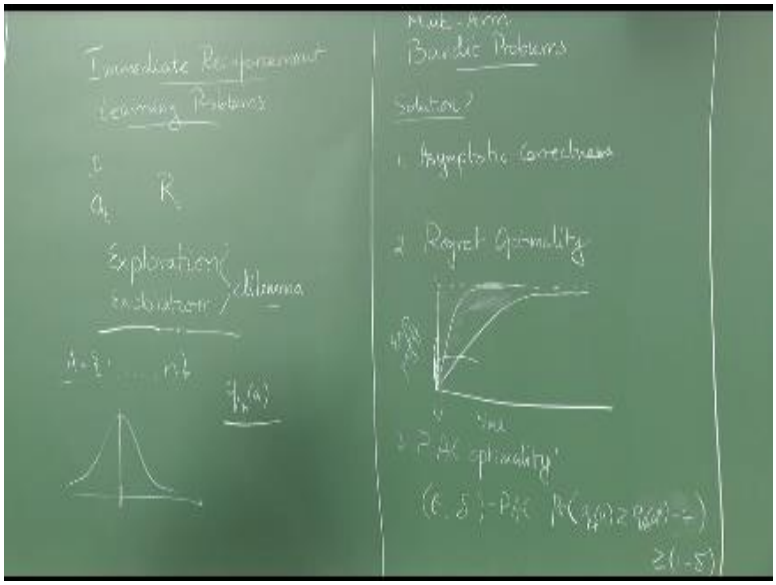
**NPTEL**
**NPTEL ONLINE COURSE**

# REINFORCEMENT LEARNING

## Bandit Optimalities

**Prof. Balaraman Ravindran**
**Department of Computer Science and Engineering**
**Indian Institute of Technology Madras**

(Refer Slide Time: 00:17)




So these kinds of problems this is a very, very simplified version of what our call with the statistics literature has bandit problems. I said you know why they are called bandit problems you know this, Deepak no, 01 you written enough of lab meetings to know these things anyway so people know what a one-armed bandit is come on man anyway so one-armed bandit is a slot machine, you know what is slot machines are you put a coin there then you pull a lever right, and then what it does it steals your money basically and all of that is mumbo jumbo you know of course all of you must realize that no casino is going to put up a machine that actually makes money for the customer, right.

So in the long run the slot machine will steal your money and then you just keep pulling that thing so that lever that you pull is called the arm and since it has one lever is called the one-armed bandit because it steals your money this is one arm bandit. So here this is very similar to the slot machine except that instead of having one arm it has n arms, right so each time you pull an arm you get some kind of a pay off right and we want to really complete the Bandit analogy.

So every time you need to pull an arm we have to pay one rupee, right sometimes you get back one rupee sometimes you get back nothing that way you know that is certainly stealing your money it is something like that, right. So but that is essentially why it is called a bandit problem sometimes it is also called the multi armed bandits, right so it is called multi on bandits because well it has multiple arms right and the dynamics is very similar to a slot machine, right.

Now we know I kept calling actions as arms, right so because the literature typically talks about arms on a bandit right, but it is really for us we do not have to worry about the Bandit connections so then it is essentially this actions, right okay any questions so far. There are many, many ways in which you can solve this multi-armed bandit problems right, but the cracks here is always that you'll have to be balancing the exploration versus exploitation, right. So I will talk about multiple solution concepts right, what do we mean when we say i want to solve a multi-armed bandit problem, right.

So one solution concept, right is asymptotic correctness so what do I mean by that I do not put any bounds on you or anything right here is this multi-armed bandit problem so give me a guarantee that eventually you will be selecting the arm which has the highest payoff, right T tends to infinity you will be selecting the arm that has the highest payoff so that is called asymptotic correctness, right.

So this is one way of solving it a lot of the older literature on bandit problems essentially concern themselves with asymptotic correctness and then of course they had some results on things like rates of convergence and so on so how quickly you reach the guarantee and so on so forth, but by enlarge the analysis was on asymptotic correctness they come up with very simple algorithms and then you show that the asymptotically they will converge to the right arm, right.

So this is a one kind of thing right, the second a popular solution concept is essentially known as regret optimality suppose I knew right, suppose I knew from time 0 which is the best arm right, suppose I knew from the beginning what is the best term to pull later and I keep pulling the arm over and over and over again right and I keep repeating this experiment multiple times what will be my expected payoff so this is time, expected payoff it is going to be some kind of a flat line, right.

So that is possibly the best expected payoff that I can achieve right because I know from the beginning I know what I see right arm, right but since I do not know this I have to do this exploration, right to figure out which is the right arm right so my payoff will look something like this right, over time it will and eventually reach that right as time becomes so may T foes to infinity I will eventually reach that point, right.

So now this reward that you see here right, this is what I could have got if I had known the right answer from the beginning, right so this is in some sense a loss that I incurred because of my learning process right, so this is sometimes colorfully referred to as a regret so I say like well as if I had known this from the beginning you know I would have done so much better so this is regret.

So another way of thinking about regret is that I am trying to maximize the total reward that I obtained okay, not just the asymptotically the payoff right, even during the learning I need to get as much pay off as possible right. So ideally I would want this slope to be pretty steamed right, if the slope is very steep so what will happen is this area will come down right, if the slope is very steep this area is going to come down.

