Social Networks
Prof. S. R. S. Iyengar
Department of Computer Science
Indian Institute of Technology, Ropar

Lecture – 05
Introduction to Social Networks
Introduction to Networkx-1

(Refer Slide Time: 00:05)

I will now be introducing this graph package called networkx, with the help of which we can do a whole lot of graph analysis. I have pre-installed a network x api on python, you may have to Google and understand how to install it on your computer. In case you find any difficulty, please let us know and we will help you out. Write to us on our email address and I will personally help you out.

So, we first say import networkx and the library function networkx gets loaded and then I say G equals networkx dot graph and then there is a graph that gets created and G gets assigned to it and now I say G add node 1 which means vertex 1 gets added. Similarly, I say add node 2 and so on, I will add some 5 nodes to it, 3, add node 4, add node 5 and when I say G dot nodes, it will show me all the nodes that I have just now added, I will add one more node; let us say 6 and then I say G nodes, it will print all the nodes that are just now added.

(Refer Slide Time: 01:31)

Now, I can start adding the edges; how do I add the edges? G; add edge 1 comma 2; G add edge 1 comma 3; G add edge 4 comma 6; G add edge 5 comma 4; G add edge 2 comma 3 and so on, I have added I did not close the bracket.

(Refer Slide Time: 02:00)

I have added all the let me do it once again it 3, 1 error; G add underscore edge, there is a mistake there, 2 comma 3 and that is it. So, I have added a few edges; I can see all the edges by giving this command G dot edges, you see the 5 edges that I just now added is here. Let me add more edge to show you that it is actually getting included, I will add the

edge 2 comma 6 which is missing here; add edge 2 comma 6 and I will display edges, one more edge gets added; 2 comma 6 got added here as you can see alright.

Now, how do I see this addition; now before that you saw that I said G equals networkx dot graph.

(Refer Slide Time: 02:55)

```
In [26]: G.add_edge(5,4)

In [27]: G.add(edge(2,3)
    ...: )
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
<ipython-input-27-864277492ee1> in <module>()
----> 1 G.add(edge(2,3)
      2 )

AttributeError: 'Graph' object has no attribute 'add'

In [28]: G.add_edge(2,3)

In [29]: G.edges()
Out[29]: [(1, 2), (1, 3), (2, 3), (4, 5), (4, 6)]

In [30]: G.add_edge(2,6)

In [31]: G.edges()
Out[31]: [(1, 2), (1, 3), (2, 3), (2, 6), (4, 5), (4, 6)]

In [32]: H=networkx.Graph()
```

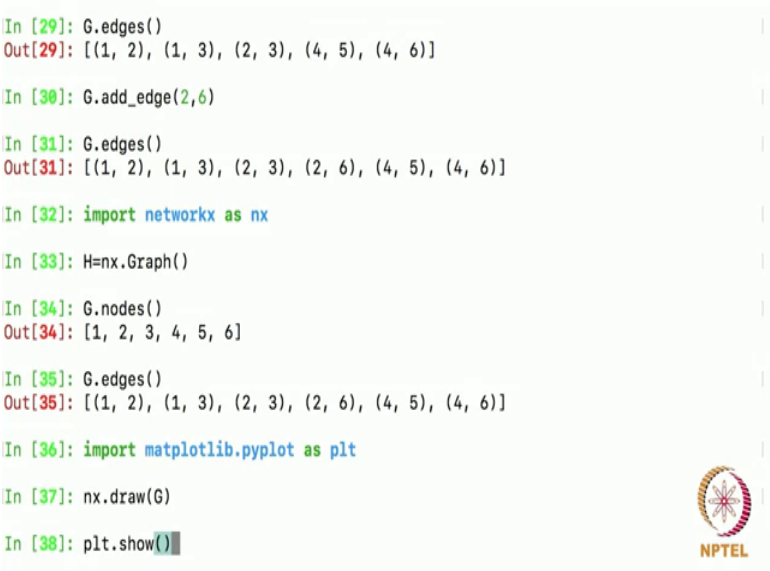
If I were to define another graph namely H; should say H equals networkx dot graph given that tell me using this network x often, there is one way to create an alias like thing; a macro like thing in c, if you remember.

