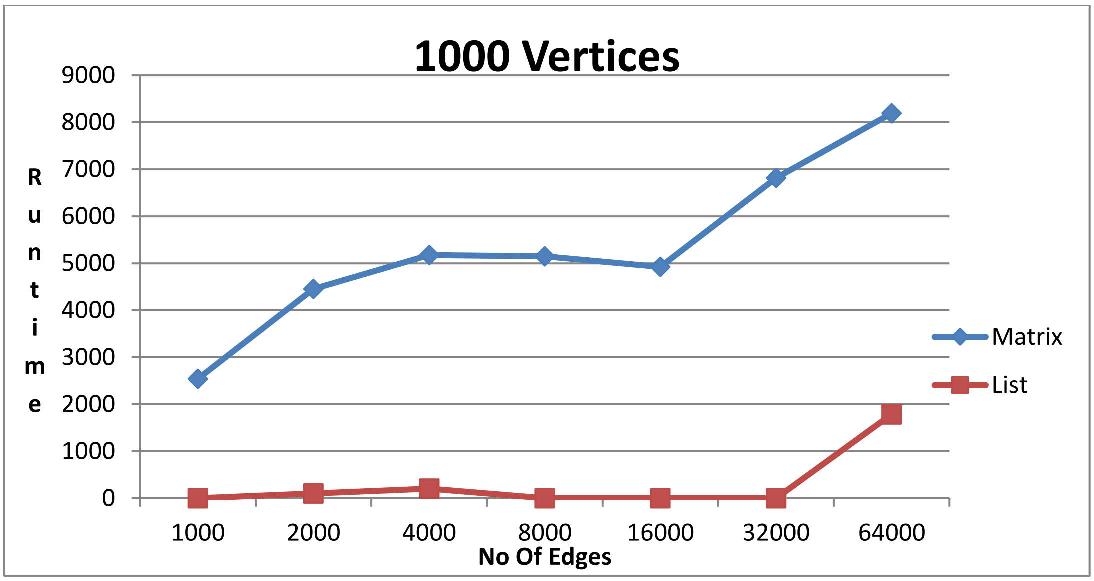
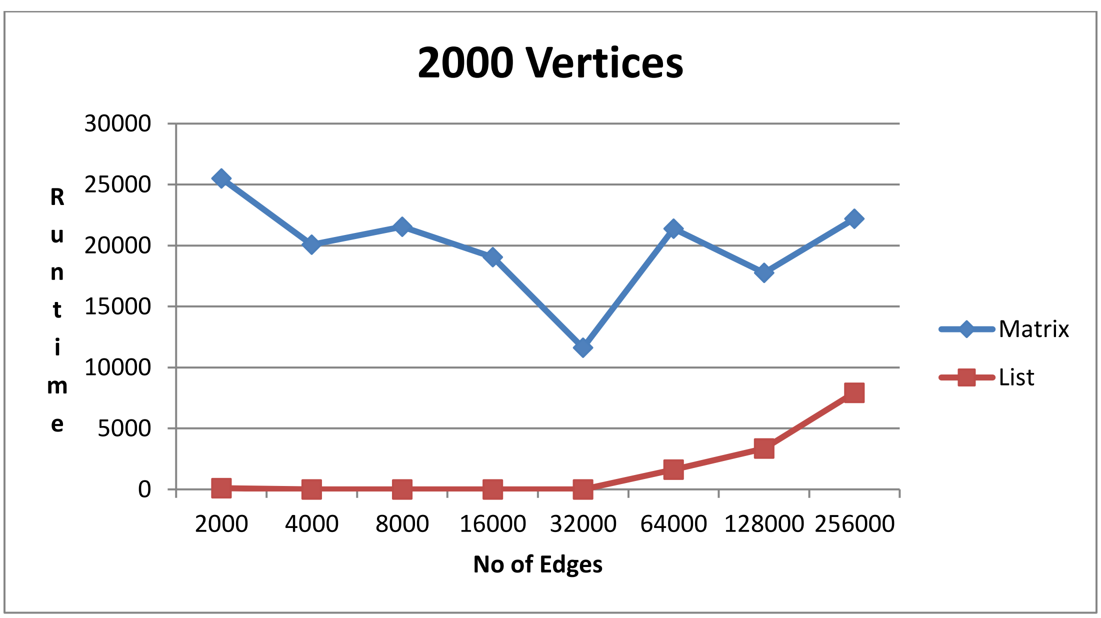
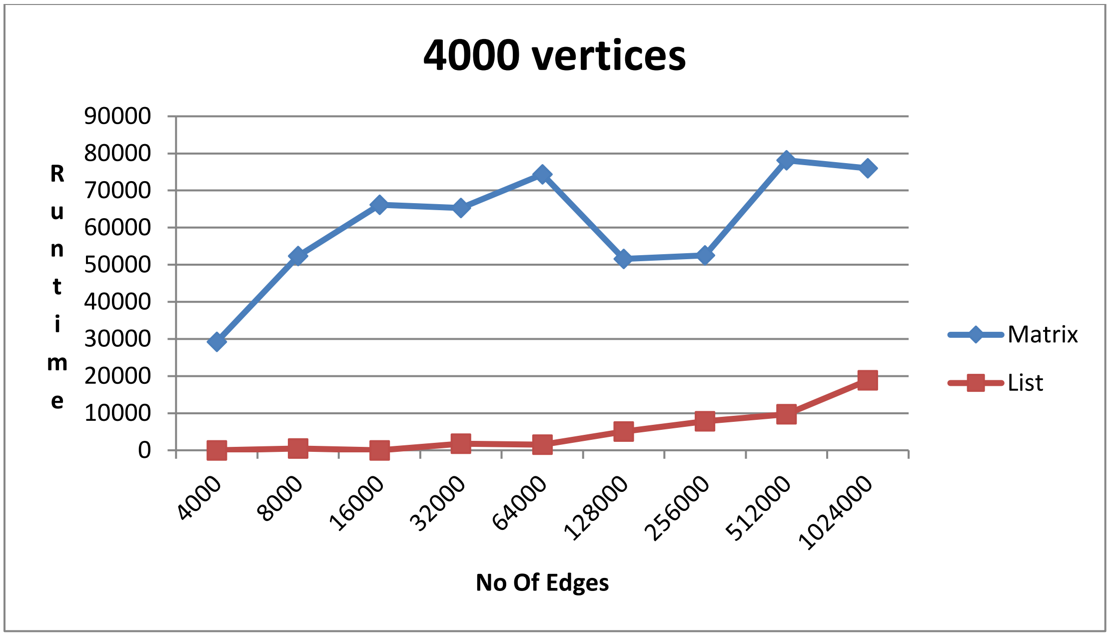
No of Vertices	No of Edges	Runtime(microseconds)	
		Adjacency Matrix	Adjacency List
1000	1000	2541	0
	2000	4454	102
	4000	5171	202
	8000	5145	0
	16000	4924	0
	32000	6813	0
	64000	8191	1785
2000	2000	25508	99
	4000	20064	0
	8000	21546	0
	16000	19054	0
	32000	11613	0
	64000	21394	1628
	128000	17768	3364
	256000	22200	7933
4000	4000	29216	0
	8000	52342	517
	16000	66161	0
	32000	65276	1784
	64000	74313	1540
	128000	51570	5083
	256000	52513	7817
	512000	78134	9749
	1024000	75964	18875
8000	8000	121017	0
	16000	244302	0
	32000	247025	1795
	64000	250997	3215
	128000	250788	4690
	256000	280523	3245
	512000	378052	14177
	1024000	398558	14723
	2048000	430979	32767
	4096000	414642	82649

