NPTEL
NPTEL ONLINE CERTIFICATION COURSE

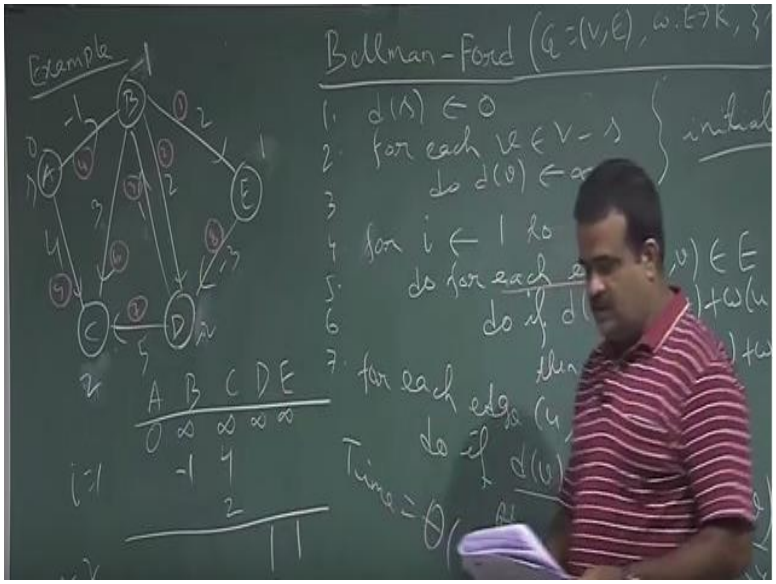
Course Name
Fundamental Algorithms:
Design and Analysis

by
Prof. Sourav Mukhpadhyay
Department of Mathematics
IIT Kharagpur

Lecture 19: Bellman Ford

Okay, so we will talk about Bellman Ford algorithm. So this is a, this can handle the negative edges, so here we are allowing the negative cycle. So like in Dijkstra's we restrict our self from the negative cycle. So for that we took our, this weights are all positive so that there should not be any negative cycle. But here this is, this can handle that Bellman Ford.

(Refer Slide Time: 00:49)



So here our input is this is also a singles of shortest problem, so we have a graph (V, E) and here our age is we are allowing negative age also. And we have a source vertex and either it should give us δ(u, v) from all other vertex or it should be put that there is a negative cycle in the graph.

Okay, so let us write the code for the Bellman Ford algorithm. This is (G = (V, E), w from E to R and we have a source findings. Okay, so initialization step is same as the Dijkstra's so we will just assign by the d(S) is 0 and the degree of any other vertices as infinity v-s that means these are not yet explored. So this is the initialization step which is same as the initialization, which is same as initialization which is same as the Bellman Ford, Dijkstra's algorithm.

Now what we do, we do the branch of relaxation, we check each edge and we will see whether we can relax that corresponding vertex, so that is the idea. So and how many times we will do, we will do this, we will have a loop from a number of vertices, we will see what is, why this is up to E times, do for each vertex, this is another loop each age (u, v) belongs to E we check whether we put relax so we take a edge (u, v) and we check whether we could relax this v or not.

Do if d(v) is greater than u+ then we do the relaxation. So this is basically what is our relaxation, this is the relaxation state, relaxing the relaxation, okay. So we have a outer loop, this outer loop is working of E times, E-1 times around E times, and in the inner loop this is also working up to the number of E times.

Basically this is we are checking each of the vertices, so we have many vertices in the graph, we are checking each of the vertices, this is one vertex and we are checking whether we could relax that corresponding v or not. And these, so number, so huge number of relaxation we are doing. So this is the way, now after doing this huge number of relaxation, if we still could relax a edge, that means there is a chance of negative cycle.

So that means definitely there is a negative cycle exist in this graph, so that check we need to do, because we are doing bunch of relaxation even though after also, afterwards, after doing this many relaxation if we could relax further then we have a problem in the graph. So that means we should report it negative cycle.

So that step we need to write, now this is the last step, this is to report the negative cycle for each age, now we just check if we can do or not time relaxation then there is a problem, d(v) is greater

than u+δ (u, v) then there is a problem because we after doing so many relaxation if we could do further relax, if you could further relax the vertex then we report negative cycle, so that means shortest path will not exist, so it will it is either giving us the shortest path δ(S,V) or it will report a negative cycle, so let us so what is the time complexity for this because these are 4, 2 loops one is this v loops and one is C loops.

