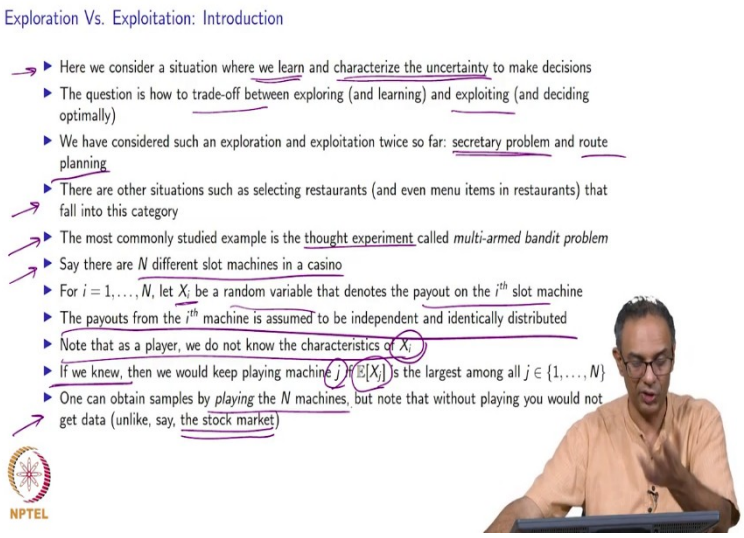
Decision Making Under Uncertainty
Prof. Natarajan Gautam
Department of Industrial and Systems Engineering
Texas A&M University, USA

Lecture – 25
Exploration and Exploitation

The next item is Exploration and Exploitation. Since we have already seen this a couple of times before I am sure- you are not really concerned by this. I would have when I first saw this, you know I would think: "oh boy, are they talking about how you know people in the 1500 and so on explored and went to the west and other places and they exploited".

(Refer Slide Time: 00:42)



No! This is not really that, but we are talking about exploring possible solutions and then taking advantage, it sounds a little bit nastier than it actually is. So, what we want to do here is: we are in a situation; this is a little bit different from what we have seen before, we want to learn and then characterize the uncertainties.

So far we were in a situation, where we already knew the uncertainty characteristics. But, what if you have to learn as you go, and as you go, you also need to make a decision; this is a different situation. We have seen a little bit of it, at least twice, but what we want to do here is: we want to find how to tradeoff between exploring and exploiting. By that, you can learn for a while and you keep learning and I will give you some examples very soon and then that

is exploring and then you exploit by: ok, you figured out which is the best, you keep hitting that.

So, we have seen so far, in the secretary problem you explore, where you look at the first n ovary of them and you reject them all. But, by exploring you at least have an idea of what numbers you can see, what is a good number, what is an odd. And then, you exploit by selecting the one that comes after that which is good. In the route planning, we did the same thing, where we tried different routes and then saw how long it took us in different routes and then we explored a little bit various alternatives and finally we liked one, we exploited- that means, we keep using that over and over again.

We do this in many other situations, we do this in selecting restaurants, we may first explore a bunch of restaurants near especially those of us that are either at a university or at a workplace, where the number of restaurants are fair limited near our offices or our universities or at schools and we explore for a while and then we kind of have a few favorites and we keep going and hitting them. We do that all the time, even do that in menu items, we some of us explore the menu a little bit and once we like some things, we keep hitting it.

So, this happens a lot. Now, the most popular thought experiment, it is called a thought experiment, this is because we really have to think of this contrived situation called the multi-armed bandit problem. Now, I am going to explain this situation and then I draw a picture in the next slide. So, let us say we have N different slot machines in a casino. So, what happens there is- you put in some money and then you play the slot machine and then it pays out a certain amount of money.

