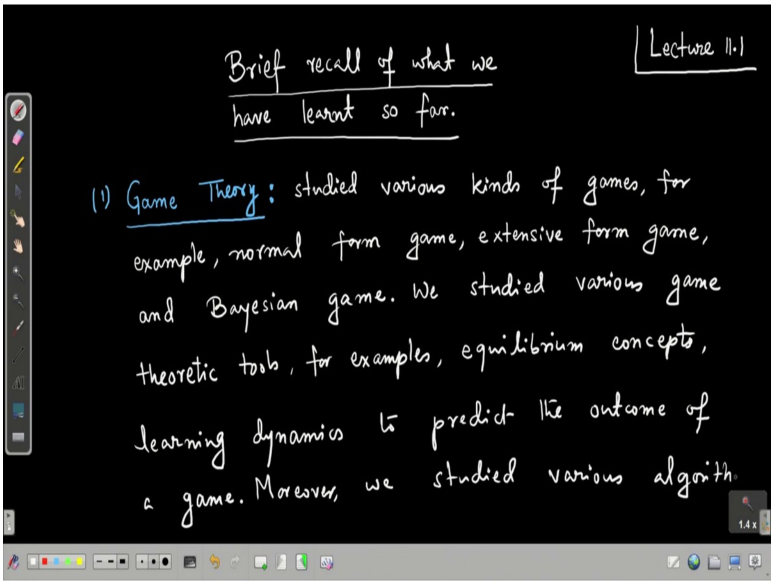
Algorithmic Game Theory
Prof. Palash Dey
Department of Computer Science and Engineering
Indian Institute of Technology, Kharagpur

Lecture - 51
Recap of Topics Discussed so Far

Welcome. So, in the last from the last couple of lectures we have been doing mechanism design. So, let us do a brief recall of what we have done till now before we start a new content.

(Refer Slide Time: 00:39)

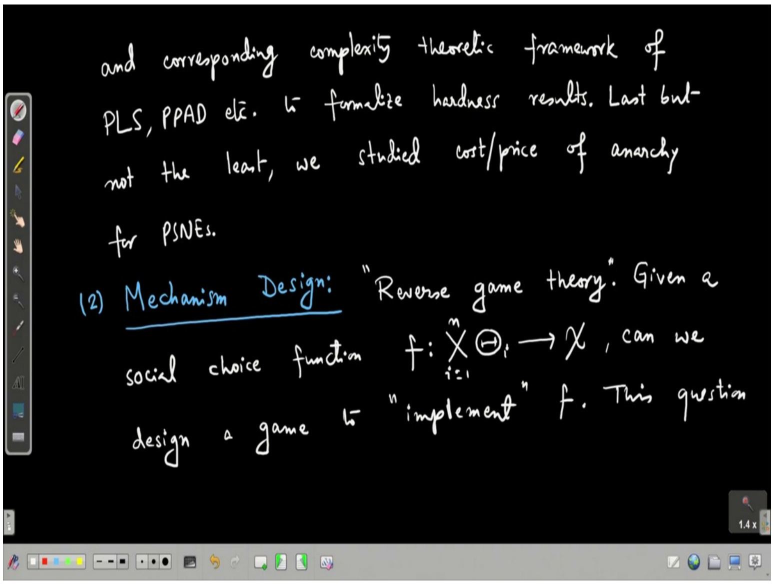


So, brief recall of what we have learnt so far. In the first part of the course we studied game theory and there we studied various kinds of various kinds of games for example, normal form game, extensive form game and Bayesian games. Actually we have dedicated most of the of most of our time and energy to study normal form games and then we defined extensive form game and Bayesian games.

And observe that extensive form games and Bayesian games can be suitably modeled by appropriate normal form games. So, although normal form games looks simpler than extensive form game and Bayesian game yet it is powerful enough this model is rich enough to capture other more sophisticated models of games like extensive form games and Bayesian games.

Not only that we studied various game theoretic tools. For example, for examples, equilibrium concepts, then learning dynamics to predict the outcome of a game. Moreover we studied various algorithms.

(Refer Slide Time: 05:41)



And corresponding complexity theoretic framework of PLS, PPAD etcetera to formalize hardness results. So, various algorithmic questions like given a game in normal form can you find a PSNE, can you find a MSNE. This these are the sort of questions and we have observed that for many class of games these questions can be polynomial insolvable, then we have learned those algorithms.

