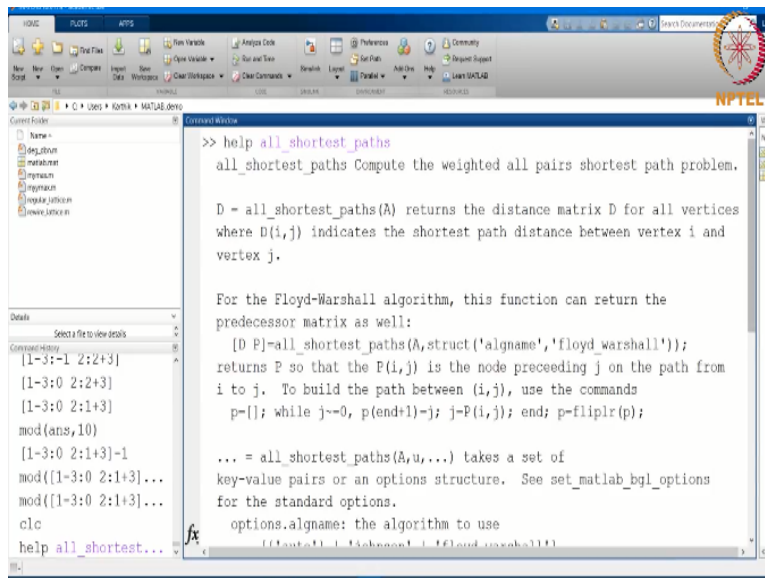


Computational Systems Biology
Karthik Raman
Department of Bio Technology
Indian Institute of Technology-Madras

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)



```
>> help all_shortest_paths
all_shortest_paths Compute the weighted all pairs shortest path problem.

D = all_shortest_paths(A) returns the distance matrix D for all vertices
where D(i,j) indicates the shortest path distance between vertex i and
vertex j.

For the Floyd-Warshall algorithm, this function can return the
predecessor matrix as well:
[D P]=all_shortest_paths(A,struct('algnam','floyd warshall'));
returns P so that the P(i,j) is the node preceding j on the path from
i to j. To build the path between (i,j), use the commands
p=[]; while j~=0, p(end+1)=j; j=P(i,j); end; p=flipr(p);

... = all_shortest_paths(A,u,...) takes a set of
key-value pairs or an options structure. See set_matlab_bgl_options
for the standard options.
options.algnam: the algorithm to use
('floyd warshall','floyd warshall')
```

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 to 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)

```

all_shortest_paths(A,struct('alignname','johnson'))

See also johnson\_all\_sp, floyd\_warshall\_all\_sp.

>> help
>>
>> help erdos_reyni
erdos_reyni Generates a random Erdos-Reyni (Gnp) graph

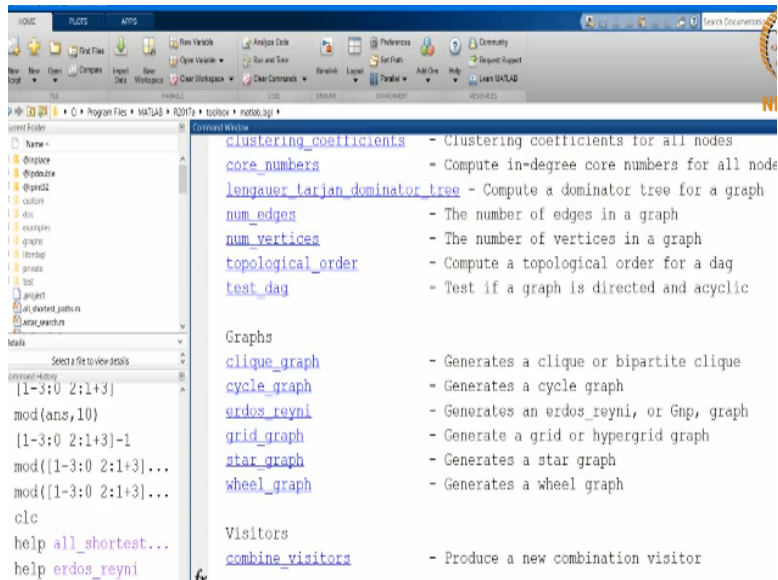
A=erdos_reyni(n,p) generates a random Gnp graph with n vertices and where
the probability of each edge is p. The resulting graph is symmetric.

This function is different from the Boost Graph library version, it was
reimplemented natively in Matlab.

Example:
    A = erdos_reyni(100,0.05);
  
