NPTEL

NPTEL ONLINE COURSE

REINFORCEMENT LEARNING

PAC Explanation and Naive Algo Proff

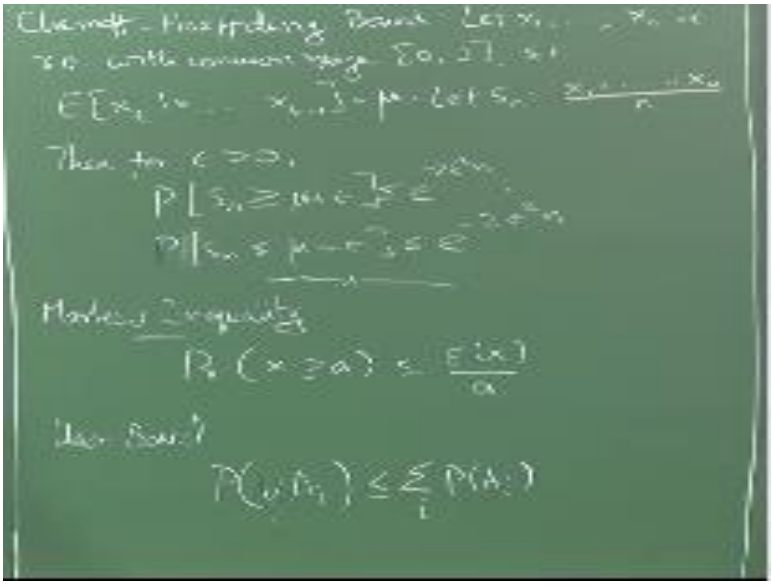
(PAC Bounds)

Prof. Balaraman Ravindran

Department of Computer Science and Engineering

Indian Institute of Technology Madras

(Refer Slide Time: 00:22)



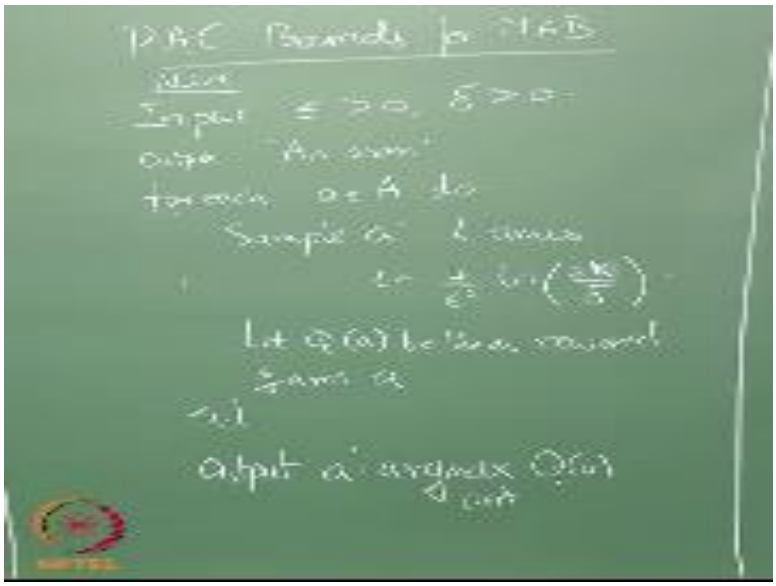
Turn of Huffing bomb, we saw that already so that is one result we looked right at the last class and you saw how we used it to manage to bond the regret of UCB right so we will be looking at yet another class of bandit algorithms today right so the pack bandits and we will be using the sheriff of ting bound again to bound a slightly different quantity but nevertheless of interest to

remember the Markov inequality elect people, people will it probability theory one form of writing it is this which is what we will need right assuming is positively expected.

So in the x support is in the positive range mean so I mean whatever makes appropriate sense for you to make sure that this is a probability right next thing you use is the Union bound this is this all of you should know the probability of the union of multiple events right this is less than or equal to the sum of the probabilities of the individual events that is this rather straightforward has a fancy name called Union bound right so whenever somebody says hope hence by Union bound we can write this result essentially all they are saying is that they are upper bounding the probability of the joint event by the sum of the probabilities of the individual events okay these are the results we will just keep using them as we will do some of the proofs today.

Okay so we talked about multiple solution concepts for bandit algorithms right so the first one was asymptotic correctness and the second when we talked about was regret optimality which we already looked at one example of a regret optimal algorithm UCB or UCB one more correctly even though popularly when people say UCB they refer to UCB one right and then what is the third one pack optimality right which was the probably approximately correct thing where you are trying to bound with some probability right that I mean you are close enough to the true solution. So that is essentially the guarantee that we want to give right.

(Refer Slide Time: 05:20)



So what we look at today are PAC so the pack bounds for multi-armed bandit problems right well start off with a very, very simple algorithm sort of with a very, very simple algorithm right this is just to get you started right it is not resorted to algorithm that you have to use and is just to get you started so I have an input which is I will give you an ϵ under δ and what I am asking for is a guarantee that at the end of this algorithm arm that a return to you will be ϵ close to the optimal armed with probability 1 minus δ.

