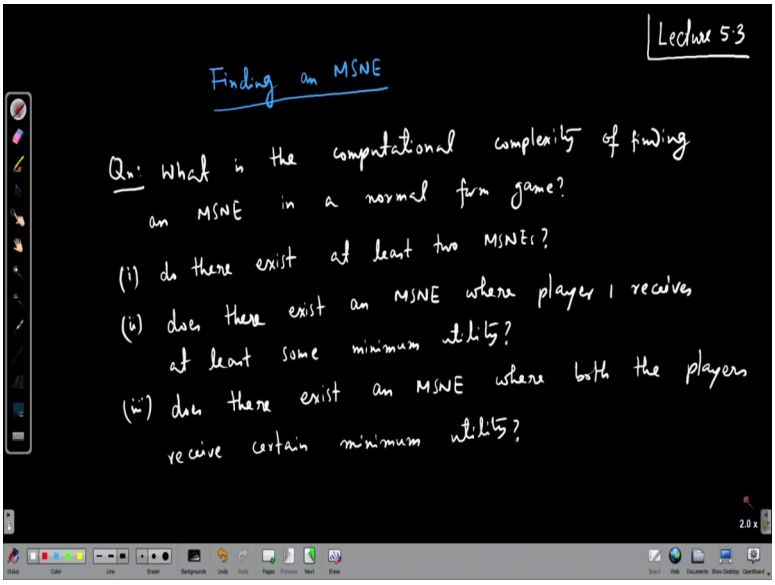
Algorithmic Game Theory
Prof. Palash Dey
Department of Computer Science and Engineering
Indian Institute of Technology, Kharagpur

Lecture - 23
Functional NP

Welcome. So, in the last class, we discussed the complexity of finding a PSNE for congestion games; we have seen that it is a PLS complete problem. So, in today's class, we will focus on the problem of come finding a MSNE, finding an MSNE in a normal form game.

(Refer Slide Time: 00:43)



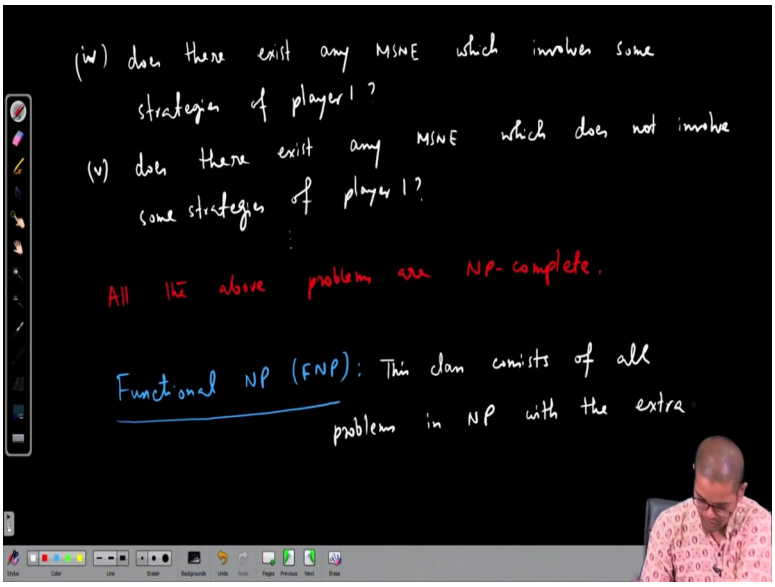
So, again we, so what is the main question? What is the complexity what is the computational what is the computational complexity of finding an MSNE in a normal form game? And again we observed that if we change this problem and convert it to a decision problem, like does there exist a MSNE for a normal form game; then the problem is not hard anymore the problem becomes trivial, because of Nash theorem the answer is always yes.

So, to tackle this scenario what people have tried is that, they are looking for some something extra over the, over what is guaranteed by the Nash theorem. For example, people have studied complexity of following questions, so and these are all decision questions; do there exist at least 2 MSNEs? One MSNEs guaranteed by Nash theorem;

so does there do there exist 2 MSNEs or does there exist an MSNE, where some players say player 1 receives at least some minimum utility?

Again only existence of mixed strategy Nash equilibrium is guaranteed by Nash theorem; but we are asking ok, so the player 1 needs to get at least certain utility, does there exist an MSNE which guarantees that? Or we can ask question like does there exist an MSNE, where both the players receive certain minimum utility?

