NPTEL

NPTEL ONLINE COURSE

REINFORCEMENT LEARNING

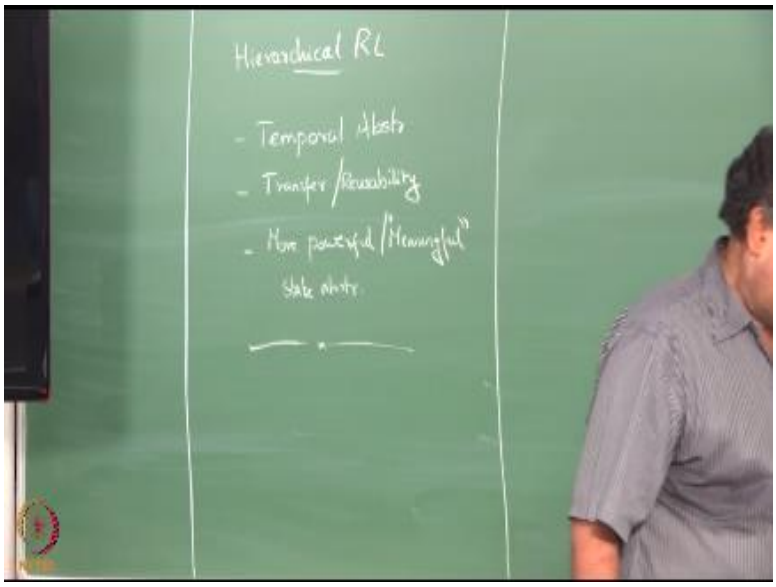
Hierarchical Reinforcement Learning

Prof. Balaraman Ravindran

Department of Computer Science and Engineering

Indian Institute of Technology Madras

(Refer Slide Time: 00:15)



And so I am going to talk about hierarchical reinforcement learning, so we talked about function approximation a lot right, so what are one of the reasons that we went in for function approximate, so what are the reasons you can try to tell me what are the reasons that you went above state space is too large generalization than, anything else okay. So essentially we are looking at trying to scale up the solution methods that we have seen so far to solving large problems, right.

So without doing function approximation we really cannot solve very large problems you will be limited to small state spaces right, where you can generate sufficient amount of experience and so on so forth we wanted to scale up, right. So there is another approach which we can take to scaling of problems right, there is another approach we can take to scaling your problems so there is of trying to find structure in the problem that you can exploit it in some way or the other, right.

So if you think about what we do with function approximation that is also trying to find structure in the state space right, so essentially we are looking at value functions that would be repeated right that is what you are looking at it is when you do generalization that is essentially what it is right, so multiple states we want them to have the same value that essentially means that you expect those values to be close together anyway right.

That is why you are clubbing them together as a single representation or you are representing a policy you expect the policy to have some structure that it will be repeated again and again in the state space and that is why you are doing some kind of parameterized representation for the policy. So you expect this kind of some structure in the in the policy space and some structure in the, and the value function representation that you are hoping to exploit may need to function approximation, right.

But we can be a little bit more clever about looking for a structure and look for more I mean some kind of higher order structure, so what do I mean by this instead of looking at small when actions repeating over and over again in the state space and I could think of problems that you solve over and over again when you are solving a much larger problem, right. So for example, think of playing some kind of a game right there might be small moves that you make repeatedly in the process of actually trying to solve the larger game right or the usual example that I give is that of navigation, right.

So there are so many problems that you solve again and again when we are trying to solve navigation problems. So for example, suppose I do not know most of my students are tired of hearing this example but suppose you want to go to central station from you, right how will you

go about doing that, how will you may go to Central Station give me directions what is that to which medium of transportation good point.

It is a good question right, so I can say that i want to minimize costs now take a local time give me directions do not say local train why do I know how the local train is I am in out of towner give me directions okay, and then get into a train okay, there you go so now you actually used a lot of sub-problems so you told me to go to the main gate right, so hey that is the problem I have to solve, right.

If somehow I had been dropped here from the sky that I would not know how to go to the main gate also right that is a problem that I have to solve so very naturally humans tend to break down long horizon solutions into sub-problems and this is something that we do naturally almost like is it is like second nature and we just do not even think about it is they say okay, solve this problem immediately try to break it down into smaller sub-problems and then try to solve the smaller sub-problems and then put the solutions together so it is.