So time is basically V, E okay now let us just execute this let us take a graph so let us take a graph A, B, C, D sand then say E okay and let us have this A with this so and then so B to this and we have B to E and here, here and here and here and let us just have the weight here we are allowing the negative weight also so based the problem with the negative weight so 1, 2 this is 2 this is -3 these are the edge weight this is 5 okay .

So now what we need to do we need number the edges because in the inner loop we are ruining this for each edge every time so we could number the edges it is easy our line our job will be easier we can follow that same numbering if not necessary you can in a second for i=1 we have to check this for the edges so if you have numbering it is easy but in the i=2 you can have different numbering also but to make it simple we are just the edges suppose this number one edge this is number 2 edge this is number 3 edge and so this I number 4 edge this is number 5 edge then this is number 6 edge this is number 7 edge and then this is a number 8 edge.

Okay so now we write A, B so these all in A, B, C, D, E D so these are the partitions we have and we have this initialization all vertices are infinity other than the source vertex this is the source vertices which is initialized by degree 0 all other vertices are in build now we will check so side is 0 all are infinity now we check with so far i=1 so this is for 1=1 step we check the vertices we check the first sorry we check the edges we check each of edge so we check edge number one and we see whether we could relax this vertex.

So this is u, this is v no because this is infinity this is infinity + 2 infinity so no gain so we check edge number 2 nothing edge number 3 nothing edge number 4 yeah for edge number 4 this is u this v so then is as a chance of relaxing this edge so this will become now -1 okay so this become -1 and then edge number 5, edge number 5 is also this will become last 4 this is C this

will become +4 then edge number 6 edge number 6 will also we can try to see that so this is -1 then+ 3 so this is 2 so 4 will now change to 2.

So this now 2 so this is 2 okay then 7 number 7 edge so this u, this is v so no change, so number 8 both are infinitive no change so this is for i = 1 again we have to take for i =2 now for i =2 again we start from h number 1 but we could use the different numbering also in this case but for simplicity we will use the same numbering edge number 1 so edge number 1 this is u this is v so the degree of this was infinity.

Now though we have it path we can go from edge to this node with a cost -1 then we can take this direct path with 2 so this is +1 so this will be +1, so EE is now +1 and then we have then we have edge number 2 edge number 2 is also change so this -2 this is +2 so this will be again+1, so this D, D is also +1 now so edge number 3 edge number 3 is this one this is u this is also this is 1 and this is 1 so we have a path from edge to this node with a cost 1.

Then you can take the direct path so this is 1, 1+1 = 2 which is not better than -1 so you will not change that then edge number 4 we check edge number also remain unchanged for edge number 5 we have this is 0 so this is 0+ 4 which is not better than 2 so we are not doing anything here so edge number 6 edge number 6 also this is -1 this is also remain same edge number 7 edge number 7 is 1 + 5 so 6 which is not greater than 2.

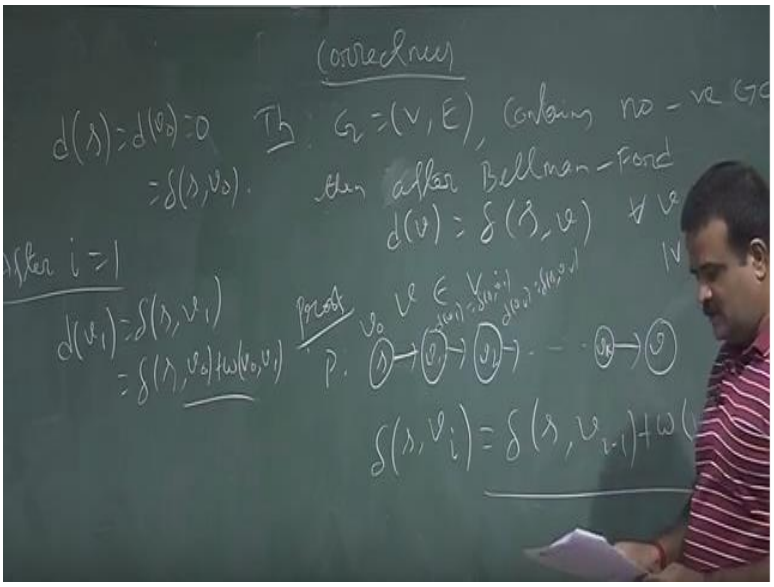
So in do the relaxation edge number 8 also edge number 8 means this is u this is v so we have a path from edge to this node with a cost 1 and then we can take this path -2 so this is where -3 so this is basically -2, so this will give -2 for due to the edge number 8 so this is now -2 okay. So now this is fro i = 2 now for i = 3 we check i = 3 again we follow the same path so 1 there is no change 2 there is no change, 3 there is no change so that's it.