(Refer Slide Time: 03:10)

```
In [27]: G.add(edge(2,3)
    ...: )
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
<ipython-input-27-864277492ee1> in <module>()
----> 1 G.add(edge(2,3)
      2 )

AttributeError: 'Graph' object has no attribute 'add'

In [28]: G.add_edge(2,3)

In [29]: G.edges()
Out[29]: [(1, 2), (1, 3), (2, 3), (4, 5), (4, 6)]

In [30]: G.add_edge(2,6)

In [31]: G.edges()
Out[31]: [(1, 2), (1, 3), (2, 3), (2, 6), (4, 5), (4, 6)]

In [32]: import networkx as nx

In [33]:
```

So, what I do is; I will say when I am importing, I will say import networkx and as nx; short form for network.

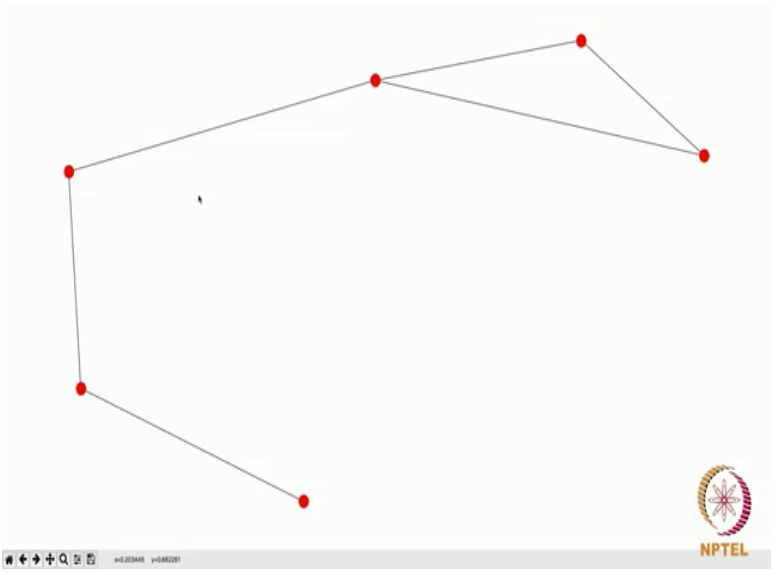
(Refer Slide Time: 03:17)



Now, I can simply say H equals nx graph; instead of network x graph, I can say nx dot graph. Now a graph gets created, anyway I just have to illustrate this usage of import networkx as nx, this is a standard way to do it. If you Google online for code using network, the first line will be import networkx as nx, I did not want you people to get confused; so I thought I will clarify it here. Going back, I created a graph G with the following nodes and the following edges. I want to see how the graph looks like, there is a package for it and I should import that package which helps me visualize the graph; it is called matplotlib, in that I only want this particular sub library of called pyplot and I am going to import that and to avoid that this big command in invoking functions within this, I will say as plt.

Now, I should visualize this graph; this is the way to visualize, it is slightly complicated but you will remember it given that you will be doing it very often. You will simply say; networkx dot draw; what you want to draw? You want to draw the graph G; networkx G and then you say plt dot show.

(Refer Slide Time: 04:50)