So that is something a very, very natural way that we have of solving very large problems I mean there are physical limitations as to why humans have to break it down into smaller horizon problems the humans cannot pay attention for I mean if the horizon is too long we kind of lose perspective the whole variety of other reasons why humans break it down, but seems to be a very useful thing to do in terms of solving these kinds of problems as well, right.

So this is one aspect of it one aspect of it is what I call temporal abstraction, so essentially now he told me go to the main gate as a single action right, go to the main gate is not a single action right it has actually has a whole sequence of policies are going to happen you said go to the main gate and turn right then obviously they are not two actions of equal time right, so go to the main gate is going to take a much longer time then turn right and then go to Kasturiba station that is going to take much longer time then maybe go to the main gate I mean depending on the traffic levels and other things that right.

So what you did when you gave me these kinds of sub-problems to solve in order to solve the larger problem is abstract away some notion of time right, because it just solve this a problem then this is a problem then that's a problem put everything together you have a solution but each of these takes differing amount of time right, so this is what I mean by temporal abstractions when I am saying temporal abstraction it kind of means both that I am solving sub problems, right that could take a varying amount of time so it kind of means both those aspects of it, right.

But where is the hierarchy here, what do we mean by hierarchy this is one level right here is his big problem go to the Central Station and here are this many, many small problems that you have to solve you put them together and you get this big problem so that is a hierarchy. But hierarchy can go however deep you want, right. So let us say okay, how do I get to the main gate there you go so that you broke down into further simpler sub-problems right, so you told me to get down now I can ask somebody the question, okay what does he mean by get down right.

I cannot obviously not jump down, right so all right so walk to the left take the left down luckily we can do that now last month it was just walked downstairs and then walk out ofthe building and then take a left take another left and then you will get to the bus stop wait for the bus get into the bus, so now you can see that you can actually start breaking these things down into smaller and smaller sub-problems to the effect then I will finally ask you okay how do I walk to the left, further now in fact you started solving this problem of going to the central station when you are about one year old walking right, so walking is a learned behavior for humans walking is not a natural behavior right so it is a learned behavior.

So as babies you learn to walk around when you are about a year or so old right, so you actually saw started solving this problem a long time back you just did not know it so that is the beauty about hierarchy so you learn solutions right, that can actually we use our and over again for many other problems so if I am leaving at the granularity of catches the train and go to park station maybe I am becoming very, very particular but then if you start thinking about the other sub problems I have to solve along the way get out of this building that is a problem that you solve every well every day we have a class right at least every day we have a class, right.

And then you decide where you want to go after that but that is a problem that you solve right you can actually afford optimize this a problem right, what should I do when should I catch the lift the lift is in the first floor I am better off walking down the stairs versus the lift is in the second floor maybe I should press the button and wait and you can start optimizing to that level of solution for this kind of hierarchical I mean reusable problems right.

So if you want to figure out whether you need to go to the park station or not you are probably going to do this once a semester so you do not really not going to figure out what is the optimal point to wait in the station to get into the train right, I mean because you are solving it only once in a while right. So these are concepts that humans use all the time right, so when do you optimize for a solution when you are happy with a single solution so on so forth none of this actually unfortunately finds a place in the RL literature that to the great extent you know.

So if you want to move reinforcement learning to be a more and more realistic model of how humans make decision making how humans do decision making right, so you have to start factoring in all of these things reusability versus so it is more like a lifetime value of the policy that you are going to have, so if I am going to use their action only once then often I am going to use a sub-problem only once I really do not want optimizer so my the effort that is spent in optimizing the solution I can save and do something else, but if I am going to use it again and again and again right, then it is better to optimize that, right. So how much you think humans optimized solutions okay, how much do you think humans off to my solutions and who do not have OCD.

In general you think we optimize solutions I mean this is a very bad audience to ask this question in right, when usually many high achievers have some degree of OCD right, there is a selection bias if you are here so you have some degree of OCD okay, so seven what seven steps a head that is a different question right, I am not even asking about that so I am asking about how much you think we optimized one, okay so people who are who do not know what OCD is an obsessive-compulsive disorder so you feel like doing the same thing again and again until you get it right or keep doing it over or not said.

