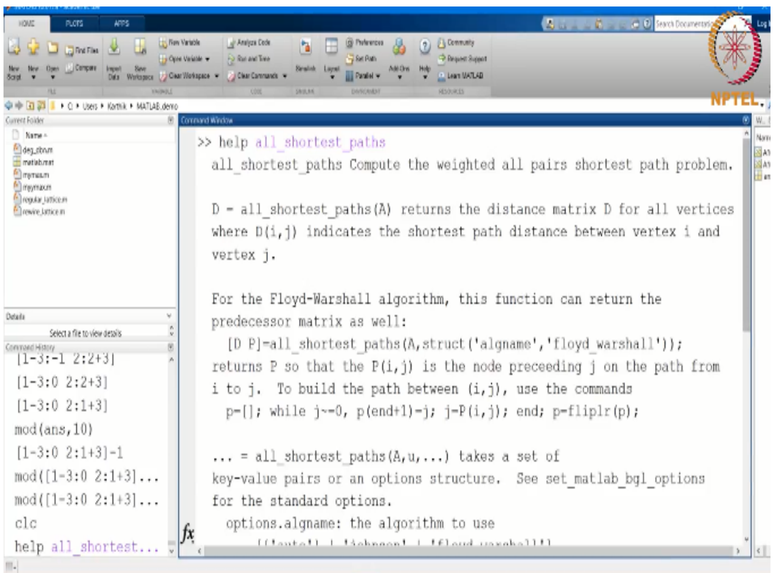
Computational Systems Biology
Karthik Raman
Department of Bio Technology
Indian Institute of Technology-Madras

Lecture – 30
Lab: Network Models & Perturbations

In today's lab, we will continue our study of network model and perturbations and focus more on perturbations as well as profiling it is a very interesting way to study a code and examine its performance.

(Refer Slide Time: 00:25)



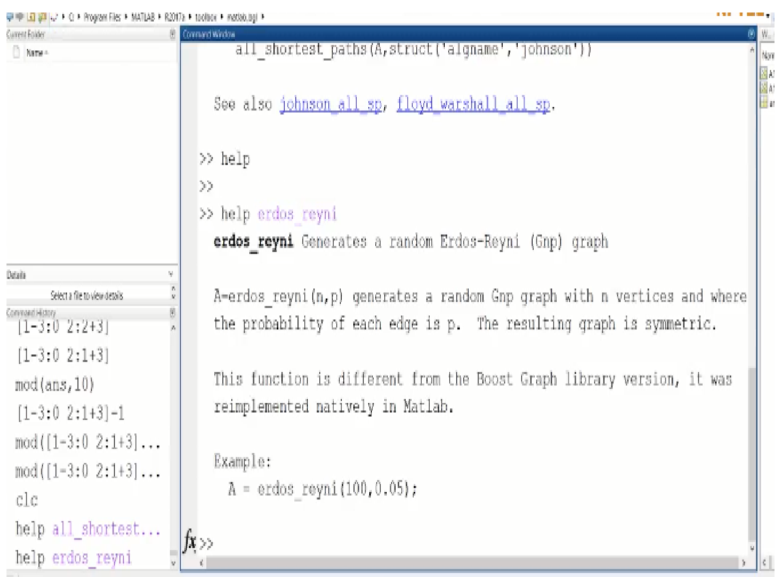
Okay now let us get back to some of the problems we were talking about in the morning today particularly trying to perturb a graph and find out what is happening and how do you perturb a graph. Even Node and its edges and then plot L and plot diameter as it changes. So, for this it can get a little tricky so what you need to understand is first is you need to have a way to perturb a graph.

Right and what you are going to do is basically inside a for loop just remove every node one by one. Or you may want to remove nodes in a particular order right in the order of between a centrality not recalculated or order of degree not recalculated that is the easiest way to start. We can do a incrementally tougher problem little later on. But initially we can start with that take a given Barabasi albert graph or something.

Or create a random graph and see how it changes as you start knocking out edges. **"Professor - student conversation starts"** delete a node means we will make a node and a column that is it because immediately the next point **t**o think about what does it mean when I say delete a node and how do you calculate diameter, the diameter will be of the maximum value yeah which matrix no shortest paths "**Professor - student conversation ends."**

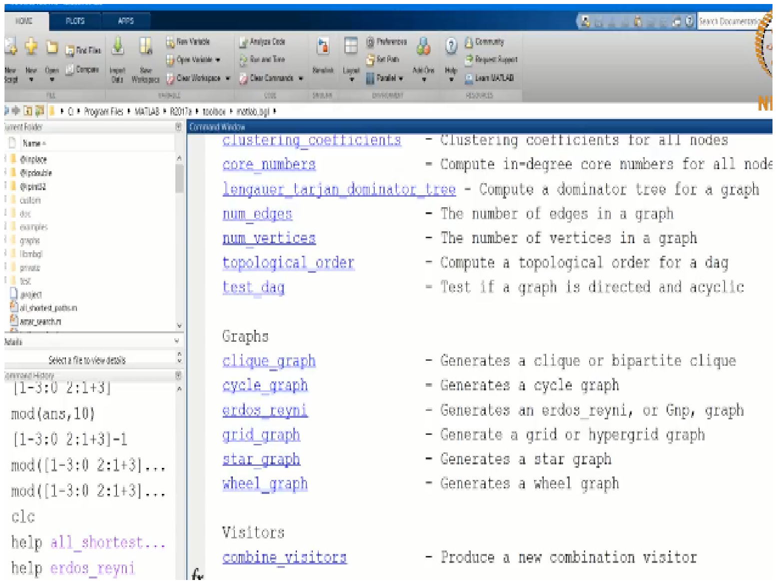
How do you get the shortest paths matrix? that is a good idea but that would not give you the distances it will only give you the connectivity. There is something called all shortest paths look this up the font size is too big. It computes the weighted all pairs shortest path so the single show shortest path problem uses the Dijkstra algorithm. This is an all pairs problem so it is using the dynamic programming algorithm that is based on that is given by Floyd and Warshall

(Refer Slide Time: 02:35)



