Complexities of Graph Operations

Let's discuss the performance of the two graph representations.

We'll cover the following

Time Complexities#

Below, you can find the time complexities for the 4 basic graph functions.

Note that, in this table, **V** means the total number of vertices and **E** means the total number of edges in the Graph.

Operation	Adjacency List	Adjacency Matrix
Add Vertex	O(1)	$O(V^2)$
Remove Vertex	O(V+E)	$O(V^2)$
Add Edge	O(1)	O(1)

Operation	Adjacency List	Adjacency Matrix
Remove Edge	O(E)	O(1)
Search	O(V)	O(1)
Breadth First Search(BFS)	O(V+E)	O(V^2)
Depth First Search(DFS)	O(V+E)	O(V^2)

Adjacency List#

- Adding an edge in adjacency lists takes constant time as we only need to insert at the **head** node of the corresponding vertex.
- Removing an edge takes O(E) time because, in the worst case, all the edges could be at a single vertex and hence, we would have to traverse all **E** edges to reach the last one.
- Removing a vertex takes O(V + E) time because we have to delete all its edges and then reindex the rest of the list one step back in order to fill the deleted spot.
- Searching an edge between a pair of vertices can take up to O(V) if all **V** nodes are present at a certain index and we have to traverse them.

Adjacency Matrix#

- Edge operations are performed in constant time as we only need to manipulate the value in the particular cell.

- Vertex operations are performed in O(V^2) since we need [illegible] d [illegible] a [illegible] columns. We will also need to fill all the new cells.

- Searching an edge is O(1) because we can access each edge by indexing.

Comparison#

Both representations are suitable for different situations. If your application frequently manipulates vertices, the adjacency list is a better choice.

If you are dealing primarily with edges, the adjacency matrix is the more efficient approach.

Keep these complexities in mind because they will give you a better idea about the time complexities of the several algorithms we'll see in this section.

In the next lesson, we will look at a special type of graph called the **bipartite graph**.

Back
Graph Implementation

Next
What is a Bipartite Graph?

Completed

Report an Issue