But you know there are various other kinds of games and for example, finding a mixed strategy Nash equilibrium even for a biometric scheme for any arbitrary biometric scheme. We do not know any algorithm and so, we need to we need to establish or need to justify the hardness of the algorithm hardness of designing the algorithm for those problems you want to tell that these problems are inherently hard.

And the way we formalize it is that through hardness framework which basically at the heart says that if you can solve these problems in polynomial time you can solve a lot of other problems in polynomial time efficiently for which there has been a lot of research effort and time has been spent and people still do not know how to solve those problems in polynomial time.

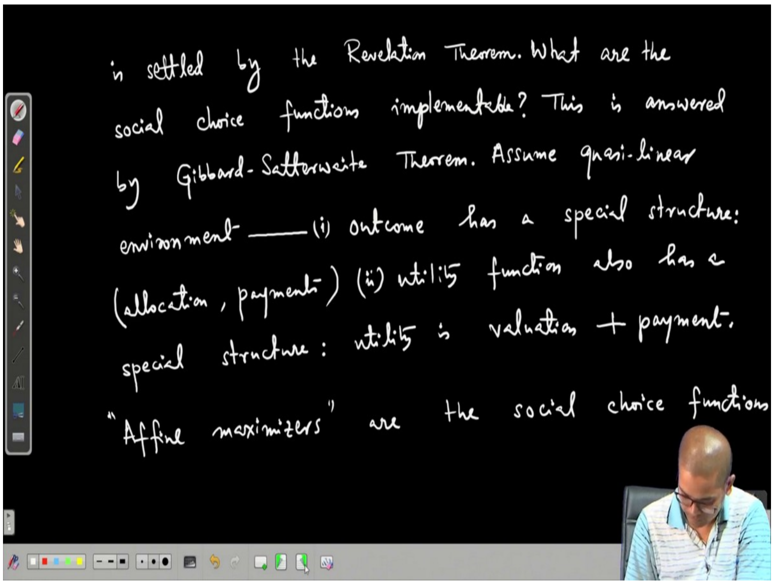
So, that is what we mean by formalizing hardness results. And also last but not the least we studied cost or price of anarchy for PSNEs. So, each player in a game theoretic environment each player is fully independent they are selfish they can the rational intelligent and so on and so forth. So, how much we lose on system performance due to this? So, the that we quantified as price of anarchy.

So, this was the first part of the of this of our course, which is which deals with game theory. The second part is mechanism design which is also sometimes called reverse game theory. Why reverse? Here the main question is that how can we design game. So, the fundamental question is that given a function which in the context of mechanism design is called a social choice function f from type profiles to outcomes can we design a game to implement f.

Now, what do you mean by implement f? It is because of the input the function of the input of this of this function f which is a type profile is not directly available it is distributed input is a tuple of n coordinates $(\theta_1, \ldots, \theta_n)$. So, it is distributed across n players and each player is rational, strategic, intelligent and so on. So, players needs to be incentivized to reveal their true type. It may not be in the best interest of the players to reveal their true type.

So, the question is can we design a game on the players so that when the players participate in the game it is in the best interest to participate in the game in such a way which enables the social planner to implement f. What do you implement f? If $(\theta_1, \ldots, \theta_n)$ is the actual or the actual inputs held by n player then mechanism designer should be able to compute $f(\theta_1, \ldots, \theta_n)$.

(Refer Slide Time: 12:47)



Now, this question is settled by is settled by the revolution theorem by the revolution theorem which basically says is that mechanism designer does not need to do any kind of work. It can simply ask each player to reveal its type and this particular mechanism is called the direct mechanism and revolution theorem says that if direct mechanism cannot implement a social choice function f then there is no mechanism which can implement these social choice function f.

So, that is revolution theorem. Then the next question is that what are the what are the social choice functions that are implementable. What is the set of social choice functions, can we characterize this? Can we find what are which social choice functions are implementable? And this question is settled this is answered by the celebrated Gibbard-Satterweite theorem.

