#Exercise 3 Graph algorithms

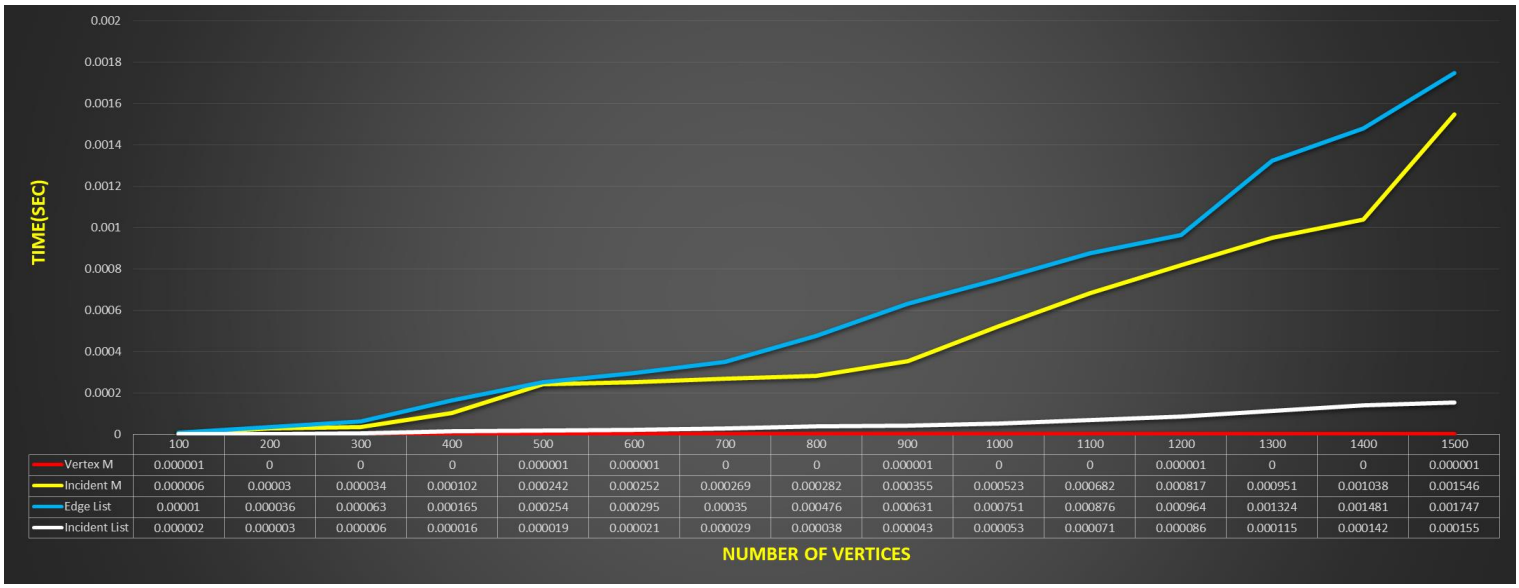
Uladzimir Ivashka, 150281

Sofya Aksenyuk, 150284

Remark:

Given results of all the below experiments are average of 10 runs.

I. Time for obtaining information about the existence of edges between a pair of random vertices for following graph representations: Vertex Matrix, Incident Matrix, Edge List, List of Incidents, for a randomly generated undirected input graph with edge saturation of 0.6.



	100	200	300	400	500	600	700	800	900	1000	1100	1200	1300	1400	1500
Vertex M	0.000001	0	0	0	0.000001	0.000001	0	0	0.000001	0	0	0.000001	0	0	0.000001
Incident M	0.000006	0.00003	0.000034	0.000102	0.000242	0.000252	0.000269	0.000282	0.000355	0.000523	0.000682	0.000817	0.000951	0.001038	0.001546
Edge List	0.00001	0.000036	0.000063	0.000165	0.000254	0.000295	0.00035	0.000476	0.000631	0.000751	0.000876	0.000964	0.001324	0.001481	0.001747
Incident List	0.000002	0.000003	0.000006	0.000016	0.000019	0.000021	0.000029	0.000038	0.000043	0.000053	0.000071	0.000086	0.000115	0.000142	0.000155

Conclusions:

