Time Complexity for Finding whether two URLs are Connected:

Adjacency Matrix:

Time complexity for finding whether two URLs are connected in an adjacency matrix is O(1), as accessing a specific entry in the matrix takes constant time.

Adjacency List:

Time complexity for finding whether two URLs are connected in an adjacency list depends on the implementation. In worst case, it can take O(V) time, where V is the number of vertices, if we have to traverse the entire adjacency list for a particular vertex.

Edge List:

Time complexity for finding whether two URLs are connected in an edge list also depends on the implementation. In worst case, it can take O(E) time, where E is the number of edges, as we may have to traverse the entire list of edges.

Most Time Efficient Graph Representation to Find the Number of Links for a Particular URL:

The adjacency list representation is the most time efficient for finding the number of links for a particular URL.

In an adjacency list, each vertex (URL) maintains a list of its adjacent vertices (linked URLs). Therefore, to find the number of links for a particular URL, you simply need to count the number of elements in its adjacency list. This operation takes O(1) time if you're using a well-implemented adjacency list.

Using Adjacency Matrix to Determine if the Graph is Directed or Undirected:

In an adjacency matrix representation, if the matrix is symmetric about the main diagonal, i.e., M[i][j] is equal to M[j][i] for all i and j, then the graph is undirected.

If the matrix is not symmetric, then the graph is directed.

The reason behind this is that in an undirected graph, if there is an edge from vertex A to vertex B, there is also an edge from vertex B to vertex A. This symmetry is reflected in the adjacency matrix. In a directed graph, edges can be one way, so the adjacency matrix may not be symmetric.