Approximation Algorithm

Prof. Palash Dey

Department of Computer Science and Engineering

Indian Institute of Technology, Kharagpur

Week – 12

Lecture 60

Lecture 60 : Inapproximability of Edge Disjoint Path

So, in the last class we have proved that there is no polynomial time alpha factor approximation algorithm for the for scheduling jobs multiple jobs on parallel, but non identical machines. to minimize the makespan of the schedule for any alpha less than 4 by 3. And then we claim that this proof can be slightly modified to prove a stronger guarantee of inapproximability for every alpha less than $\frac{3}{2}$. And towards that we modified that construction and we removed the jobs corresponding to X and of one type of dummy job we introduced one type of dummy job for every x and then we define the processing times cleverly and we claim at the end that we can prove using this reduction that there is no alpha factor polynomial time approximation algorithm for scheduling task for any alpha less than $\frac{3}{2}$. So, let us finish that proof. scheduling jobs on multiple non identical machines.

So, here is the theorem that we that if the 3 dimensional matching instance is a yes instance, then there is a schedule with makespan 2. Otherwise the makespan of every schedule is at least 3 proof. So, in one direction suppose the 3 dimensional matching instance is a yes instance. suppose the 3 dimensional matching instance is a yes instance, then there exists a subset S' of S with cardinality n, where every element of X, Y and Z appears exactly once ok.

Now, in this case we want to show that there is a schedule of makespan exactly 2. So, for all tuple $(x, y, z) \in S'$ schedule the jobs corresponding to y and z to the machine corresponding to X. So, this way exactly one machine of the $n|x|$ machines got scheduled and their load is 2. then for all $x \in X$ schedule the $n|x| - 1$ dummy jobs of type X on the machine corresponding to X. The makespan of this schedule is 2. You see all the machines corresponding to S' their load is 2 and all the machines which are scheduling those damage of their load is also 2. So, the makespan is 2. So, this proves the one direction the first part if part now the otherwise part.

So, we want to show that the makespan of if the 3 dimensional matching instance is a no instance. then the makespan of every schedule is at least 3. What we will show is contrapositive of this statement that is if there is a schedule of makespan 2, then the 3 dimensional matching instance is a yes instance. So, for the for otherwise we will show that if there is a schedule of makespan 2, then the 3 dimensional matching instance must be a yes instance. So, suppose there is a schedule which makes span 2.

Now, because each dummy job needs at least 2 units of time. So, let us name this schedule, schedule say A. So, in A, the dummy job corresponding to each $x \in X$ must be scheduled the $n|x|-1$ dummy jobs must be scheduled to the $n|x|-1$ machines. among n x machines of type x. So, this leaves exactly one machine of type x free and only in total only n machines to schedule the 2 n jobs.

this leaves only n machines free to schedule the jobs corresponding to Y and Z ok. So, it is easy to show it is easy to see or take it as a homework to prove that homework prove it that the n tuples corresponding to the n machines which execute the jobs corresponding to Y and Z forms a three dimensional matching. So, this finishes the proof of the theorem and the immediate corollary what we have there is no polynomial time. alpha factor approximation algorithm for scheduling jobs on multiple parallel non-identical machines to minimize makespan. for any alpha less than $\frac{3}{2}$ because any such algorithm will find out is forced to output a schedule with makespan 2 whenever it exists and that is enough to solve the 3 dimensional matching problem.

Our next problem for which we will show in is edge disjoint path. So, because we will prove inapproximability let us write down the optimization version. an undirected graph $G=(V,E)$ and key pairs of vertices $\{s_1,t_1\},\{s_2,t_2\},\ldots,\{s_k,t_k\}$ ok. The goal is to so, find or $S \subseteq [k]$ such that cardinality S is maximized and there are or there is a path P_i from s_i to t_i for every $i \in S$ and this paths are edge disjoint ok. So, these paths are edge disjoint.

So, among this k pairs of vertices find as many edge disjoint paths as possible. So, for this problem we will show a strong inapproximability bound. So, there is a simple greedy algorithm which gives $O(\sqrt{m})$ approximation algorithm gives $O(\sqrt{m})$ approximate solution what we claim is that that is pretty much the best approximation guarantee we can hope for in polynomial time. So, here is a theorem for any or for every alpha greater than there is no $\Omega(m^{\frac{-1}{2}+\epsilon})$. So, this is $\frac{1}{\sqrt{m}}$ not $\sqrt{m}$ this is $m^{\frac{-1}{2}}$.

