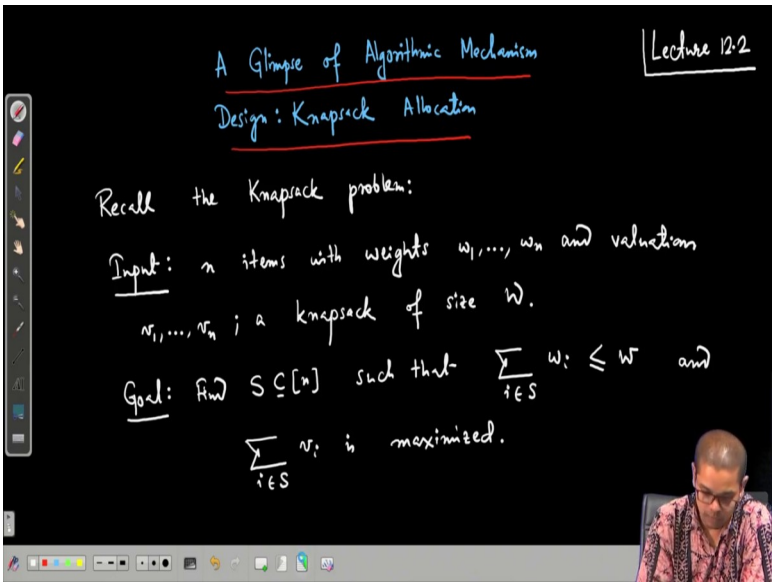
Algorithmic Game Theory
Prof. Palash Dey
Department of Computer Science and Engineering
Indian Institute of Technology, Kharagpur

Lecture - 57
Algorithmic Mechanism Design

So, till the last we have seen implementable social choice rules and implemented implementable allocation rules in various single in various (Refer Time: 00:37) environment. In today's lecture we will see a glimpse of Algorithmic Mechanism Design this is not even a tip of iceberg.

This is just a glimpse just to give you a flavor and this algorithm algorithmic mechanism design is currently a very advanced field of active research in theoretical computer science and game theory.

(Refer Slide Time: 01:03)



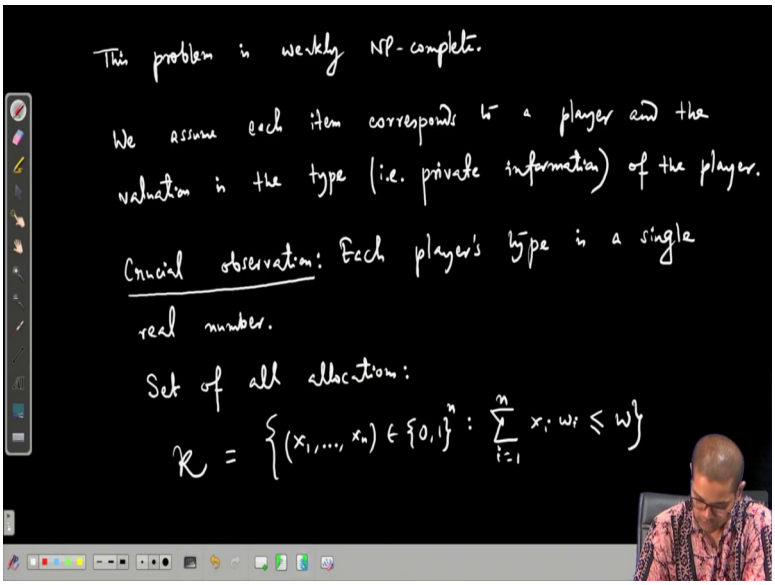
So, today's topic is a glimpse of algorithmic mechanism design and for that we will take an allocation rule which is called Knapsack allocation in a Knapsack problem we will study the Knapsack allocation rule. So, we all know the classical Knapsack problem. So, let us recall the Knapsack problem what was the problem? What are the inputs?