```

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)



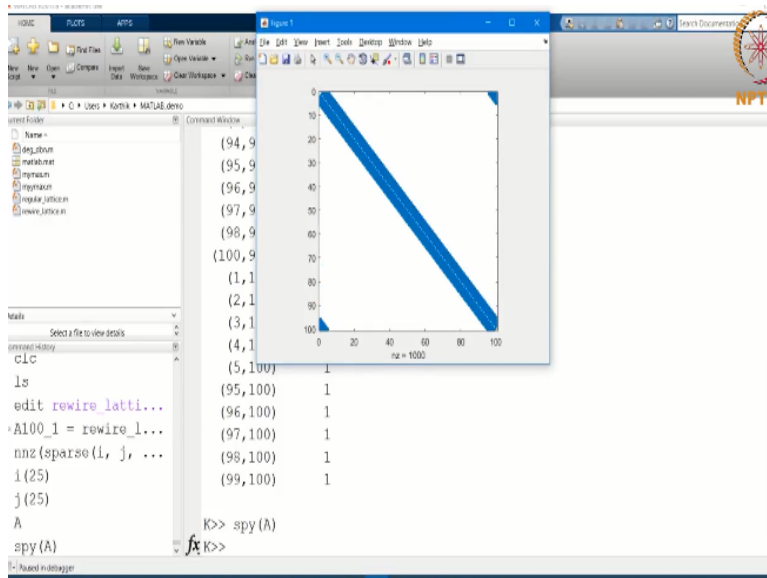
It has got a clique, cycle, Erdos Reyni, grid So, we have to make our own graphs okay let me see if I can anybody has got rewiring right.

(Refer Slide Time: 04:41)

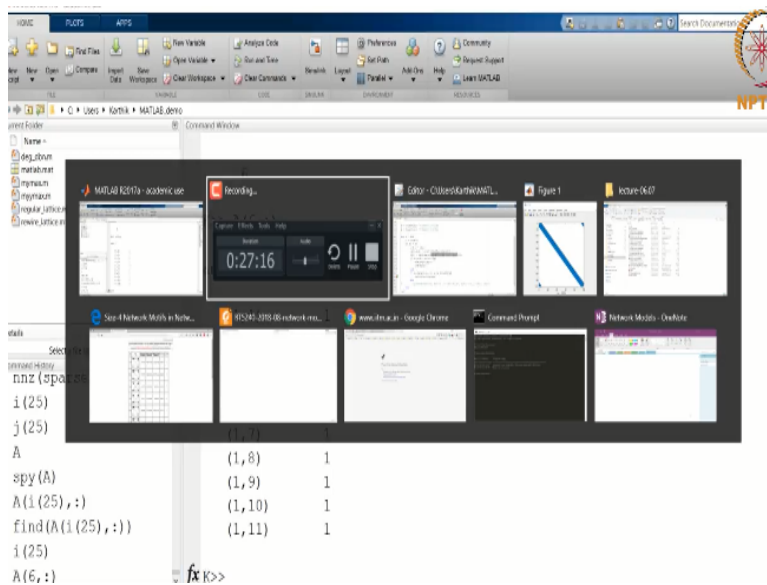
```

6
7- N = length(A); % Number of nodes
8- E = length(l); % Number of edges
9
10- for k = 1:E
11-     x = rand();
12-     if (x < p)
13-         old_j = j(k);
14-         curr_nbrs = [i(k) find(A(i(k),:))];
15-         j(k) = randsample(setdiff(1:N, curr_nbrs),1);
16-         if any(curr_nbrs==j(k))
17-             disp(curr_nbrs)
18-             disp(j(k))
19-         end
20-         nnz(
21-             fprintf('Rewiring [%d %d] ==> [%d %d]\n', i(k), old_j, i(k), j(k));
22-         end
23-     end
24
25- ANSW = sparse(i, 1, 1, N, N);
  