So, X_i is a random is a random variable that tells you how much the i^{th} slot machine is going to spit out. Now of course, you have to pay something to play the game. So, let us not worry about how much you pay? I mean we should be worried, but I am just saying that we are just calling X_i as how much it pays out, it could be 0, it could be a certain other amount. Now, we assume that the payouts are IID. Now, this is an assumption, I do not know how the slot machine actually works, we are going to pretend like it is IID.

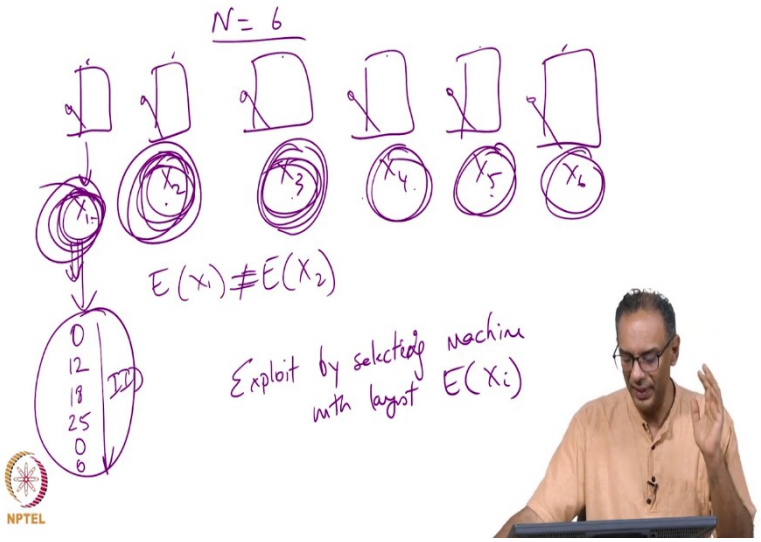
Now, as a player, when you go and show up at one of these slot machines, you do not know what is the characteristics of X_i, you know nothing about it, you just put your money in and you pull the lever and see if you are lucky. Now, if we know what are the characteristics of X_i, then what you will do is- you will take the one that maximizes the expected value for X_j

and keep playing it. Of course, provided the expected value of X_j is larger than the amount of money you put in.

If that is the case, you will keep playing it and then you become really rich very quickly. However, we actually do not know that. Now, why is this multi armed bandit such a big deal, why not some other system? Well, here is the key difference- you will not get any data, unless you play the machine. So, for you to find out how much payoffs you get, you put in the money, you pull the lever and you see what happens. So, without playing, you are not going to get. Now, if you look at the stock market on the other hand, look at the stock prices for various companies, you may have invested in one stock, but you can see the other stocks as well.

You can go and look at it, you can see how much it has grown, and you can hypothesize: "if I put my money in this stock, how would it look like right now? Because you do not have to play, you do not have to put in your money in this stock, that information is already there, but that is not the case here. Here, we are looking at a slot machine, where you are going to get information only if you play or if you have somebody who goes and plays for you.

(Refer Slide Time: 05:33)



So, one more time, let me just formalize this by the following. So, let me pick a number, let us say N = 6. So, let us say I have 6 slot machines all right. So, I have 6 slot machines and there is a lever in each of these, and there is a lever here that I need to pull. So, I pay money and I have to pull and I am sitting there in a casino and I say: ok, I have these 6 machines and

I have some money with me and I want to put my money in and play this and hoping that the payouts are good. So, let us say this one has a payout of X_1, this one has a payout of X_2, this one has a payout X_3, this is X_4, this is X_5, and this is X_6.

Now, how much ever money comes from here, is how much I collect? So, I pay some money and then whatever it spits out. I am going to assume that these numbers, although these numbers are different, these random variables are not IID, they are just independent. However, what comes out, so my first one for example, first one- I could have probably got 0, then I got 12, then I got 18, then I got 25, then I got 0, then I got 0.

