Linear Programming and its Applications to Computer Science

Prof. Rajat Mittal
Department of Computer Science and Engineering
Indian Institute Of Technology, Kanpur

Lecture – 39
LP for mass flow problem

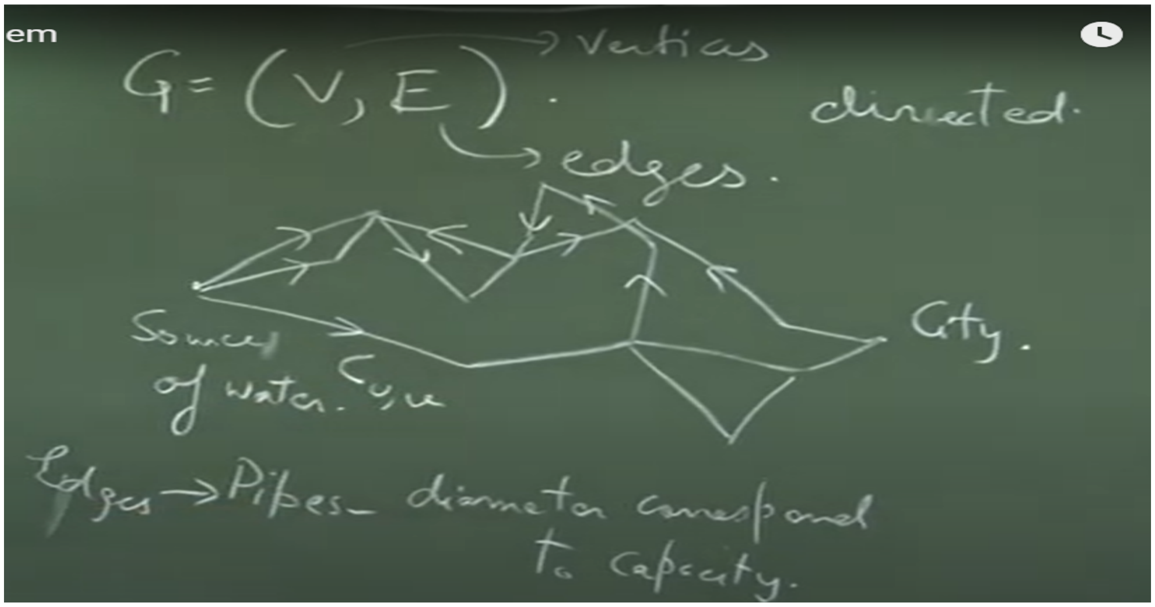
Welcome to another lecture on linear programming. We have looked at linear programming from multiple perspectives, we have looked at convexity theory, we have looked at duality theory and we saw some applications of duality theory.

Right From now on you will keep seeing more and more applications of linear programming in diverse areas. The first area we are going to cover is network flows and probably you might have seen such algorithms in the course. So, we already saw one example in the exam where we talked about matching and covering right. So, matching and covering correct me you might have heard of maximum flow problem is that correct.

These are examples of network flow problems, fallen network flow problems and were kind of standard problems for which people came up with nice algorithms. And Some of the things which you will do might be familiar to you, but we will look at them from the perspective of linear programming that is what we want to do. So, just by show of hands how many people know what Max flow problem is? So, good. So, it seems that most of you are familiar with what Max flow problem is that makes my problem easy.

