[ECE650]

Final Project Report

By

Shuting Lian

Sifan Huang

December 1, 2017

Final Project Report

1. The plot about running time and approximation ratio

After running 100 times for each value of V, the results are shown in the following plots. Figure 1 shows the running time of the CNF-SAT thread, and Figure 2 is the result of the running time of the Approx-1 thread and Approx-2 thread. Each vertical line in the two plots indicates the deviation of 100 data of each number of vertices.

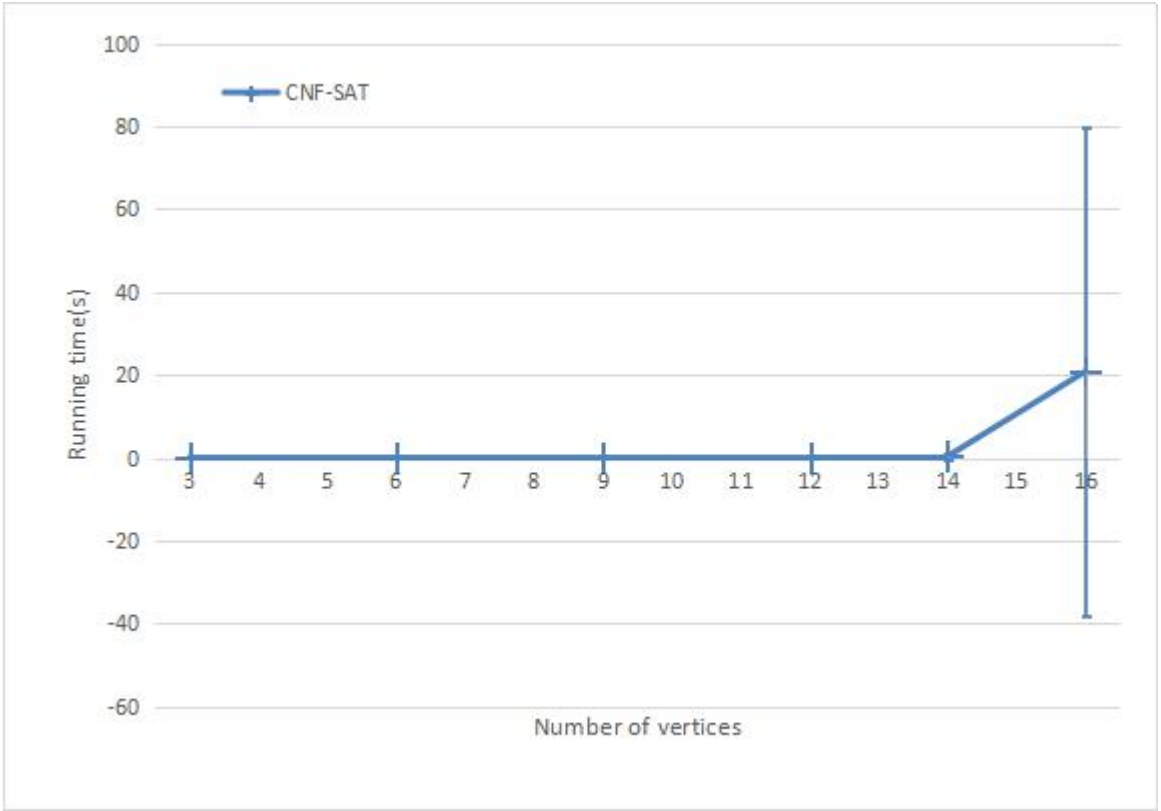

Figure 1: Running time of the CNF-SAT thread.