If you like doing something perfectly that is what do you think is OCD is it okay, a compulsive not perfect you just keep doing it again and again and again not necessarily doing it perfectly okay, anyway so it turns out that human is really the optimization that we do is not visible to the extent that you would expect from an artificial reinforcement learning agent mainly because we do not do that number of repetitions, right.

So there is this story about the Cuban cigar rollers you know so you know people roll cigars right so you have tobacco we have the tobacco leaf or paper or whatever you put the tobacco in it and then you roll it and then you see let me paste it you sear it and then the back to the cigar is ready, right so Cuba supposedly makes the best cigars in the world right, so and had no idea about that but they do make the best cigarettes in the world apparently so how long do you think it takes somebody to roll a cigar roughly an hour a minute half an hour close to, close to a minute maybe less than minute.

So obviously there is something here right, I am not telling the story otherwise so any other guesses one hour oh man, maybe one hour sure I am not I mean going for a Guinness Record for cigar make the world's largest cigar or something like that yeah so people who have been doing this and so those are the gate roots of all this fantastic worlds best industries right, there are humans behind its slaving away, so people who have been doing this for years and years and years like 20 years have been rolling cigars for 20 years or so can do it in less than seven seconds, seven seconds just count yourself seven seconds take a paper spread it out put tobacco on it but the right measure right I mean the quality control has to be there if you are making cigars that do not satisfy the Cuban quality then they throw it out, right so seven seconds so humans to optimize given sufficient number of trials, right.

So but then we do not usually get to do 20years of cigar rolling right so you do notget to see that much of optimization but you will see that in sports persons right, so the sportspeople do a lot of optimization right so Sachin Tendulkar does like 14 hours of nets a day for 35 years and then he basically he can play the same shot again and again and again with the exact same precision with or without a small deviation that is because it is optimized the stroke to that extent, right so yeah so we do significant amount of optimization.

So they are going back to the point so we do not try to optimize things we do not repeat of, right and quite often we solve problems once, quite often we solve problems once and so we do not try to optimize the solutions for that but then we try to optimize the sub component of these problems right, so that is another thing that gives hierarchy RL really gives us is this ability to you know I am going to say ability to transfer so you make this smaller problems you solve those smaller problems and then if I give you a completely different problem to solve right, there might be something from the smaller problems that you solved already that you can reuse, right.

And I do not know so this is these are something some of these are my thoughts about hierarchical RL right, so hierarchical reinforce planning also gives you meaningful modules to optimize you know trying to say that here is this very large problem I am going optimize how I am going to solve it does not make sense quite often there is this huge disconnect between you know the planning community AI there is a very large community that is what they call planning.

So the planning problem is I give you a start position and I give you a goal position right and you have to figure out a sequence of operators that they have to apply to get from the start position to the goal position right, and each operator has a set of preconditions so you can apply those operator only if the preconditions are satisfied, right. So for example I can put a phone down here only if the table is empty is a precondition so put something down requires the condition that the table should be empty at the point where I am trying to put it down.

Let us say I am not stacking up things but let us say I just want to put it down somewhere so it has to be empty so those are the things like that right, so i have preconditions and then I have post conditions things that hold after the operator has been employed right, so once I put it down what is going to happen the table will no longer be free, because I put something down on the table so that is a post condition.

Now because we some post conditions have become true some other operators might now become feasible, right. So I go and pick some of those operators or it could be some operators was earlier already was feasible that is still feasible so I look among all operators that are

feasible I will find another action and then I do this right, so quite often people in the planning community are happy to find a solution.

Because if you think about the complexity of the problem it is a really hard problem to solve because it can be a combinatorial explosion of possible paths that you can take right, so to figure out even one path that takes you from the start to go people are all happy they say okay, here I solve the problem so why do you want to do this again right, I want to stack above I want to clean the table I want to put all this away in a single file, right I have done that once okay, why do i want to repeat it again so it does not even make sense to them that you want to repeat things again, right.

So if you are talking about a very large problem maybe repeatability is an issue and why do you want to learn becomes an issue you happy solving things you see this kinds of planning operations, if I have an efficient way of searching through this operator space I could just do it is all the problem and get away with it. So now the question becomes why learn at all we cannot you just do planning and be happy with it, right.