```

(Refer Slide Time: 07:20)



(Refer Slide Time: 09:00)



(Refer Slide Time: 10:52)

```

MATLAB R2014a - academic.mex
File Edit View Insert Window Help
New File New Script New Function New Class New Live Script New App
Open Recent Open Recent
Save Save All
Clear Workspace Clear Command Window
Run Run & Debug
Preferences
Get Path
Add-ons
Help
Learn MATLAB
Search Documents
NPTEL

Current Folder: C:\Users\Korik\MATLAB\demo
Name
log_schum
matlab.m
my_maths
my_maths.m
regular_lattice.m
rewire_lattice.m

Command Window
66
32
32
4
5
7
8
9
10
11

K>> find(A(i(k),:))
ans =
     1     2     3     4     5     7     8     9    10    11

K>>

```

(Refer Slide Time: 12:30)

```

This file can be opened as a Live Script. For more information, see Creating Live Scripts.
1 %% Study How Parameters Change with Rewiring
2
3 A100 = regular_lattice(100,5);
4
5 nSteps = 10;
6 [CC, L, D] = deal(zeros(nSteps, 1));
7 k = 0;
8 for k = 1:nSteps
9     beta = k/nSteps;
10    A_rew = rewire_lattice(A100, beta);
11    CC(k) = mean(clustering_coefficients(A_rew));
12    % L(k) = char_path_length(A_rew);
13    L(k) = diameter(A_rew);
14 end
15 plot(beta, CC, beta, L, beta, D)

```

“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)

The image shows a MATLAB Command Window with several error messages. The script 'study_rewiring.m' contains the following code:

```

>> A=hilb(4)
>> study_rewiring
Error: File: study_rewiring.m Line: 15 Column: 28
Unbalanced or unexpected parenthesis or bracket.

>> study_rewiring
Undefined function or variable 'char_path_length'.

Error in study_rewiring (line 12)
    L(k) = char_path_length(A_row);

>> study_rewiring
Cell contents assignment to a non-cell array object.

Error in study_rewiring (line 11)
    CC(k) = mean(clustering_coefficients(A_row));

>> study_rewiring

```

The Command History shows the following commands:

```

j(1:31)
i(1:31)
j(i==i(k))
find(A(i(k),:))
A100_1 = rewiring_1...
spy(A100_1)
clc
edit
study_rewiring

```

(Refer Slide Time: 20:46)

The image shows a MATLAB Command Window displaying the output of a script. The output is a column vector of values:

```

CC =
    0.4882
    0.3578
    0.2665
    0.1769
    0.1492
    0.1184
    0.1016
    0.0967
    0.0951
    0.0957

```

The Command History shows the following commands:

```

spy(A100_1)
clc
edit
study_rewiring
D
L
CC
plot(beta,CC)
scatter(beta,CC)

```

The Command Window shows a new error message:

