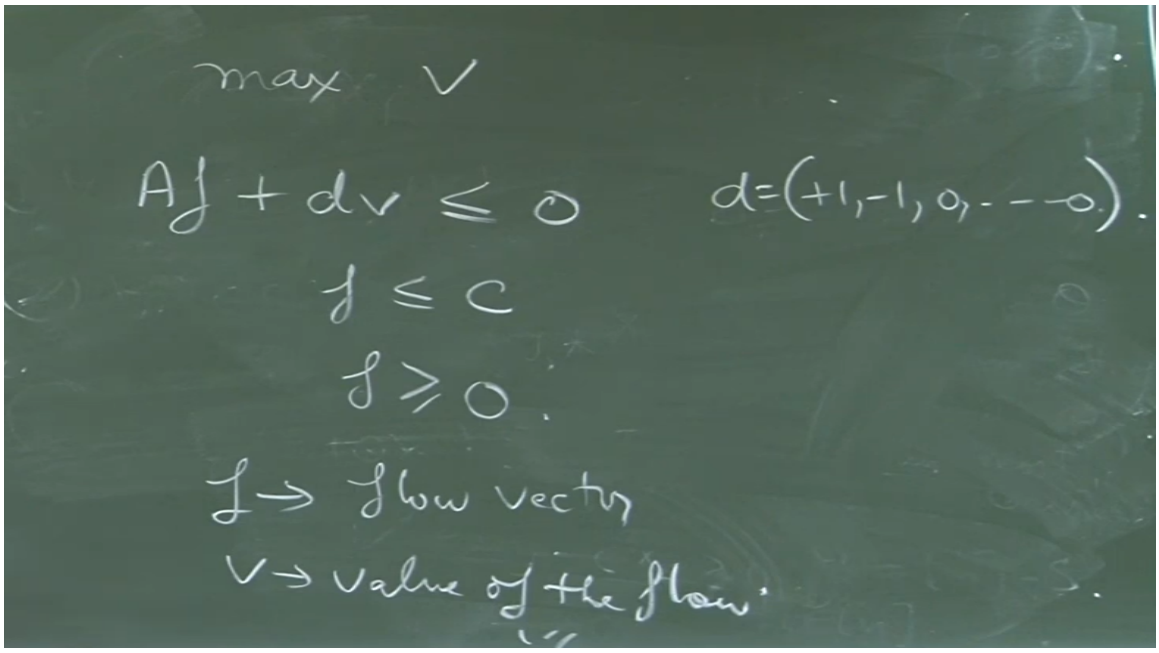
Linear Programming and its Applications to Computer Science
Prof. Rajat Mittal
Department of Computer Science and Engineering
Indian Institute Of Technology, Kanpur

Lecture – 43
Primal Dual for Max Flow

So, let me write the primal LP for this, in the way, that you do not identify. So, F is the solution of the flow, V is the value of the flow, right. So, F is the flow vector, this is the flow in every edge, V is actually the, sorry oh this will be your constraints right the connection the adjacency graph.