(Refer Slide Time: 05:11)

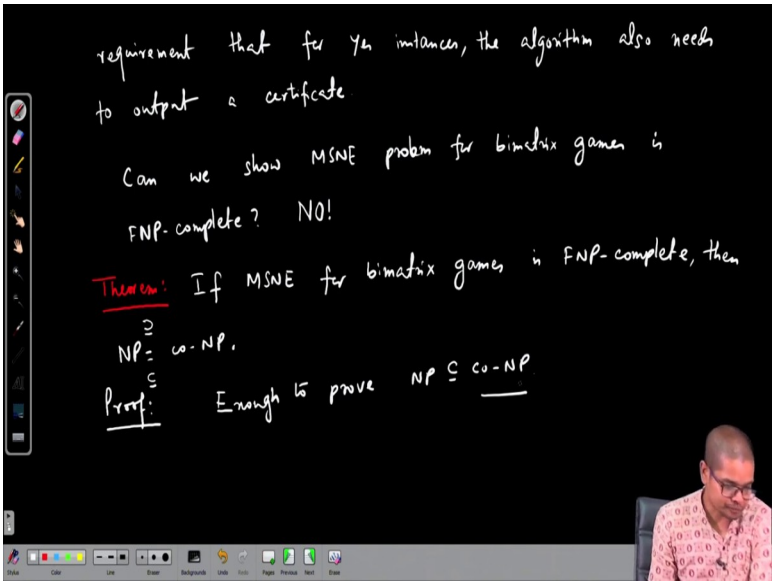


Or does there exist an MSNE, which involves some strategies? That means, some strategies must have non-zero probability; does there exist any MSNE which involves some strategies of say player 1? Or does there exist any MSNE, which does not involve some strategies of player 1? Say some set of strategies must receive 0 utility and so on this continues.

There are list of problems and what is striking is that, all the above problems is red color, all the above problems are NP complete. So, this hints that you know maybe the problem of finding an MSNE is also hard; but how will you formally prove it? So, towards that we introduce complexity class called functional NP, in short FNP. So, what is the idea? Why NP did not work? Because if I pose the MSNE problem as yes no problem is a decision version; then the answer is trivial, the answer is yes and that is because of the Nash theorem.

So, if we want that you know only yes no answer is not enough, you also need to give me a Nash equilibrium, a mixed range Nash equilibrium; then you know then you can simply not say with yes or no, you cannot simply say only yes, because it follows from Nash theorem. You need to compute you need to find a mixed in Nash equilibrium that is what we want.

(Refer Slide Time: 09:52)



So, to capture that that intricacy, what we do is that, we define a functional NP and you know this is FNP; this class consists of all problems in NP with the extra requirement that for yes instances, the algorithm also needs to output a certificate ok, so that a verifier can verify.

So, the a set in this case is called functional set, all the problems and you use the adjective functional and those problems are below, those problems belong to that functional NP class, FNP class. For example, functional set where give where the input is a set formula and you have to find whether it is satisfiable or not and if it is satisfiable, you need to give unit output satisfying assignment or some certificate, ok. So, this is the setting.

Now, can we show MSNE problem for say bimatrix games is FNP complete? Can we show this; can we show that this FNP complete? Why bimatrix game? Because you know for more than 2 players, 3 players onwards; we have discussed that even if all the utilities are rational numbers, it is possible that the all the mixed Nash equilibrium

involves some irrational numbers. And so, we it is difficult for it is yeah, there is some representation issue and we may need to settle for epsilon MSNE.

So, let us not get into them and let just let us just focus on bimatrix games, where it is guaranteed that if all the utility values are rational numbers; then there exist a MSNE, whose all the probabilities are rational numbers only, ok. So, for bimatrix games at least can we show that it is FNP complete? The answer is no, why?

No and the reason is this theorem; if MSNE for bimatrix games is FNP complete, then NP equal to co-NP. What does it say? What do you mean by NP equal to co-NP? NP is the set of problems intuitively speaking for which it is easy to convince one that it is a yes instance; it is there exist a certificate easily or (Refer Time: 14:12) verifiable certificate for yes instances, for convincing that the answer is yes.

For example, if I have a set formula and if it is satisfiable; then I can simply output a satisfying assignment and anyone can check that assignment and put that assignment values to the formula and check that whether it is it evaluates to true or false. So, if it is a yes instance, then it is easy to certify; but if it is a no instance, then it is bit difficult or it is not clear how to certify no instances effectively or efficiently. So, those are the problems we which belong to NP. And what is co-NP?

co-NP is exactly opposite, the set of problems for which there exist certificate, which is not; there is certificate for no instance, it is easy to is easy to verify that the instances are no instance given a certificate, but not the yes case. For example, unsatisfiability is a canonical co-NP problem; given a Boolean formula is the formula unsatisfiable. If the answer is no, then you can to convince someone; you just give a satisfying assignment and one can check very easily that the satisfying assignment indeed satisfies the formula and hence the formula is not unsatisfiable.

