Approximation Algorithm

Prof. Palash Dey

Department of Computer Science and Engineering

Indian Institute of Technology, Kharagpur

Week – 01

Lecture 05

Lecture 05 : Primal dual method for Weighted Set Cover

Welcome to this lecture. So, we have been seeing an approximation algorithm design and till now we have seen 2 methods one is linear programming rounding the iterative the deterministic rounding to get an f factor approximation algorithm for set cover. And then in the last class we have seen dual rounding again to get an f factor approximation of for the set cover problem. So, we will continue our journey and see a brief overview of various algorithm design techniques. through set cover and then we will delve into each of these techniques into more detail. So, today we will see another technique which is called primal dual method.

So, today's topic is primal dual method for designing approximation algorithms. So, this is a combinatorial algorithm in the sense that we do not need to solve the linear programming relaxation unlike the earlier methods of linear programming rounding or dual rounding. This is method to design combinatorial algorithms this term is loosely used to denote the fact that we do not need to solve a linear program and get an optimal solution for a problem. programming relaxation and dual of it only used.

for analyzing the algorithm. These type of algorithms often much faster often runs much faster than algorithms which needs to solve linear programs and get an optimal solution. So, again let us see this technique through the example of weighted set cover. So, let us recall what was the LP relaxation of weighted set cover. for each set we have a variable and that I want to minimize the set of the weight sum of the weights of the sets that I pick.

So, $\sum_{j=1}^{m} w_j x_j$ such that the idea is x_j will be 1 if the j th set is picked otherwise it is 0 such that for every element $i \in [n]$ $\sum_{j \in [m]: e_i \in S_j} x_j \geq 1$. And each $j \in [m], 0 \leq x_j \leq 1$ ok. So, this was the linear program for weighted set cover and its dual LP for every element we have a we have a variable which is its price which what we

charge $\sum_{i=1}^{n} y_i$ and that we want to maximize max subject to for every set no for every set the sum of its elements, sum of the charge charges of its elements should not exceed the weight of that set. So, we have a constraint for every set for every $j \in [m]$, $\sum_{i \in [n]: e_i \in S_j} y_i w_j$ and for all $i \in [n], 0 \leq y_i \leq 1$. Again let me remind you there is a purely mechanical way to derive the dual of a linear program and if you just follow this that mechanical method and try to obtain the dual of the linear programming relaxation of weighted set cover you will get this linear program. and once we have the dual you can interpret the variables give suitable names. So, that it fits the human interpretation, but mathematically this can be done purely mechanically. Now, in the primal dual method what we do in the primal dual method, we do the following.

We maintain a dual feasible solution. And we maintain a primal infeasible solution $(x_j)_{j \in [m]}$ ok and we ensure one part of complementary slackness. So, we ensure that x_j greater than implies jth constraint jth dual constraint is tight . So, whenever x_j is greater than 0 the jth dual constraint is tight and this is the one part and the other way also that ensure this is for all j.

and y_i greater than 0 implies i th primal constraint is tight . Now, if this holds for both primal and dual feasible solution, then from complementary slackness we can say that both x and y are optimal solution, but you know this x is infeasible. So, and we iteratively make primal solution, primal assignment more and more feasible. So, this part is not always this is sometimes this is ensured sometimes. not always and this is typically ensured for polynomial time algorithms.

So, primal dual method is also used for successfully for designing polynomial time algorithms for various other problems like mean cost flow, matching and so on. So, not only approximation algorithms but primal dual method primal dual framework is also useful for designing polynomial time algorithms for various important problems for example, mean cost matching, mean cost flow, maximum matching etcetera ok. So, with this high level framework let us see how we can apply this framework for designing approximation algorithm for set cover weighted set cover. So, what is the first step? The first step so, we will maintain a dual feasible solution. So, we need to pick an initial dual feasible solution.

need to find a an initial dual feasible solution. Here also in the next step we iteratively make the primal assignment more and more feasible and update dual and update dual variables ok. So, and here primal infeasible solution which is integral. Recall that this linear program primal LP is a relaxation of the ILP and so, at the end we need to find a integral solution which corresponds to a solution in the of the original problem. So, our

initial primal dual feasible solution is all y_i equal to 0 for all i in n see if I set all y_i to be 0, because all weights are non negative all these constraints are satisfied and y_i is in between 0 and 1.