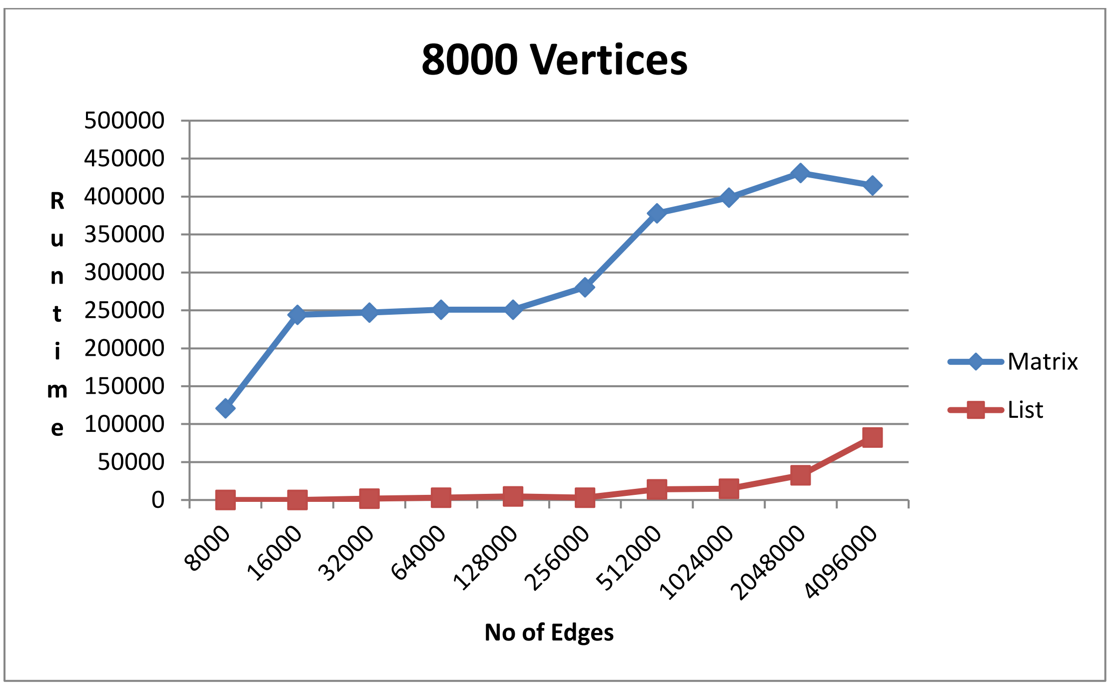
16000	16000	716866	0
	32000	932052	3311
	64000	942687	3458
	128000	898989	6485
	256000	864177	10935
	512000	874906	9786
	1024000	868421	25375
	2048000	1091678	38869
	4096000	1032200	58709
	8192000	1135614	166216
	16384000	1532651	328197