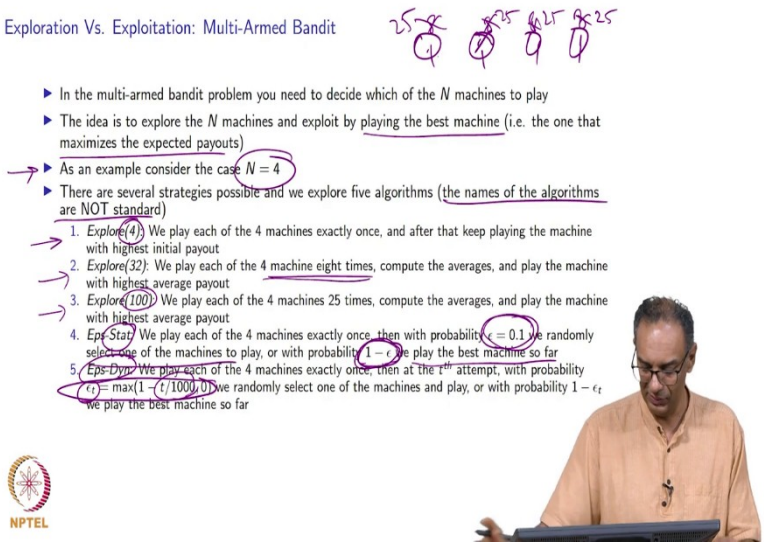
Now, these numbers are assumed to be IID, Independent and Identically Distributed. So, this X_i has a mean. So, each one of this is going to have a certain mean and variance, each one of these values. Now, these values are different. So, expected value of X_1 is typically not equal to, the expected value of X_2, it does not have to be, it could be, but it does not have to be.

So, the distributions could be something very different. So, we are going to assume that all these are random variables according to a distribution that we do not know- we do not know the distributions of each of those. But, we are just going to get samples of that, if we play we get a sample, if you play this next one, we get a sample and we use samples in order to get the mean.

So, for example, if I want the sample mean and that is what we want to do right, we want to pick the one that is the largest. So our ultimate when we exploit; we will exploit by selecting the machine with the largest expected value of X_i. So, whichever is the largest among the expected values, I would select that.

So, how will I explore? Well, I will try this a few times and then I will get something, I will take the average, I will use that as an estimate for average, I will play this a few times, use that as an estimate for my average. I will play this a few times, use that as an estimate. So, I will play these a few times to get some type of an estimate of the expected value and then I will exploit.

(Refer Slide Time: 08:20)



Now, the question is when do you stop exploring and start exploiting? So, how long do you want to explore, how many samples you want to take in each of these is a question. So, the question is, again you want to make sure that you are picking the best machine- the one that maximizes the payoffs. Now, we are going to consider another example, where N equals 4, it is a slightly smaller example. Now, we are going to look at 5 different strategies.

To figure out when should you stop exploring and start exploiting. The first one is what is called explore 4. That means, I have 4 machines: 1, 2, 3, 4. I will explore by trying one number in each of these, whatever random number comes from there, the one that is the largest, I will go with that. So, explore 4, the number 4 would mean: I will explore 4 times, one for each machine. Explore 32 is- I will explore 8 times each machine. So, I had explored this 8 times, this 8 times, this 8 times and this 8 times.

I get 32 sample points, I will compute the averages and the one that is the highest average, I will play that. Explore 100 is the same thing- instead of 8, I will do 25, I will explore this 25 times. So, that if I take average of 25, I am somewhat more confident that I have a representative sample, whose averages I can take and believe and go ahead and pick the one with the largest pair.