Input n items with weights $w_1,\ldots,w_n$ and valuations $v_1,\ldots,v_n$ and Knapsack of size w and the goal is to find a subset S of items $\{1,\ldots,n\}$ such that the $\sum_{i\in S} w_i \leq W$. The size

of the Knapsack and the valuation is maximized and subject to this weight constraint the sum of weights is at most W subject to this constraint the valuation of S the what is v of S?

It is $\sum_{i \in S} v(i)$ is maximized ok. This is the classical Knapsack problem and we know that this is weekly NP complete.

(Refer Slide Time: 05:01)



So, this problem is weekly NP complete what does weekly NP completeness means? That this problem is NP complete, but if the input if the in input integers are encoded in unary, then it is polynomial and solvable. Say in the same thing in other words there exists pseudo polynomial time algorithms for it. Now what is the game theoretic masala in this problem? We assume each item corresponds to a player and the valuation of the items say i evaluation v_i of item i this is the type of player.

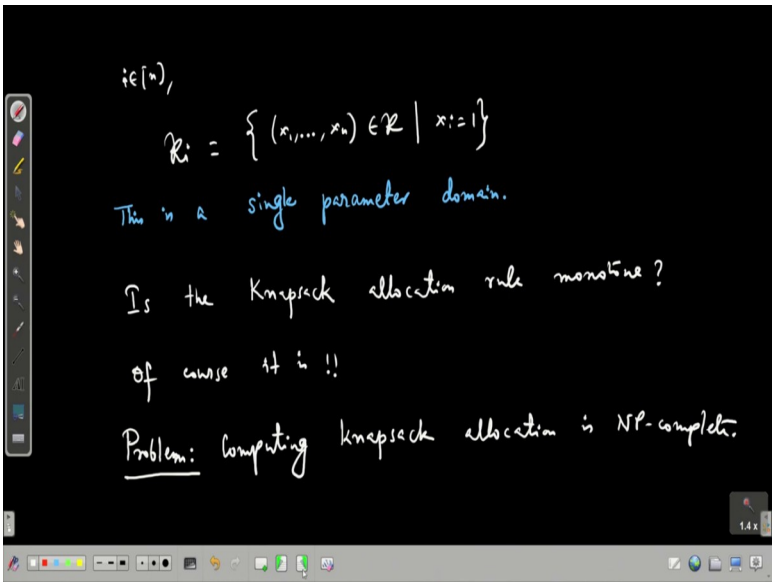
Valuation is the type which is that means, the private information known only to player i is the type that is private information the valuation is the type of the player. Now to solve this problem we first need to design a mechanism to a to design a game or mechanism so, that it is the best interest of the players to reveal the true type that is what we mean by implementability.

So, why players would reveal their true type? First observe that the type the private information is a single real number is the crucial observation is each players type is

single real number ok what are the set of all allocations? The set of all allocations let us write it set of all allocations k is you know this this vectors $(x_1,\ldots,x_n)\in\{0,1\}^n$ such that this is the; this is the sort of the allocation vector if $x_i=0$; that means, the i-th item is not picked if $x_i=1$; that means, the i-th item has been picked.

So, the constraint is $\sum_{i=1}^{n} x_i w_i \leq W$. So, these are the set of all possible all valid subset of item that can be picked and what is the condition for validity? It is that the sum of the weights must be at most W.

(Refer Slide Time: 09:55)



And for a particular player $i\in[n]$ what is K_i? K_i is all those vectors allocation vectors in k whose i-th component is 1. And you see that players as far as player i is concerned for player i, it on only depends on the whether the whether the allocation chosen belongs to K_i or not from player is perspective its utility depends only on whether its item has been picked or not. So, this is exactly what is the what is our basic model of single parameter domain.