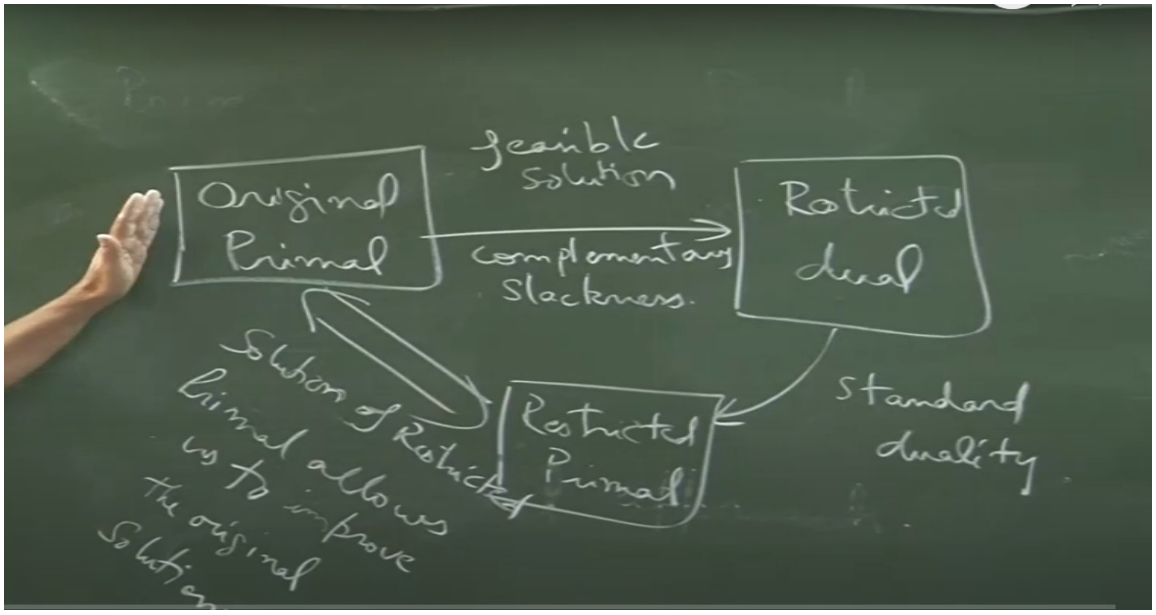


This will be some version of adjacency matrix which will give you right. So, the summation x from every direction all the edges it will I think be some small version of edge adjacency matrix. As an exercise you can figure out what A is like you write the four things you will figure out A right.

You can figure out which one is source which one is sink. If this still seems mysterious remember we had the flow conservation constraints instead of equality we had change it to less than equal it did not matter right. So, the previous constraints look like A F less than equal to 0 for all the vertices except S and T. Here we have included S and T that is all. Flow this is the capacity constraints flow has to be positive great right.

And now if you remember our approach we never have to worry about this right because this was only used to give you the idea of the algorithm. The algorithm runs just

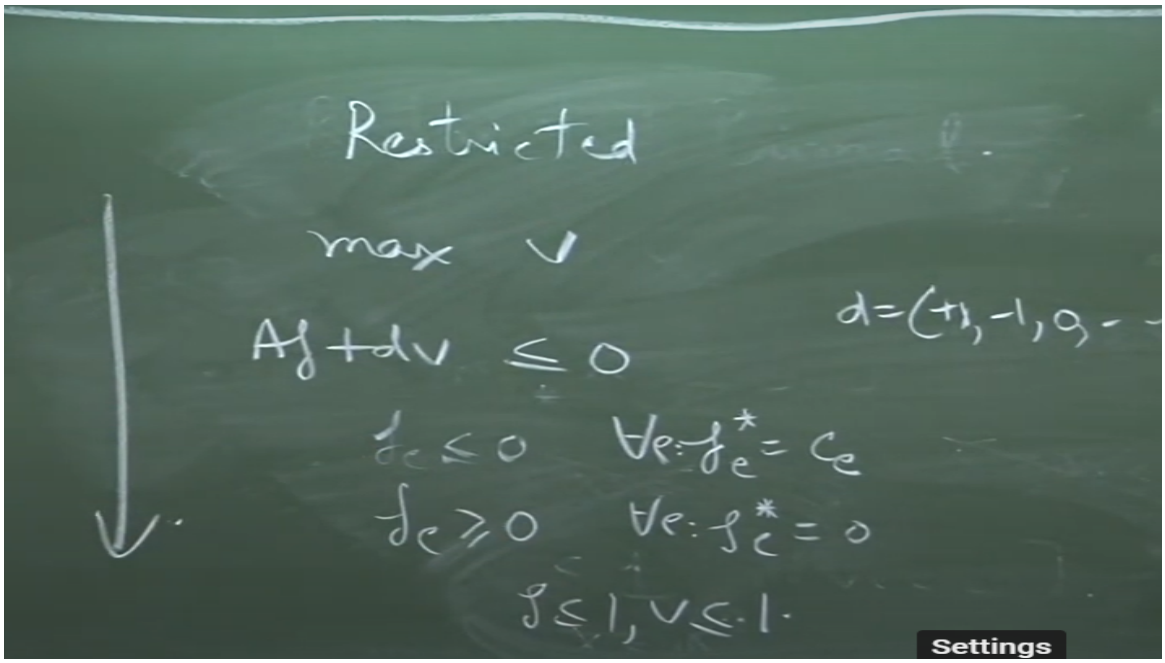
here find the feasible solution give me the restricted primal solve restricted primal it will allow me to improve my original solution. Again this will allow me to create the restricted primal again this will again lead me to improve. So, algorithm just goes this way it has nothing to do with restricted dual. So, why anyway you are very bad at taking dual.