But what is the trade-off typically you will have to give up is it will take a longer time to reach optimality usually there is a trade-off right, because you typically to do this right you will be giving up some amount of exploration right, so there are some corner cases where you might actually miss out on important exploration because you are trying to be very optimistic with respect to the surrogate thing right so I want it to be I want to have very little regret so what my what I am at end up doing is I met miss out on certain key exploration that I should have done.

So essentially what will happen is so in some cases I will never reach optimality also because I would have ignored certain important outcomes along the way so this is the trade-off that you have to worry about. But regret optimality is essentially looking at how steeply you learn at the outside of course that does not mean I can be really bad right I mean does it have a small regret right, but I did learn very fast I actually went up the y-axis right but then I am going to incorrect a constant very, very large constant regret okay.

So that is no so you have to balance it right, so because you keep accumulating even though this is reached here but you still keep accumulating regret right, so it is not it come to the optimal case in fact we know that no algorithm can guarantee that your regret will grow small, I mean essentially regret will fall right, faster than log T so it has to grow at least test log T right, suppose you have taken T times steps the regret you have accumulated till that point will be proportional to log T, right.

So and as T becomes larger the rate of growth will become smaller and smaller right but that is the best rate of growth that you can achieve all that you can fiddle play around with this some a times log T will be the rate right, so that a is what you can fiddle around with so those constants you can fiddle around with but log T itself is non-negotiable so there are results that show that log T is a lower bound so you cannot do better than log T in achieving regret, right.

So the essentially so if you think of this area above this curve and between this dotted line in this curve so that area will keep growing at some rate right, as T becomes larger and larger that area keeps growing at some rate so the rate at which it will grow will be at least log T, so that is the result that we have, so I am not going to show you that result but I will talk about a couple of other things okay, so is it clear so people understand what regret is right, good.

So third thing that I want to talk about is what is called PAC optimality right, or not is not I should not say we call it PAC optimality but it should be more PAC complexity right, so it is a little tricky thing. So PAC stands for probably approximately correct, okay sometimes in a very loose fashion we tend to use these as interchangeable things yeah, he probably right and he is approximately right, right so but they are not interchangeable in the very different context very,

very different things and he say somebody is probably right that means he is either right or wrong okay, so he is right with some probability and he is wrong with some probability right. When he say somebody is approximately right is almost surely not right, right but he is very close to being right this is essentially what approximately means, so it turns out that both of these concepts are applicable in the Bandit setting. So when I say somebody is approximately right in the Bandit setting what do I mean that I give you an arm right, finally you know what is the goal at the end of the day I am supposed to give you an arm back, right and this is supposed to have the highest expected payoff.

When they say I am approximately right that means that arm I am going to return to you will be very close in pay off to the best possible suppose arm, right. Suppose I return some A to you so Q*(a) will be very close to see Q*(a*) which is the best arm, right is it make sense so I will return some arm a to you at the end of my algorithm Q*(a) will be very, very close to Q*(a*) that what is Q* again the true expected payoff which I do not know about right, the algorithm des not know what Q* is but it will return an arm and the guarantee I give you is the Q* of that arm the unknown Q* of the top will be close to the Q* of the best arm, okay this is approximately correct okay.

So what is the probability connect path here, no it is either approximately correct or not, because we already only guaranteeing you approximately correct guarantees right, so with a very high probability it is approximately correct, right it is some small probability it might give you an arm that is more than some distance away from the best arm yeah that is what PAC is probably approximately correct.

Oh, I say okay, probably correct would mean yeah or two is an optimal or not optimal yeah probably approximately correct is well with some probability you are approximately correct some probability you are not, right so typically there are many ways in which people talk about this there is a 1 popular way of specifying this is called the epsilon delta PAC where epsilon refers to the approximately part and the Delta refers to the probability part right.

So with probability of 1-Δ okay, the solution I returned to you will be within ε of the best arm right, so this essentially means probability that probability that $q_*(a)$ that is arm written to you is greater than that is one way of writing it but that is not what the PAC framework guarantees, okay. So what are the difference between the first one and this one that is the first one was relative guarantee this one is absolute guarantee, so we could think of either way we can think of absolute guarantee or related as a relative guarantee but this is essentially what PAC optimality means right, so for a given epsilon delta if i can give that guarantee and then you say there is PAC optimal for this.

But what is the interesting part here right, if you allow you to draw an infinite number of samples from the arms right, I can always guarantee this they given give me whatever epsilon delta I want i can just keep drawing arms okay, then I some point I can say okay now I have satisfied this okay. The optimality part comes in when you want to minimize the sample complexity right, so given an epsilon and Delta what is the smallest number of times I have to select arms says that I can give you that epsilon delta PAC guarantee, okay does it make sense.

So that is essentially what we are looking at here, so this is a sample complexity question that this is a the correctness question right, this is a kind of rate of convergence question and this is a sample complexity equation so these are all slightly different notions of solutions when I say you are solving a bandit problem these are different notions of solutions and the kind of algorithms that you come up with for addressing each of these questions would be pretty different.

You do not know but I give you the guarantee this is the, if you know q*a* then this will be as close as that. Exactly so these are questions that we will look at as we go along I have not told you what the image is telling you what the solution concepts are right I am not even told you how you actually solve these problems right so when we look at those I will tell you how to go about doing this. In fact it will turn out that the algorithms themselves are very simple okay, but to analyze it to show that this kind of guarantee holds is where all that took place.

Funded by
Department of Higher Education
Ministry of Human Resource Development
Government of India

www.nptel.ac.in