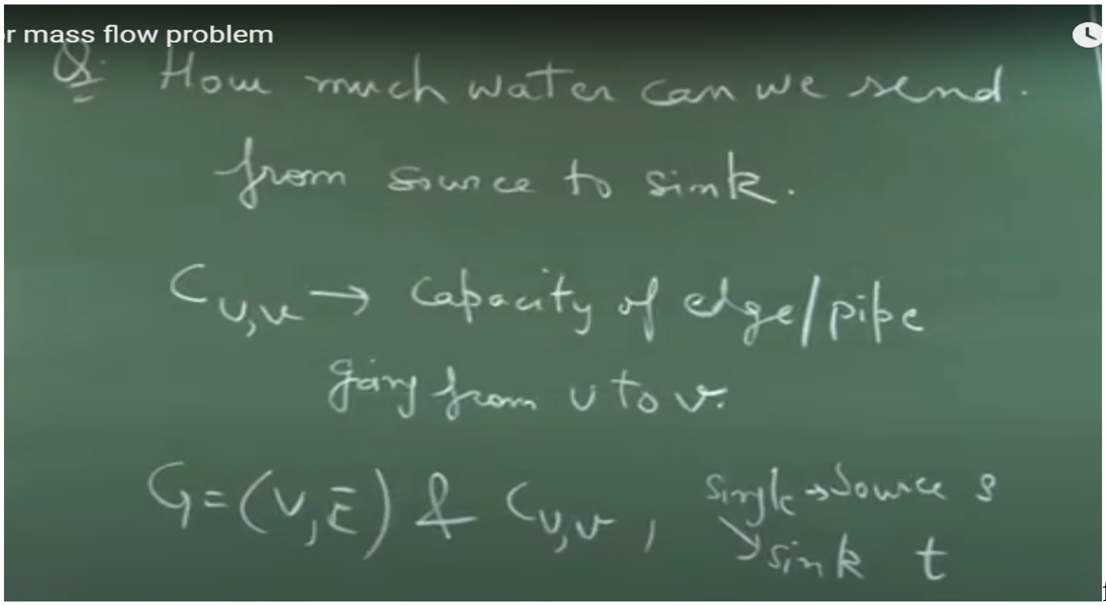
The idea is that you are given a graph. This is going to be a notation, vertices, e is edges and in this case the graph is director or undirected? Directed, right. In this case it is a directed graph. And probably before we start explaining this problem in Mathematical detail, let just say why it is a max flow problem. It has a very nice real life application to it which appears in many situations.

What is the situation? Let us say you have some source of water and there is some city where the water is needed. Right and then you might have different connections of pipes which might take you to these places right. Let us say this is where you know the water can be pumped in these directions or something. And obviously, each of these pipes have a diameter. And we can say that the diameter.



So, I should say that the edges correspond to pipes. And I can say that diameter correspond to some kind of capacity. So, we can send 3 liters of water per minute or whatever. So, we have these capacities and then the idea is that how much water you can send from this source to this city. So, what we remember is now if you want to abstract it out Mathematically, we are given a graph which has vertices and edges.

The edges are directed. We are in a directed graph domain and then each of the edge is given some capacity. So, if this is an edge between U and V then I am saying that it has capacity U V edge. This could be water, this could be oil, this could be blood, there could be you know from heart to some organ or something. We can ask in many situations, we can ask how much of the flow of some liquid, something, some material can go from the source to the sink.



r mass flow problem

Q: How much water can we send. from source to sink.

$C_{u,v} \rightarrow$ capacity of edge/pipe going from u to v.

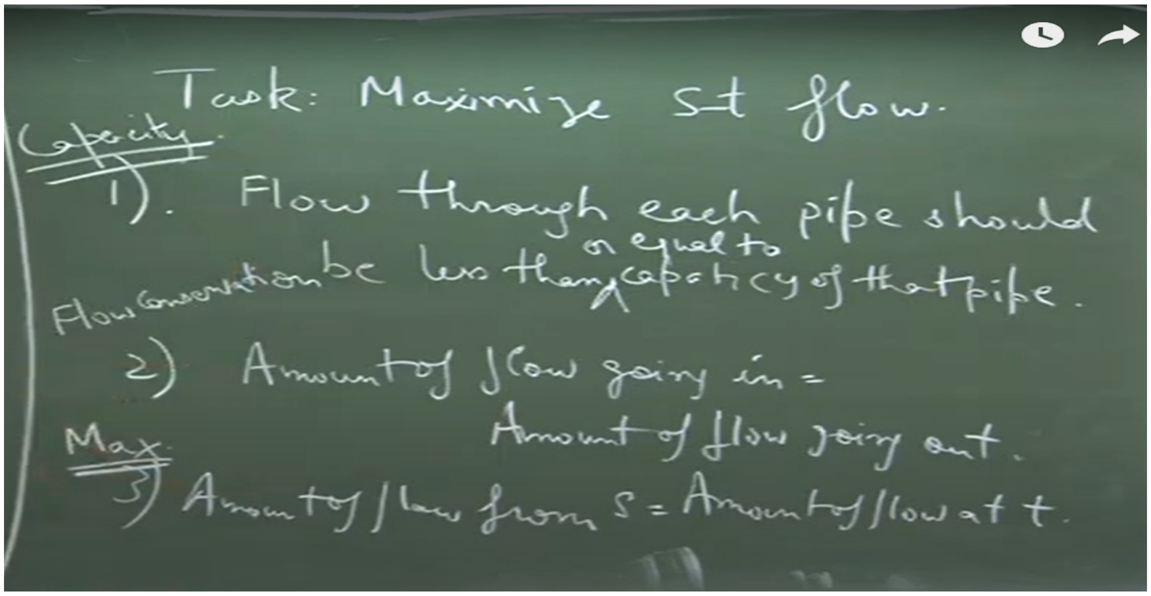
$G=(V,E)$ & $C_{u,v}$, single → source s ↘ sink t