Now, NP equal to co-NP it is bit it is bit unlikely, it is not what we believe to be true and why because; you know intuitively saying we do not know or it is at least not clear how to how to succinctly represent or convince someone that a Boolean formula is not satisfiable.

So, if NP equal to co-NP; that means it is easy to easy to verify or certify that a Boolean formula is not satisfiable, which is a bit unlikely. And this theorem says that if there exist

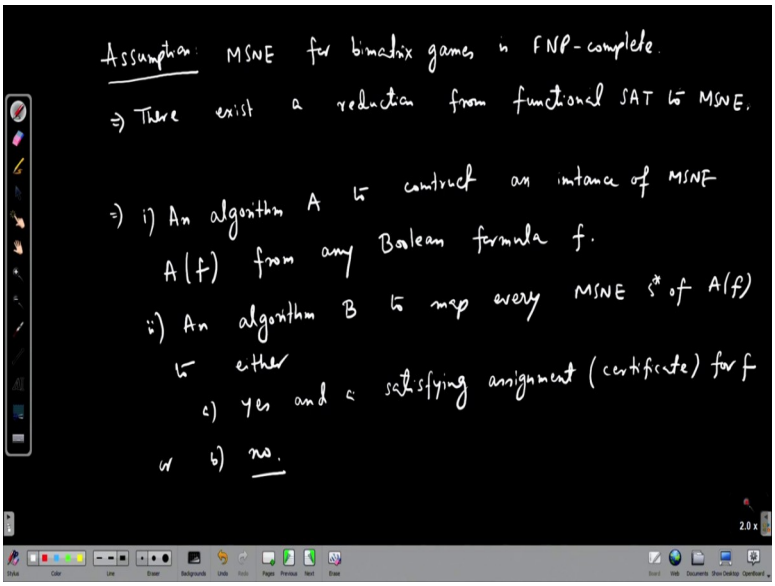
a proof that MSNE for bimatrix games is FNP complete; then we have NP equal to co-NP, which is unlikely. And in this sense we expect that we think that, it is unlikely to have a proof that MSNE belongs MSNE is MSNE for bimatrix games is FNP complete; we do not, we do not expect this sort of statement.

So, proof. So, first we observe that it is to prove NP equal to co-NP, typically when we. So, NP is a set of problems, co-NP is another set of problem; when we want to show 2 sets are equal, we typically show containment on both side that, NP is contained in co-NP and NP contains co-NP.

But we claim first observation is that enough to prove one containment, say NP is a subset of co-NP. What does, why it is enough? If NP is a subset of co-NP; that means a can the canonical NP problem namely set for which we know how to how to certify yes, but if it also belongs to co-NP, that means we know how to certify no and then for the set problem, the satisfiability problem, we know how to certify both yes and no.

Now, why this implies co-NP is a subset of NP? Because the canonical problem of co-NP is unsatisfiability and if we know how to how to how to certify both yes and no instances of satisfiability; then yes instances of satisfiability is nothing, but no instances of unsatisfiability and no instances of satisfiability are nothing, but yes instances of unsatisfiability. So, we immediately know how to certify yes and no instances of unsatisfiability and hence co-NP yes will be a subset of NP. So, it is enough to prove that NP is a subset of co-NP, ok.

(Refer Slide Time: 19:15)



Now, what is the assumption of this theorem? That MSNE is what is the assumption let us clearly write; what assumption do we have? MSNE for bimatrix games is FNP complete. Of course, it belongs to FNP that is not a problem, but it is FNP hard; that means, so this implies that there exist a reduction from MSNE to functional SAT.

Now, again here also like PLS reductions, it has 2 algorithms; same flipper it is same like PLS reduction. So, this implies that there is an first an algorithm A to construct to construct any to construct an instance of MSNE say A of f from any Boolean formula f.

So, f is nothing, but a input of satisfiability set and it is a Boolean formula; the first step is that we there should be an algorithm to construct from through map from f to bimatrix games, for which we need to find a MSNE. And there should be another algorithm which will convert the from, which we should be able to construct or which will be able to construct a satisfying assignment if it is an yes instance or it should be output no from MSNE of A f.