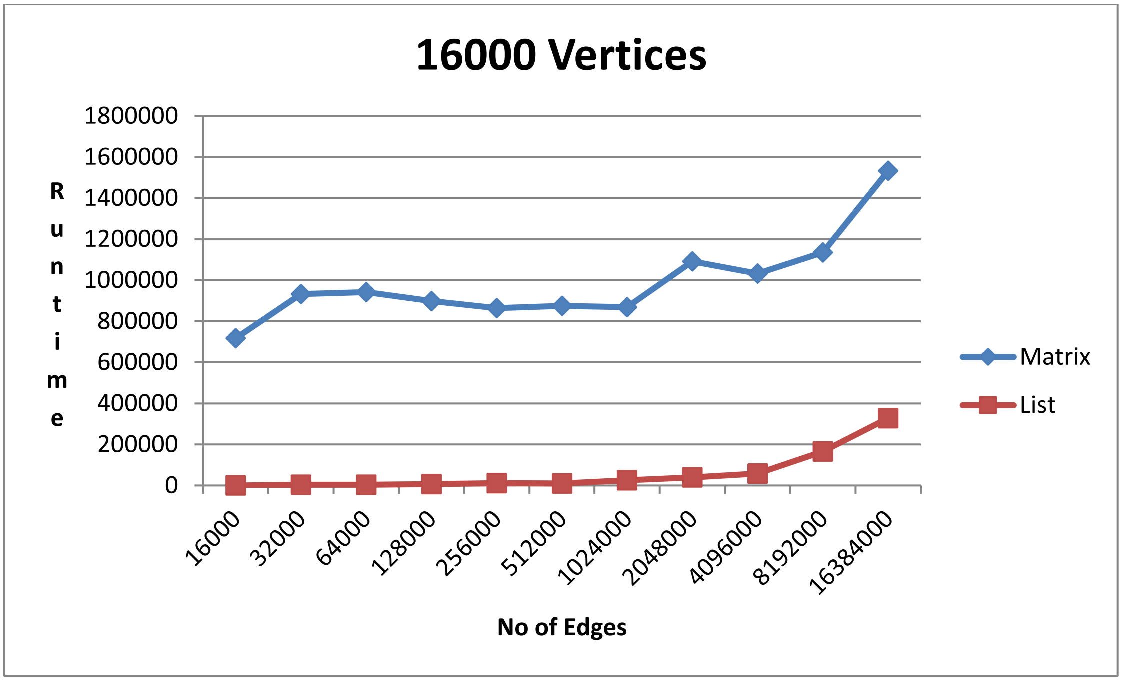




4000 vertices
Runtime
90000
80000
70000
60000
50000
40000
30000
20000
10000
0
4000
8000
16000
32000
64000
128000
256000
512000
1024000
No Of Edges
Matrix
List



8000 Vertices
Runtime
500000
450000
400000
350000
300000
250000
200000
150000
100000
50000
0
8000
16000
32000
64000
128000
256000
512000
1024000
2048000
4096000
No of Edges
Matrix
List



Question Answer

1. What is the impact on runtime if we keep |V| unchanged and double |E| for adjacency list? Why is it so?

   Ans. For adjacency list representation, if we double the number of edges (|E|) keeping the number of vertices same the runtime will increase. I do have very less information to give verdict about will the runtime be double of the previous one or not because of the randomness but in most of the cases it increases surely. We know, the running time of BFS for adjacency list representation is O(|V| + |E|). So, running time will incease keeping pace with |E|.

2. What is the impact on runtime if we keep |E| unchanged and double |V| for adjacency list? Why is it so?

   Ans. For adjacency list representation, if we double the number of vertices (|V|) keeping the number of edges same the runtime will increase. I do have very less information to give verdict about will the runtime be double of the previous one or not because of the randomness but in most of the cases it increases surely. We know, the running time of BFS for adjacency list representation is O(|V| + |E|). So, running time will incease keeping pace with |E|.

3. What is the impact on runtime if we keep |V| unchanged and double |E| for adjacency matrix? Why is it so?

   Ans. From our collected data, it is quite sure that the runtime of BFS have a very slight effect or no effect at all if we double the number of edges keeping the number of vertices same. It is because of the fact that, the runtime of BFS for adjacency matrix representation is O($|V|^2$). The fact is clear that, the runtime has no effect of the cardinality of the edges.

4. What is the impact on runtime if we keep |E| unchanged and double |V| for adjacency matrix? Why is it so?

   Ans. It is evident from our data that, doubling the number of vertices keeping the number of edges constant does effect the runtime significantly. It is because, the runtime of BFS in a graph having adjacency matrix representation has the runtime of O($|V|^2$). In our experiment the runtime did not increase in a quadratic manner but $|V|^2$ is surely an upper bound of it. As the runtime depends on the number of vertices, increasing the number of vertices increases runtime eventually.

5. For the same |E| and |V|, why are the runtimes for adjacency list and adjacency matrix representation different? Which one is higher and why?

   Ans. For same |E| and |V|, the adjacency matrix representation will give higher runtime. For running BFS in adjacency list representation, we have to first initialize all the vertices and then have to run a loop which covers all the edges. So, it's runtime is O(|V| + |E|).

   On the other hand, when it comes to adjacency matrix representation to run a BFS we have to first initialize all the vertices as before but on the next step we have to cover all the edges. For this reason we have to iterate over the 2D |V| x |V| matrix that is used to store the information about the edges. So, the second step takes O($|V|^2$) runtime alone. Thus the total runtime in this case is O($|V| + |V|^2$) or O($|V|^2$).

   That is why the runtime is much higher when it comes to adjacency matrix representation.

Zaber Ibn Abdul Hakim

#1705015