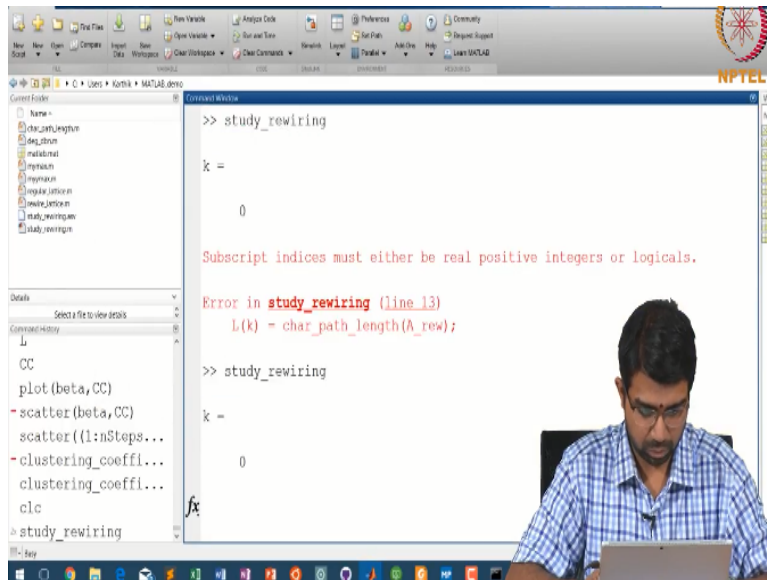
```

>> plot(beta,CC)
>> scatter(beta,CC)
Error using scatter (line 61)
X and Y must be vectors of the same length.

```

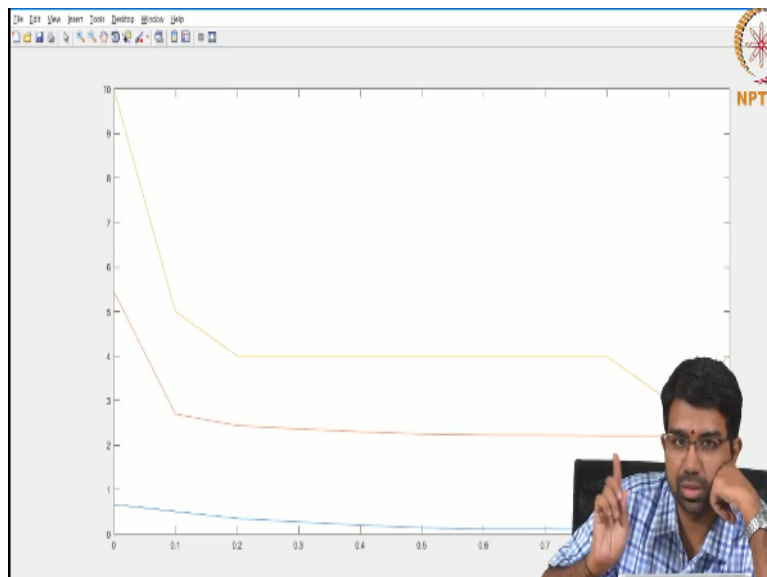
In the background, a person is visible, looking at a laptop screen.

(Refer Slide Time: 21:36)



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)



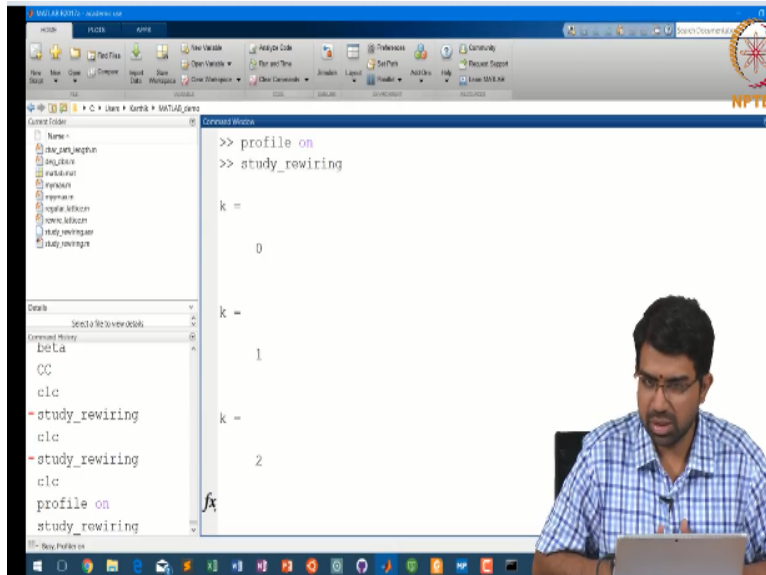
Here are the different things, I think this is diameter this may be characteristic path length this is clustering coefficient “**Professor - student conversation starts**” X axis is beta, Y axis is any of these parameters, rewiring regular lattice, I think this is characteristic path length, this is diameter, this is clustering coefficient 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 coefficient no no that is compared to random. Rewiring compared to random oh well wait, wait, wait no, you are right clustering coefficient 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 coefficient 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.

