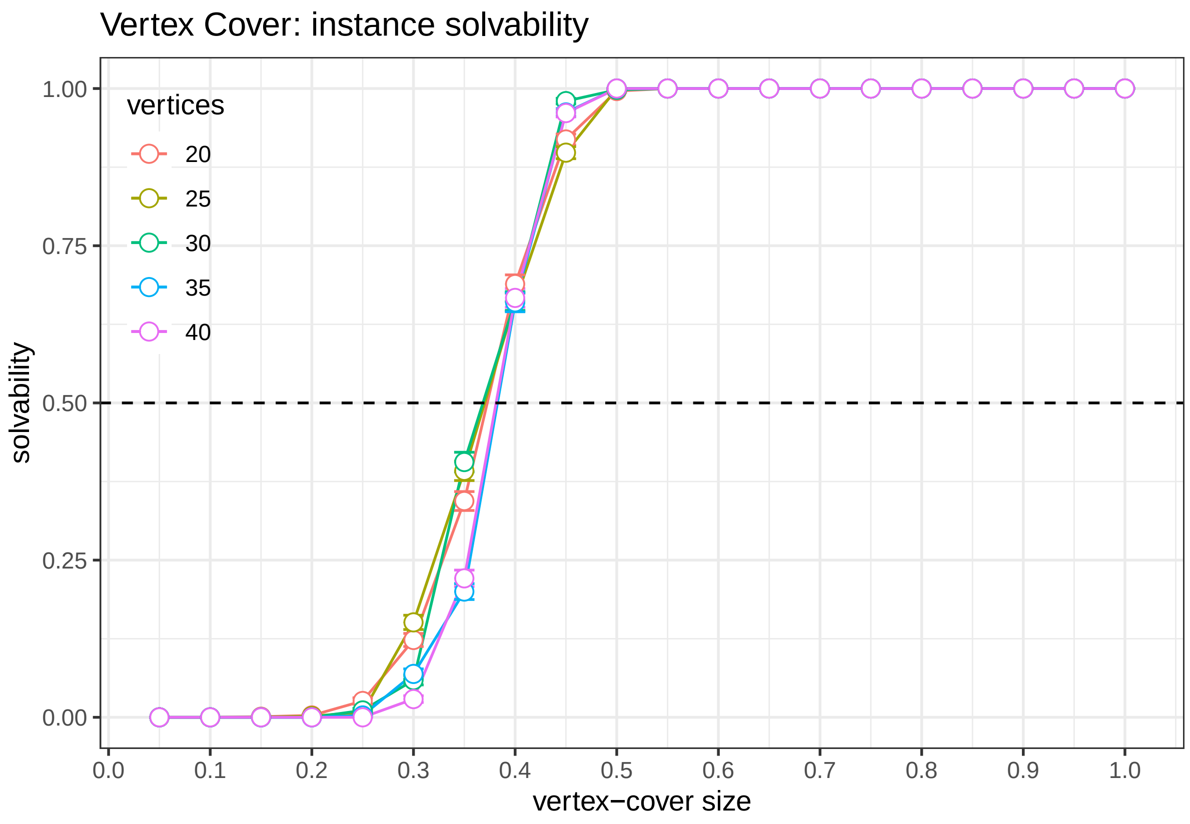

Figure 2: Fraction of solvable instances for increasing size of the vertex cover (expressed relative to the number of nodes).

4 Discussion

The results of the tests are consistent and very accurate with these theoretical predictions. The vertex cover problem is NP-complete, meaning that there is no known algorithm that can solve it exactly in polynomial time. This implies that the difficulty of the problem grows exponentially with the size of the input.

Searching for other algorithms to resolve this problem, I have found that some algorithms are better in practice than others. The backtracking algorithm is the simplest algorithm, but it is also the least efficient. Greedy algorithms and local search algorithms seems to be more efficient in practice, but they are not guaranteed to find the optimal solution.

The choice of an algorithm depends on the specific application. If it is important to find the optimal solution, then a more sophisticated algorithm should be used. However, if it is more important to find a solution quickly, then a simpler algorithm may be sufficient. That means that it is important to consider the trade-off between efficiency and optimality when choosing an algorithm.

A Appendix

A.1 Data Summary

Table 2: Summary of the experimental results for different number of vertices and sizes of vertex covers. The mean and standard error are provided for the number of nodes explored and the fraction of solvable instances.