f

This is the abstract problem which you want to solve. And as mentioned before, you have seen it before. We will see that linear programming gives us very nice interpretation of the results known here and gives us nice algorithms. So, coming back to the description, the problem will be specified by the graph that means vertices and edges capacities. And then for our case for simplicity, we will assume there is single source called S and single sink called T.

The task is to maximize ST flow. Make sense? This is what we want to do. What are the strains here in English? Sorry. So, I cannot put more flow than a capacity. So, what should be my variables? Or let us just write constraints first and then you can write mathematically these constraints.

So, first thing is flow through each pipe should be less than since I am sure one of you will point it out. So, less than or equal to capacity of that pipe. Correct? I think else I have to keep in mind.



Task: Maximize S-t flow.

Capacity.

1). Flow through each pipe should be less than or equal to capacity of that pipe.

Flow Conservation

2) Amount of flow going in = Amount of flow going out.

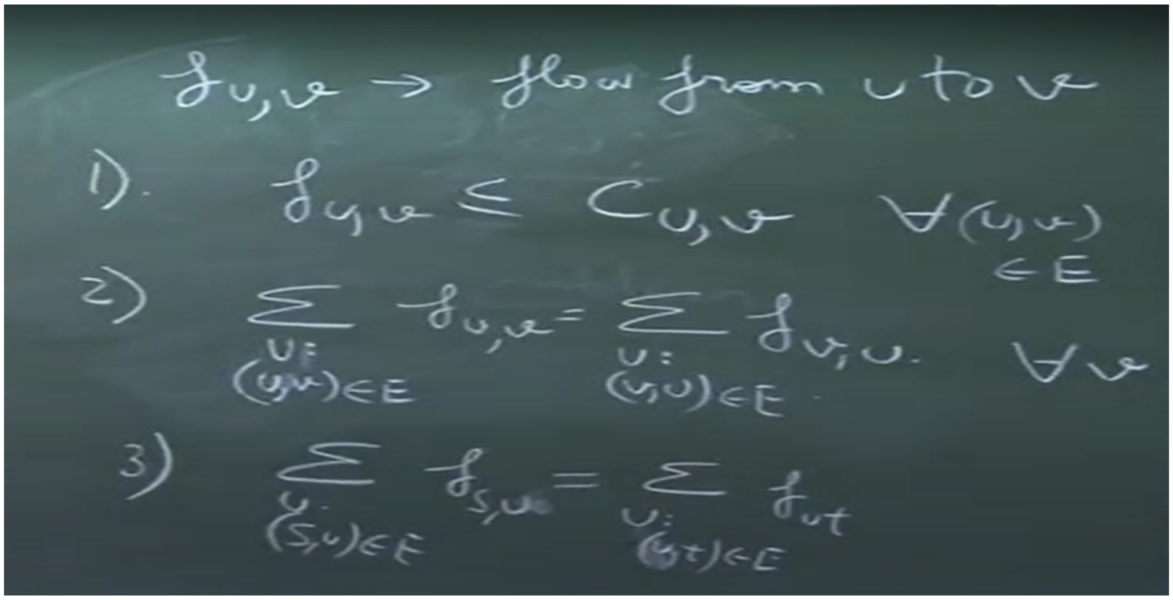
Max.

