**Course: Introduction to Graph Algorithms**

**Professor: C Pandu Rangan**

**Department: Computer Science and Engineering**

**Institute: IISc**

**Week: 01**

# Lecture 04 Undirected Graph

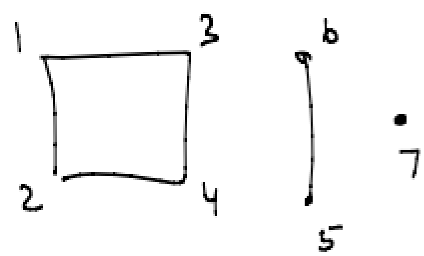
Namaskara, in this session we will briefly discuss about undirected graph. We have seen already basic definitions of directed graph and way to represent them. We will see the analog remarks related to undirected graph. An undirected graph is also a pair, vertex set and edge set except that this edge set consists of unordered pairs.

$$G = (V, E)$$

So, technically you can represent an unordered pair by ab, ordered pair was written like this, unordered pair, because sets will have no order among their elements, pair of elements. So we can, put in this notation ab and write in a curly bracket but that would be little cumbersome, we can do that, but the moment we say that it is an undirected graph, we can understand that even with this notation, okay.

$$\{a, b\} \qquad (a, b)$$

I can represent the undirected graphs. So, for example if I have the picture 1, 2, 3, 4, 5, 6, 7 notice that I have not added any arrowheads, there is no direction here, okay.




So you can traverse an edge in both direction you can go from 1 to 4, a path, 1 to 3, 3 to 4, the same path you can walk in the reverse direction, right. You can interpret this like a two-way street, the directed edge is like a one-way street, undirected edge is like a two-way street, you can go this way or in the opposite direction also.

$$1 - 3 - 4$$

$$4 - 3 - 1$$

So you can go from 4 to 3, 3 to 1, you can go from 1 to 3 to 4, 4 to 3. Undirected edges are like two way street, directed edges are like one way streets, okay. So you can traverse in both directions. Therefore you see that we assume that the edge 1 3 as well as 3 1 exists, because you can go from 1 to 3 or 3 to 1, okay. So in the adjacency matrix, we would fill both 13 and 31 as 1.

So, if ij belongs to E, strictly speaking I should write like this, but I will continue to use this notation, with an understanding that I am now dealing with undirected graph, therefore that represents unordered pair, okay.

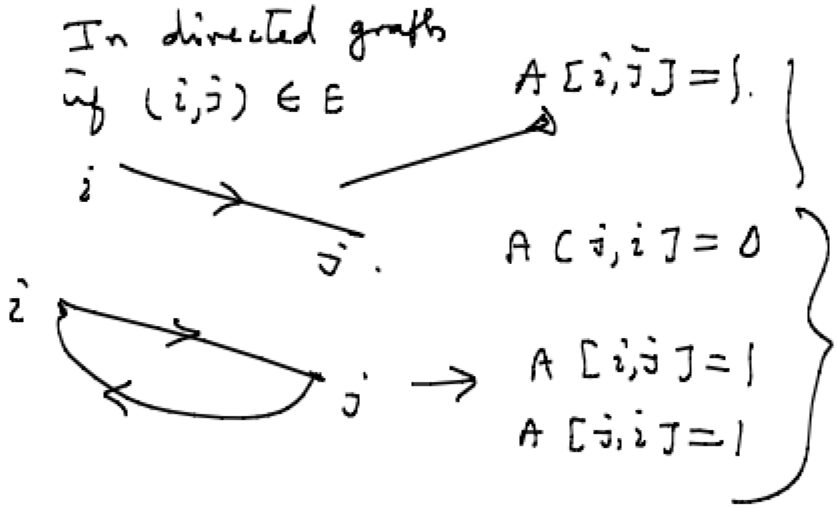
$$\text{if} \quad (i,j) \in E$$
$$\{i,j\} \in E$$

You can avoid that ambiguity by writing like this. But we write, we use the same notation with the understanding that we are currently dealing with unordered pairs or undirected graph. Then, A ij equal to 1 and A ji is also 1. There will be 2 entries that are set to 1.

