NPTEL

NPTEL ONLINE COURSES

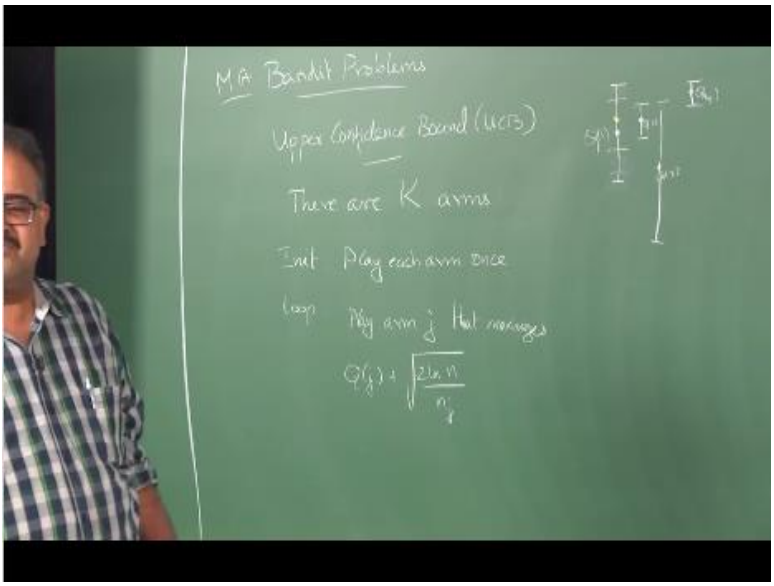
REINFORCEMENT LEARNING

UCB1 Explanation (UCB1)

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

So we are looking at right looking at multi-armed band it problems, we looked at a couple of ways of solving it so essentially you try to keep an estimate of the value function around, right the value function is the expected payoff that you get for pulling in all, right you keep an estimate of the value function around and then you be have some expletive fashion with respect to that value function that you have right.

(Refer Slide Time: 00:14)



So either you do epsilon greedy or you do soft max or something and, you get some kind of asymptotic guarantees correct, yes okay good, we also spoke about other forms of optimality so one of them we spoke about was regret optimality so regret optimality is what, reducing the total

I mean or increasing the total reward that you get over the process of learning right, so the initial loss that you get due, to the exploration you want to minimize.

So that I come as close as possible to the optimal case as quickly as possible right so that is essentially what regret is all about and, so we will start by looking at one algorithm which in some sense has become the popular most popular bandit algorithm around right now, because it's so easy to implement and also gives you not too bad the great bounds okay is called the upper confidence bond, the upper confidence bound algorithm or the UCB algorithm, okay.

So I will use slightly different notation today, right not not too different I will try to as much as possible I will try to translation the fly, okay to the notation that issued in the textbook right so even though I will give we will be giving you some papers to read, we can take care of linking the UCB paper and the median elimination paper and other things right, on what will be giving you some papers to read and the notation in the papers will be very different okay.

You probably have to spend like half an hour just trying to understand the notation, but when I am explaining it in class to the extent possible I will try to map it to the notation is used in the book rights, so that you have one uniform set of things that you can keep track of throughout right, the changes or the following right, I am going to assume that there are capital K arms right, earlier I was assuming there were n arms but the terms of the reason I want to change n is we are going to now talk about a notion of time right.

So now so they are going to be pull set every time instant and so on so forth and this should be familiar to electrical engineer, so if time is discrete you denote it by n if time is continuously noted by T right so I want to be able to denote discrete time, so I want to use n for discrete time okay and so we will save n for discrete time and I will not use it for the number of arms, so that is the reason I switch to K okay, assume there are K arms and so as before with each arm that is associated some arbitrary probability distributions.

I don’t know that whenever I pull an arm I will get a sample drawn from that probability distribution it could be where only it could be Gaussian right it could be poised on we do not

know what distribution it is but according to that distribution we will get a, payoff right and what else do we did we assume last time there is the expectation given for that distribution right now yeah we assume that the distribution is stationary.

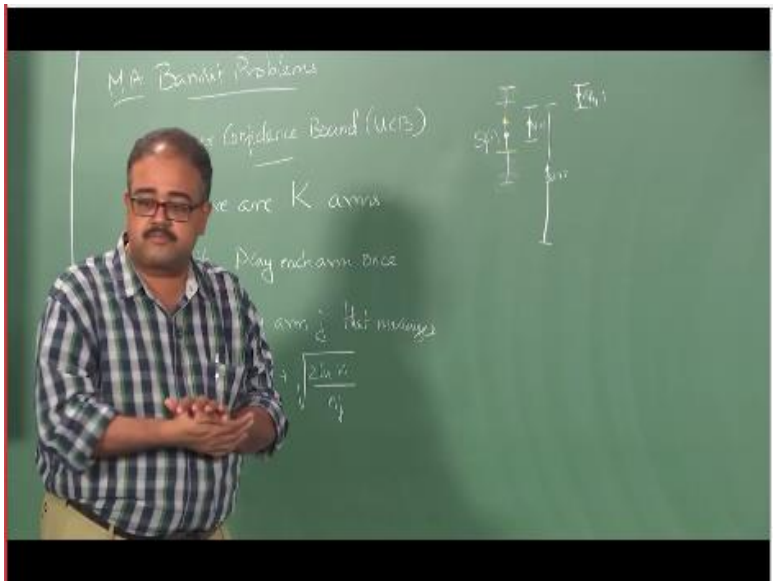
It is not going to change the time right and that there is an expectation associated with that distribution which will denote by Q* of A right, when you say Q* of A it is the expectation for pulling arum A ok. Let's clear here is a UCB algorithm so there is assume there are K arms so the initialization phase is play each arm at least once let play each arms once right, you have to pull every arm at least one time you cannot do better than that agree it, I never pull the arm I do not know anything about the arms.