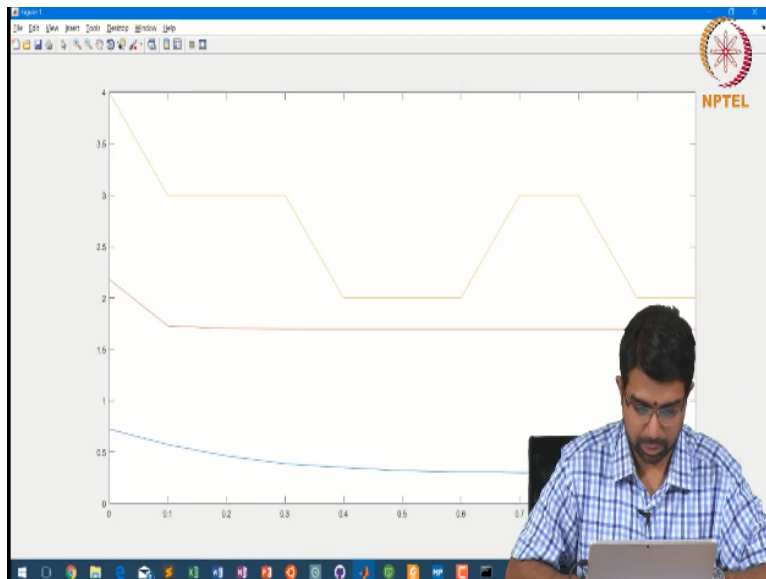
So, let us I will just show one interesting concept, so this is just a simple piece of code that does rewiring. When I have done it I feel it is quite slow it takes about 5 seconds roughly for 5 set of to study 1 set of rewiring **Professor - student conversation ends**”.

(Refer Slide Time: 31:08)



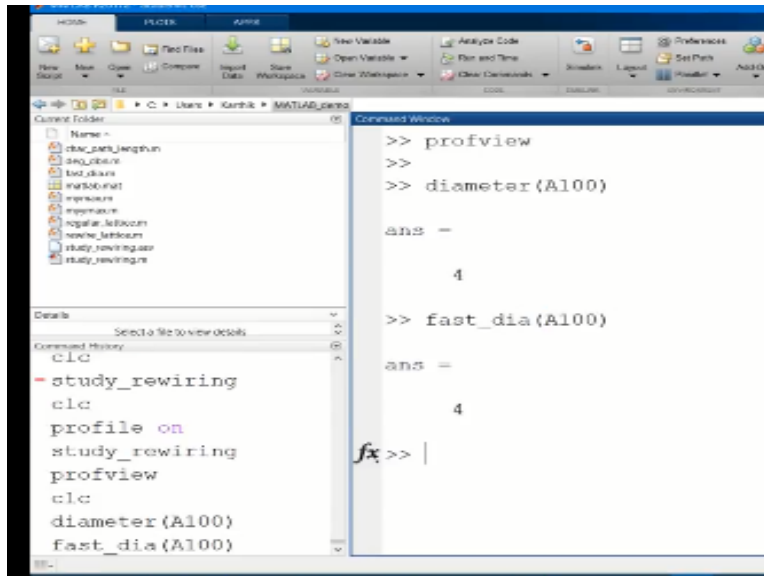
It is a very interesting way to study a performance of a program you can profile your program now I just save this I said profile on and I am now running the program, it is running as usual, so it is now almost finished this is perceptibly plotted.