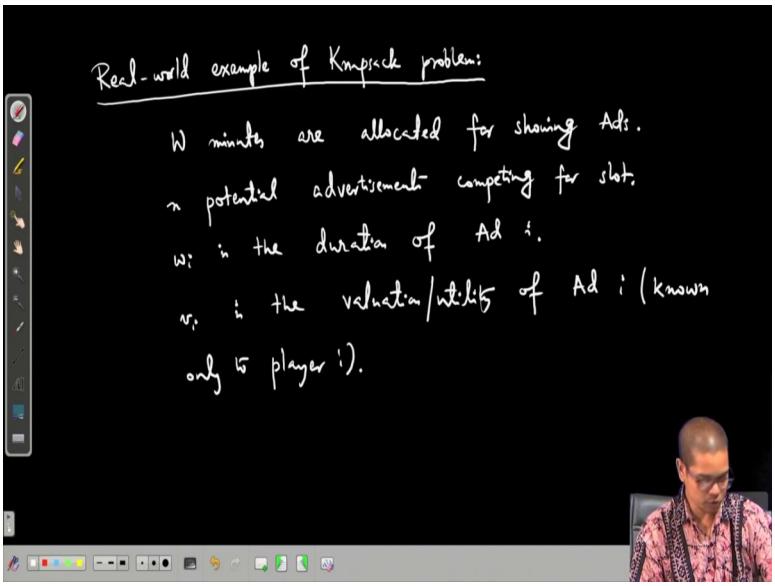
Each type is a real number the set of all types you can think of a set of all real numbers may be. So, it is an interval and there is a set of allocations and among the set of allocation there is a subset of allocations K_i where player i wins and in the other player i loses this. So, this is. So, this is single parameter domain ok.

This is a basic single parameter domain and you know we know that in this thing guess if we if the if the allocation rule is monotone then it is implementable. So, is the allocation rule monotone is the Knapsack allocation rule monotone in each component. Of course, it is clearly it is. How? What does monotonicity says, if the type profile of all the all other players.

So, let us focus on a player i and if a type profile of other players remain same in this particular context it means that; that means, that the valuation of all other items except i remains same and for a particular type of player i θ_i or in this particular example with a particular valuation for item i if player i wins; that means, if that item is picked then if player i increases its type if its valuation increases, then we will continue to picking up this object in the in our solution. So, this is clearly monotone.

So, what is the big deal because then then we can apply they then we can apply the critical value and that will be the payment and everything is fine. The problem is the problem is computing Knapsack allocation is NP complete its computationally intractable. And you know this Knapsack problem is not just a problem which is only of theoretical interest, this is very practically motivating problem it appears in many practical examples.

(Refer Slide Time: 14:40)



For example so, let us write. So, real life real world example of Knapsack problem. So, think of a say a popular TV show and there is a certain amount of time allocated for

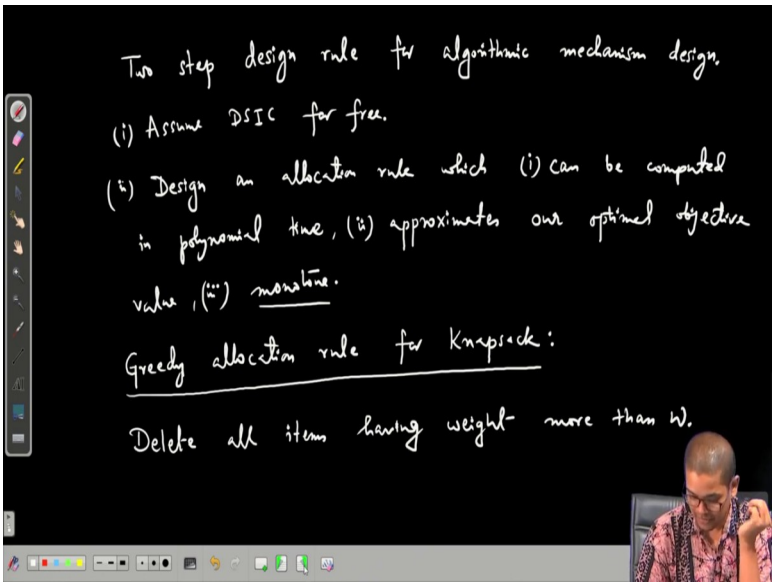
showing ads. So, W minutes are allocated for showing ads and there are n potential advertisements n potential advertisements competing for slot.