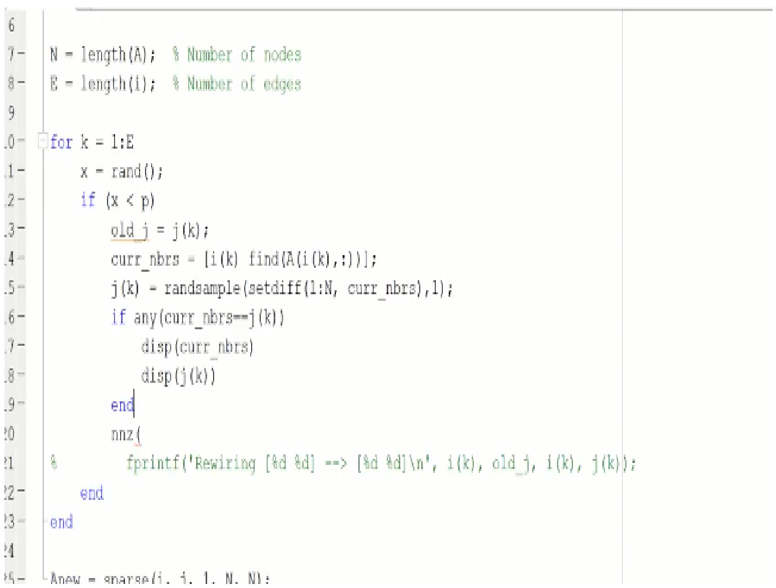
So, efficiently finds the entire matrix of shortest paths distances **"Professor - student conversation starts"** you can use mostly we are using handmade graph you can use anything "**Professor - student conversation ends"**. So, BGL has this misspelt so make sure you get the spelling right NN Y0 Y and N. Let us see what are the other graphs that we can generate easily using BGL and we can use one of them for perturbation.

(Refer Slide Time: 03:41)

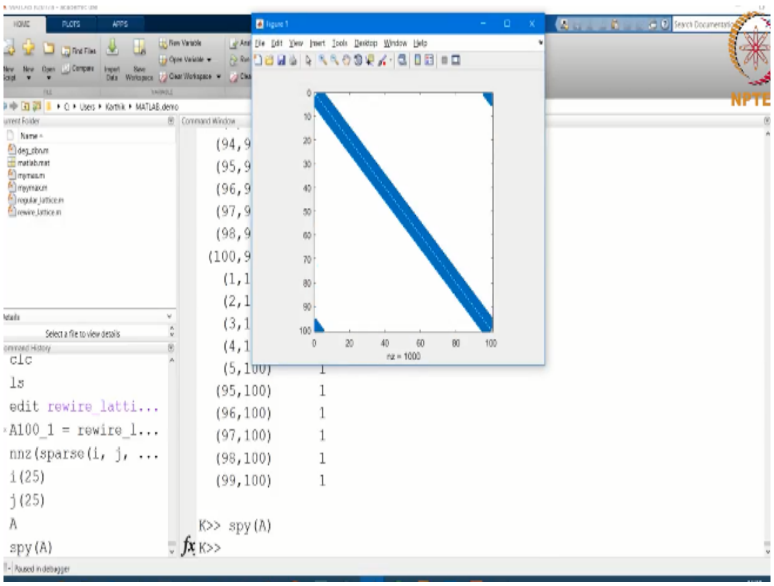


It has got a clique, cycle, Erdos Reyni, grid So, we have to make our own graphs okay let me see if I can anybody has got rewiring right.

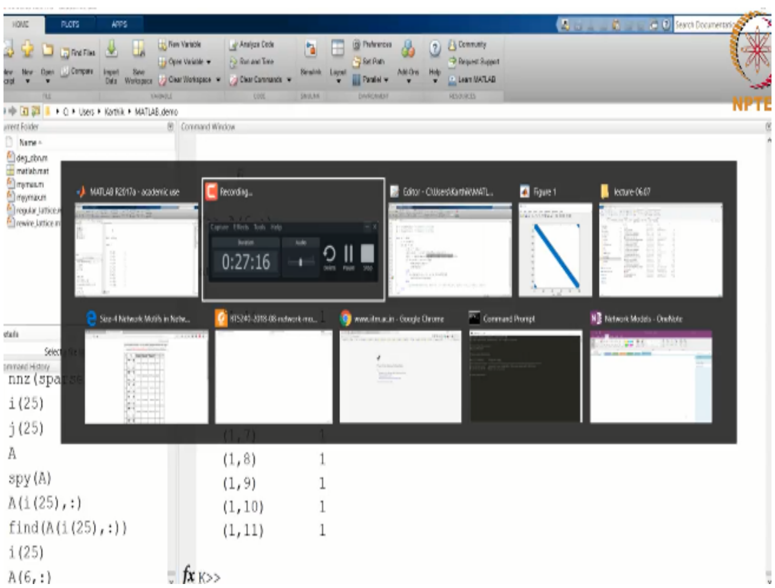
(Refer Slide Time: 04:41)



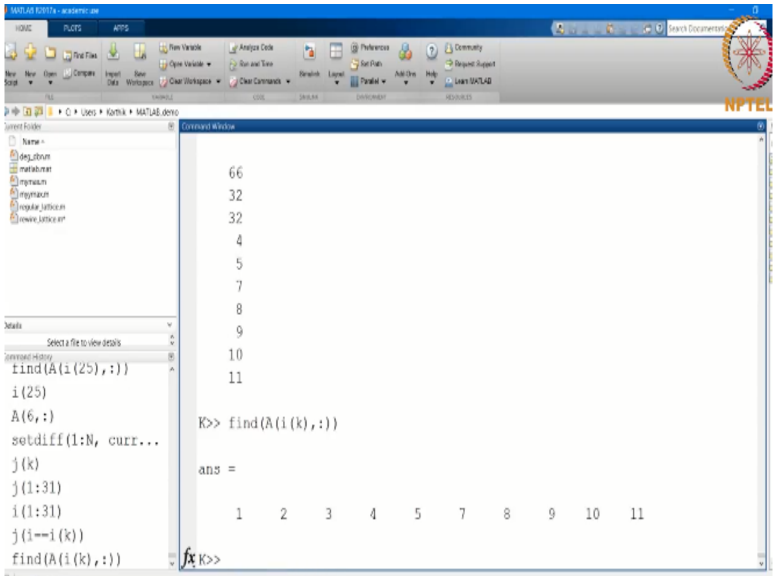
(Refer Slide Time: 07:20)