Now, those 3 schemes are one in which the number is fixed. Now, there is another thing, that I call “Eps-Stat”. Now, these names are not standard. I made these names up, these are not standard. So, then let us say we do epsilon static, that means- I will play each of these

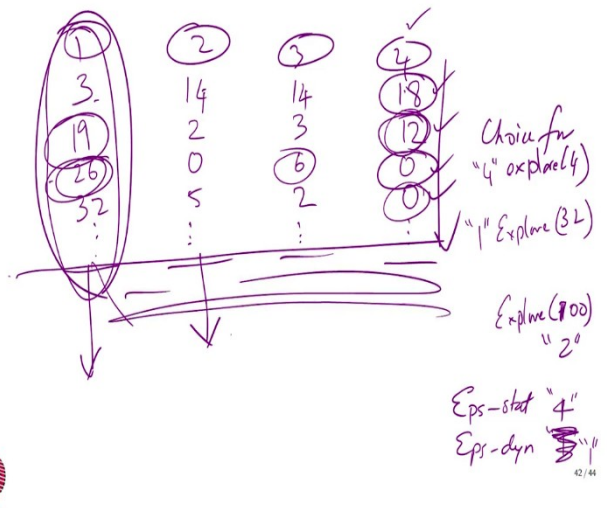
machines exactly once. I will play each of these exactly once, then the one that gave me the largest, I will play that with probability $1-\varepsilon$, we play the best machine so far.

And with probability ε, I will randomly select one of these 4 machines. So with probability 0.4, I will randomly select one of the machines and with probability $1-\varepsilon$, I will pick the best machine that I have seen so far.

Now, why I say so far is because let us say I randomly picked one of these machines. Now, it gives you one more sample point. So, the number of sample points keep increasing and so it keep adjusting your values and at that point of time we are saying: "ok, this is the best machine so far" and you keep exploiting at that point. So, you keep doing both, you explore a little, exploit. So, in this case 90 percent of the time you exploit, 10 percent of the time we explore but, this stays static.

Now, why should we be static, there is an epsilon dynamic algorithm. In this dynamic algorithm what we do is: in the beginning, we explore a lot and later in time we exploit a lot. So, in the beginning, we will explore with high probability, epsilon t is here. So, when t is 0, you will explore with probability 1, you explore with high probability. As t goes closer and closer to 1000, then you are thinking: "ok. Now, I will go ahead and start exploiting". So, that is actually that is an algorithm which is dynamic over time. Now, I do want to do a quick example of these 5 algorithms.

(Refer Slide Time: 11:55)



So, let us say I have my 4 machines: 1, 2, 3 and 4. Let us say- the first machine gives me 3, this one gives me 14, this one gives me 14 and this one gives me 18 for example. Clearly, this one did the best. So explore 4, we will keep using this, this is the choice for explore 4. After 4, it will just keep exploiting. So, explores for 4 times and then it will exploit. Now, if you were to look at explore 32, it will explore 8 of these. Ok. So, 3, 19, 26, 32 and so on. And this one maybe 14, 2, 0, 5 and so on; 14, 3, 6, 2 and so on; 18, 12, 0, 0 and so on.

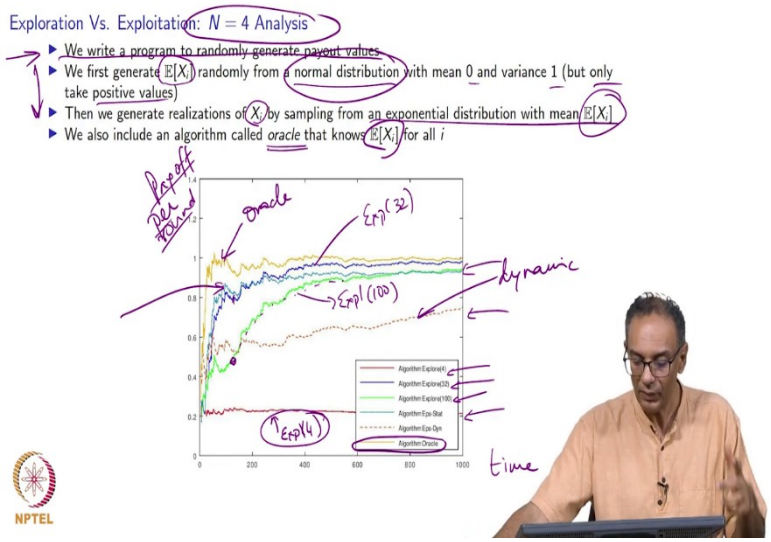
Now, if you look at it, maybe this guy is better and I did not put on all the numbers, it appears to be better. So, explore 32 will probably pick this. So, let me call this as machine 1, machine 2, machine 3, machine 4. This one will pick 4, this one will pick 1 because that one is highest.