So hierarchies gives you one answer for that so you want to learn the solutions for some of these sub operators because you want optimize them, so that next time i use this right I will be better at it. So for example put something on the table right, so put on the table so maybe the exact sequence I did to build a stack i might not repeat it but i am solving a different planning problem I still might be picking and putting objects, right I might still be picking and placing objects know if I figure out a way to do this optimally then there is good.

But normally in planning community they assumes its operators themselves or given to you I mean they do hierarchical planning but typically they assume that the operators are given and you can just execute them. But what hierarchical all will says no, no you have to learn what the operators are how to go about solving them as well. So I hope I have convinced you that hierarchies are good right, and then they give you a couple of things transfer slash it gives you temporal abstraction gives a transverse reusability.

So what do I mean by that, so what do I mean by state abstraction gives you in the first place remember state abstraction is essentially the ϕs, right so when we do function approximation other things we talked about using for the function ϕ and it allows you to aggregate states together right, or you know I mean how operated as an indicator variable of some source we are talking about the right. So let me go back to the problem of getting to the Central Station so you define a solution for let us say the problem of getting to the main gate, right.

When I want to solve the problem of getting to the main gate do I care about where I am going to after that, right the optimal solution for getting to the main gate really there is not depend on where I want to go after that, right and how did you explain the solution to me to get to the main gate, right you said get out of this building right getting out of this building the solution to that does it depend on where I want to go after that.

So it does not even matter if I am going to the main gate or to the valley civil maybe to some extent but assuming that there is only one way all of us obey the vehicle rules and go out only through the out gate it really does not matter whether I am going to the valley city gate or or going back to my hostel or whatever it does not matter, right so it is just the same solution so it turns out that when I start breaking things down into these kinds of sub problems I can start focusing on things that are only needed for solving the sub problems right, this gives me two things what are the two advantages it gives me, radio state phase means fewer parameters right so I have a simpler problem to solve right.

In the context of going to central station, right it might be a very large state space if we even stop and think about it right, here will be mind-boggling the state space is mind-boggling it got to worry about the state of the vehicles on the road well you walk to the Kasturiba train station you have to worry about the timetable a lot to worry about where the where the platform is there and I mean whether you can stand here without getting run over by the train I mean there are so many things you have to worry about right, the state space is so humongous.

So if you just think of putting together a vector that represents the entire state space okay, and then trying to plan how to exit the building in that state space right you are going to fail, this is so

many extraneous variables which are not needed and I cannot abstract all of them away and I need to know where the vehicles are going on the road when I am working on the river when I am walking on the Sardar Patel rod it I cannot ignore it, so I need to know that to stand on the platform when the train comes in so I cannot ignore that variable as well, right.

But I can ignore them when I am actually solving how to get out of this building right I can ignore a whole bunch of variables and I am trying to solve for to get out of this building so it gives me more meaningful representation, right. In the sense of what is needed at that point to solve a problem okay, anything else it gives me this kind of more powerful state of the abstraction this is what really enables the transfer, right.

If I had solved the problem with the vector space that included states that talk about the central station I cannot reuse the solution for exiting the building to go to the value Velacheery gate because I will say Central Station variables in here right, which is absolutely irrelevant to this problem in fact they do not even exist in this problem domain, right. So I am actually talking about going to the Velacheery gate, right so I cannot really take the solution as it is defined in that problem and use it here, but I can use it if the solution is defined only in terms of the state variables with respect to this room or this building right, so it allows me to have greater reusability it reduces the state space so that I can do efficient learning it also allows me to have greater reusability right. So that is essentially there are three things that hierarchy is really enable for you right, and so what we will do this class and the next please look at different hierarchical reinforcement learning frameworks that have been proposed okay over the years and so I am a huge believer in hierarchies, right.

So I think from 96 I have been working in hierarchical RL and so I am a huge believer in hierarchies and how people use hierarchies for problem solving and so many other aspects so many other 20 years okay, there are so many other know in this just a moment of feeling old soul so many issues that to do with the hierarchies that people really have not started you know exploring in depth in detail and in fact I can do a whole semester course on hierarchical RL.