So, this is a is a dual feasible solution. For maintaining the primal for maintaining the primal infeasible solution which is integral instead of maintaining this variables x_j 's we can maintain the solution which corresponds to a set cover the subset of elements that we are subset of sets that we are picking. So, that we are initializing it to empty set. So, what is the corresponding x_j 's all x_j is 0 that is the initial primal solution, but it is primal infeasible solution. So, we can say it is a primal it is an assignment to primal variables it is it is infeasible it does not satisfy the constraints.

So, next what we do we it is an iterative algorithm while there exists an element e_i in the universe which is not covered by I. Then what do we do? We the idea is we pick if there is an element e_i which is not covered, then the corresponding y_i we increase that dual variable until as long as we can until some dual constraint becomes tight. So, keep increasing y_i until some dual constraint involving y_i becomes tight . So, observe that we are also maintaining ah this condition that whenever x_j is greater than 0. So, these conditions are also maintained at the beginning both of them and we are increasing y_i .

So, if we increase y_i . So, all this initially all y_i 's are 0. So, the all this constraints were not tight an inequality is called tight if it is an equality. So, initially all y_i 's are 0, we can assume without loss of generality that all w_j 's are greater than 0. All sets have some positive weight because if some set has weight 0, I can always pick those let us pick those sets in my solution and we can reduce the instance.

So, without loss of generality assumeis w_j greater than 0 without loss of generality ok. So, if you keep increasing y one of the constraint will become tight. So, Suppose $l \in [m]$ be such that suppose the l-th constraint becomes tight that means, $y_i^{'}$ such that $e_i^{'} \in S_l$ this is equal to w_l . So, what I do? I pick that in my solution $I = \cup S_l$ or let us keep the indices only l or S_l . and this we continue and this loop exits when I is an set cover.

So, return I ok. So, this is the algorithm here you see it is a purely combinatorial algorithm in the sense that we are not solving the linear program ok. Now, let us analyze. So, here is a claim that the above algorithm has an approximation ratio. of at most f, where f is the frequency maximum frequency of an item, it is the maximum number of sets that an element can belong to. So, proof the proof is quite similar to the dual

rounding algorithm.

So, what is ALG? is the sum of the weights of the sets output by the algorithm. So, this is $\sum_{j\in[m]:S_j\in I} w_j$ ok. Now, for these sets so, let why $(y_i)_{i\in[n]}$ be the dual feasible solution when the algorithm terminates ok. So, for each set recall you see we are picking the set S_l whenever this dual constraint Lth dual constraint is tight. So, for this we can write this is w_l equal So, this constraint becomes tight.

So, for this w_l is equal to $\sum_{j\in[m]:S_j\in I} w_j \sum_{i\in[n]:e_i\in S_j} y_i$ star ok. Again we have a double sum and this is a recurring theme that whenever you have a double sum try to switch the order of the sum. So, now let us look at from elements point of view $e_i\in S_j$. y_i^* and how many sets in the solution and this is cardinality of the sets such that $e_i\in S_j$ that many times y_i^* appears in the sum this number is at most f.

So, this is less than equal to $f\sum_{i=1}^{n} y_i^*$ ok all elements appear. So, here also i equal to 1 to n. because it is a set cover I is a set cover. And this is and at this step this is a dual solution and this is less than equal to primal opt. which is less than equal to opt.

So, this shows that this is again an f factor approximation algorithm purely combinatorial where linear programming and duality we are using only in the analysis of the algorithm ok. So, let us stop here. Thank you.