#vertices	cover size	nodes (mean)	nodes (stderr)	solv. (mean)	solv. (stderr)
20	0.05	3	0.01	0.000	0.0000
20	0.10	6	0.04	0.000	0.0000
20	0.15	11	0.11	0.001	0.0010
20	0.20	21	0.24	0.003	0.0017
20	0.25	38	0.49	0.026	0.0050
20	0.30	67	1.01	0.123	0.0104
20	0.35	106	2.00	0.344	0.0150
20	0.40	118	3.19	0.689	0.0146
20	0.45	98	3.88	0.919	0.0086
20	0.50	56	2.82	0.996	0.0020
20	0.55	31	1.37	1.000	0.0000
20	0.60	21	0.65	1.000	0.0000
20	0.65	18	0.24	1.000	0.0000
20	0.70	17	0.07	1.000	0.0000
20	0.75	17	0.06	1.000	0.0000
20	0.80	17	0.06	1.000	0.0000
20	0.85	17	0.07	1.000	0.0000
20	0.90	17	0.06	1.000	0.0000
20	0.95	17	0.06	1.000	0.0000
20	1.00	17	0.06	1.000	0.0000
25	0.05	3	0.01	0.000	0.0000
25	0.10	11	0.10	0.000	0.0000

25	0.15	20	0.22	0.000	0.0000
25	0.20	36	0.47	0.002	0.0014
25	0.25	67	0.96	0.008	0.0028
25	0.30	216	3.82	0.151	0.0113
25	0.35	312	6.60	0.392	0.0154
25	0.40	376	11.02	0.661	0.0150
25	0.45	342	15.96	0.898	0.0096
25	0.50	93	7.32	1.000	0.0000
25	0.55	43	2.38	1.000	0.0000
25	0.60	27	0.76	1.000	0.0000
25	0.65	23	0.33	1.000	0.0000
25	0.70	22	0.10	1.000	0.0000
25	0.75	22	0.07	1.000	0.0000
25	0.80	22	0.07	1.000	0.0000
25	0.85	22	0.07	1.000	0.0000
25	0.90	22	0.07	1.000	0.0000
25	0.95	22	0.07	1.000	0.0000
25	1.00	22	0.07	1.000	0.0000
30	0.05	6	0.04	0.000	0.0000
30	0.10	11	0.10	0.000	0.0000
30	0.15	34	0.42	0.000	0.0000
30	0.20	63	0.86	0.000	0.0000
30	0.25	205	3.46	0.011	0.0033
30	0.30	385	7.13	0.059	0.0075
30	0.35	1003	23.59	0.406	0.0155
30	0.40	1220	39.93	0.662	0.0150
30	0.45	658	44.22	0.980	0.0044
30	0.50	319	27.70	0.998	0.0014
30	0.55	59	5.83	1.000	0.0000
30	0.60	34	1.06	1.000	0.0000
30	0.65	27	0.27	1.000	0.0000
30	0.70	26	0.10	1.000	0.0000
30	0.75	26	0.08	1.000	0.0000
30	0.80	26	0.08	1.000	0.0000
30	0.85	26	0.08	1.000	0.0000
30	0.90	26	0.08	1.000	0.0000
30	0.95	26	0.08	1.000	0.0000
30	1.00	26	0.08	1.000	0.0000
35	0.05	6	0.04	0.000	0.0000
35	0.10	19	0.20	0.000	0.0000
35	0.15	33	0.39	0.000	0.0000
35	0.20	104	1.53	0.000	0.0000
35	0.25	349	5.93	0.003	0.0017
35	0.30	1195	23.22	0.069	0.0080
35	0.35	2084	43.66	0.200	0.0127
35	0.40	3909	133.93	0.660	0.0150
35	0.45	2454	190.57	0.962	0.0060
35	0.50	483	65.45	1.000	0.0000
35	0.55	233	23.49	1.000	0.0000

35	0.60	52	3.79	1.000	0.0000
35	0.65	34	0.94	1.000	0.0000
35	0.70	31	0.08	1.000	0.0000
35	0.75	31	0.08	1.000	0.0000
35	0.80	31	0.09	1.000	0.0000
35	0.85	31	0.08	1.000	0.0000
35	0.90	31	0.08	1.000	0.0000
35	0.95	31	0.09	1.000	0.0000
35	1.00	31	0.08	1.000	0.0000
40	0.05	6	0.04	0.000	0.0000
40	0.10	19	0.20	0.000	0.0000
40	0.15	57	0.71	0.000	0.0000
40	0.20	176	2.73	0.000	0.0000
40	0.25	591	10.74	0.000	0.0000
40	0.30	2034	40.86	0.029	0.0053
40	0.35	6272	151.97	0.221	0.0131
40	0.40	11677	446.77	0.667	0.0149
40	0.45	8091	623.91	0.961	0.0061
40	0.50	1857	211.21	1.000	0.0000
40	0.55	308	41.05	1.000	0.0000
40	0.60	89	13.51	1.000	0.0000
40	0.65	38	0.54	1.000	0.0000
40	0.70	36	0.10	1.000	0.0000
40	0.75	35	0.09	1.000	0.0000
40	0.80	35	0.09	1.000	0.0000
40	0.85	35	0.09	1.000	0.0000
40	0.90	36	0.09	1.000	0.0000
40	0.95	35	0.09	1.000	0.0000
40	1.00	35	0.09	1.000	0.0000