|  |  |  |
| --- | --- | --- |
| **Project report** | | Academic year:  **2020/21** |
| Subject:  **Algorithms and data structures** | |
| Project name:  **Hamilton and Eulerian cycle** | | Subject takes place at:  **N/A** |
| Faculty, field of study, semester:  **FC, AI, II** | Full name:  **Jan Gruszczyński** | Grade: |
|

**Exercise 1**

Finding Eulerian cycle in randomly generated connected graph, that contains an Eulerian cycle.

The graphs containing said cycles were generated using the algorithm presented during the lecture.

At each measuring point, search time was measured 10 times on consequently new generated graphs, then an average was taken as representative of obtained values.

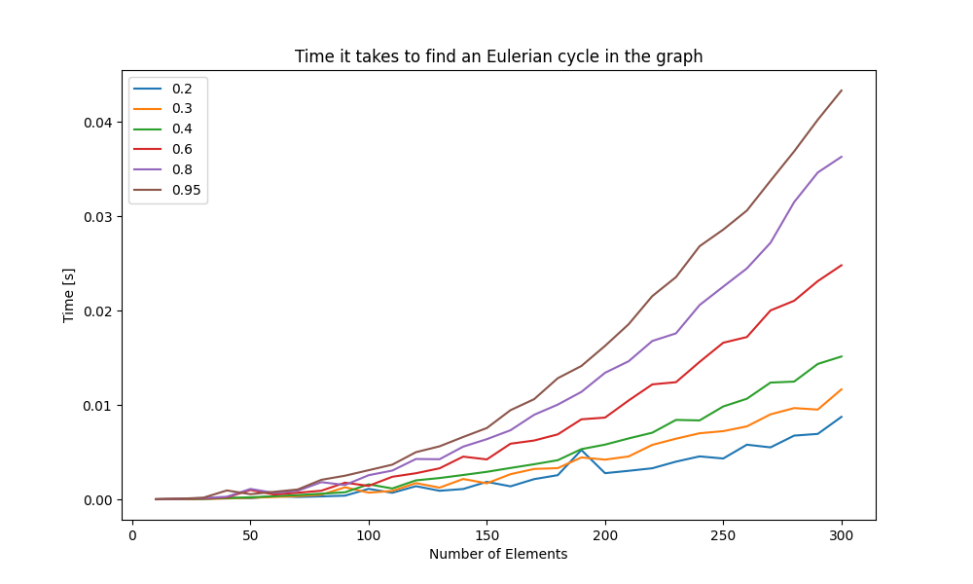


Figure 1: Time it takes to find Eulerian cycles in the graph, with given saturation and number of elements. (Legends presents saturations)

List of incidents is employed as a graph representation, because it takes the least amount of memory (space complexity O(|V|+|E|)), it also allows to directly access all adjacent vertices, although it prolongs the time it takes to check if a given vertex exists.

The higher the saturation the longer it takes for the algorithm to find an Eulerian cycle.

As the implemented algorithm is based on Hierholzer’s algorithm, it computational complexity is linear and equal to O(number\_of\_edges). Thesis is supported by the chart, as more saturated graphs (containing more edges) with the same amount of vertices need more time for the algorithm to find an Eulerian cycle in them. Also, lines in the chart are growing somewhat linearly.

**Exercise 2**

Finding Hamilton cycle in a randomly generated connected graph.

At each measuring point, search time were measured 10 times on consequently new generated graphs, then an average was taken as representative of obtained values.

After 30 seconds, if the cycle wasn’t found, time-out was instantiated.