3) Amount of flow from S = Amount of flow at t.

Sorry. Right. So, I do not want water to be lost at any of these vertices. That means for each vertex, the amount of flow coming in should be equal to amount of flow going out. And then so this is the quantity which we want to maximize right. This is the flow conservation constraint. And this is capacity constraint.

Correct? Great. So, this is the English description of the problem and it is not hard to convert into the mathematical description. Right we just need the variables FU, fV. What does FUV denote? Flow from U to V. Then what is my first constraint saying? Correct? What is the second constraint saying? How can I write the second constraint? Why do not you write down pen and paper the constraint? So, amount of flow going into a vertex, let us say I pick a vertex V. So, amount of flow going in.

Correct? So, this is saying look at all the pipes which are going towards V. Let us say their starting point is U. So, look at all U's such that there is a pipe going from U to V. Look at the flows there. This is the amount of flow coming into V.



$f_{u,v} \rightarrow$ flow from u to v

1). $f_{u,v} \leq C_{u,v} \quad \forall (u,v) \in E$

2) $\sum_{\substack{u: \\ (u,v)\in E}} f_{u,v} = \sum_{\substack{u: \\ (v,u)\in E}} f_{v,u}. \quad \forall v$

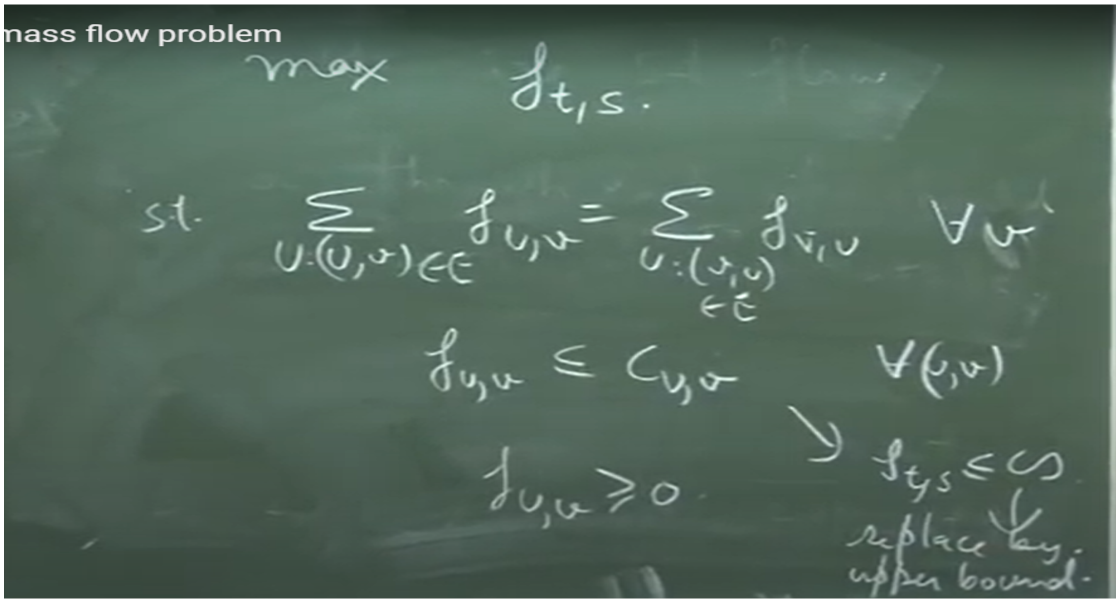
3) $\sum_{\substack{u: \\ (s,u)\in E}} f_{s,u} = \sum_{\substack{u: \\ (u,t)\in E}} f_{ut}$

So, if this was V, then how many edges are coming into it? S V 1 V 4. So, if I want to write down the equation for V, I am going to sum up the flows from S to V and V 1 to V. This should be equal to the amount of flows going out of V to V 3 and V to V 4 because no water can stay here. This is the equation. How do I write the third equation? Summation over U such that this is the amount of flow going out of S.