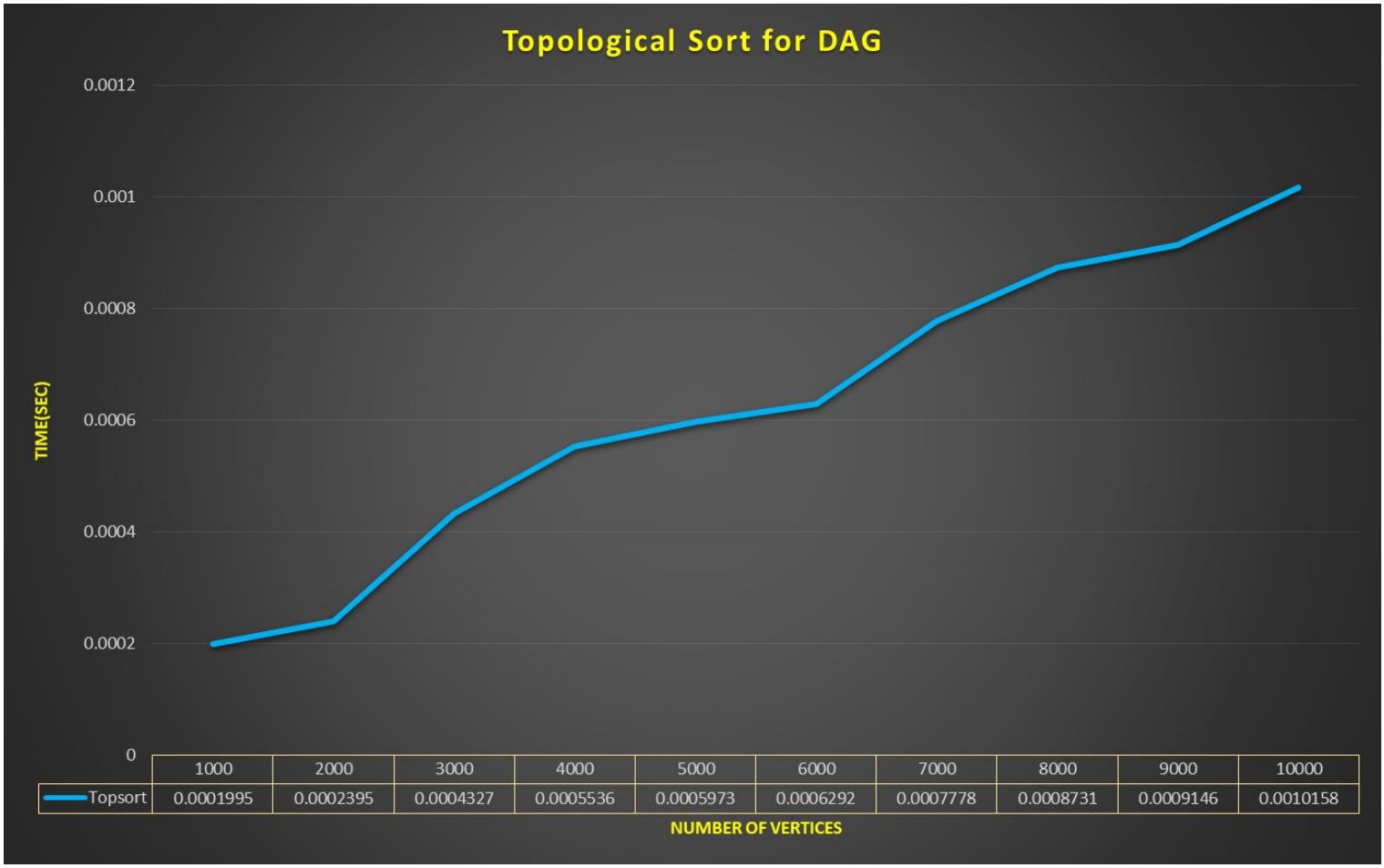
According to the chart above, it can be concluded that Vertex Matrix is the fastest graph representation for checking edge existence among all presented ones. Such behavior of the algorithm was expected due to the construction of the matrix that allows us to “access” requested element directly so that it gives time complexity of O(1).

Incident List algorithm shows results a bit slower than Vertex Matrix. It can be explained by its work principle that parses list of first requested vertex’s connections until it finds the second one. Thereby, Incident List structure has search time complexity of O(V) (V – number of vertices).

It is better to group Incident Matrix and Edge List due to the noticeable difference of computational time comparing with Vertex Matrix and Incident List. Nevertheless, Incident Matrix performs slightly faster than Edge List. The way Incident Matrix is structured slows down the procedure: it checks all of its columns to obtain an information of a relation between two vertices that.

Despite the time complexity of O(E) (E – number of edges) which is equal to Incident Matrix one, Edge List shows the worst results. Edge List simply goes throughout the whole list of edges checking for the combination of both requested vertices which is not an effective approach.

II. Time for topological sorting procedure for a randomly generated DAG graph with edge saturation of 0.3.



	1000	2000	3000	4000	5000	6000	7000	8000	9000	10000
Topsort	0.0001995	0.0002395	0.0004327	0.0005536	0.0005973	0.0006292	0.0007778	0.0008731	0.0009146	0.0010158

Conclusions:

For the presented computations, the main choosing criterion for a graph representation for DAG graph was the speed of an algorithm. That is why Incident List was chosen: in comparison with the graph algorithms described in point (I.) above, it has one of the highest searching speeds. Apart from that, the structure of Incident List is needed to apply DFS.

Since DFS is used and it takes O(V+E) with O(1) to insert an element into a linked list, the time complexity of Topological Sort is equal to O(V+E) as well, which supports our theoretical background and the chart above.