So I need to pull each I'm at least one so that is the least so we start off with that and then we do the following in a loop, remember what QJ is estimated payoff as expected payoff right so this is average reward that we'll be maintaining right there is a value function right for arm J so QJ is the estimate that I have a time n right QJ estimate I have a time n for arm J.

So essentially what I am doing is I am taking whatever is the current estimate right, and I am adding this expression to that right, and then I am playing the arm that gives me the highest value for this total expression right, so the way to think about this is following I will come back to this it will show things more precisely later, but let us assume that this is my, Q of arm one right whatever is some scale this the x-axis, mean the x-axis let us assume is armed index and the y-axis is the expected payoff right.

So that is Q1 then I have Q2 say Q3 and Q4, let us say I have four arms right so what the idea behind upper confidence bound is to say that hey, I am not going to use just the estimated expectations so far right, so I've actually drawn many samples right I can use the samples that I have drawn so far to figure out, what is the confidence with which, I can say that this is the expectation right.

(Refer Slide Time: 07:17)



So you can come you can think of giving some kind of a bound around this you can say that ok, this is the expectation I have right but the true value of Q1 or Q*1 right the true value of Q*1 is going to lie within that band right, so intuitively you can see that the more number of times have sampled arm one the smaller is this going to be, right if I have taken a lot of samples then I'm more confident about the expectation that I am giving you tell you right we know that as the number of samples tends to infinity this QQ will converge to Q*.

So we know that that Q1 will converge to Q*1 if number of times have sampled one goes to infinity, so the more the number of samples I draw the narrower will be this band of uncertainty right, so likewise I will have a band like this for Q2 right I will have a band like this for Q3, another band like this for Q4, so what arm do am I suggesting that you take now, I am saying take arm one that is what essentially UCB tells you right.

So why do you think this kind of scenario meta happened I met a chosen arm for a lot of times because it seems to have a higher value, so my uncertainty and the value of arm 4 has come down all lot right, but arm one I'm not take n of times, arm 3 obviously I have not taken n of times, but I am really not inclined to take arm 3 anymore because given my current state of

uncertainty right, the probability that arm 3 will be higher than this end right, a minute it is very small that even though man certainty arm 3 is very high I am not inclined to take that arm, because the chance that it will truly be high a chance that it will truly be high is very small right.

While for arm 1 the chance that it might be better than arm 4 it fairly decent so this is interval I am Telling You is some kind of bound, that says it with very high probability like the true value of Q 1, like within this bond, that means that is a chance I did lie here which case it can be higher than Q4 right, so I will k take this if I take this a few more times and let us say after some time, my estimate moves up like this but my estimate moves up a little bit but my bounds also shrink right.

Now I'd my tonight it might not look attractive compared to Q4, so I will go back to taking Q4 right, so that is why you can see in this expression I have the total number, of samples I have drawn off arm J right, were NJ is the number of times I have sampled, I am j so the larger the NJ the smaller this expression is going to become, ok a larger the number of samples I draw the smaller this expression will become, therefore this interval will keep coming down.

What that yeah that is more like a normalizing term right it is there in all the expressions all the values right all the intervals, that I compute so it is not relatively it is not going to matter but you will need it to show some results later, so there is algorithm itself clear right, so they explain things here the n is essentially the number of times I have played arm so for any of NJ a number of times have played arm J okay.

So you have that pictorial description of what this algorithm is doing great, so what is nice about this algorithm it is very simple algorithm in saw that right is nothing about it all you need to do is apart from keeping track of QJ you have to keep track of NJ, as well which you are anyway doing if you are doing that incremental update right, if you remember last class we wrote the incremental update so the step size was any way related to NJ.

So the only additional overhead would be if you have been using constant alpha for your updates you will have to remember NJ in addition otherwise it is exactly the same thing like we did for

epsilon greedy, so instead of random lead deciding which action to take for exploration you do not do any exploration also the random color the random number generator also has gone now that this is a very deterministic algorithm right.

So what is so great about it yeah, QJ no because the other term might be a very small term Q might be a very large then matter, why does it matter, they like QJ like the expected payoff scale me something like ah that way okay, fine fine fine fine fine yeah, so all of this this this particular form of expression does assume that the rewards are bounded between 0 and 1 right, so yeah I will come to that in a minute.

So yeah you are right, so then you can rescale the rewards if you want to write as long as all the rewards are positive so you can rescale them to lie between 0 and 1 but if you have negative or reverse you'll have to think about always not do this okay.

IIT Madras Production

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

www.nptel.ac.in