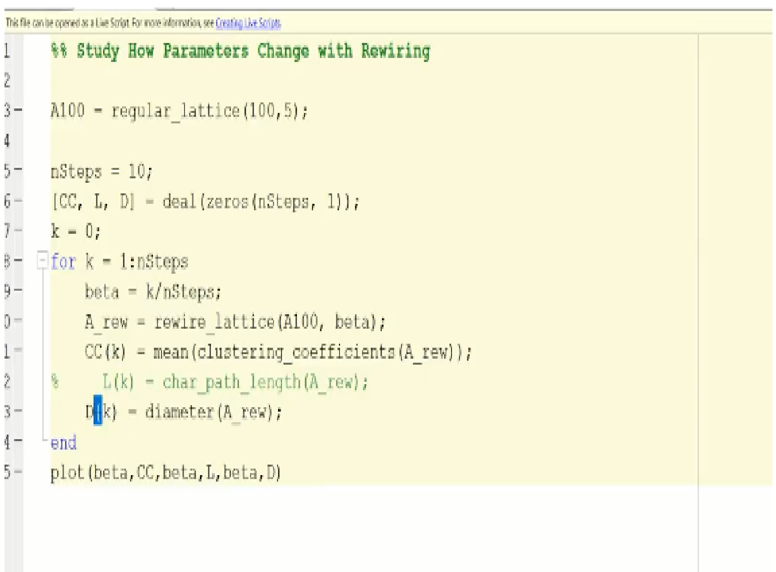
(Refer Slide Time: 09:00)



(Refer Slide Time: 10:52)

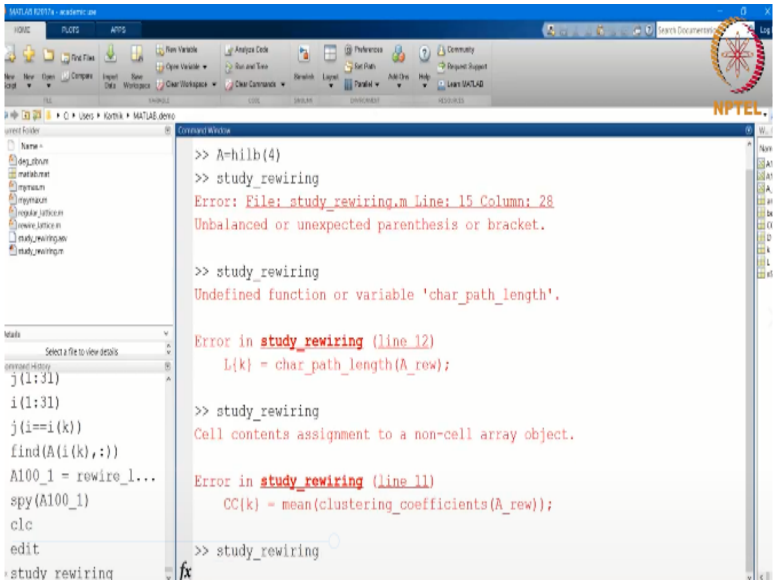


(Refer Slide Time: 12:30)

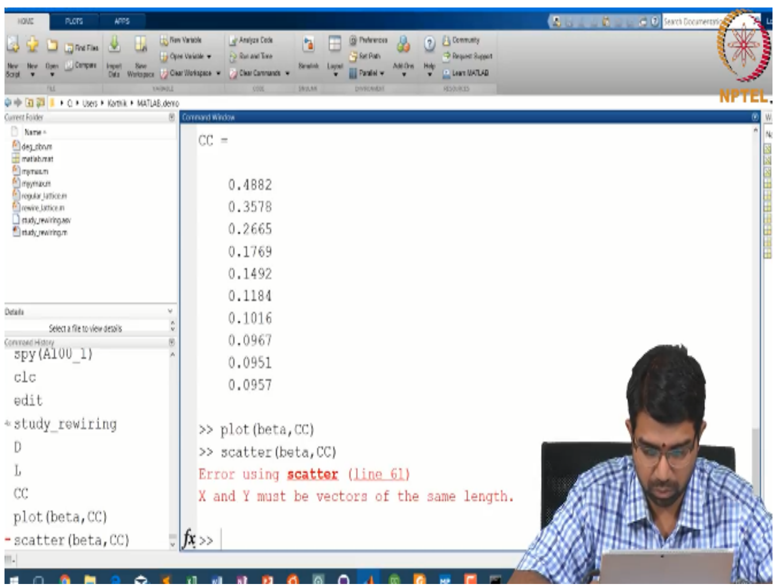


"Professor - student conversation starts" you have only one laptop Ranjith no laptop or not. You do not have a laptop, what about you you have a charger or not, no I did not bring Rohini you have a charger "**Professor - student conversation ends.**"

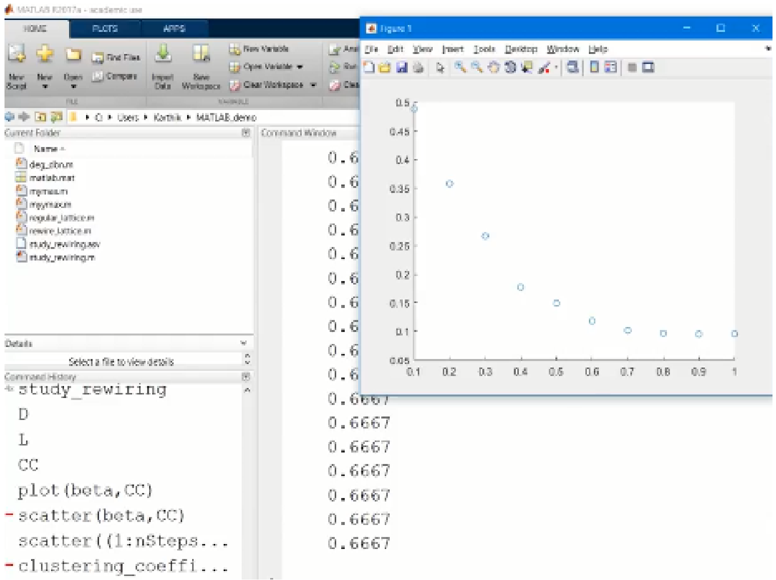
(Refer Slide Time: 19:39)