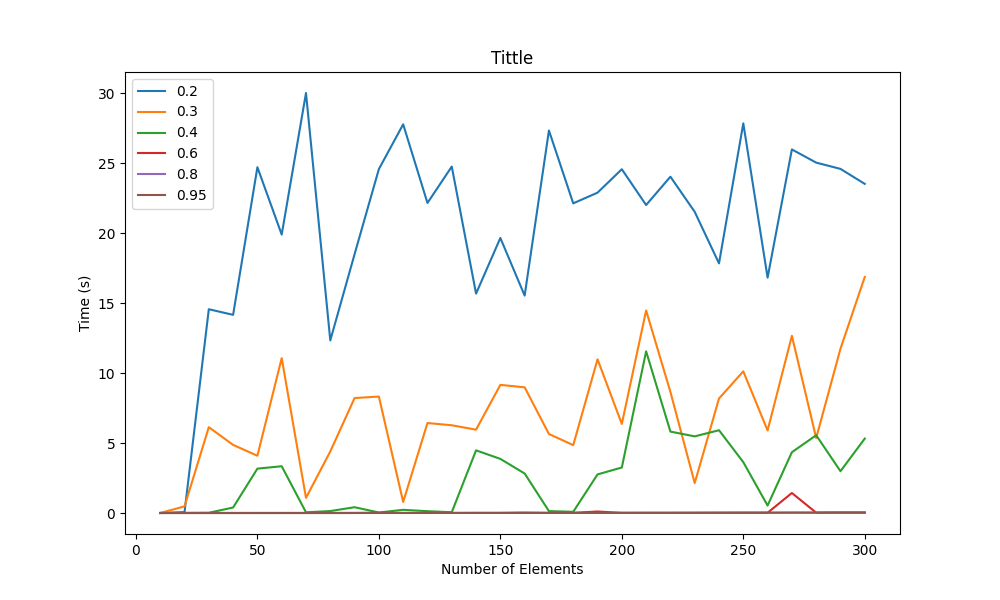


Figure 2: Time it takes to find Hamilton cycles in the graph, with given saturation and number of elements. (Legends presents saturations)

**(More text below)**

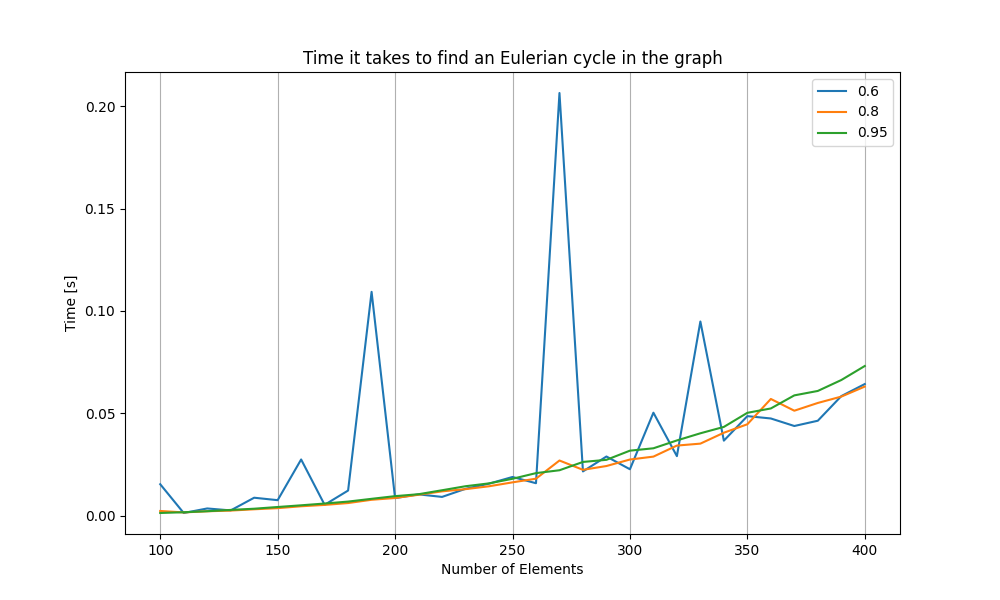


Figure 3: Time it takes to find Hamilton cycles in the graph, with given saturation and number of elements. (Legends presents saturations). Troublesome saturations omitted.

Because we need to check n\*(n-1)\*(n-2)[…] vertices, computational complexity of the backtracking algorithm appears to be O(n!)

The lower the saturation, the more time it takes to find a Hamilton cycle by the algorithm in the graph.

Conclusion:

Finding Hamilton cycle is an NP-complete problem, because there is no algorithm that can solve it in polynomial time.

As presented algorithm can find Eulerian cycle in polynomial, linear time, it is not NP-complete. It belongs to the P problems class.