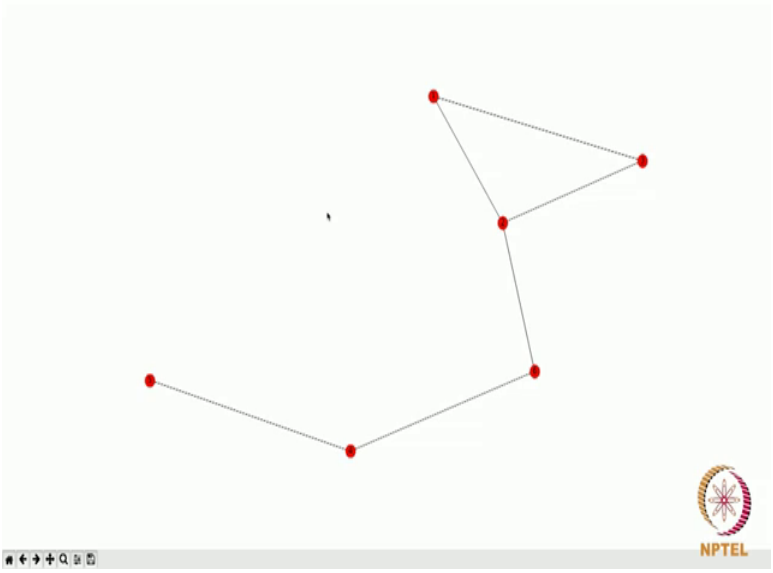


You see the graph that we just now created and we do not know which node is what.

(Refer Slide Time: 04:57)

So, what will do is to display labels I just issue the same command nx draw G and I say with labels equals true. If I include this, you will get; you will see what you will get, you will also get the vertex labels and now I issue the command plt show.

(Refer Slide Time: 05:14)



And you will see that look at this, the graph that I input they are with labels right now, they were without labels in the beginning as you saw.

(Refer Slide Time: 05:24)

Networkx has a lot of built in functions, I will show you one such built in function; let me say Z is equal to nx complete graph 10, this means it generates a complete graph on 10 vertices and assigns it to the variables Z. Let us see what happens, I say enter now a graph on 10 vertices is created, complete graph means it puts all possible edges between these 10 nodes a little bit of thinking will tell you that all possible friendships between 10

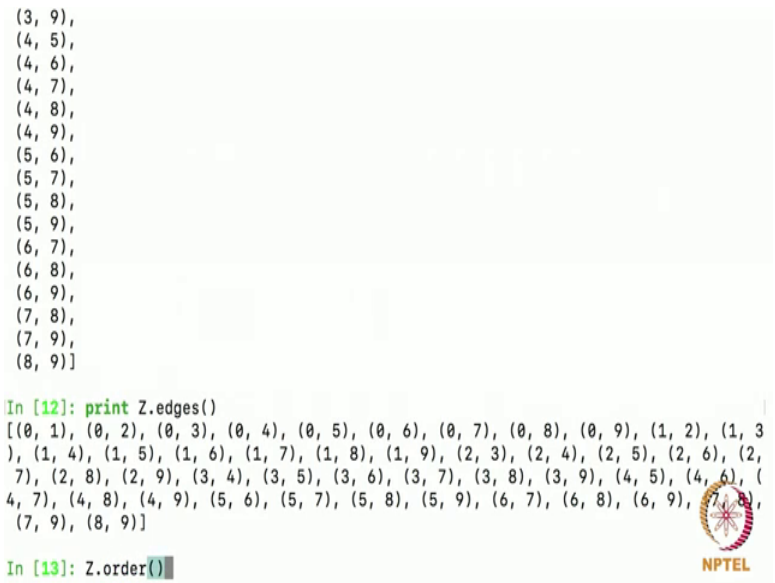
people is actually 45 alright. You can do the computation, you can take it as a homework and do it is a very straightforward question.

So, let me look at the number of nodes in Z; what are all the nodes in Z of course, there are 10 nodes here and the edges; print Z edges or simply Z edges, both of them will be output the edges; here are the edges.

(Refer Slide Time: 06:28)

If I simply say that Z dot edges, it will put it in a line like this; that is why I said print Z dot edges; it will neatly show it like this.