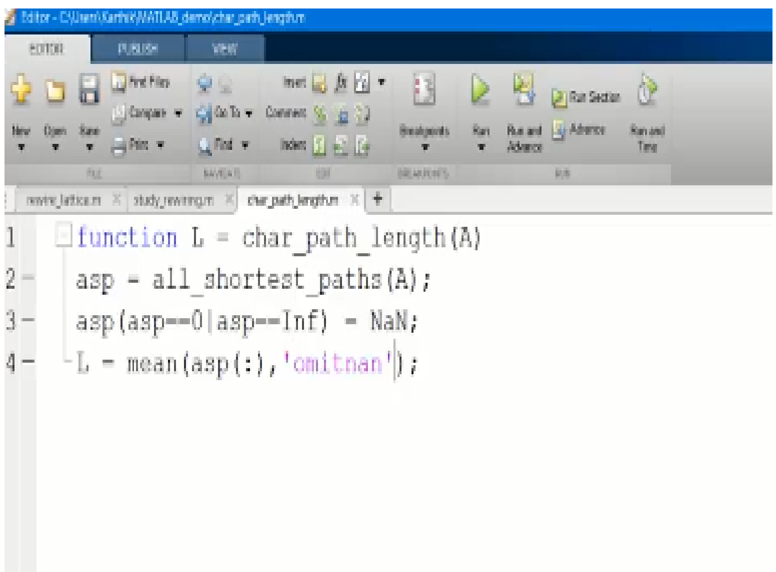
(Refer Slide Time: 20:46)



(Refer Slide Time: 21:36)

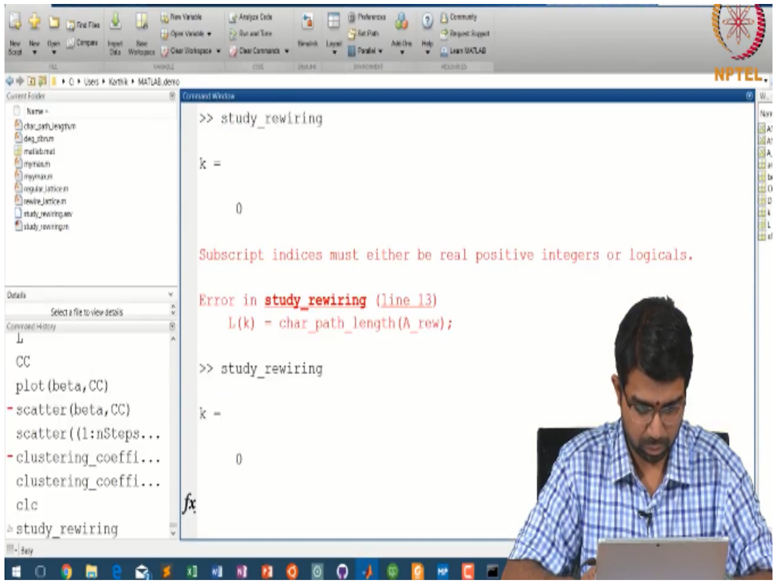


(Refer Slide Time: 22:43)



Yes, yeah tell me how is it going **"Professor - student conversation starts"** you ranked by degree or by betweenness centrality first do degree "**Professor - student conversation ends."**

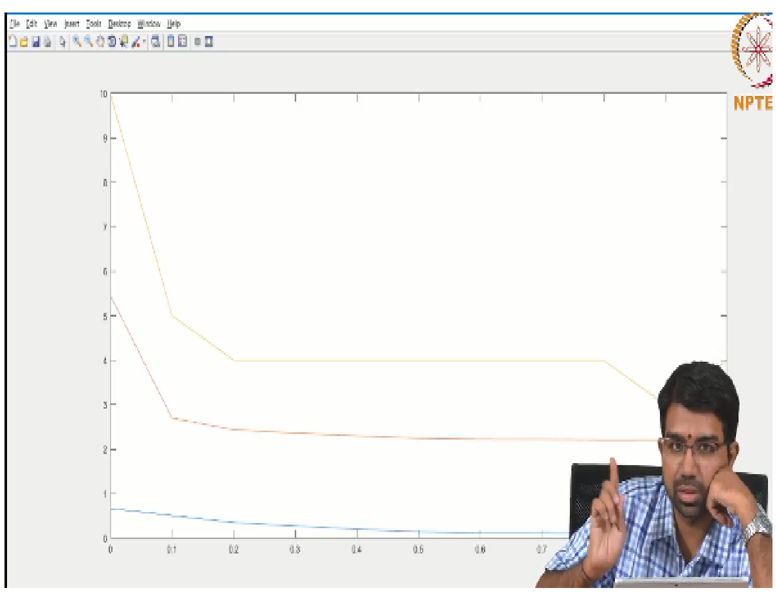
(Refer Slide Time: 25:17)



How is it going? It is fine how did you understand it **“Professor - student conversation starts”** (()) (27:21) so basically I want you to create adjacency matrix then you need to see what are all the so for constructing adjacency matrix you need to find out what are all the nodes that you need to connect to. So, if your node is numbered 5 and you need to connect to 3 on each side. You need to connect 432 on the left hand side and 678 on the right hand side.

Which you can get by MATLAB indexing. So, 678 is basically 5+1 to 5+K and 5-1 to 5-K+1 something like that and you will have a problem for nodes in the end where you will get negative numbers for which you have to use MOD function “**Professor - student conversation ends”.**

(Refer Slide Time: 28:09)



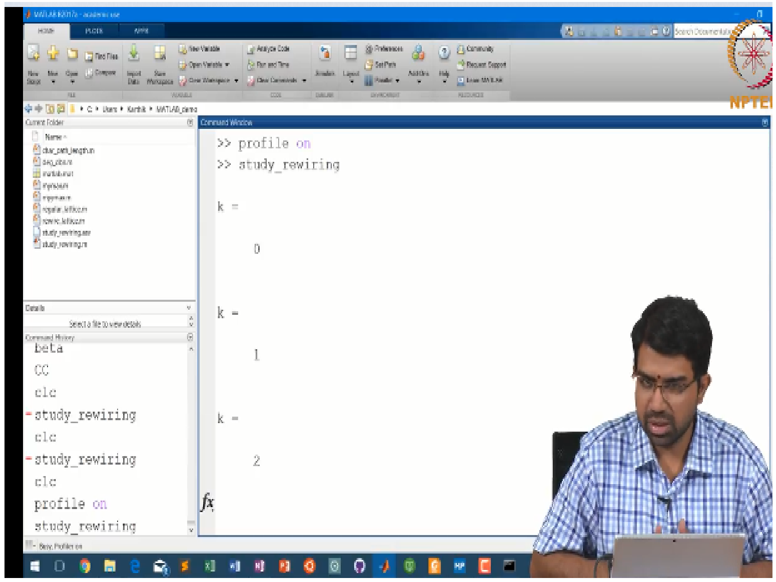
Here are the different things, I think this is diameter this may be characteristic path length this is clustering co efficient **“Professor - student conversation starts”** X axis is beta, Y axis is any of these parameters, rewiring regular lattice, I think this is characteristic path length, this is diameter, this is clustering co efficient what is the x axis beta increasing probability of rewiring, it is 100%rewired 0% rewired**.**

But there should be a significant increase in the clustering co efficient no no that is compared to random. Rewiring compared to random oh well wait, wait, wait no, you are right clustering co efficient decreases may be this is not a good example to look at maybe we look at larger values I perceive, code is too slow, I have to optimize it. So, clustering co efficient how does it change from a regular lattice.