(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)



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)

| Function Name | Calls | Total Time | Self Time | Total Time (incl. stack frame self time) |
|----------------------|-------|------------|-----------|--|
| main_topLevel | 1 | 3.945 s | 3.929 s | |
| main_topLevel | 11 | 3.848 s | 3.809 s | |
| test01 | 85592 | 2.544 s | 3.251 s | |
| test01out@R2017a | 85592 | 2.509 s | 3.235 s | |
| main | 85598 | 1.375 s | 3.706 s | |
| test01@ | 85592 | 0.802 s | 3.005 s | |
| main_topLevel@R2017a | 85596 | 0.809 s | 3.149 s | |
| main_topLevel | 85596 | 0.811 s | 3.011 s | |
| main_topLevel@R2017a | 85592 | 0.471 s | 3.471 s | |
| main_topLevel@R2017a | 85596 | 0.449 s | 3.449 s | |
| main_topLevel | 22 | 0.128 s | 3.012 s | |
| main_topLevel@R2017a | 22 | 0.114 s | 3.114 s | |
| main_topLevel | 11 | 0.092 s | 3.011 s | |
| main_topLevel | 11 | 0.091 s | 3.004 s | |
| main_topLevel@R2017a | 11 | 0.009 s | 3.009 s | |
| main_topLevel | 11 | 0.007 s | 3.006 s | |
| main_topLevel | 1 | 0.024 s | 3.004 s | |
| main_topLevel@R2017a | 1 | 0.017 s | 3.002 s | |
| main_topLevel | 22 | 0.016 s | 3.011 s | |
| main_topLevel | 1 | 0.015 s | 3.004 s | |
| main_topLevel@R2017a | 44 | 0.009 s | 3.009 s | |
| main_topLevel@R2017a | 1 | 0.006 s | 3.006 s | |
| main_topLevel | 1 | 0.005 s | 3.005 s | |
| main_topLevel@R2017a | 22 | 0.005 s | 3.009 s | |
| main_topLevel@R2017a | 1 | 0.006 s | 3.006 s | |
| main_topLevel@R2017a | 11 | 0.004 s | 3.009 s | |
| main_topLevel@R2017a | 22 | 0.003 s | 3.009 s | |

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)


```

1 % Implements a simple version of the Dijkstra shortest path algorithm
2 % Returns the distance from a single vertex to all others, doesn't save the path
3 % INPUTS: adjacency matrix (adj), start node (s)
4 % OUTPUTS: shortest path length from start node to all other nodes
5 % Note: works with a weighted/directed matrix
6 % GB, Last Updated: December 13, 2004
7
8 function d = simple_dijkstra(adj,s)
9
10 n=length(adj);
11 d = inf*ones(1,n); % distance s-all nodes
12 d(s) = 0; % s-s distance
13 T = 1:n; % node set with shortest paths not found
14
15 while not(isempty(T))
16     [dmin,ind] = min(d(T));
17     for j=1:length(T)
18         if adj(T(ind),T(j))>0 && d(T(j))>d(T(ind))+adj(T(ind),T(j))
19             d(T(j))=d(T(ind))+adj(T(ind),T(j));
20

```

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

(Refer Slide Time: 37:31)

```

1 function Anew = rewire_lattice(A, p)
2 % rewire_lattice Rewire a given regular lattice, with a given
3 % probability p (beta in WS model)
4
5 [i, j] = find(A);
6
7 N = length(A); % Number of nodes
8 E = length(i); % Number of edges
9
10 for k = 1:E
11     x = rand();
12     if (x < p)
13         curr_nbrs = [i(k); j(i==i(k))];
14         j(k) = randsample(setdiff(1:N, curr_nbrs), 1);
15     end
16 end
17
18 Anew = sparse(i, j, 1, N, N);