There are so many issues that that come up and there are so many interesting things so probably next semester in that course we will spend some amount of time investigating hierarchical RL other than I mean apart from the other methods that we will be talking about and let us see okay, good question how do we spit sorts of problems maybe that is one criteria you should put in when you are splitting sub problems split it so that I can use it for other problems, who tells you that this is hard question.

So I do not think I will have time to get to how people actually figure out these hierarchies there have been some attempts in the literature not just in the reinforcement learning literature in the AI literature in general there have been some work on trying to figure out these kinds of hierarchies and this satisfying at best so many issues I mean exactly the question that you asked right there is no good answer for them how do we figure out how an option or how a sub problem is reusable right, there is no good answer for that.

Typically the thing is oh the domain expert will tell you right, or there is another hypothesis that says that there are points of inflection in the state space you know like finals for example the door to this building is a final right so it could come to any part in you could come to the NPTEL office you go to ICSR you could go to the alumni office all of these people, right enter through the main door right so the door is like a final right so once I understand that there are these kinds of final states then the sub problem could be how do I get to that final state how do I get to the door that could be a solution which could potentially be reusable across multiple domains, right. But now I told you that this is a final state with by cause of my knowledge about this building right, how will you normally discover this you know in a hypothetical situation.

I mean geography geometric navigation problems is always easy and if I am having non geometric problems then how would you define these kinds of bottlenecks it becomes final stage or bottlenecks a call becomes a little tricky so they have been atoms from try to do this from data and so on so forth there is nothing that says that okay, here is a very robust method just throw it at your algorithm for throw out for your problem for example I cannot go and define these kinds of hierarchies in Atari even though we are trying but there is no nothing that will successfully

find these kinds of sub problems in Atari games for you to solve for example because it is really complex.

And even though it has some kind of geometric navigation side to it but there is significant amount of dynamics there that makes the things very muddy right, so just comparing similarity between problems sure how many does increase so you are trying to do some kind of a navigational problem from the outside the building to like multiple points inside the building solve all these problems separately and then try to compare similarities between trajectories and then separate oh yeah, so there are few issues here right if I am already solved multiple problems so what am i doing in finding these hierarchies after that I mean the assumption should be that okay, I have like a distribution of problems defined on the same state space I sample a few problems from that and then.

I will solve those problems and then there is still a lot more problems that I have not seen you know so I have not gone to every nook and cranny of the ICSR building so but then I have gone to 15 different places in the ICSR building from that I will figure out what the options are, right what the bottlenecks are so it is fine even if your situation support such a setting then it is great I mean that is typically in fact that is what would happen also and if you as a human you are trying to optimize navigating into ICSR.

So you really need to come to multiple places in this building before you think of optimizing your trajectory through this thing or if you will come to a single place in the building it will always coming only to the NPTEL classroom you are not going to find sub problems you are going to optimize the problem of coming to the NPTEL classroom efficiently optimally that is all you are going to solve right. But if you keep coming here say you have to go to the Alumni Office repeatedly and then you have to come to NPTEL conference room or the studio for classes and then you start optimize okay.

Let me see okay I will figure out how to get to the staircase efficiently and then from there I will solve these two problems, right so only then you start doing this it is little tricky as to how many sub problems how many sector is allowed to generate, so can you do this where is Jane just

looking at one trajectory or solving one problem instead of solving multiple problems in the domain, see the whole planning takes it under different context right here that I am assuming I know the operators I know the preconditions I know the post conditions that is not typically what the RL folks assume right.

So they know the in fact our planning assumes you know what the goal is not only do not even assume what the goal is right, so people do this kind of sub problem discovery in planning also right, that is done so not done means not in sense of it is solved and people attempt to solve the problem and they are still there are solutions and there are not too many satisfactory solutions out there, in fact whatever you said I mean there are a couple of papers out there that actually look at that looking at trajectories and trying to figure out commonalities in the trajectories and then based on that trying to figure out which are the which are the final states right.

So to do this I am not scalable right, and it is not clear if what is a tradeoff between the number of trajectories that you use and the quality of the solutions you get and it of course all interesting problem I should probably do this lecture as the first lecture in the RL course so that people can do RL projects on hierarchical level a lot of interesting questions to explore in this place, anyway,

IIT Madras Production

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

www.nptel.ac.in