So, why write you are taking the straight dual. This again the constraints remain the same in the restricted primal. The variables have constraints I just removed the thing, but so this is where your restriction will come. Sorry, you raise a good point if you remember in the restricted primal some of the constraints were removed, correct. Why have I not removed any constraint here? Sometimes you pose problems by asking the question, because I will ask the question to you again, right.

So, if you remember this is a constraint whenever there I am putting them up only when this was tight, because of that I have changed the constraint. And this was tight I have changed the constraint. I have never mentioned about this. If this was not tight I should have dropped those. Why have not I dropped those constraints here which are not tight? This has nothing to do with linear program this has to do with max flow.

Remember originally this was AF plus DV equal to 0. We said remember this is a flow conservation constraint, right. So, for a feasible flow this will always be equal to 0. It is a flow conservation constraint we said that we can relax it to be less than equal to 0, but will remain equal in the any solution any feasible flow solution this has to be 0 it cannot be anything else. So, nothing is going to be dropped from here.

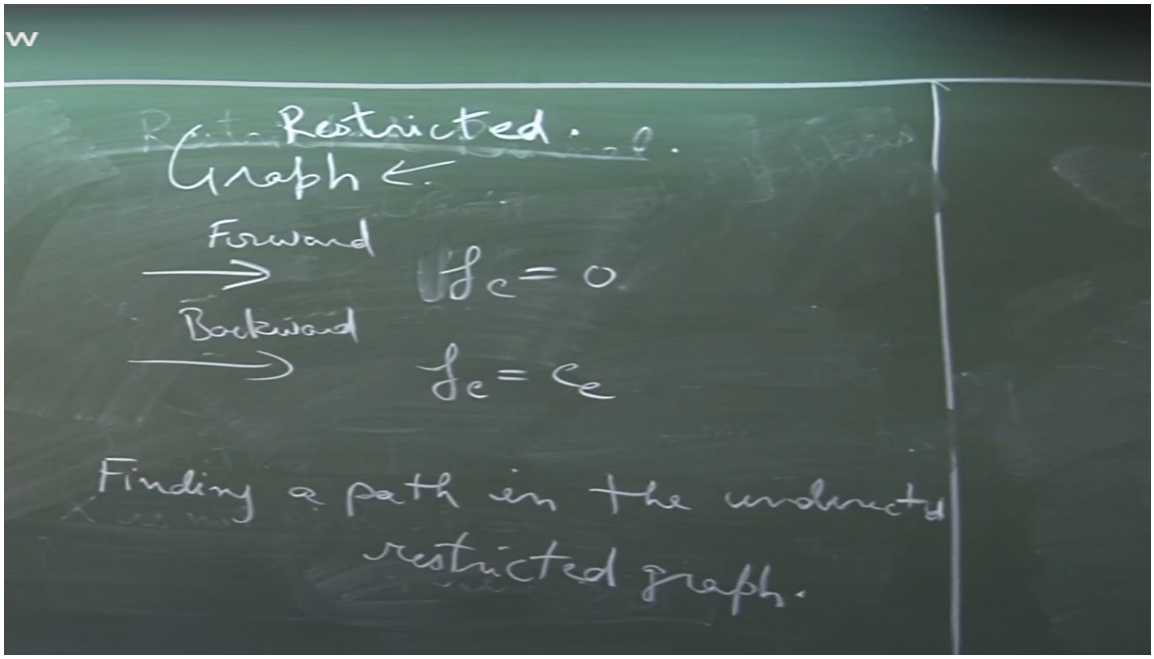