When you rewire it should increase well but the regular lattice is already very highly clustered.it is just clustered in a different way. No what is the most interesting decrease drastically that is something you clearly see, see in one shot in first stimulation it is half 10%rewiring and characteristic path length has become half which makes sense. I think this makes sense this is far too slow.

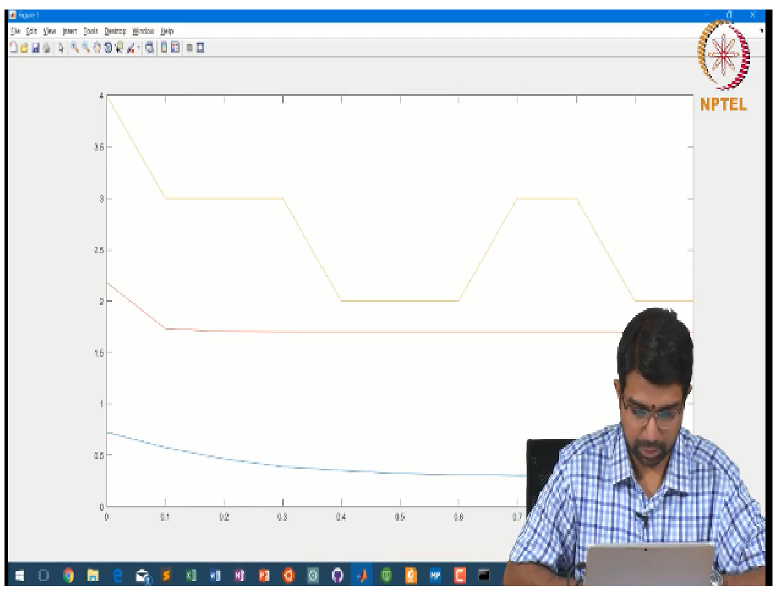
So, let us I will just show one interesting concept, so this is just a simple piece of code that does rewiring. When I have done it I feel it is quite slow it takes about 5 seconds roughly for 5 set of to study 1 set of rewiring **Professor - student conversation ends”.**

(Refer Slide Time: 31:08)



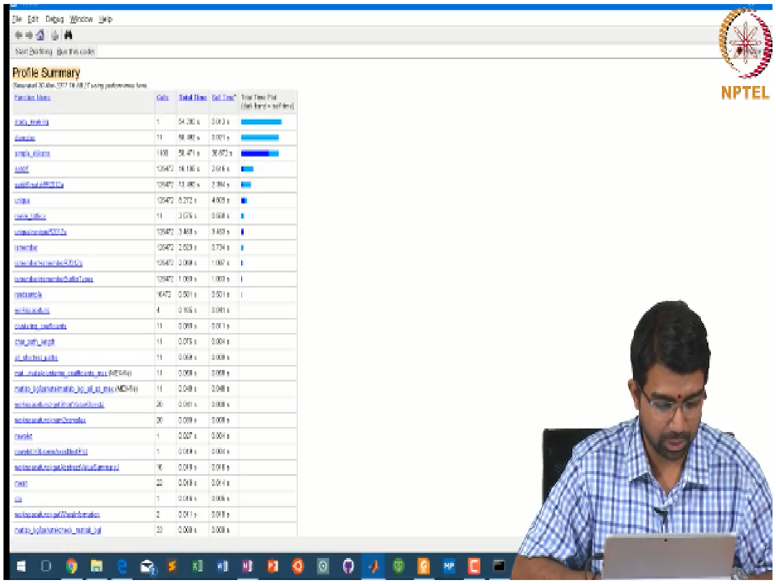
It is a very interesting way to study a performance of a program you can profile your program now I just save this I said profile on and I am now running the program, it is running as usual, so it is now almost finished this is perceivably plotted.

(Refer Slide Time: 32:16)



The characteristic path length is sort of fluctuating here in this case, but what you can now do is.

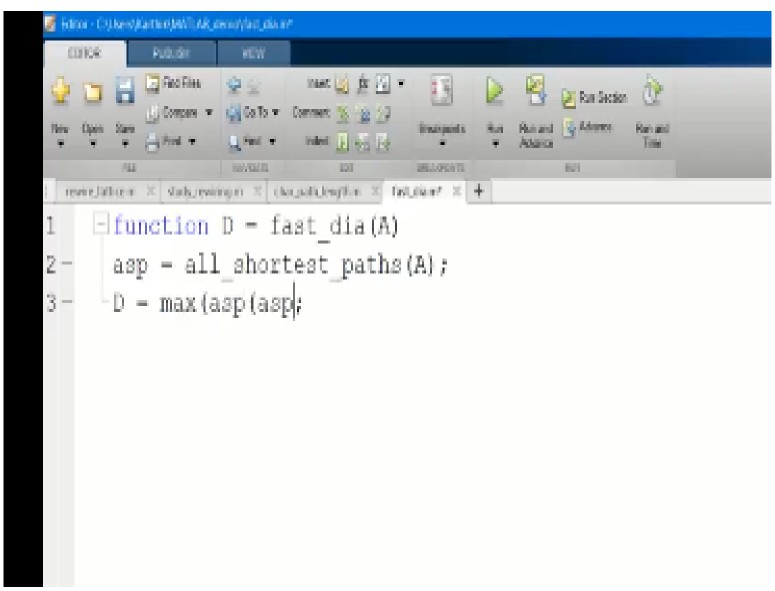
(Refer Slide Time: 32:26)