This is the amount of flow going into V. So, I will keep this picture because this is kind of instructive. So, the first thing to remember is that we really do not need the third constraint. So, Dev is nodding his head vigorously. Anyone else except Dev will tell me why? Since the flow is already conserved, then we know if at everywhere there is no flow leaving in or out, then if I am pushing 10 units of water, it cannot stay anywhere else.

So, it has to go to the sink. So, it is just part of the second constraint in some sense. So, I will not put it explicitly. So, then this problem if I want to write it, this is the amount of flow. This is what I want to maximize.

So, this is the second constraint. Since, this constraint it out, the problem straight forward become this. This is the quantity which you want to maximize such that it follows the flow conservation. It follows the capacity constraints and each flow obviously has to be positive. There is a direction associated with it.


nass flow problem

$$\max \; f_{t,s}.$$

$$\text{s.t.} \quad \sum_{u:(u,v)\in E} f_{u,v} = \sum_{u:(v,u)\in E} f_{v,u} \quad \forall v$$

$$f_{u,v} \le C_{u,v} \quad \forall (u,v)$$

$$f_{u,v} \ge 0. \qquad \searrow f_{t,s} \le \infty.$$

replace by upper bound.

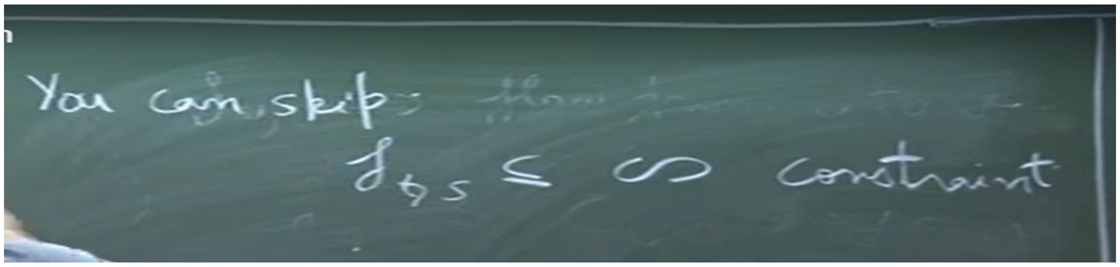
So, the flow should be positive here. Right and why are we happy? Exactly, we are very happy because this is the linear program. Correct Just once more, this is a linear program where the parameters or the input would be the graph and the capacities and the variable is U V. and actually, in the input you also have to specify which one is your source, which one is your sink, correct. So, once you are given all those descriptions, you can make this linear program.

Just to make sure that we are on the same page, how many constraints are there? How many variables are there? Number of edges, right. How many constraints are there? So, number of edges plus number of vertices minus 2. Correct Now, since it kind of seems weird that we do not have flow constraints for 2, it feels like a non uniform kind of a problem, right. So, there is a small standard trick to make this uniform. What is that trick? Exactly, we create an edge from T to S. And we can say that we want to maximize F T S.

So, whatever is the flow from S to T, we are saying it goes back. Then, we have flow conservation everywhere. What capacity should I give it? At this point, I do not know it. So, let us say infinity.

Let us start with infinity. Notice that now, I have an extra edge. This contains T, S also. Sounds good? Do you agree that these are equivalent programs? I see that there is some slight unease, because yes they are equivalent programs, but then you are going to have a constraint here, which is this allowed in linear programming? Is this not allowed in linear programming? Yes, no. Are you happy with it? Can I leave it? Can I put it in linear programming solvers? Why cannot I put it in the linear programming solver? Infinity is not a number, Right. So, what is the small fix? Exactly, very nice.

I can replace infinity by a very big number, right. For example, just sum up all the capacities. Definitely, that is an upper bound on my flow. Replace by a suitable upper bound. Then, you are not worried about actually putting this thing as a linear program.



Then, this makes sense. Why are we putting this constraint? That is also fine. So, if you are not happy with an upper bound, if you want, if you just think deeper, then you realize that you can skip less than infinity constraint. But then, someone says that it does not uniform. You know there is one edge.

Sorry C t s, I want to keep it as infinity. The constraint is this. The F t s less than C t s is the constraint. I wanted to put infinity for C t s. So, this constraint I will avoid. So, this was the maximum flow problem and there was an LP for it.