With a very high probability if when this algorithm finishes running with a very high probability I will give you an arm which is ϵ close to the true the payoff of the arm that I give you will be ϵ close to the true so if you remember the back guarantee that we wrote right so most pack algorithm pack style algorithms essentially take this ϵ and δ as inputs right and they determine some parameters of the algorithms based on this ϵ and δ so I am writing the algorithm in the same style as it is in the paper that will give you to read so that it is easy for you to map it.

Unfortunately I will still do the online translation of notation so that you can compare across multiple algorithms okay, and this paper is way, way better written than the UCB paper so it is actually you can understand the paper when you read it okay so this is the on the median

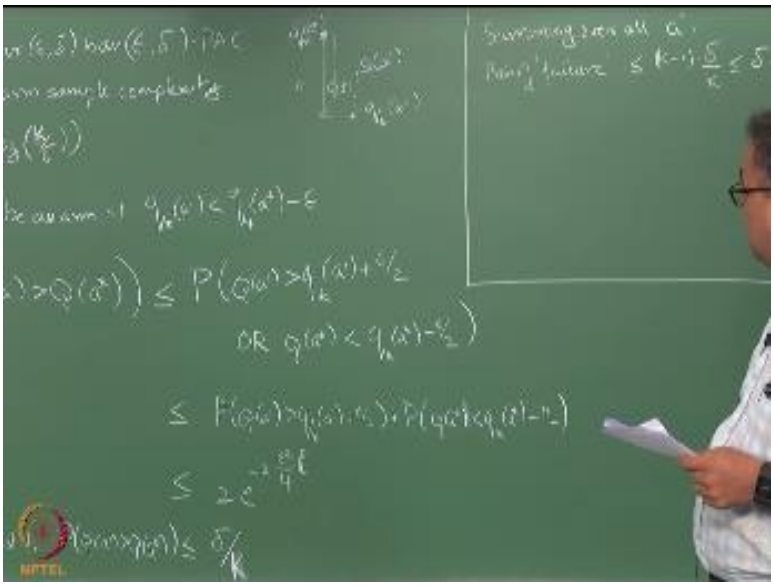
elimination algorithm by Eail Evanda Shymanor and Ishai Mansoor okay, its title action elimination for reinforcement learning okay so the nice thing about this I can tell you of the next things about the paper later.

But the really nice thing about the paper is it is well written so you can easily very simple algorithm right algorithm is very simple algorithm what is any the log what is l inside a log N is a number of arms right so what did I say I change it to K rights over here all right the last class last lecture I change it to K yeah right is a very simple algorithm all I am saying is take each arm sample at some number of times okay we can think about it right this is this is work see if I if I get a good enough estimate for each arm right if you get a good enough estimate of the reward of each arm then I can say that with a very high probability I will be ϵ optimal right.

So this of course there is always a small chance that I will go wrong right you see what I am saying it so the idea is very simple so I will take an arm keep pulling it many many times right until I have gotten a good estimate of the of the expectation right so remember the probability with which it is going to come up with one or whatever right so whatever is my expectation parameter I will just keep pulling them multiple times until I get a good estimate of the expectation I will do that for each of the arms that I have right.

And once I have a good enough estimate of the expectation all I have to do is now go and play the arm that has the highest expectation right that is essentially what I am doing here now the magic here is in that number which I have written down there right I mean given K ϵ and Δ that is actually a number okay remember I take ϵ and everything as inputs given all of this is actually it is actually a number right so that magic number how did I produce somewhere I all of this I wrote all of these things so that you know that you are going to use them to produce that magic number so it is very simple.

(Refer Slide Time: 11:13)



So let us make things a little formal so we call this algorithm new so then algorithm name which takes as input ϵ δ is an ϵ δ pack algorithm remember whatever what is ϵ δ pack we defined it in the earlier lecture right ϵ δ pack gives you that whatever guarantees you are talking about right ϵ δ pack algorithm with arm sample complexity this is the number of times you pull the arms given by or of n by ϵ squared into log N/δ right so this part is obvious and I pull each arm 1 by ϵ square times lon K/δ right so for n arms I have to do that n times so my overall sample complexity is this okay, so the interesting part now is to show that it is actually ϵ δ pack was it obviously here for today's lecture if I write n by mistake right please point it out to me.

So I will correct it to K right okay so how do we prove it well the sample complexity path is obvious we do not have to prove that we just have to prove the ϵ δ pack so for that we will assume that let A' P and arm say slit we do this Q* A' so A' prime is an arm which is not ϵ optimal write a norm that is ϵ optimal is ϵ close to the best term right so Q* A* is the reward corresponding to the best top we know that right so I am saying then I will be pick in arm A' says that it is at least ϵ away from Q* A* right.

So the Q* of A’ is at least ϵ away from Q* A* right so that means it is a bad round right so arms about more than I meanϵ close to the true oh my do not care right if I select them I am happy because my guarantee is satisfies I really have to worry about making a mistake which in this case is selecting a and that is more than ϵ away from the true optimal okay so basically we are looking at the event right so this is what we want we are looking at right so this is the case then the algorithm will output A’ instead of A* correct.