```

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)

| Function Name | Calls | Total Time | Self Time | Total Time Plot (Self time + self time) |
|---------------------------|-------|------------|-----------|---|
| study_rewiring | 1 | 3.545 s | 0.225 s | |
| rewire_lattice | 11 | 3.648 s | 0.593 s | |
| setdiff | 10096 | 2.544 s | 0.951 s | |
| randperm(100000) | 10096 | 2.193 s | 0.336 s | |
| randi | 10096 | 1.175 s | 0.706 s | |
| rand | 10096 | 0.692 s | 0.096 s | |
| randperm(100000, 'int32') | 10096 | 0.598 s | 0.148 s | |
| randstr | 10096 | 0.511 s | 0.511 s | |
| randperm(100000, 'int32') | 10096 | 0.471 s | 0.471 s | |
| randperm(100000, 'int32') | 10096 | 0.440 s | 0.440 s | |
| randi(100000, 'int32') | 22 | 0.128 s | 0.012 s | |
| randi(100000, 'int32') | 22 | 0.114 s | 0.114 s | |
| randi(100000, 'int32') | 11 | 0.092 s | 0.011 s | |
| randi(100000, 'int32') | 11 | 0.091 s | 0.004 s | |
| randi(100000, 'int32') | 11 | 0.060 s | 0.060 s | |
| randi(100000, 'int32') | 11 | 0.067 s | 0.006 s | |
| randi(100000, 'int32') | 1 | 0.024 s | 0.004 s | |
| randi(100000, 'int32') | 1 | 0.017 s | 0.002 s | |
| randi(100000, 'int32') | 22 | 0.016 s | 0.011 s | |
| randi(100000, 'int32') | 1 | 0.015 s | 0.004 s | |
| randi(100000, 'int32') | 44 | 0.008 s | 0.008 s | |
| randi(100000, 'int32') | 1 | 0.006 s | 0.004 s | |
| randi(100000, 'int32') | 1 | 0.005 s | 0.005 s | |
| randi(100000, 'int32') | 22 | 0.005 s | 0.003 s | |
| randi(100000, 'int32') | 1 | 0.005 s | 0.006 s | |
| randi(100000, 'int32') | 11 | 0.004 s | 0.003 s | |
| randi(100000, 'int32') | 22 | 0.003 s | 0.003 s | |

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)

rewire_lattice (Calls: 11, Time: 3.648 s)

Generated 30-Nov-2017 17:09:26 using performance time.
Function in file C:\Users\Bart\My Documents\rewire_lattice.m
Click to show analysis for children in function.

This function changed during writing or before generation of this report. Results may be incomplete or inaccurate.

Parent (calling functions)

| Function Name | Function Type | Calls |
|----------------|---------------|-------|
| study_rewiring | script | 11 |

Lines where the most time was spent

| Line Number | Code | Calls | Total Time | % Time | Time Plot |
|-----------------|----------------------------------|-------|------------|--------|-----------|
| 14 | setdiff = setdiff(a, b) | 10096 | 3.208 s | 89.1% | |
| 15 | randi(100000, 'int32') | 9999 | 0.971 s | 93.2% | |
| 11 | n = randi(100000, 'int32') | 33008 | 0.007 s | 0.2% | |
| 16 | rand | 11 | 0.006 s | 0.2% | |
| 15 | j(0) = randperm(100000, 'int32') | 33008 | 0.006 s | 0.1% | |
| All other lines | | | 0.008 s | 0.2% | |
| Totals | | | 3.648 s | 100% | |

Children (called functions)

| Function Name | Function Type | Calls | Total Time | % Time | Time Plot |
|--------------------------------------|---------------|-------|------------|--------|-----------|
| setdiff | function | 10096 | 3.208 s | 89.1% | |
| randi(100000, 'int32') | function | 10096 | 0.511 s | 14.0% | |
| Self time (self-ans, self-ans, etc.) | | | 0.630 s | 15.3% | |
| Totals | | | 3.648 s | 100% | |

Code Analyzer results
Mx Code Analyzer snapshots.