(Refer Slide Time: 06:33)

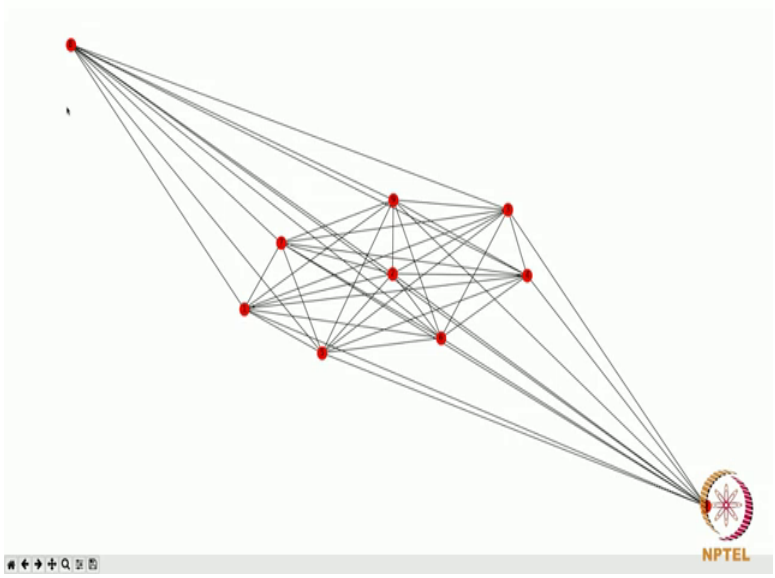


There are 45 edges here, how do I say it is 45? There is another command called Z dot order which stands for the number of nodes in the graph.

(Refer Slide Time: 06:42)

And Z dot size which tells you the number of edges in the graph, there are 45 edges. Now, let me visualize this graph Z; how do I visualize it, nx draw Z; I want it with labels, so I say with labels equals 1 and then I say plt show.

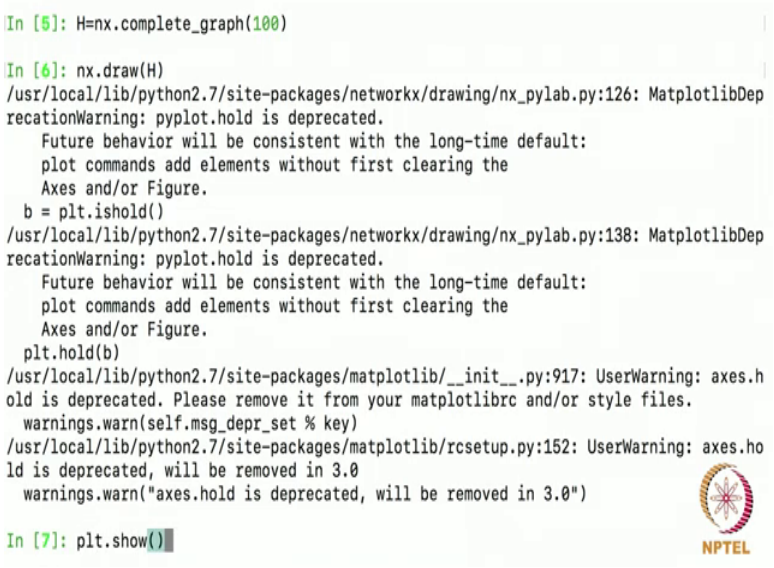
(Refer Slide Time: 07:02)



It will show me the graph and here is my complete graph on 10 vertices, as you can see every node is connected to every other node; that is what you mean by a complete graph

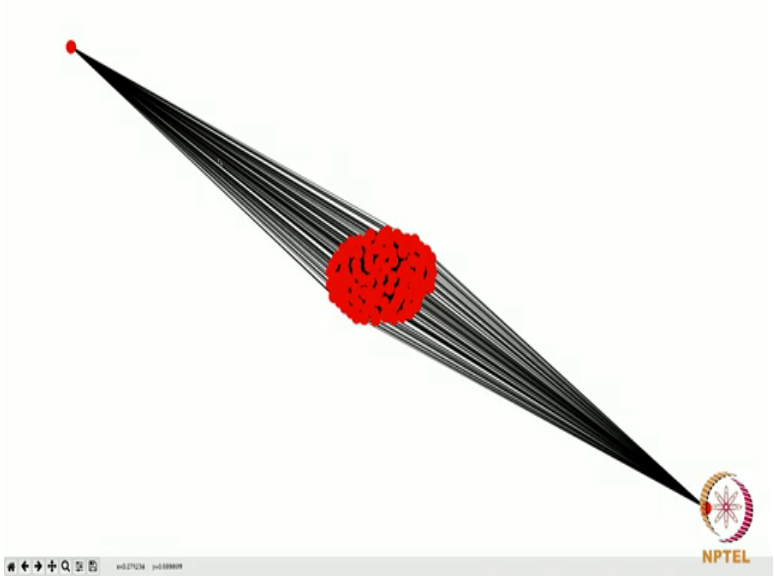
correct. Now, let us try seeing a complete graph on 100 vertices, let us see how it looks like; how does a complete graph on 100 vertices look like?