So this is the, these are the basically delta so it is converging so even for i is equal what is the number of vertex? Number of vertex is 1, 2, 3, 4, 5, so we are doing this up to 4 times even for i = 2 times it is converging but if there is a negative cycle are even after running this outer loop 4

times even though we have a we could able to do the relaxation for some vertex, so that is the indication that there is negative change.

But in this case there is no negative cycle so it is converging and these are the deltas, so this is the execution of this Bellman Ford algorithm and this is the runtime, okay. Now the question is why we are going for this many times, this outer look, why it is order of p type? So that we have to check that is the correct mate of this code, so let us just write the correctness of this Bellman Ford algorithm and try to see why it is B type.

(Refer Slide Time: 14:09)



Okay so correctness is coming from this theorem, theorem is telling you have a graph and suppose there is no negative cycle it is graph point end no negative cycle then after running the Bellman Ford then after running the Bellman Ford we should have this degree of each matrices is basically delta of S some of E for all E belongs to this, okay. So to prove this what we are doing, we take a vertex V and we take a path form A: V suppose this is v1, v2 like this…vk like this is a simple path from A: V.

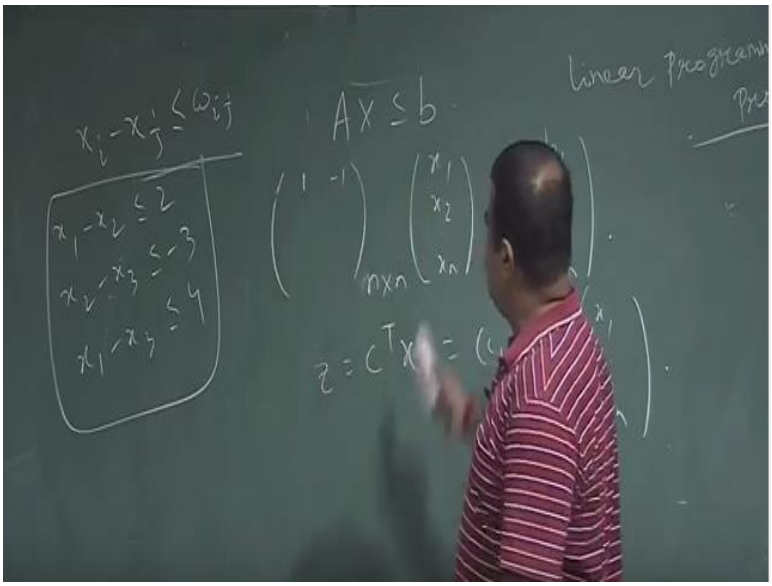
And since there is no negative cycle so that means we can just this is shortest path so can you write this? Delta of AS, BI is basically delta of AS, VI -1 + W (VI -1, VI) this is coming from optimal sharp structure if we have a shortest path from A:B then sub path will give us the shortest path from A: that node and this is also shortest path from edge to this, okay. So this will give us the true idea, so we are trying to see why we are the outer loop is why 5 times, okay so now we can claim that δ(s) which is basically we can denote at this 0, δ(s) is basically 0 which is basically δ(s,v0) because since there is not negative cycle, so to be so if we are in s is greater why to remain at, at s because there is no other way we can minimize that cost, because there is no negative cycle, okay.

So now, so after the first pass after i=1 this means after first path the outer y=1, so this is an edge in the graph so this edge numbering we have so at some point of type if this edge number is 1, then we are lucky, so it will be immediately it will change but it need not be this edge number is 1, so after first path after i=1 so this edge will change and this degree will be basically change it to δ(s,v1) because it is basically δ(s,v0)+w(v0,v1) and by this result it si basically δ(s,v1) because we do not know the number in, but we know that if we taking the all edges then this edge number will come in i=1 in the first sort.

So this will give us, this after first sort, this degree of this will be δ(s,v1) then after second sort i=2 since this is done then this will be again i=2 sort this will be δ(s,v2) degree because it will change because this, edge is also arrange in the graph so this edge number will come they are not be in the next immediately but it will come so once it will come, it will update this degree by the δ. So this is the, this is the correctness so, so now how many in a simple path what is the maximum number of vertices, maximum number of vertices is v so that is why we have t-1 that is the middle maximum so that is why our outer loop is from 1 to -1.