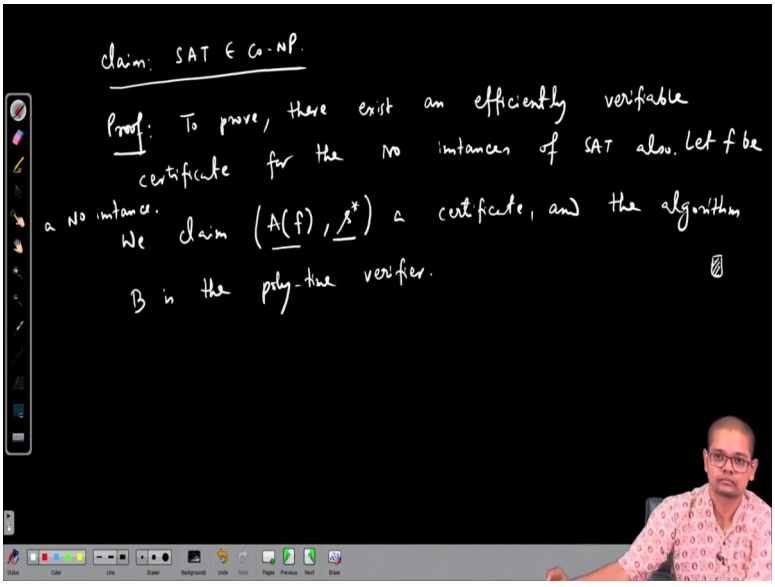
So, an algorithm B to map every MSNE say s star of A f, such that an algorithm B to map every instance of A f sorry to either yes; that means the formula f is satisfiable yes and a satisfying assignment, satisfying assignment which is nothing, but the certificate, yes and a satisfying assignment for yes or no, for no I do not need any satisfying assignment.

So, either of the either of these 2 things, we the algorithm B should be able to do. So, basically you should you should see reductions to be the machinery inlet, so that given an algorithm for MSNE, I can design an algorithm for solving the functional SAT problem.

So, what is the solution? Given a formula f, you apply algorithm B, get a MSNE instance A f, then solve or; that means find a MSNE s star of A f using the using the claimed algorithm hypothetical algorithm, then use the algorithm B to construct or to or to answer the Boolean formula f. Tell whether it is satisfying satisfiable or not; if it is satisfiable, output a certificate a satisfying assignment may be or you output no, ok.

So, this is this what we have this what do we mean by having a reduction from ms from MSNE to SAT, sorry from functional SAT to MSNE sorry yes. So, reduction from not MSNE to functional SAT, functional SAT to MSNE; from functional SAT to MSNE. Now, what goes wrong if there exist 2 algorithms? We will show that if there exist these 2 algorithms, then you know then NP is a subset of co-NP; then it is easy to certify a no instance of SAT.

(Refer Slide Time: 26:39)



Let us see. So, we claim that claim is SAT belongs to co-NP why? So, proof. So, basically to proof, there exist there exist an efficiently verifiable certificate for the no instances of SAT also. Of course, for yes instances there exists a certificate, you simply output satisfying assignment; but what about for no instances, why it is a certificate? We

claim that suppose f is a no instance. So, let f be a no instance; we claim that you know that MSNE instance A of f and along with the an MSNE s star, this is a certificate and the algorithm B is the polynomial time verifier.

What do you mean by that? What is B? The job of B is to look at A of f and s star and correctly a discover that f is a no instance. Now, $A(f)$ is small is poly size; because the algorithm A runs in polynomial time, s^* is a MSNE. And so, $(A(f), s^*)$ is a poly size certificate and the because B is correct algorithm; it should correctly be able to can able to say whether this is a, this is these are no instance and this is exactly what we mean by certificate.

So, what went wrong, why this approach failed? Intuitively speaking you know the main hurdle for MSNE problem is finding an MSNE; the yes no is the yes no part is superficial, the answer is always yes, which is not the case for a for typical FNP problems like functional SAT or so on.

Because you know the there are non-trivial no instance also; it is not that the main hard part lies only in finding the certificate, even deciding whether the whether the instance is at the decision version is a yes instance or no instance there lies the hard part, not finding the certificate. Actually there is something called a self-reduction of SAT or it can be applied for any NP complete problem and it is enough if you have an algorithm for finding yes, no answer; if you have an algorithm, it can be converted to find a certificate.

So, we will see next what further refinement is required to capture the complexity of MSNE problem; this we will discuss in the next class, ok.

Thank you.