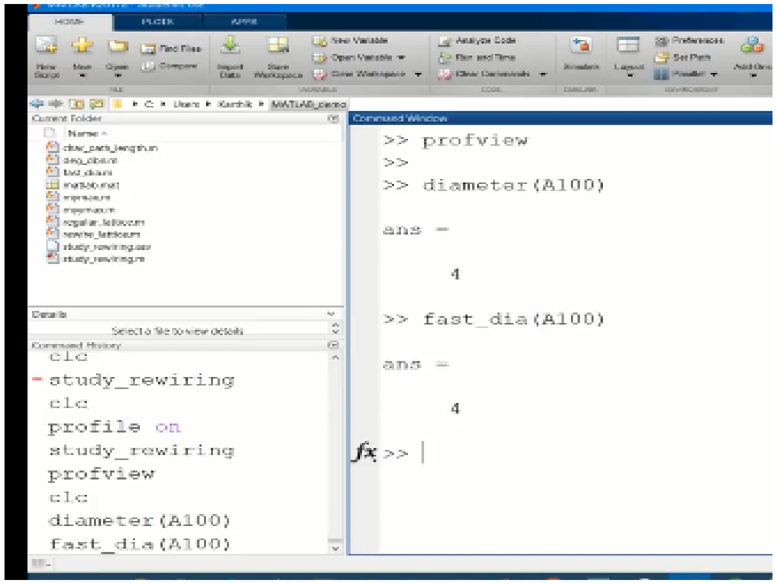
You can now view the profile which tells you where your program is spending its time.so, it took a total of 76 seconds to run this program. Out of which 76 is between the time we started and finished it. Looked at the plot and all that. So, study rewiring ran for 54.28 seconds right and it almost spent no time in itself self-time is 0.13 seconds. It spent a lot of time doing its code called simple Dijkstra it is used for diameter.

That seems a little slow for example all shortest paths seem even faster. So, looks like this is the killer here. So, what we can do is let us.

(Refer Slide Time: 33:44)

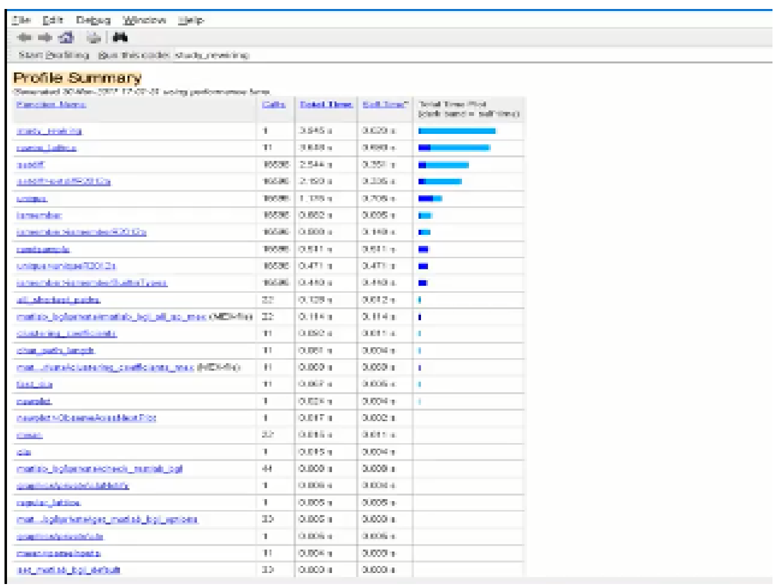


(Refer Slide Time: 34:22)



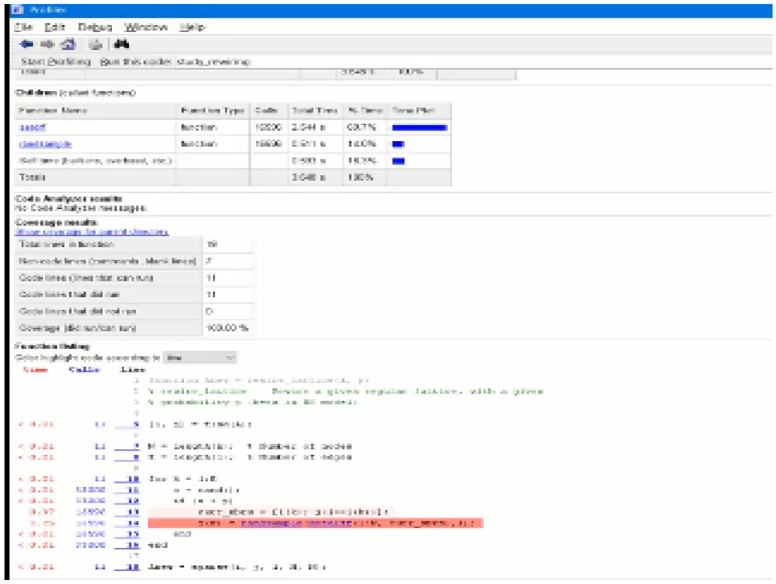
Okay now let us straight away repeat it first let us test it out, it already takes a few seconds it takes all the time. Okay diameter is 4 is 4 this is fast.

(Refer Slide Time: 34:51)



Okay, Now let us re run the code, I did a typo 4seconds so now where is all the time going in you are doing set diff.

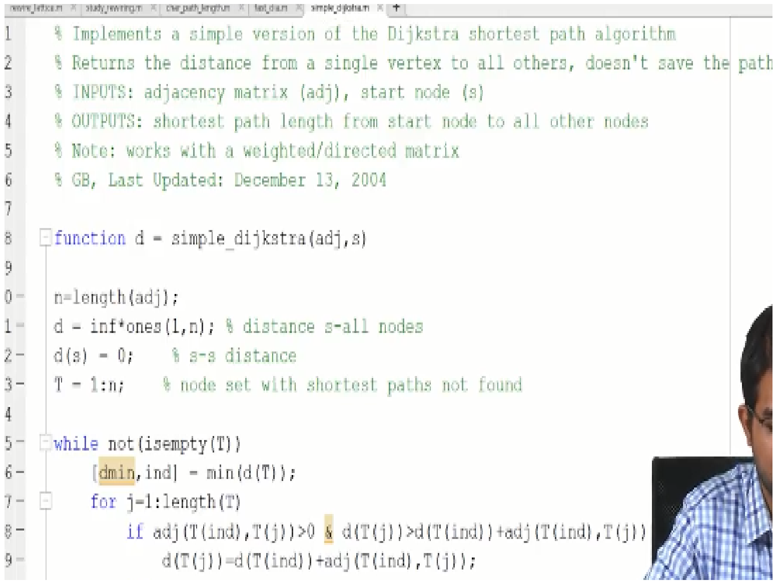
(Refer Slide Time: 35:36)



Cannot really help some of these right so I will show you one moment so you click study rewiring it tells you have much time spent in each of these lines so it made 11 calls to this 1 call to many of these it spent 3.65 seconds here out of the 3.65 seconds here it spent 2.35 seconds in rand sample and set diff and it spent 70% of the time in set diff and 14 15% of the time in rand sample.