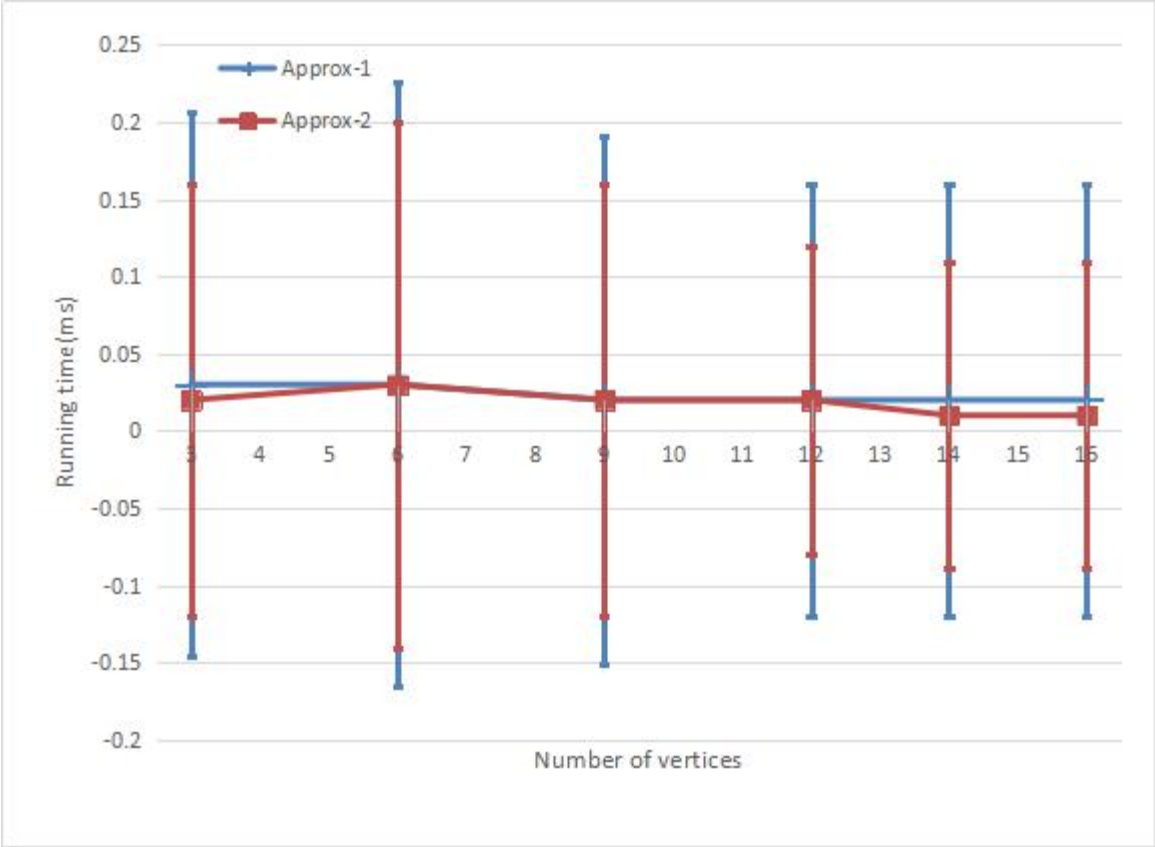

Figure 2: Running time of the Approx-1 thread and Approx-2 thread.

See from Figure 1, the running time of a graph in CNF-SAT thread stays small when the number of the vertices is less than 14, after that, when it increases to 16, the running time increases to 20.83 seconds, that is, when the number of vertices of a graph is larger than 14, the running time climbs sharply. And the deviations are also small when the number of vertices is smaller than 14, but as large as 59.16s when the number of vertices is 16.

As the Figure 2 shows, either in Approx-1 thread or in Approx-2 thread, the running time stays nearly constant, at about 0.02ms or 0.03ms, which is very small. At the same time, the deviations also remain in a limited range, from 0.0995ms to 0.1959ms.

2. Analysis of running time

For the CNF-SAT thread, it is a classic NP-hard optimization problem to calculate the minimum vertex cover of a graph, that is the reason why the running time of calculating doesn't increase linearly according to the number of the vertices.

In order to compute the optimal vertex cover, it is mandatory to satisfy some clauses. According to the formula in *a4_encoding.pdf*, the number of clauses in the reduction is $k + n\binom{k}{2} + k\binom{n}{2} + |E|$, the running time depends on the number of the vertices and the number of minimal vertex cover.

As for the Approx-1 thread, the algorithm of it is to pick a vertex which has the most incident edges. Add it to the vertex cover list and throw away all edges incident on that vertex. Repeat until no edges remain. It is a more efficient way to compute a vertex cover, but the result of it cannot be guaranteed to be the minimum, that is because sometimes it cannot pick the right vertex at the time more than one vertices have the most incident edges. It is efficient because every time the thread just need to compare and choose one of the highest-degree vertices, so the running time would be polynomial in the size of its input E.

Approx-2 algorithm is also an efficient algorithm, unlike Approx-1, it randomly picks the edges and removes them, and stops when no edges remain. Its running time is linear in the size of input E, which could be denoted in big-O notation as O(|E|)[1]. So in general, the running time of Approx-2 would be faster than that of Approx-1.

Either Approx-1 algorithm or Approx-2 algorithm has large deviations comparing to the mean of the running time. That is because the running time of each graph in these two threads depends on the size of E, which means, even if the |V| are the same, because the structure of the graph (or say, the number of the edges) is different, the running time would be different, too. So, because the running time of CNF-SAT algorithm depends on the number of the vertices and the number of minimal vertex cover, so the deviation would be small. But when |V| equals to 16, the deviation is large because the number of minimal vertex cover of each graph is different.