So, there is no $\Omega(m^{\frac{-1}{2}+\epsilon})$ approximation algorithm for edge disjoint path unless $P=NP$, this condition also should be there in our previous proof this is unless $P=NP$ ok. So, for that what we will show we will give a reduction the from edge disjoint path to itself. This edge disjoint path problem is NP complete even for $k=2$. So, edge disjoint path paths problem is NP complete even for $k=2$.

That means, given 2 pair of vertices $\{s_1,t_1\},\{s_2,t_2\}$ it is NP complete to decide whether there exist 2 edge disjoint paths. one from s_1 to t_1 another from s_2 to t_2. What we will show is that if there is a $\Omega(m^{\frac{-1}{2}+\epsilon})$ factor approximation algorithm for edge disjoint path, then we will be able to have a polynomial time algorithm for 2 disjoint path ok. So, let $G=(V,A)$ and $\{s_1,t_1\},\{s_2,t_2\}$ be an instance of two disjoint paths. from this we will construct an instance of k disjoint path.

So, for that we set k to be equal to cardinality a and ceiling of $\frac{1}{\epsilon}$. So, I am given an epsilon greater than 0 arbitrary. So, we take k to be this and we will construct an instance of k disjoint path problem. from this. What we do is this to distinguish from this 2 disjoint path we will name the pairs of k disjoint path as a 1 b 1 a 2 b 2 and so on.

So, we will have this construction So, suppose this 2 disjoint path we denote it by this notation. So, here is s_1, here is t_1 and here is s_2, here is t_2. What we do is we have a vertex a 1 connected with a copy of G we will have multiple copies of G and connect it to s_2 here ok. So, here not undirected graph we will have directed graph let us have directed graph because that is important ok. So, this is a_1 and then a_2 is connected to this.

a_3 is connected here and a_k is connected here, here means that means, a_i is connected a_2 is connected to s_1, a_3 is connected to s_1 of the below graph and so on. And from the top graph from the t_2 I connect I put an edge from t_2 to s_2 and so on this And here also we have another copy, we have an edge from here t_1 to s_1, we have edge from t_1 to s_2 and so on. Let us draw the a_4 also then the pattern will be very much visible. we have another copy and here is b_1, here is b_2, here is b_{k-1} here is b_k. Now, if there is there are two edge disjoint paths one is from s_1 to t_1 another is from s_2 to t_2 then you see we have an edge disjoint path from every a_i to b_i.

So, for a_i you can from a_1 you can take this edge, then this edge disjoint path, this edge, this edge disjoint path, this edge, this edge disjoint path and so on till b_1. So, you get a path from a_1 to b_1. Similarly, you can get a path from a_2 to b_2 like this a_2 to this edge, then take this edge disjoint path, take this edge, this edge disjoint path, and so on. So, what we have seen is that we have basically proved this lemma that if there is there are 2 edge disjoint paths. in G then there are k edge disjoint paths in $G^{'}$.

So, let us call this graph $G^{'}$. On the other hand, if you see if you see that there are no two edge disjoint path, then in $G^{'}$ there can be at most one edge disjoint path. So, whenever you can have a path from a_i to b_i that is it because that will not that blocks all the paths. Otherwise that means, if there are no two edge disjoint paths in G. otherwise there is only one disjoint path. So, again here you see that if the number of edge region paths is k which is high which is some you see k is some high number.

And this that is set in such a way that if the number of edge region paths is k then an polynomial team algorithm even with this approximation guarantee this weak approximation guarantee needs to output at least 2 edge disjoint paths. So, I will fit this $G^{'}$ as an input and ask that approximation algorithm to out give me edge disjoint paths. Now, if there are k edge disjoint paths the approximation ratio is set in such a way then that the algorithm will is forced to output more than one edge disjoint path and then from that information I know that that the two edge disjoint path problem is a yes instance. Otherwise if G has exactly one edge disjoint path then $G^{'}$ will also have one edge disjoint path and in that case the algorithm will output one edge disjoint path. And hence existence of such an algorithm gives me a polynomial time algorithm for the NP complete problem to edge join path thereby proving $P=NP$ ok.

So, you finish this proof formally as a homework ok. So, let us stop here. Thank you.