There are n advertised and advertisements and each want their Ad to be shown each ad has a certain interval it needs certain amount of time to be shown. So, w_i is the duration of Ad i and each particular ad has a certain valuation its call it a goodness or appeal or utility of that ad to a society, but that is known only to player i.

So, v i is the valuation or utility of Ad i known only to player i and so, the idea the question is how will I how will I choose these ads and what algorithm I should use so, that not which not only picks the best ad, but ensures that players reveal their true valuations and that is very easy because if you pick any monitor allocation rule and the Knapsack allocation rule is monotone, but the problem is it is NP complete.

So, the whole play or the challenge of algorithmic mechanism design is to come up with approximate good approximation of the desired allocation rule which are still monotone. And which approximates the optimal solution closely and why and so, and why we need monotronicity? To ensure that players reveal their true type in the best of their interest.

(Refer Slide Time: 18:40)



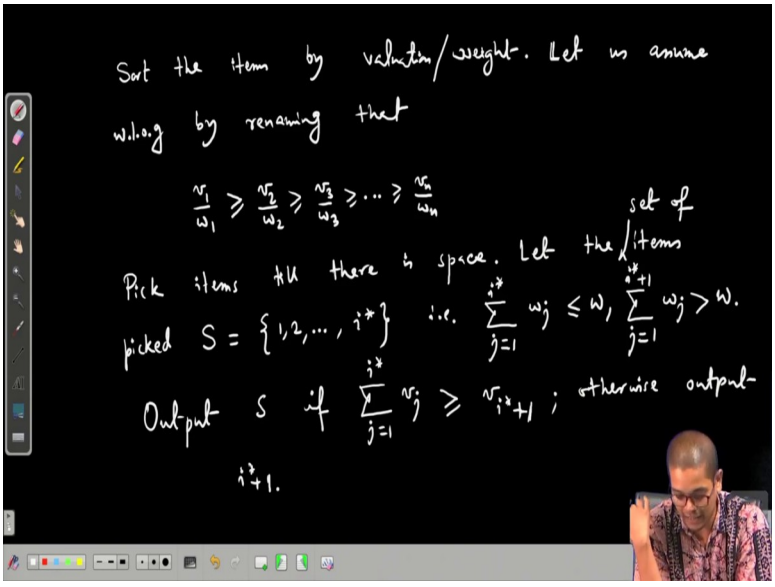
So, here is a two step paradigm design rule of two step design rule for algorithmic mechanism design. The first tip is assume DSIC for free. Assume that players reveal their true type in the best of their interest that is why what DSIC means and then second

step is design and allocation rule which of course, can be computed in polynomial time, approximates our optimal objective value and monotone.

This is the extra requirement that we have in this algorithmic mechanism design and only because of this monotonicity we can assume that the DSIC is free this monotonicity justifies our assumption in step 1 how we could assume DSIC for free. So, in the context of in the context of Knapsack problem let us see how we can design an allocation rule which satisfies all these three properties.

And the allocation rule for Knapsack problem is very simple, it is called there is a very simple greedy allocation rule. So, greedy allocation rule for Knapsack what does it say? First discard all items whose weight is more than w because those items are never participate in any outcome in any solution delete all items having weight more than W. So, we can assume that without loss of generality that the weight of all the items is weight or size is at most W.