[1] |E| means the quantity of E.

3. Analysis of approximation ratio

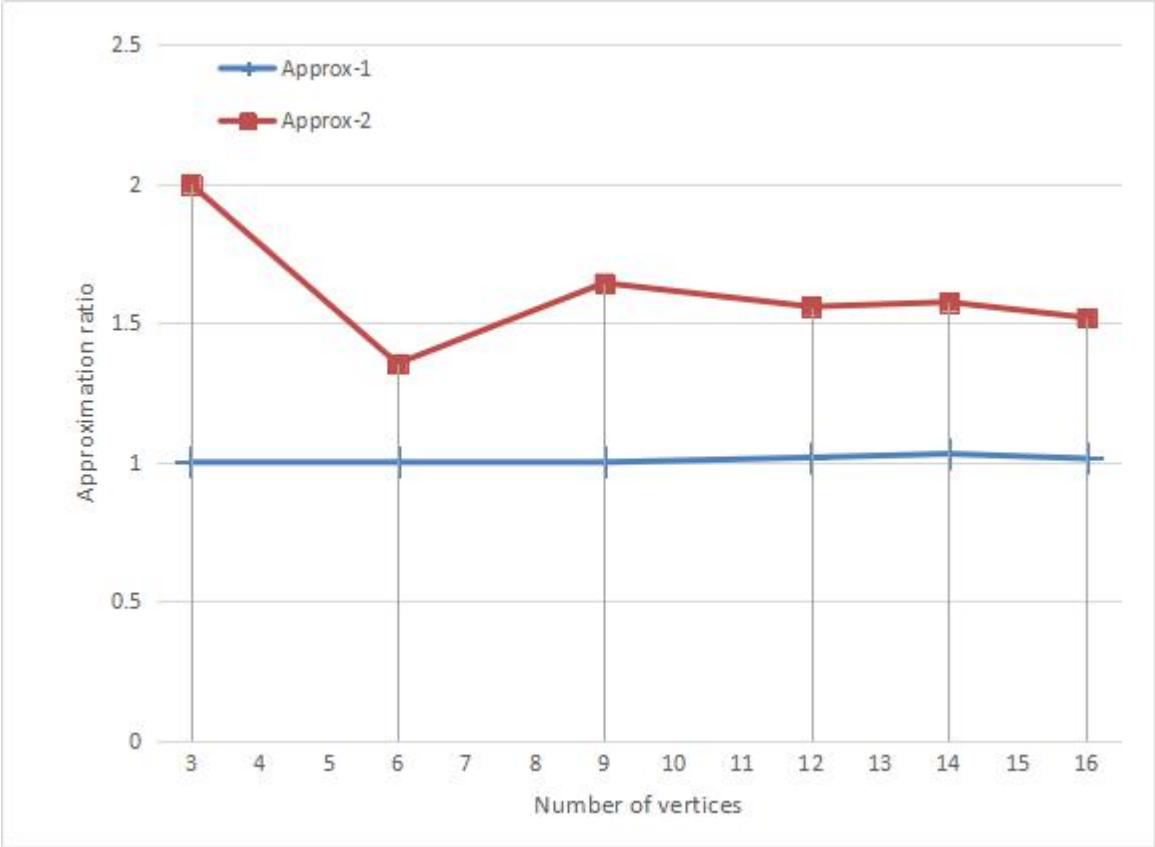

Figure 3: The approximation ratios of the Approx-1 thread and Approx-2 thread.

Figure 3 represents the approximation ratios of the Approx-1 thread and Approx-2 thread. The approximation ratio is the ratio of the size of the computed vertex cover (comes from the Approx-1 thread or the Approx-2 thread) to the size of an optimal vertex cover (comes from the CNF-SAT thread). The ratio is more close to 1, the corresponding algorithm is more accurate.

Observe the plot and we can find out that the approximation ratios of Approx-1 are much smaller than that in Approx-2 thread, which remain nearly 1, which means the results coming from Approx-1 thread are similar to the optimal result. As for Approx-2 thread, the ratios keep changing around the 1.5.

The reason for the high approximation ratio of Approx-2 is on the algorithm. The algorithm of Approx-2 asks to pick edges randomly, without any condition or restriction to make the option, so the quantity of vertex cover coming from it would be much

bigger than the optimal one.

Although the vertex cover coming from Approx-1 could not be guaranteed as an optimal one, the algorithm is more accurate than Approx-2, because it does have some restrictions -- Pick a vertex of “highest” degree each time, which is also a reasonable condition, so the accuracy of the algorithm is high.