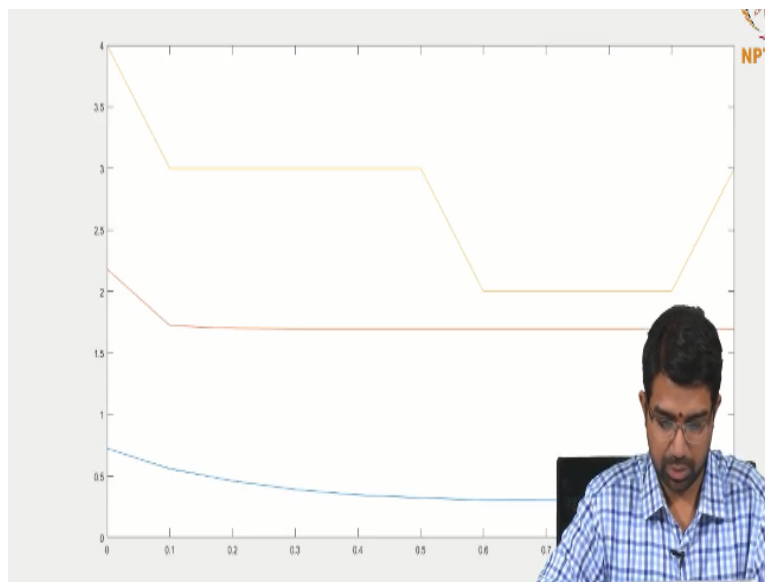
Coverage results
Show coverage for selected functions.

Total lines in function: 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 rand sample 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.

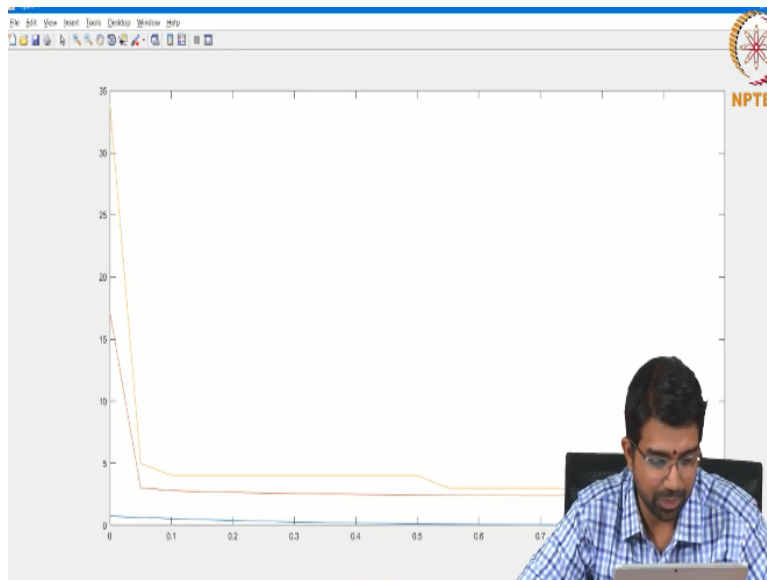
What do you mean by already an edge? We are going to rewire it you mean we have already rewired it there are 2 edges no there is no 2 edges there is only 1 edge we are worrying about. You pick a node we find its edges, there are say 4 edges correct 2 edges are less than the random like okay you want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

Recap

Topics covered

- ▶ Network Perturbations
- ▶ Profiling (MATLAB)

In the next video ...

- ▶ Reconstruction of Gene Regulatory Networks
- ▶ What is a GRN?
- ▶ Algorithms for reconstruction (overview)
- ▶ Synthetic lethality

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 Gene regulatory networks and you know some algorithms for reconstruction and the concept of synthetic lethality.