But, now one more question to you even though you did not make it happen to yourself I inflict the pain do you recognize this LP? This should remind you of something S very good you remember this we did in the when I we solved that dual of the I LP and LP were equal. We talked about residual graph which was we only keep the edges where there is capacity and where there is equal to 0. So, we are going to create a residual graph and then see if there is a connection that residual graph will allow us to move the flow. So, we look at the feasible solution wherever for an edge Fe star is equal to C we say that we can push a negative flow there. Wherever Fe star is equal to 0 we say that we can put a positive flow there and these are just artifact of that restricted primal F less than equal to 1 yes we had X less than equal to 1 remember in the restricted primal.

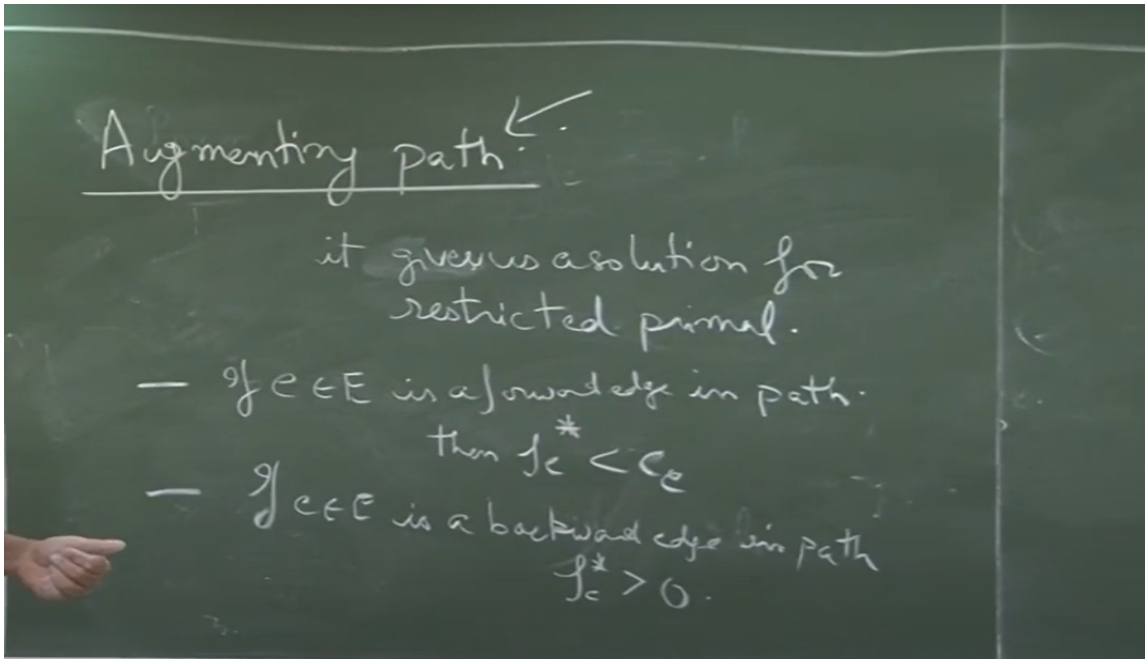
Now, we have two residuals F and V was all X were less than equal to 1. So, this and again most of the time if you are going to apply a an LP solver here this is mostly going to be a waste. Because, this is an LP this is an LP an LP solver cannot recognize that there is a difference between that most of the time this approach is good if we have a combinatorial algorithm if we have a nicer algorithm to solve this. But, that we saw last time that we will have a margin because they are not tight. So, now what we are doing is we take the original graph and in that graph we have a kind of a forward edge if the oh sorry what I have done wrong is I think that is why probably there was a confusion this was my original solution of this call it Fe.