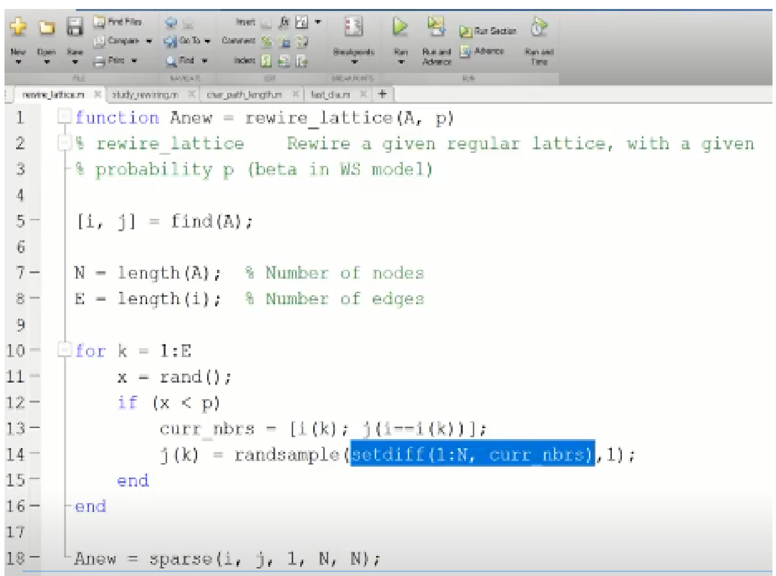
Can you speed it up can you avoid using set diff can you so something else I think I can set up some sort of a mask. Let us try that out so this is how you actually debug a piece of code and make it faster. So, you incrementally find out what is the bottle neck and try to get it of the bottle neck. Sometimes you would not be successful let us see if we are successful here. So, it turns out the Dijkstra using was (()) (37:11) for whatever reason. So, I do not even know where the Dijkstra is coming up from let us see

(Refer Slide Time: 37:21)



Yeah this is cobra tool bool It is somehow a slowish Dijkstra fair enough.

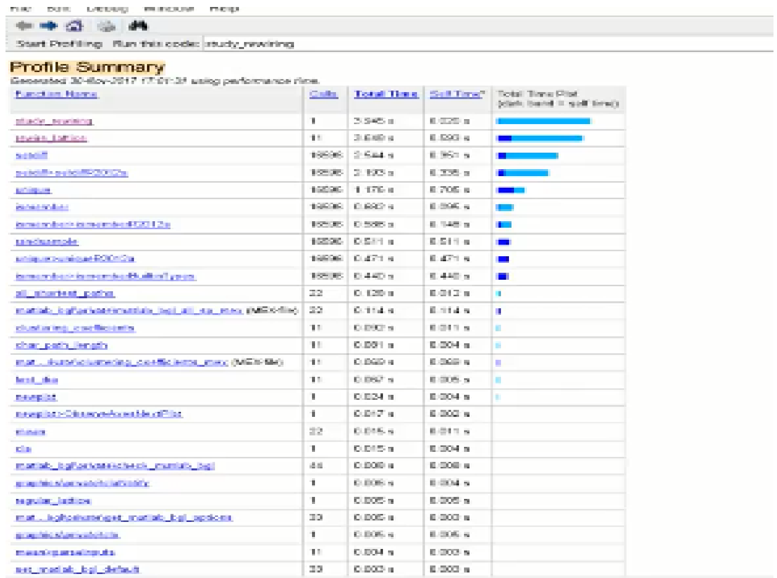
(Refer Slide Time: 37:31)



Now let us see we want to speed up setdiff what is that I am achieving here I am trying to find neighbors that I do not have at the moment. Right I want to rewire every node to a different neighbor. So, I first find the existing neighbors **"Professor - student conversation starts"** What is setdiff setdiff A-B set difference set A-Set B the written elements on set A is not in set B **"Professor - student conversation ends."**

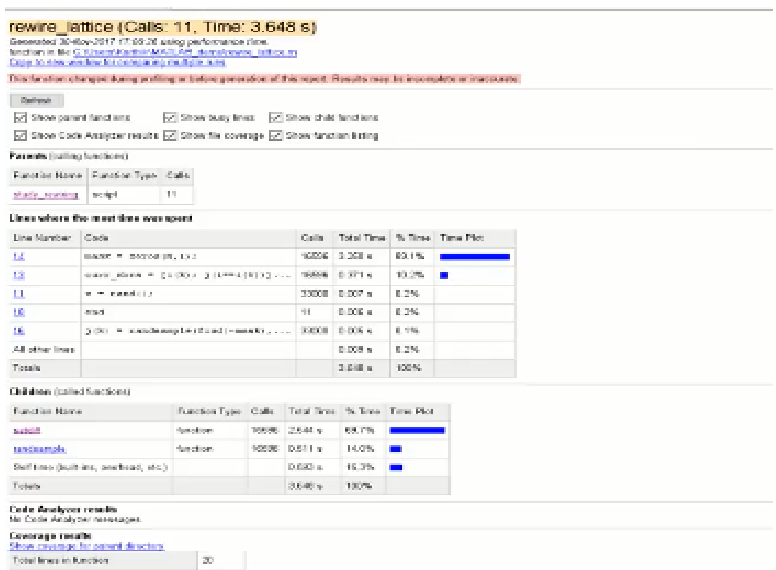
So, this is the list of all nodes from which I want to get rid of those which are present in my current neighborhood. Right so one way of doing this would be so let us see how much time this is taking.

(Refer Slide Time: 40:09)



So setdiff 2.54 seconds so what I can do is I can let us go to rewire lattice that is where all the time was.