So this is the correctness of this Bellman Ford algorithm, so now we will have an application of this Bellman Ford in a linear programming problem. So how we can solve the LPP or linear programming problem using a Bellman Ford.

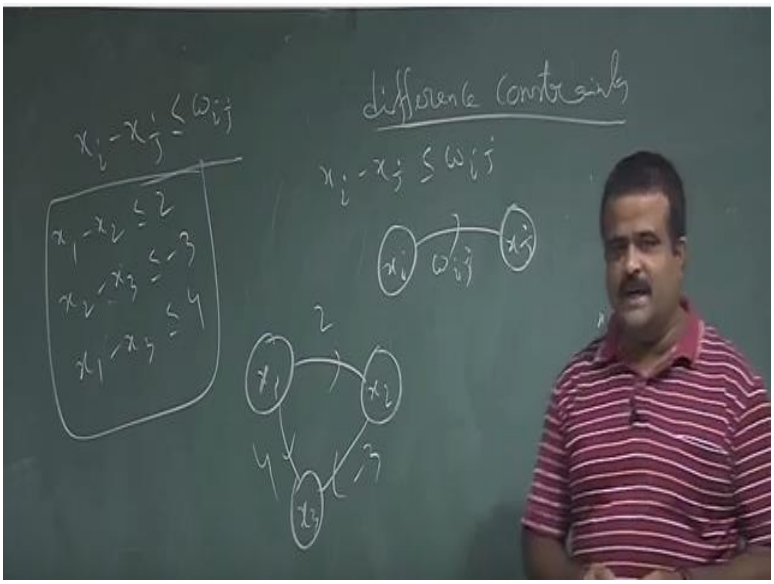
(Refer Slide Time: 19:14)



So we may linear programming problem basically what we have, we have a matrix A and we have some variable X which is less than equal to b and basically we have a matrix m/n of if this implicitly m/n and we have a vector x1,x2,xn and which is less than equal to b1,b2,bn so this is the constant and then we have to maximize or minimize some objective function C^T x so C1, C2, Cn so this is the general LPP problem, linear programming problem, programming problem.

So here we will take some particular case of linear programming problem here this matrix is having only two element +1 and -1, remaining are 0. So that means only two element +1 and one is -1 that means our this inequality is of the form x_i-$x_j \leq$ sum w_{ij}, so for example suppose we have three variable x_1- $x_2 \leq$ - 2, x_2-$x_3 \leq$ -3, x_1-$x_3 \leq$4 so this type of form. We can have, we have restricting our self by this, this is also called linear constant, so we will see how we can solve this linear constant with the help of Bellman Ford algorithm.

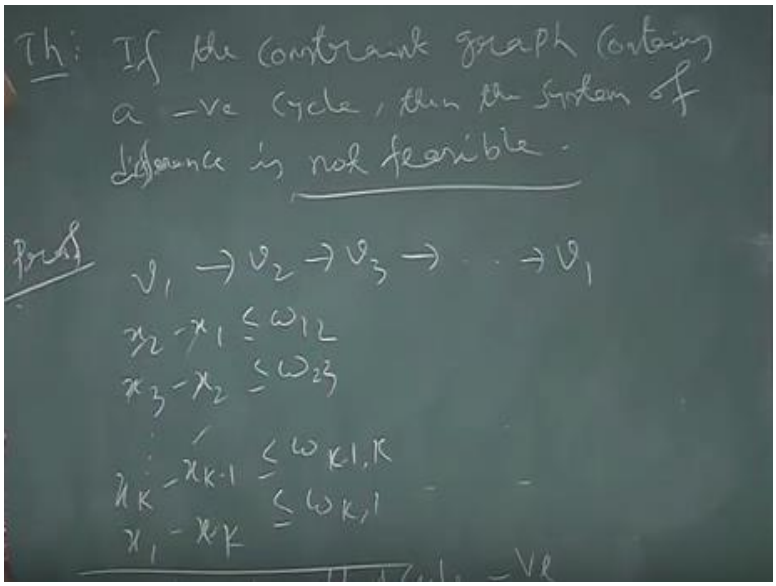
(Refer Slide Time: 21:06)



So this is also call difference constant, constants okay, so how we can suppose we have some inequalities like this, how we can view this as a graph so basically suppose we have this x_i-x_j is the w_{ij} so we will create two nodes like this here I and then we will have a matrix from this node to this node with w_{ij} is this, so for this graph so we have three vertex x1, x2, x3 now this will give us connection over here with 2 and 2 to 3 so this will give us the connection over here -3 and 1 to 3 so this is – 4.

