| Operations | Adjacency Matrix | Adjacency List |
|---|---|---|
| Storage Space | This representation makes use of VxV matrix, so space required in worst case is **O(\|V\|$^2$)**. | In this representation, for every vertex we store its neighbours. In the worst case, if a graph is connected O(V) is required for a vertex and O(E) is required for storing neighbours corresponding to every vertex .Thus, overall space complexity is O(\|V\|+\|E\|). |
| Adding a vertex | In order to add a new vertex to VxV matrix the storage must be increases to (\|V\|+1)$^2$. To achieve this we need to copy the whole matrix. Therefore the complexity is **O(\|V\|$^2$)**. | There are two pointers in adjacency list first points to the front node and the other one points to the rear node.Thus insertion of a vertex can be done directly in **O(1) time.** |
| Adding an edge | To add an edge say from i to j, matrix[i][j] = 1 which requires **O(1)** time. | Similar to insertion of vertex here also two pointers are used pointing to the rear and front of the list. Thus, an edge can be inserted in **O(1)**time. |

| Operations | Adjacency Matrix | Adjacency List |
|---|---|---|
| Removing a vertex | In order to remove a vertex from V*V matrix the storage must be decreased to $\|V\|^2$ from $(\|V\|+1)^2$. To achieve this we need to copy the whole matrix. Therefore the complexity is **O(\|V\|$^2$)**. | In order to remove a vertex, we need to search for the vertex which will require O(\|V\|) time in worst case, after this we need to traverse the edges and in worst case it will require O(\|E\|) time.Hence, total time complexity is **O(\|V\|+\|E\|)**. |
| Removing an edge | To remove an edge say from i to j, matrix[i][j] = 0 which requires **O(1)** time. | To remove an edge traversing through the edges is required and in worst case we need to traverse through all the edges.Thus, the time complexity is **O(\|E\|)**. |
| Querying | In order to find for an existing edge the content of matrix needs to be checked. Given two vertices say i and j matrix[i][j] can be checked in **O(1)** time. | In an adjacency list every vertex is associated with a list of adjacent vertices. For a given graph, in order to check for an edge we need to check for vertices adjacent to given vertex. A vertex can have at most O(\|V\|) neighbours and in worst can we would have to check for every adjacent vertex. Therefore, time complexity is **O(\|V\|)** . |

Report An Issue

If you are facing any issue on this page. Please let us know.

Dash

All

Articles

Videos

Problems

Quiz

Contest