Then explore 100. So, notice that once it picks 4, explore 4 will only go along this line, it will not even see what is going on in the others. Now, explore 32 we will, we will go to 32 and then after that, we will pick this and keep sticking with this throughout. Explore 100 would do the same up to 100 and let us say it picked 2 as the best one, it will just stick with 2 after that, after 100 it will stick with 2.

Now, I do want to say a little bit about Eps-Stat and Eps-Dyn and Eps-Stat what happens is it will still pick one. So, Eps-stat will pick one. It will pick one as it starts as its initial choice. Now, it will explore it one more time, it still likes it- that is good, the next time it picks- it will give a 19. Now, 19 seems lucrative, but it will still like this better. Now, it tries it again, it will now give you a 0, then it let us say it tries again gives me a 0, next time it comes back, it picks 26 and let us say it randomly picks 6, it keeps trying here and there and finally selects, keeps keeping tab of what is the average payoff for each of this and it might switch around a little bit.

Now, the epsilon dynamic will mostly be exploring in the beginning. So, it will explore all this. It says the static one will pick 4, the dynamic one will pick 1, it will pick 1 and then it will stick with it, but then it will keep trying the other. So, it will keep exploring for a long time.

(Refer Slide Time: 15:22)



So, let us see how the performance of these guys are for the N equals 4 situation. Now, I am going to write, I mean I am not going to show that to you because it is a fairly long program in octave. So, but I did write that program and then I randomly generated some payout values. Now, as the program it does not know, it does not know any of what I am going to say here, but what I am going to do is, I randomly generate expected value of X_1, I picked that by selecting from a normal distribution with mean 0 and variance 1. So, I could pretty much get any number from let us say negative 3 to positive 3. Now, usually any of those numbers, but I only take positive values. So, once I select the positive values that is my mean. Now, I do not know what gets selected.

So, remember we generate random numbers, I just generate a random number. Then, I create other random numbers- sample from an exponential distribution, with mean given by this expected value. So, that beta that we have is selected randomly. So, we have select a random beta and give it to the exponential and it will start spitting out X_1, X_2, X_3 and X_4. I am also going to throw in a 6^{th} algorithm called oracle.

This oracle is going to pretend like it knows this. So, the moment it knows the expected value of X_i, it will just pick the best one and keep hitting it. So, notice that it is this guy. That is the algorithm- oracle, which is the yellow one. So, that is the oracle. Clearly, it is doing the best. So, this is this graph although it does not say. So, this is time, this is the payoff per round.

So, each time you pull, you get the average. So, your average, this is the average payoff, it is payoff per round. So, this guy obviously is doing really well. So, oracle clearly does extremely well. Let us look at how explore 4 does. So, that is this last one, this is the explore 4. Now if you look at explore 4, it does poorly because initially what happened is- it picked the one that was the largest and then stuck with it and then afterwards you start to get a lot of 0s and did not do very well.

