Approximation Algorithm

Prof. Palash Dey

Department of Computer Science and Engineering

Indian Institute of Technology, Kharagpur

Week – 09

Lecture 43

Lecture 43 : Integer Multicommodity Flow

Welcome. So, in the last class we have started seeing the integer multi commodity flow problem. It is an NP complete problem and we have first written the integer linear programming formulation and then we relaxed it to linear programming formulation. In this class we will see how we can use that LP formulation, LP relaxation for that problem to get an approximation algorithm for integer multi commodity flow. integer multi commodity flow. Let us recall the LP relaxation we had a variable c which denotes the congestion.

So, which we want to minimize subject to we had flow variables. So, for every edge e and commodity i we had a variable $f_{e,i}$ which is 1 if that edge is carrying that commodity otherwise it is 0. So, for every $i \in [k]$ and for every vertex $v \in V \setminus \{s_i, t_i\}$. So, because edges do not have any capacity only flow conservation needs to be ensured that means, at every vertex v total flow in for i-th commodity should be same as total flow out at the for the i-th commodity.

So, this is $\sum_{\{u,v\} \in E} f_{\{u,v\},i}$ this should be same as this is total flow in for i-th commodity same as $\{v, w\} \in E$. So, this is the flow conservation property and at s_i and t_i at s_i total flow out should be 1 and at t_i total flow in should be 1. So, for all $i \in [k]$ $\sum_{\{s_i,v\} \in E} f_{\{s_i,v\},i} = 1$ which should be same as total flow in $\sum_{\{v,t_i\} \in E} f_{\{v,t_i\},i} = 1$. So, these are flows and we need some other variables for every edge x_e it should denote the it should denote its load and the c should be maximum load.

So, these inequalities we missed in last classes ILP formulation. So, for every variable for every edge e we have a variable x_e which simply denotes the load of that edge recall load of an edge is the is the number of commodities that are passing through that edge. So, x_e is equal to $\sum_{i=1}^{k} f_{e,i}$ is 1 if edge e carries the ith commodity. So, x_e denotes the load of the edge e and the congestion is the maximum load. Now, because I am

minimizing c.

So, if I put a constraint that congestion should be as high as the load of every edge then the then I need x_e is less than equal to c. So, if I want to minimize c then c must be equal to minima or maximum of all the loads of the edges. So, these are the constraints and of course, we have the usual constraints that is that that this variables should take $\{0,1\}$ value for ILP formulation and should take only positive values greater than equal to 0 values for. So, for all edge e and $i \in [k]$ $f_{e,k} \geq 0$, $x_e \geq 0$. So, this is the LP relaxation for integer multi commodity flow.

Now, as usual so, we will design a randomized rounding based algorithm. So, we solve the be an optimal solution. Now, this f^* gives gives you a fractional solution each commodity did not be sent along one each along one path from s_i to t_i because this need not be need not take $\{0,1\}$ values $f_{e,i}$. So, but because summation f is 1 roughly along all paths, then we use this f values as probabilities.

So, if some edge has higher f value, then the linear programming solution LP optimal solution is indicating to use that edge more often or more likely than other edges. So, what we do here? It is not completely clear how to how to use this (f^*, x^*) to select one path from s_i to t_i for every i. So, we need to choose one path P_i from s_i to t_i for all $i \in [k]$ ok. Somehow using f^* and x^* as our guide. So, for that we it would have been natural if instead of f^* we have distribution over all the paths from s_i to t_i.

So, if I look at s_i-t_i there could be multiple paths they can share some part or they could be disjoint and so on and the number of paths could be exponentially many. So, but we do not we are not given paths we are given flows on every edge. So, here the useful thing is the flow path decomposition. So, what is flow path decomposition? Lemma flow path decomposition. So, what is it? It says that if I am given a flow, flow means every edge carrying certain amount of flow and some values on every edge which satisfies flow conservation property and capacity constraint, but here edges do not have any capacity they have infinite capacity.

So, only we need flow conservation property. So, let if it is a function from edge set to non-negative real numbers R s to t flow. That means, at every vertex except s and t flow conservation property is satisfied flow in equal to flow out at s total flow in total flow out is f of summation of the flow values of the edges and at t total flow in should be same as total flow out at s. So, this is the definition of flow. there exists s to t paths because we have we will use P_i in our algorithm let us denote the paths as Q_1 ,…, Q_k or $Q_{l,k}$ is there in our problem definition of multi commodity flow.

Then there exists s-t paths Q_1 to Q_l and values and non-negative values f_1 to f_l such that for every edge if I look at the flow value defined by f or if I look at the flow value as if along path Q_1 I am sending f_1 unit of flow from s to t along path Q_2 I am sending f_2 units of flow and so on and so forth. Then the flow values in every edge matches they such that for every edge $e \in E$. $f(e)$ this is the flow value as per this function is same as all the i's in $[l]$ this edge e belongs to this path Q_i. So, think of here is s here is t here is an edge e and if the flow value passes through this edge suppose this Q_1 passes suppose Q_{10} passes. So, total flow value of edge e is the sum of the flow values of all these Q_i's that uses e this is f_i.