So this is if we have a linear constant how we can drew the graph okay so now the question is suppose we convert so we have a linear system of constant equation we convert into a graph now the question is if this graph is having a negative cycle so that also can be detect by Bellman Ford, we can run the Bellman algorithm on this graph and we can if we it is detecting in negative cycle. Then there is no solution exist for this system of constant so how to prove that, so this theorem is telling.

(Refer Slide Time: 22:50)

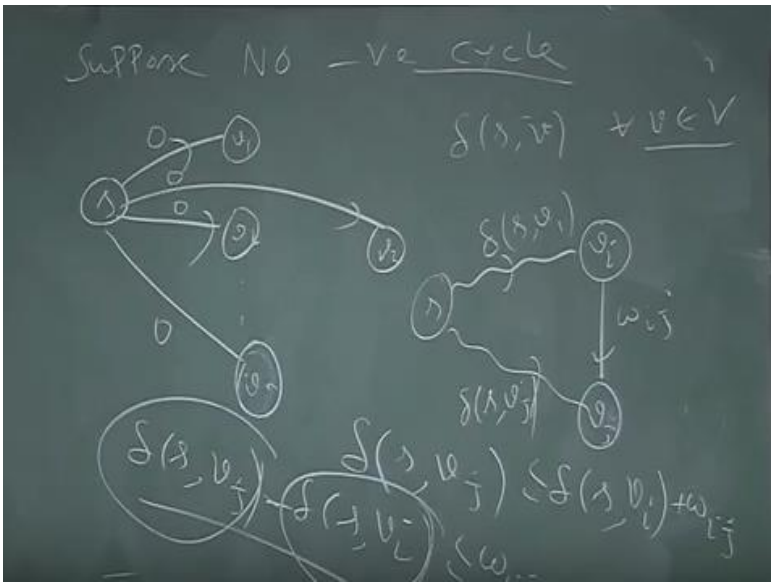


If the constant graph, if the constant graph contain a negative cycle then the system, system of difference then system of this linear is un satisfy, it is not feasible that means there is no solution for this, is unfeasible no there is no solution so how to prove that? Suppose there is a negative cycle in the corresponding graph so that means suppose this is a negative cycle V_1, V_2, $V_{3...}$ again it will come to V_1. So weight of this path is negative, so this is starting from V1 it is going to V_1 with the negative way.

So these will correspond suppose these are the weight so this will corresponding to the equation like $x_2 - x1$ is less than = w1 2 $x_3 - x_2$ is less than = w 2 3 dot $x_k - x_k - 1$ is less than = w(k – 1, k) then finally $x1 - x_k$ is less than = w a, 1 now if you add this, this term will be 1, all this term will be gone, here we have 0 and this is here we basically, basically weight of this path, weight of the cycle which is basically negative.

So that is the contradiction which is less than 0 so that means this system should not have a solution because it is contradicting this, so this is the proof, if a graph is having a negative cycle then it is not having a solution but the way round suppose we are having a, we are having no negative cycle then can you get the solution for this linear constant, yes so that will so.

(Refer Slide Time: 25:39)



So suppose there is no negative cycle in the corresponding graph, no negative cycle then we should get the solution. So how to get the solution, so let us just introduce a new vertex is for any other vertex V_1, V_2, V_3 like this so we just put the h with all other vertices from this vertex and the age we are going to put, 0. So we are introducing a new vertex, so we have a corresponding graph we are introducing a new vertex. Now this, if the original graph is having no negative cycle this graph is also having no negative cycle because we are only introduce a 0 weight.

So if it is no having no negative cycle so that means if run the Bellman it will give us a δ, so δs, v for all v belongs to this, okay. So now we take two vertices V_1 and V_2, so this is basically δ s, V_1 and this is basically suppose there is a direct age this age we are denoting by w_{ij} and this basically δ of s, v_j. Now by triangular inequality we can say like this δ of s,v_j sorry this is i and this is j, we check a just a inequality then we take the corresponding graph part, then δ base v_j is less than = δ of s, this path δ of s, vi.

Last w_{ij} okay, so this is basically giving us δ s, v_j – δ s, vi is less than equal to w_{ij} so this is the solution, so this δ's are basically xi, x_j so this is giving us the, this δ's basically giving us the

solution for the, this different constant. So if we can solve it, if we have a, if we run the Bellman Ford on this we got the δ's and that will give us the solution for the linear constant. Thank you.