# Complexity Analysis – Big Screen Graph Implementation

This program is using **Adjacency List** representation.

Since there is access between vertices and edges in both direction, performance is better than edge list structure.

Say, a graph with **n vertices and m edges**, then,

1. Space requirement of Adjacency List structure: is O(n+m)
2. Time Complexity:

Assignment Questions:

1. List the number of movies and the actors represented in the graph

**To list number of movies and actors, need to traverse the adjacency list once.**

**Hence O(n) time complexity**

2. List the names of the movies in which performer A has acted.

**To find the actor vertex - O(n)**

**To display incident movie vertices to actor,**

**O(deg(v)). Since deg(v) limited to 2 in this problem, Constant complexity.**

**O(deg(v)) = O(1).**

**Total complexity will be O(n)**

3. List the names of the performers in the movie X

**To find the actor vertex iterate the adjacency list once.**

**To display incident nodes, O(1), as the vertex count is 2.**

4. Consider the following relation R on the movies

“*Movie A is related to Movie B if there is at least one actor common in the movies A and B. In this case, we write R(A, B)*”.

Given any two movies A and B, verify if R(A, B)?

**Steps involved:**

1. **Identify vertex for Movie A - O(n)**
2. **Identify vertex for Movie B – O(n)**
3. **Check Actor in A is present in Actor B, Since no of vertex is 2, it will constant time, maximum 4 comparisons O(1) – constant complexity.**

**Total is O(n) + O(n) + O(1) ~= O(n)**

5. Consider the following relation T on the movies “

*a. T ( A,B ) if R (A , B)*

dis

*(or )*

b. *T ( A, B) if there is a movie C such that R(A,C) and also R (C,B).* ”.

Given any two movies A and B, verify if T(A, B)?

**Using BFS to find the path between vertices.**

**The logic used in the assignment is :**

**Step 1: Find path between v1 to v2 with BFS**

**Step 2: Find path between v2 to v1 with BFS**

**Step 3: Iterate the paths to determine common vertices**

**Output: The common vertices forms the relationship between 2 vertices.**

**Complexity Analysis:**

**Step 1: To find incident edges, will need O(n+m) time complexity; as it is BSF**

**Step 2: same as step 1**

**Step 3: Will need to iterate through both paths to see the common vertices. Then O(n^2) time complexity.**

**Since O(n^2) is the highest degree, the time complexity is O(n^2).**