So, if this holds then we say then that this is $Q_1, \ldots, Q_l$ are alternative representation is a flow path decomposition of this flow this $Q_1, \ldots, Q_l$ these are called flow paths. Not only that moreover l is at most the number of edges. in the graph that is 1 and $Q_1, \ldots, Q_l$, $f_1, \ldots, f_l$ can be computed in time poly of number of vertices in number of edges and the value of $\log \sum |f_e|$ basically in polynomial time of the input this flow paths can be computed. So, the proof is quite standard in any standard book on algorithm which discusses maximum flow problem discusses this theorem along with the proof. So, proof I am leaving it as a homework.

So, now, what I do for every commodity i I have a flow and I get the flow path decomposition. So, let for every $i \in [k]$ that means, for every commodity let $P_{i,1}, \ldots, P_{i,l}$ and $f_{i,1}, \ldots, f_{i,l}$ be a flow path decomposition of f_i the flow for i-th commodity. So, because we are sending to sending total 1 unit of flow for every commodity we have $f_{i,1}, \ldots, f_{i,l}$ equal to 1 and all this $f_{i,1}, \ldots, f_{i,l}$ these are greater than equal to 0. So, it is very natural to treat them this f_i s values as probabilities and pick one of this paths one of this l one among this l paths as per this probability distribution and send one unit of i-th commodity along that path.

So, pick one of $P_{i,1}, \ldots, P_{i,l}$ with probabilities $f_{i,1}, \ldots, f_{i,l}$ and send one unit of commodity i along that path ok. So, is a very natural algorithm. So, let us analyze the algorithm. So, let c^* be the optimal congestion. So, what we will show is that the load of every edge e is not too far from c^* with high probability.

So, you see x equal to define $x_e = \sum_{i=1}^{k} x_{e,i}$. So, e is an edge. So, where x_e is the load of edge e and $x_{e,i}$ is the indicator random variable whether edge e is used for sending ith commodity. So, indicator random variable for the event that edge e is used. to send commodity i from s_i to t_i.

So, congestion is max of x_e. which is ALG. So, what we will show what is expectation of X_e? Expectation of X_e by linearity of expectation $i=1,\ldots,k$ expectation of $X_{e,i}$. Now again because of flow path decomposition this is the expectation of indicator random variable is the probability that the event happens.

So, the probability that edge e is used to send commodity i from s_i to t_i is exactly $f_{e,i}$ this is because of the flow path decomposition and so, this is $f_{e,i}$ ok. So, you see expectation of or so, this is less than equal to congestion. You see for every edge we had a variable X_e which was denoting the congestion. So, this is small x_e here we have capital X_e. So, this is equal to x denotes the congestion of edge e in the linear programming formulation and capital X is the congestion which is an indicator random variable of edge e in our routing decided by the algorithm.

So, $x_e \leq c_e^*$. What we will show is that the probability that the load of edge e is far away from expectation of X_e which is at most c^* is very small. And here we use Chernoff bound because you see X_e is sum of random variables and these are indicator random variables that is why they are $\{0,1\}$ random variables and they are independent also. across for every $i \in [k]$ they are independent. So, probability that X_e is greater than equal to $(1+\delta)c^*$, c^* is an upper bound on expectation.

So, this is less than equal to $\left(\frac{e^{\delta}}{(1+\delta)^{1+\delta}}\right)^{c^*}$ ok. So, this is for a 1 edge now. So, this is from Chernov bound. Now, we apply union bound to get the probability that there exist at least 1 edge whose load is high in the sense more than greater than equal to $(1+\delta)c^*$. So, by union bound probability that the probability that there exist an edge e such that X_e is greater than equal to $(1+\delta)c^*$ is less than equal to ah suppose there are $m\left(\frac{e^{\delta}}{(1+\delta)^{1+\delta}}\right)^{c^*}$. So, probability that for every edge the load is less is 1 minus this probability that ALG is congestion of the network ALG is less than $(1+\delta)c^*$ which is exactly probability that for all edge $e \in E$ X_e is less than $(1+\delta)c^*$ which is 1 minus probability that there exists an edge $e \in E$ X_e greater than equal to $(1+\delta)c^*$, but this is greater than equal to $1-m\left(\frac{e^{\delta}}{(1+\delta)^{1+\delta}}\right)^{c^*}$.

Now, we just need to put some appropriate value of δ. So, for $\delta = 2\left(1+\frac{\ln m}{\ln \ln m}\right)$ you see

the value of δ we are putting is greater than 1. So, we cannot use the simplified version of the Charnoff bound that is why we have used the general version of the Charnoff bound which works for every δ greater than 0. This works for all δ greater than 0 this is very important. So, for this what we have is probability that ALG is less than 1 plus delta times c^* this is less than equal to or this is greater than equal to $1-\frac{1}{m^{c^*}}$.

So, this means that with high probability because sister is at least 1 that means, with high probability this ALG is less than equal to $(1+\delta)c^*$. So, with high probability in algorithmic or in randomized algorithms we say some event to happen with high probability if the probability is greater than equal to 1 minus 1 by poly of input parameters here the number of edges is an input parameter. So, with high probability the probability congestion of the routing output by the algorithm is if you put $\delta = 2\left(1+\frac{\ln m}{\ln \ln m}\right)$ factor approximate solution. So, let us stop here. Thank you.