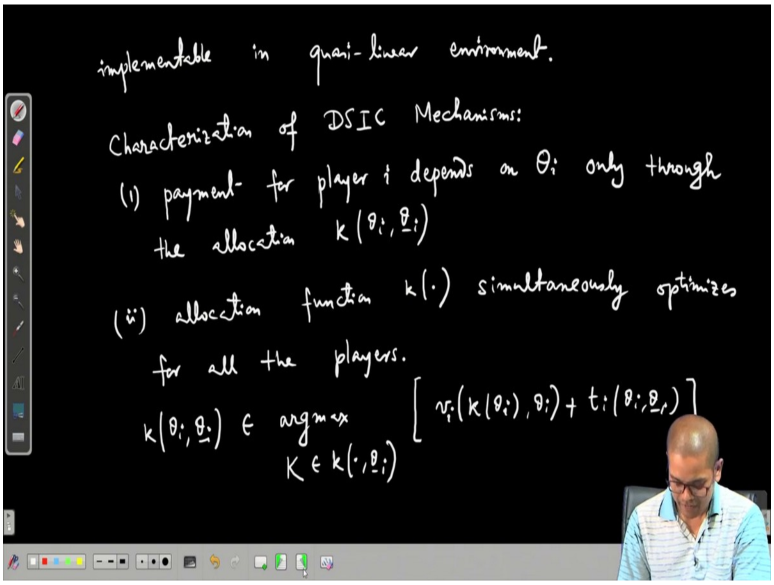
Which essentially says that if there are at least three outcomes in the set of all possible outcomes x and if the function this social choice function is unanimous and it has at least three outcomes then the only social choice functions that are implementable in dominant strategy equilibrium are the dictatorial choice functions. So, this is very negative, it is a sort of a very pessimistic result that only dictatorial social choice functions are implementable.

Then we asked ok then what we can do and there are various approaches to tackle this and one very successful approach is that assume quasi linear environment. Now what do you mean by assume quasi linear environment? It basically says that the outcomes each outcome cannot be arbitrary.

So, first is that outcome has a special structure, it is allocation and payment. Outcome has two parts. And the second is the utility function; the utility of a player at a particular outcome at a particular type utility also has a special structure; utility function also has a special structure. What is that the structure? Utility is valuation plus payment.

So, in this particular environment then we ask the same questions what are the set of what is the set of all implementable social choice functions. And by Groves theorem and generalized Groves theorem and at the end due to Robert's theorem we the final result is that you know basically affine maximizers is the affine maximizers are the social choice functions implementable in quasi linear environment.

(Refer Slide Time: 19:06)



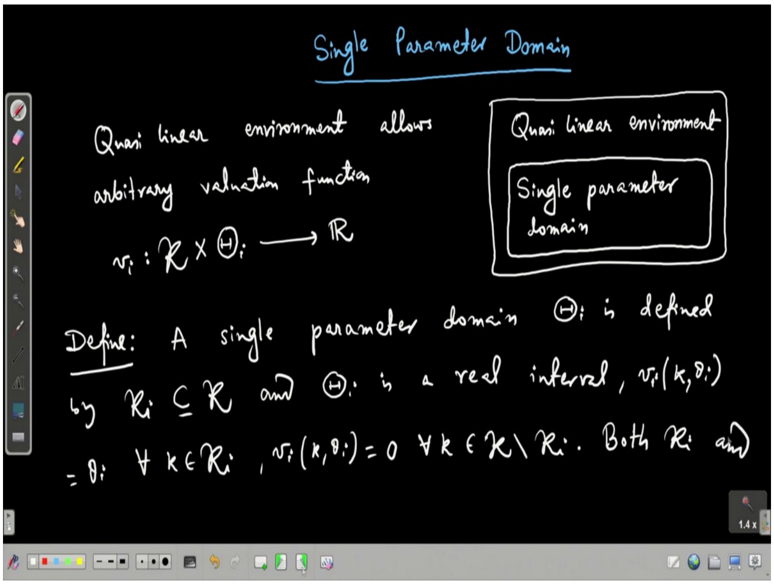
So, after this we now began to study further the implementable social choice functions in the quasi linear environment from some other perspectives. And in the last class we stopped with this theorem that we studied a characterization of DSIC mechanisms. What is the characterization? It basically says that any social choice function is implementable in dominant strategy equilibrium or equivalently it is dominant strategy incentive compatible if and only if the payment and the allocation has certain structure.