And this is my restricted condition on the restricted primal for for a main restriction. So, now in my new graph I only I going to have edges within my feasible solution either Fe star was equal to C or Fe star was equal to 0. So, then this is opposite of like right. So, we are going to have a graph where I will have a forward edge if Fe was equal to 0. And I

am going to have a backward edge if Fe was equal to C correct.

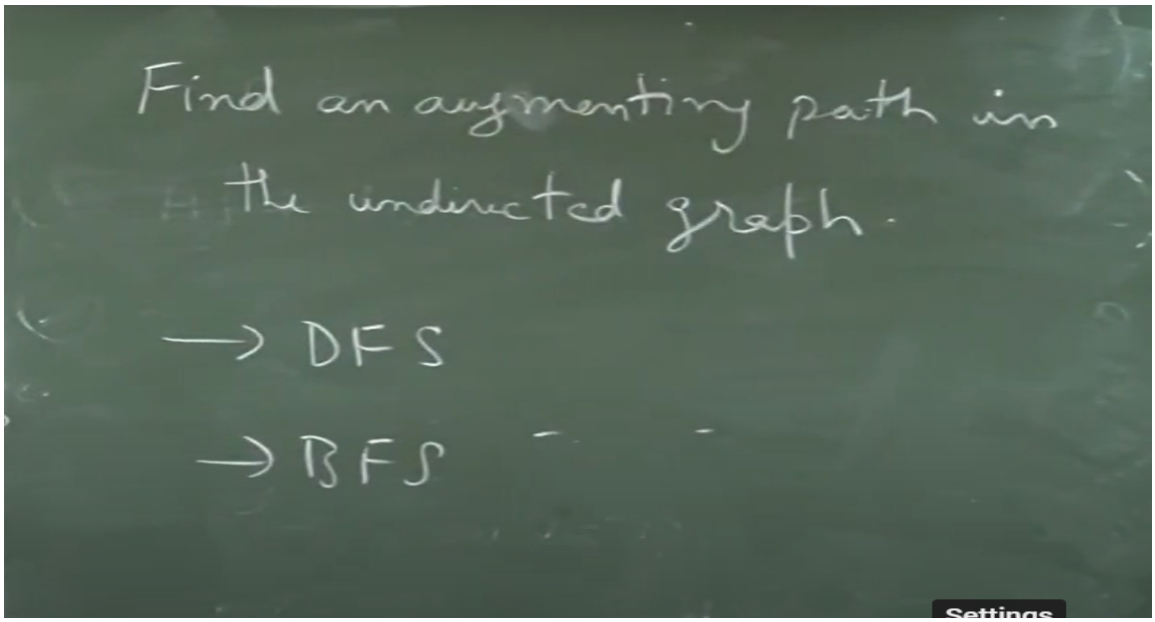


Now, this is the restricted graph and my question is what I am asking by this LP in the restricted graph commutability. Remember I want to show that this V is not equal to 0 if 0 then optimal if V is not equal to 0 then there is a flow in this graph right. That means there is a path between S and T. So solving the state primal finding a feasible solution to it is basically same as finding a path in the undirected restricted graph. I can just I can just find a path from S to T and if I find a path in S to T then I am done.



So, what you will show in the exercise is that an augmenting path and I will tell you what