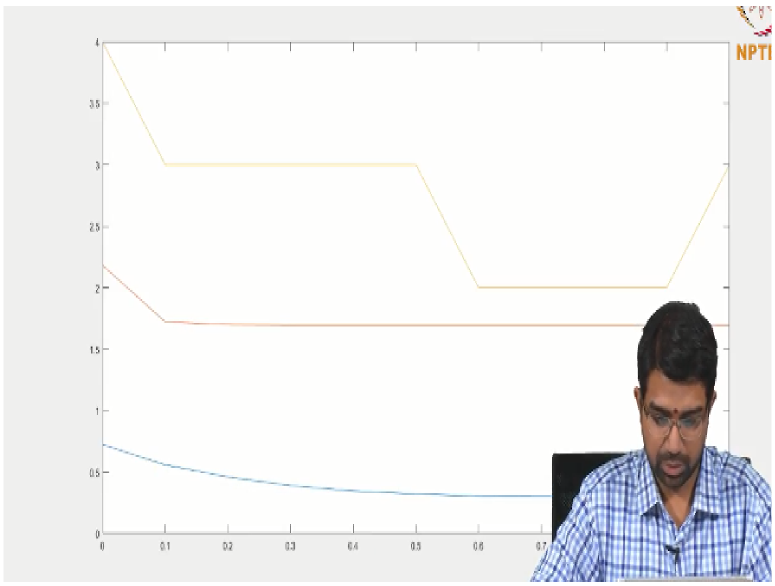
(Refer Slide Time: 40:30)



It has already changed anyway copy to new window for comparing multiple runs. Anyway this number you can imagine it spent 89.1% in set diff. Right so let us try to run it up again. Can you see the difference it has now gone down to from 3.25 to 1.732 seconds and now it is spending time in randsample which we really cannot afford to you cannot avoid it? Right you need to generate random numbers you need to rand sample I think now it will hit the ceiling.

I do not think we can make this code faster. But essentially had so what I am doing here I am first getting the bunch of current neighbors and then I am setting up mask which is all zeros and then I am essentially trying to find out what are all the nodes that are not currently in my neighborhood. So, I am saying I am first assuming that all nodes are not in my neighborhood and getting rid of those currently in the neighborhood in a sense. So, let us run this code now so that was really fast.

(Refer Slide Time: 42:13)



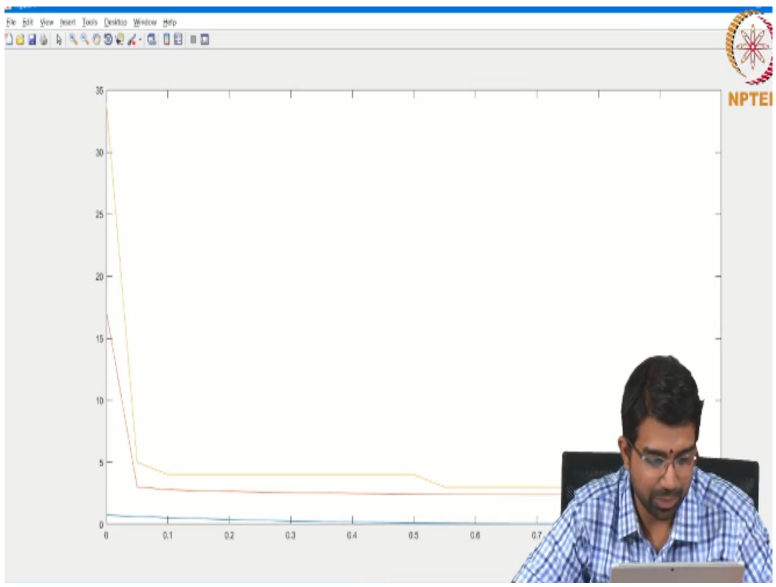
Looks fair enough looks very similar to what we started off with but we can study it for with more Grammarly. So you can use a larger lattice let us re ambitious more steps it is still respectable right compared to what it was this is going to take half a minute may be this is not too bad given value started of **"Professor - student conversation starts"** sir if there Is already an edge.

What do you mean by already an edge? We are going to rewire it you mean we have already rewired it there are 2 edges no there is no 2 edges there is only 1 edge we are worrying about. You pick a node we find its edges, there are say 4 edges correct 2 edges are less than the random like okay you want to rewire 2 of those yes what if I rewire the 2 edges that is also rewiring.

No, it is already a neighbor right you would not rewire it for this node you are saying okay let us spell it out you have A and you have C and D. Let us say as neighbors of A these two so when you rewire C you will never rewire it to D because D is already a neighbor that is what I am trying to do. So, find a node that is not in my current neighbor list and require it to that. So, revoke this X+K X-K find something else across the table and lattice and rewire it to it.

Professor - student conversation ends." Your question gave us enough time to finish the separation let us look at**.**

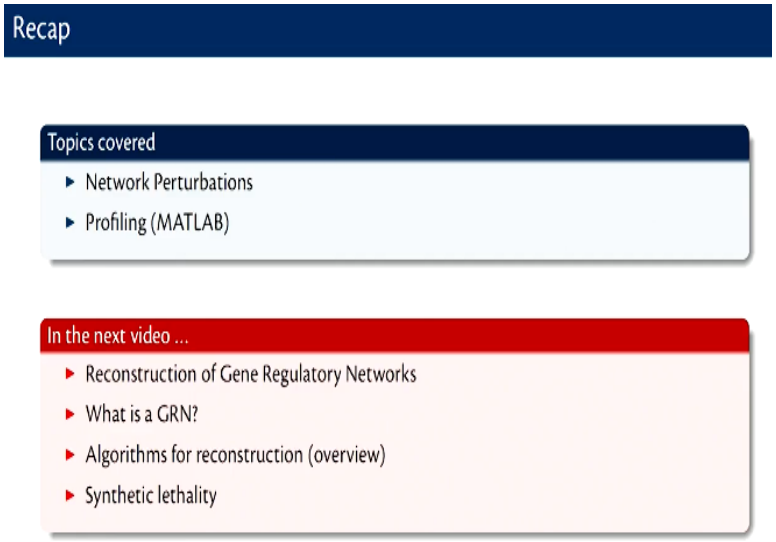
(Refer Slide Time: 44:29)



Look at this all the action happens in the first shot right the first rewiring makes a sea change. The initial diameter characteristic path length is 35 it comes down crashing to 5. For a rewiring of 1/25 4% **"Professor - student conversation starts"** (()) (45:03) well yeah it depends how it is not large because for a function of NK it is kind of complex but yes. So, you can see if you keep varying K.

How does this graph change if you keep N? how does this graph change and so on **Professor - student conversation ends"**

(Refer Slide Time: 45:20)



In today video I hope you had a good picture of how we perturb networks and how we profile MATLAB codes to try and improve how fast they run and so on. But the basic concept is try to avoid loops because MATLAB never does loops efficiently. In the next video we will switch gears and move on to reconstruction of biological networks. We studied a lot of tools and tricks to study biological networks.

But how do you build these networks in the first place yes you can get something of the string database but how do you build these networks in the first place. So, we will look at how do we reconstruct Gean regulatory networks and you know some algorithms for reconstruction and the concept of synthetic lethality.