So, what structure? Payment for player i depends on its type θ_i only through the allocation $k(\theta_i, \theta_{-i})$. And the allocation function which is k of type profiles allocation function simultaneously optimizes for all the players. What do you mean by that? It is $k(\theta_i, \theta_{-i})$.

So, for each player $k(\theta_i, \theta_{-i})$ if you fix θ_{-i} and vary θ_i then this belongs to $arg\,max_{k \in k(.,\theta_{-i})}$. You vary θ_i and see what are the various case available and $k(\theta_i, \theta_{-i})$ should be one of those outcomes which maximizes the utility of player i which is $v_i(k(\theta_i, \theta_{-i}), \theta_i) + t_i(\theta_i, \theta_{-i})$. So, this we have stated and we have seen in the last class.

So, with this characterization we now go further deep into further specialization of single peak domain. We move we put more structure on single peak domain and ask again what are the social choice functions which are implementable in the hope that you know in many real world applications although they are a quasi linear environment, but they have more additional structures, additional constraints, additional properties and due to which we may have more social choice functions which are implementable.

(Refer Slide Time: 24:30)



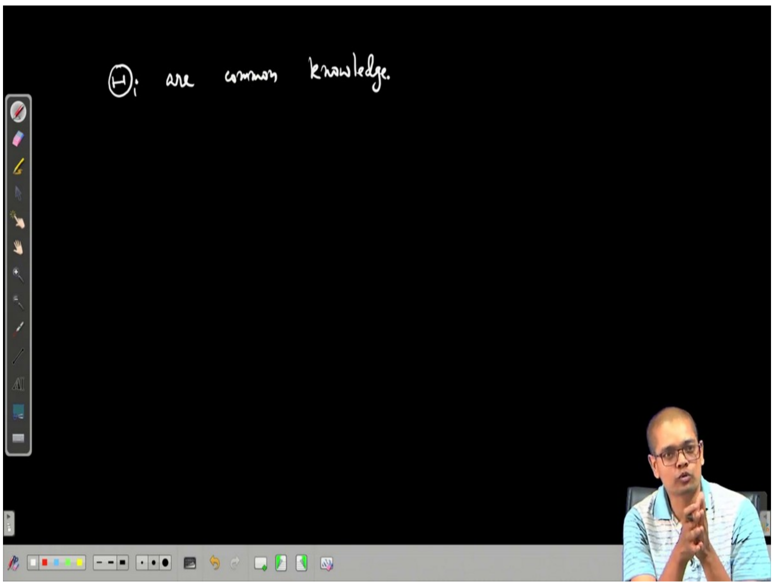
So, that is the first one such structure we are studying which is called single parameter domain; single parameter domain. So, first this picture should be clear that we have quasi linear environment and among quasi linear environment we have certain assumptions like outcome should have certain structure utility must have certain structure we put more assumptions and what we get is single parameter domain.

So, in some sense quasi linear environment allows arbitrary valuation function $v_i : K \times \Theta_i \rightarrow \mathbb{R}$. But we observe that you know many such settings many for example, an auction scenario, it all depends on whether the player valuation of an allocation depends on whether the player wins the auction or loses the auction. So, this sort of extra structure is sort of exploited or these are a taken to define single parameter and domain.

So, let us define single parameter domain. A single parameter domain Θ_i is defined by a subset $K_i \subseteq K$. So, k is the set of all allocations and think of K_i where a subset of allocations where player i wins in some sense. And so, subset of allocations and this Θ_i the set of types is not arbitrary, but this is a real interval. For example, in auction scenario the type is the private valuation which is some real number which is often assumed to be a within 0 and 1.

So, theta i cannot be something abstract sit, it is a it is a real interval ok. And how does the valuation function defined? And valuation $v_i(k, \theta_i) = \theta_i$ for all $k \in K_i$. It is like when player i wins; that means, $k \in K_i$ then its valuation is the type itself. $v_i(k, \theta_i) = 0$ for all k in for all other allocations where player i does not win ok. And both K_i and Θ_i are common knowledge.

(Refer Slide Time: 30:06)



So, like auction for each player all players know for what outcomes that player wins and all player knows the set theta i capital theta i is always a common knowledge the set of all types is a common knowledge, but what is the actual type it that is only known to the player i ok. So, we will stop here today. We will continue in the next class from here.