People onboard with me on this right so if Q A’>QA* then my algorithm will output at least is not output a star for me to select A’ as the best or at least this condition has to be satisfied okay let us put it that way right for my algorithm to output A’ as the best arm for sure this condition must be satisfied in fact A’ has to be higher than every other arm right only then I will output that but at least this condition has to be satisfied so essentially I am going to look at the probability of this happening all right and then we will try to bound that right.

So people can see why we need this condition right so if a perm has to be reported as the west arm it has to be at least better than A* great so that is the condition was already we are the being little loose here okay, this is not really the probability that yes A’ will be the best on okay but it is only looking at some sub B vent of that it is not all the events together but this will give me at least a bound on the probability so that is essentially what you are looking at this event right.

So what is happening here, let us say that is my that is my Q*A* okay that is why and then this gap is at least this gap is at least ϵ, okay so that is this is a setup that we have now so and I am making some estimate of Q* A’ prime which is QA’ right and I am making some estimate of Q*A* which is QA* so the condition I am asking is that Q’ A’ should be higher than I mean QA’ should be higher than QA* a star.

So essentially I am saying that I should have QA’ here QA* somewhere lower right so this is the case I am looking at right so this all automatically means that either QA’ has to be more than ϵ by two from Q* A’ it should be at least ϵ by more than ϵ /2 above our QA* should be ϵ/2 below Q*A* if they are exactly ϵ / 2 there will be equal I mean there won't be one above the other nights if one is ϵ by two below others ϵ by two above so you can think of cases where this is

actually here right so this could be here in which case Q* is the argument Q*A* has to be even below right it will be more than ϵ away forget about $\epsilon/2$ right so likewise for cases where you are overestimating QA*.

Then this will also have to be higher than that in which case again it will be more than ϵ away right so the interesting case is when it is in between the two right then it has to at least one of them has to be ϵ /2 away from the true estimate right either as a overestimate or as an underestimate right is it clear so why I wrote this now, right one of these two events has to be satisfied right I am just taking them as independent events and summing them up and that still be an upper bound on the total probability I have rights and making things simpler and simpler for mean now what happens here it kind of in the Sherman off space right.

So I can apply sure enough bound here so what would be the case here $e^{-2\epsilon^2}/4$ x l okay, yeah so let us do that so this will be less than or equal to $e^{-2\epsilon^2}/4$ x l what about the other term the same right it is a symmetrical bound so there is plus ϵ bit or minus ϵ me too it is going to be the same right so is it clear take so now plug in this L there so do you get so essentially what we need to do now is adjust this yell so that all those things get cancelled out which is basically what we are looking for right and you will be left with what should I be left with.

What should you be left with what do you think it should be left with no I want this probability Δ this is the probability of making a mistake right so it should be Δ I should I be left with Δ not quite this is the probability of me choosing one non optimal arm by mistake what if all my arms are non optimal except one optimal arm right everything is away from worst case so I should have δ / k or k -1 so we will take δ/ k it is easier than and so this whole thing should simplify to δ x k right so that is what I should be looking for so what should be my l right.

So this if we plug this in what will I get so therefore by ϵ square will go so I will get 2 lon $2k/\delta$ is not quite right and what do I get then the 2 x 2 K / Δ so I will get δ base I will get δ x k right is it fine I simplify this I get δ / k so now you know how we actually come up with this magic number L right, by actually they are having this bound and then figuring out what else should I plug in

here so that I get the necessary error for the probability so that just for one A' well as a two outside right to ϵ okay.

So this is clearer this is a naïve algorithm the analysis also is pretty simple pretty straightforward it's all of you are happy with this right Sophie if I give you an algorithm with a slightly different way of selecting actions you can all derive the complex the L that is required basically great it turns out that so we have produced this complexity here right it turns out that it is almost impossible to get rid of this $1 / \epsilon^2$ right and log $1 / \delta$ right, so when I try to come up with another pack algorithm right I want to remember what I said about pack analysis.

So I will give you an ϵ δ and now the interesting part is telling me how you can achieve this ϵ δ with as few samples as possible correct so in this case I am telling you that this is the sample complexity right and k figures in two places you have a k / ϵ^2 and log k / δ right it turns out that this $1 / \epsilon^2$ and log of $1 / \delta$ is very hard to get rid of right but then people now come up with and can you get rid of this k it should depend on K why at least once right.

I have to try every am at least one so I cannot get rid of the K also right so the only thing I can try and minimize hear this well I can if I and get rid of this K right so I can try to get rid of the log k dependency right and then have k / ϵ^2 log $1/ \delta$ roughly I can try to move in that direction and once I reach there I will probably be in the left with trying to get rid of this to hear right and try to make the 2 2 x 3 / 2 or 1.77 or something let me show you know how these things proceed right so you can try to optimize the constants but more or less this $1 / \epsilon^2$ and log $1 / \delta$ is something that will have to live with okay.

IIT Madras Production

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