$$\left.\begin{array}{l} A[i,j]=1 \\ A[j,i]=1 \end{array}\right\}$$

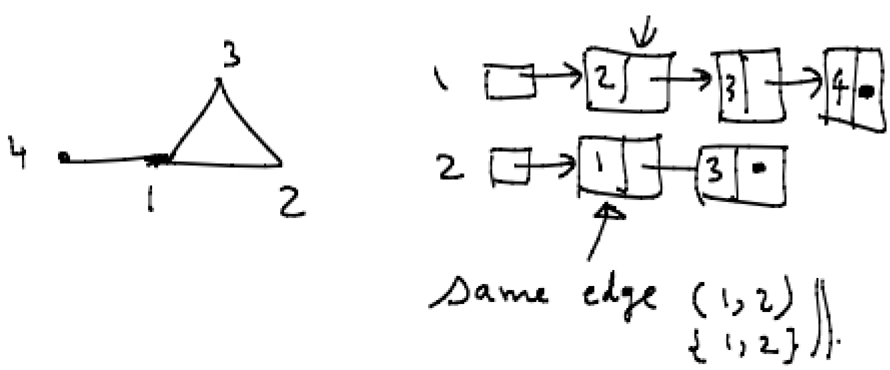
In the directed graph, if ij belongs to E only

A j is set to 1, you do not touch the other one, the A ji could be 1, could be 0, depending on, so this is this says, from i to j an edge exists. Therefore only one entry, what about the other entry? other entry could be 0 or could be 1. In this case it is 0, in this case A ji is 0 but if it is like this, from i to j there is an edge, from j to i there is an edge, in this case A ij equal to 1 and A ji is also 1, both edges are present. This is the essential difference, because of the direction you see that this happens here, there is no direction here. For directed graphs, if ij exists we set only A ij to 1, A ji maybe 0 or maybe 1 depending on ji exists or not. In undirected graph, if ij exists, we assume j also exists and we set both A ij and A ji to be 1.




Therefore the adjacency matrix of an undirected graph is always symmetric. If there is a 1 in A ij, there will be a 1 in A ji. If there is a 0 in A ij, there will be a 0 in A ji. Therefore aij equal to aji in all cases, that means it is a symmetric matrix. The adjacency matrix of a directed graph need not be symmetric, I may have 1 there and 0 here, 1 here and 0, you can have one at both 0 at both, it is completely independent. Therefore, arbitrary structure, the matrix is an arbitrary Boolean matrix. Similarly, the adjacency list, each edge will give rise to 2 boxes. So if you have an edge ij,


in adjacency list of i, we include j and in adjacency list of j, we include i. So there will be 2 boxes that would correspond to an edge for an undirected graph, okay? So typically 1, 2, 3 adjacency list of 1 will have, let us say adjacency list of 1 will have 2, 3 and 4. adjacency list of 2, it will have 1 because of this edge, because of the edge 1 2 to 1 and 3. So you can see this box and this box, they correspond to same edge, 1 2 or 1 2.



A single edge will give rise to 2 boxes, one box in the adjacency list of i, another box in the adjacency list of j. Therefore the size of the adjacency list, size of adjacency list of directed graph is exactly E.

$$|E|$$

Size of adjacency list data structure for undirected graph is 2E, because each edge gives rise to 2 boxes.

$$2|E|$$

In adjacency list of i, we include j and in adjacency list of j, we include i whenever ij is present, okay? We also do not have in degree and out degree, there is no direction, there is only the notion of degree of a node is number of edges incident on it. We do not say incoming, outgoing and all, incident, it is falling on it. That edge contains the vertex as an end vertex, it is incident on it. That is the reason why, sigma degree of v, v belongs to V is twice the number of edges, because each edge is counted once in the degree of i. again it is counted in the degree of j.

$$\sum_{v \in V} d(v) = 2|E|$$

So if you have an edge ij this edge is counted in the degree value of i the same edge is counted again in the degree value of j you can see that every edge is counted twice therefore the degree sum is twice the number of edges.

Recall that for the directed graph recall for directed graph it is sigma in degree of  v, v belongs to V is cardinality of E Sigma out degree of v we do not have the notion of in degree and out degree therefore for undirected graph the formula is different.

$$\sum_{v \in V} \text{in-degree}(v) = |E|$$

$$\sum_{v \in V} \text{out-degree}(v) = |E|$$

So all I have done is introduced the basic definitions related to undirected graphs and directed graphs compared and contrasted them and discussed about the way to represent them in computer, how do we represent that so that we write programs based on these kind of representations. This concludes my brief discussions on directed and undirected graph, thank you.