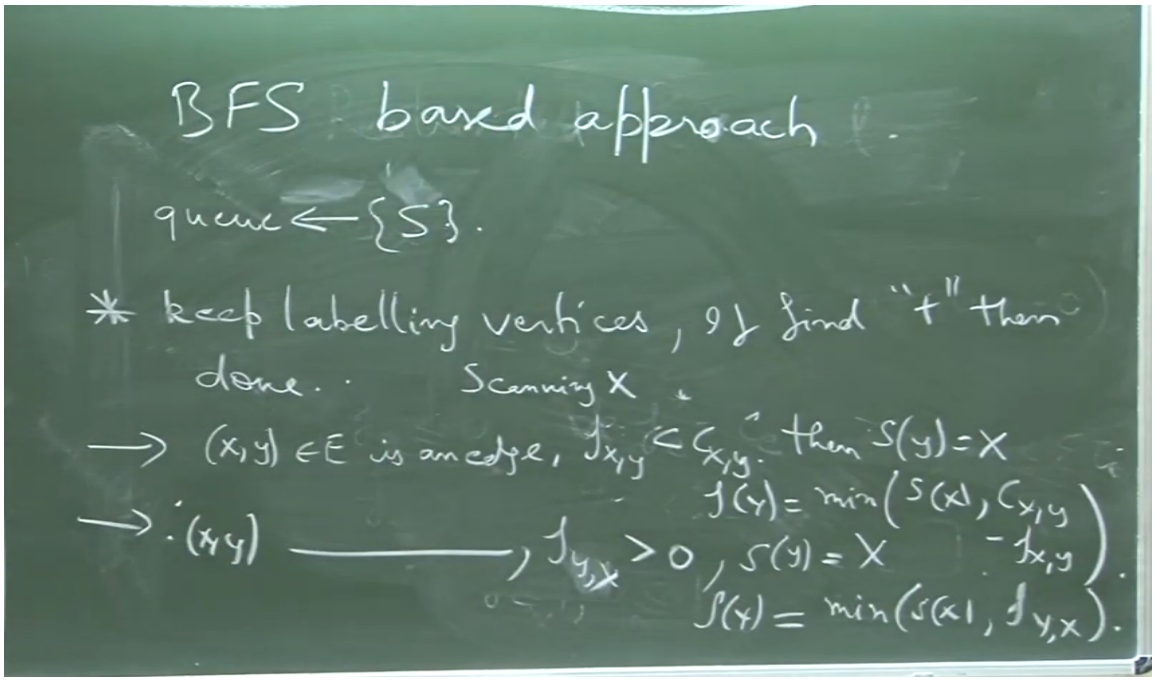
an augmenting path is this will again lead back to the solution the original definition of a residual graph. If you have an augmenting path it gives us a solution for restricted primal. So, what is an augmenting path? So, what is an augmenting path? So, this equivalence you can you can show and the point is that solving this restricted primal amounts to finding an augmenting path. So, then your problem reduces to the undirected graph and how do you find a path in the undirected graph? This you can solve by DFS, BFS whatever is your favorite method. The only thing is now you will have to modify the DFS or BFS slightly.



So, that you keep track of the path and how much flow you can push it in sounds good. So, once you modify this and depending upon what you choose sometimes your algorithm will be very efficient it can stop in required number of times sometimes it can just not converge. So, things are simpler when your capacities are integer ok. You start with a integer flow then you can assure that every time you will get a flow of at least one and then you can give better bounds here. If the values are irrational in some of these cases you might get into keep improving the result, but not in a good amount of time.

So, I will just outline the BSF based approach. You create a queue and you start with the source being in it, ok. So, this is your starting thing and then you start labeling vertices. If finally, t becomes labeled that means there are no labels, ok. This queue is just to keep track of which neighbors to scan next. This is not the list of labeled vertices that you can keep in some array or something or whatever.

This is just telling you which what to label. Now, I will put all the unlabeled vertices of s in the queue. I am going to label them. How am I going to label them? So, let us say I am scanning x. I am looking at neighbors of y and now this is an edge and this was in our graph because f x y was less than equal to c x y.