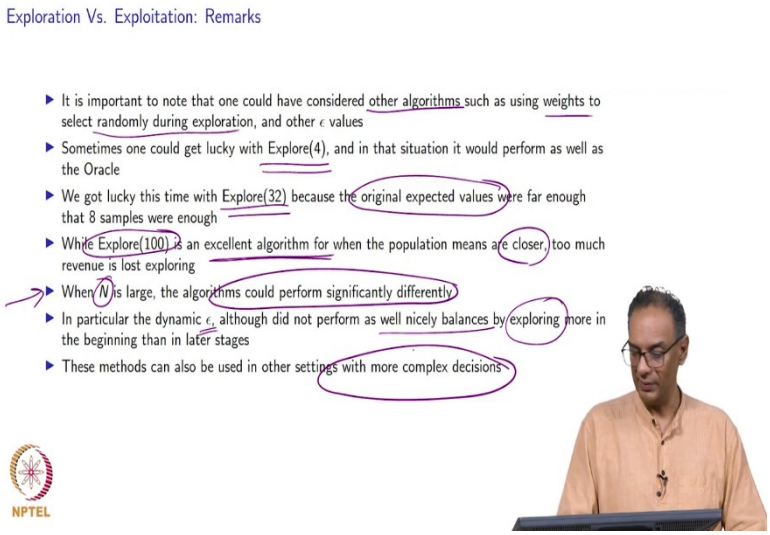
So, that is the problem and it kind of got stuck there because its mean is roughly. So, you can actually see the averages, it goes to these 4 averages, you notice these numbers are actually the average- the expected value of X_1, X_2, X_3, X_4. So, it kind of asymptotically goes there. Next one let us look at 32; 32 does reasonably well. Notice here, at the beginning it is a little bit slow and then it catches up and does phenomenally well.

So, in fact, this was explore 32, then after that explore 100 does not do very well because it takes a very long time, this is the explore 100, takes a very long time, keeps exploring for a very long time, takes a while for. So, actually it only explores till 100, but it is already so much below these other guys, it takes time for the average to catch up, that is basically what is happening, that stopped exploring at 100.

Now, look at the static and the dynamic. So, turns out very unfortunately although it is an excellent algorithm, this is the dynamic. The dynamic is not doing very well and one of the reasons is that, the algorithm for dynamic is not very aggressive, that is it is very slow it moves very slowly. So, another update would be better, that is one of the problem there are some other issues as well, which I will talk about in the next slide.

However, if you look at the static actually, it starts out really being promising. It starts out extremely promising and then eventually loses out to the 32 and that is because you know, it really exploits early enough that 32 needed some time before it could actually catch up. So, that is essentially what is happening here.

(Refer Slide Time: 19:25)



So, as the final set of remarks I want to say that we could have considered many other algorithms. And if you go look at the literature there are ton of others, they use weights for example, to randomly select, we selected with equally likely probability. So, 25 percent chance they will pick one, 25 percent of 2, 25 percent of 3 and 25 percent of 4. Well, I could have just used weights, which is my averages. So, I could have been more greedily selected one with which has a higher value and gone with, that will do well too.

Sometimes, one could get lucky with explore 4, that is the first one itself is a good one and then it will perform practically as good as the oracle because in the first attempt itself we would pick the right one. We got lucky with 32 because the original expected value were in our favor, but there is a good chance especially when you had a large number, this could be pretty bad. Explore 100 is an algorithm, which is generally considered excellent, where you know you explore for a while and then you exploit. But, what happens is when the population means are close, which is sometimes the case in casinos and so on, you are not going to have one machine giving a ton of money versus the other. Then you are wasting too much time just exploring.

Now, the other thing is when N is large, which is many practical situation, we are talking about not the casino case, but when N is large in a more generic setting, where you do not know what values are, the algorithms could perform very differently than what we saw. In

fact, the dynamic algorithm would perform extremely well, when N is large that is because it will kind of explore well in the beginning and stop exploring as we go towards the end.

Like I said, the dynamic epsilon did not do very well, but it is one that nicely balances exploring more in the beginning and less in the later stages. And we could use this for making more complex decisions under uncertainty, where even the data is not easily available. So, actually this is a very powerful methods and lot of people are researching this topic as we speak and there is a lot of work going on in this topic. So, it is a very nice hot problem and I would encourage you all to explore and exploit this option.

Thank you.