(Refer Slide Time: 07:22)



I will say H equals nx, complete graph on 100 vertices then I will draw this H; I will not label it labeling will make it look clumsy, the typical warning message I told ignore this; plt show.

(Refer Slide Time: 07:51)



So, this is a complete graph on 100 vertices, so there are 100 vertices and all possible edges. You can ask me, what is this vertex and this vertex? These are all visualization

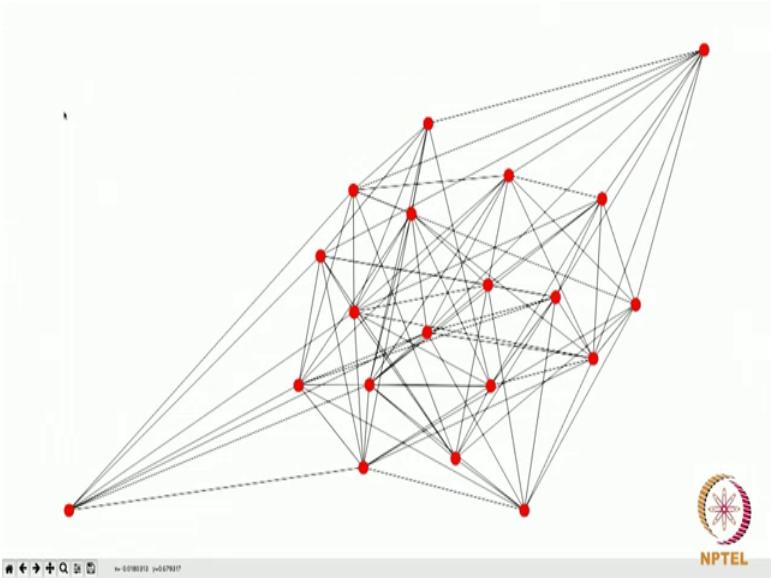
styles that are in built into this matplotlib. You can in fact change this visualization to; there are ways to do that. But as of now, all I want to show you people is that you can see that there are 100 vertices and they are connected to each other and this is how we visualize it.

(Refer Slide Time: 08:18)

Networkx actually has a lot of built in functions, I will show you one such built in function which you may want to use more often. So, look at this I am going to generate a graph let us say G; which is equal to, the command is called gnp random graph; it is an nx; gnp random graph on 20 vertices and I say 0.5 here, What does this mean? This means, I am generating a graph on 20 vertices and edges I am putting them with probability 0.5, which means for every edge; I put the edge with probability 0.5, I do not put the edge with probability 0.5; by that I mean, I toss a coin; if I get a head I put that edge, if I get a tail I do not put that edge.

How does the graph look like? A nice way to explain this is there are 20 people and there are friendships forming between them, they decide whether 2 people should become friends or not by tossing a coin; this is such an experiment that is what you mean by gnp graph. More details, I will be covering in my lecture do not worry much if you did not understand what I said here; all I am trying to do is generate a graph G with 20 nodes with some random edges; how does this graph look like? Lets us draw and see this plt show.

(Refer Slide Time: 09:50)



This is how the graph looks like on 20 nodes, where edges are put uniformly at random. Unlike the previous case, this is not a complete graph as you can see; for example, there is no edge between this vertex and this vertex, this is not a complete graph; this is just a graph on 20 nodes with random edges.