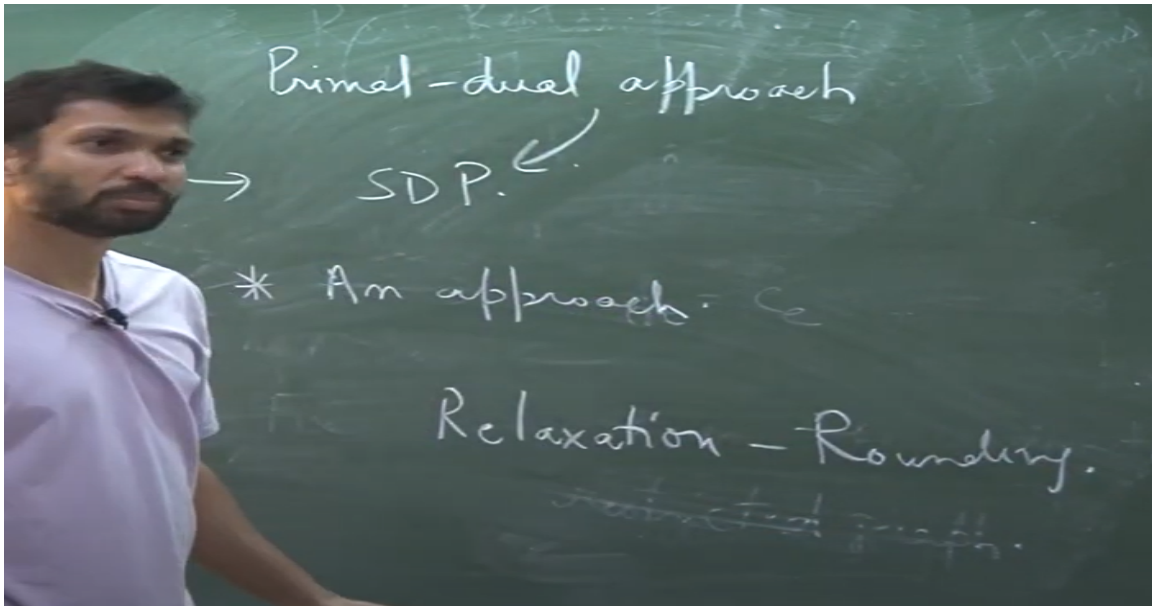


Then I say that I am going to reach y from x and the flow which can go in y is minimum over x comma f x y, ok. So, on this path the amount of flow I can put in is how much flow I can get up to x either that or the capacity of this edge the minimum of that I can make it flow. If x y is an edge because f x y was greater than equal to 0 sorry f y x was greater than equal to 0. Then s y is equal to x you will reach it from x this is an undirected graph finding a path we are saying will reach from x to y and f y is going to be. So, this is just a modified breadth first search which allows us to find an augmenting path.

Once t is found then I can stop then I have a feasible solution I have a feasible augmenting path then I have a feasible solution for the restricted primal. Now using this augmenting path I know this is very common it is very intuitive right on every path I have seen there is this much gap I can push and move things. So, the final f t here will tell me how much flow additionally I can put it. So, this will tell us even actually the improved primal. So, this primal dual approach works here there are primal dual approaches for even semi definite programs.

We are not going to cover this in the course, but even works for semi definite programs I do not know this paper by Satyen Kale which I had given as a project. So, you probably will talk about this, ok. So, how to apply this in the semi definite program there is also. So, that is why this primal approach is kind of nice. It is not a solution for everything again important point is it is an approach I would not call it an algorithm in some cases it is worthwhile to take this approach in some cases it does not help. Specifically in max flow when we take this approach the restricted primal turns out to be something which we know from before this idea of augmenting path.

This is exactly that is what I am saying this is Ford Fulkerson algorithm with the perspective of primal dual. It is not at all different from Ford Fulkerson what the things which I told you here this is literature on Ford Fulkerson. I am just giving you a twist on how to view Ford Fulkerson algorithm and this twist is important because this can be done in other linear programs semi definite programs and so forth. So the take away from today is the complementary slackness conditions sometimes can help us give a simpler LP which will let us improve the solution. This kind of approach is known as primal dual approach and there are many cases when this turns out to be beneficial for us.



This is it for this primal dual approach now I want to come up with cases when actually the integer linear program and linear program have a gap. Even if they have a gap can we do something about it there is a very interesting idea of relaxation and rounding which we start from tomorrow.