(Refer Slide Time: 23:17)



Then you sort the items by valuation by weight and by renaming. So, let us assume without loss of generality by renaming that $\frac{v_1}{w_1}$ whichever item has the highest valuation by weight. Let us call that item 1. So, $\frac{v_1}{w_1} \geq \frac{v_2}{w_2} \geq \frac{v_3}{w_3} \geq \ldots \geq \frac{v_n}{w_n}$ and the algorithm is pretty

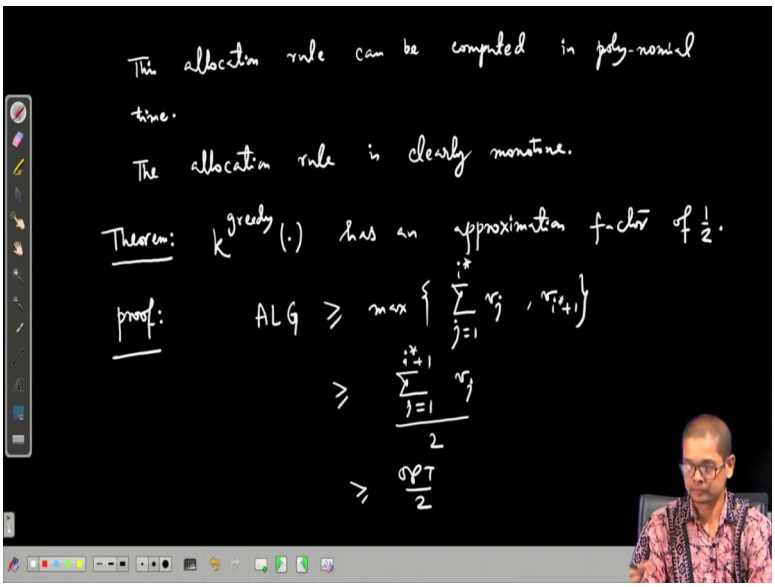
simple keep picking items according to this order 1, 2, 3 till there is space in the there is space in the back in the Knapsack.

So, pick items till there is space let the items picked let the set of items picked be $S=\{1,2,\ldots,i^*\}$ that is the sum of weights of the first i^* items $\sum_{j=1}^{i^*} w_j \leq W$, but if I go till i^*+1 the weight exceeds $\sum_{j=1}^{i^*+1} w_j > W$ ok.

And then what should i output? So, output S if the total valuation of the items in S $\sum_{j=1}^{i^*} v_j$ this valuation exceeds or greater than equal to the valuation of (i^*+1)th item. On the other hand if the valuation of all the items in sum of valuations is less than the valuation of (i^*+1)th item because each item fits in the Knapsack the weight of (i^*+1)th item is also less than equal to W.

So, otherwise if; that means, if this if the valuations of items in S is strictly less than valuation of (i^*+1) then otherwise output otherwise output (i^*+1). So, this is clearly polynomial time algorithm because it just involves some divisions and sorting which can be done in order $O(n \log n)$ time.

(Refer Slide Time: 28:44)



So, this allocation rule this allocation rule can be computed in polynomial time this allocation rule can be computed in polynomial time and it is clear that it is monotone. So, to check that if you see that if the output is S; that means, the set contains the first i^*

items and if the valuation of some item increases, then the same set will continue to out to be output.

On the other hand, if v_{i^*+1} is output and if that item the its valuation increases while the valuation of all other items remaining same then that item continues to be picked. So, it is again the allocation rule is clearly monotone clearly monotone. How good is the approximation factor?

So, let us see. So, here is the theorem let us call it k greedy has an approximation factor of as an approximation factor of half; that means, its valuation the quality of the solution should be at least half the optimal solution proof ok. So, what is k greedy? Let us let us call k grid the alk the output of the value of the output of the algorithm l this is greater than equal to $max\{\sum_{j=1}^{i^*} v_j, v_{i^*+1}\}$.

So, this is greater than equal to max of 2 numbers is greater than the average these the is the averaging principle $\frac{\sum_{j=1}^{i^*+1} v_j}{2}$. And this numerator is an upper bound for OPT

. So, this is greater than equal to $\frac{OPT}{2}$ which shows that this is a half factor approximation algorithm which concludes the theorem. So, we will stop.