NPTEL

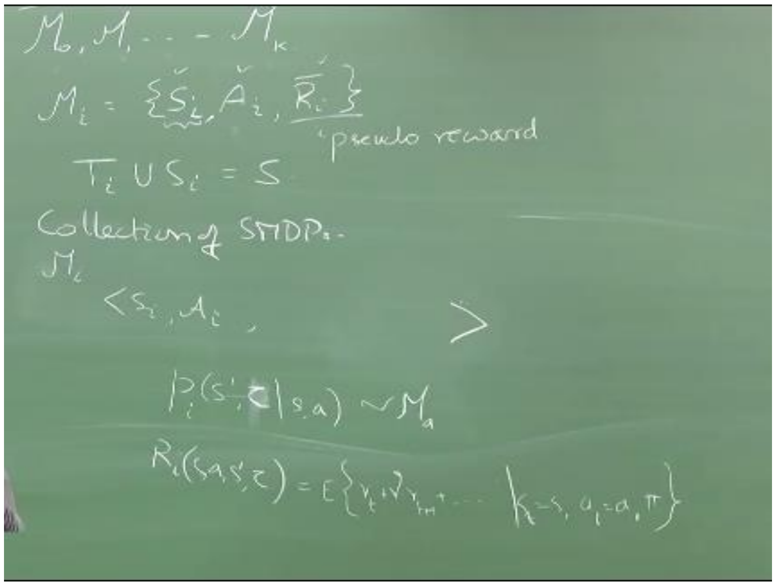
NPTEL ONLINE COURSE

REINFORCEMENT LEARNING

MAXQ Value Function Decomposition

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

(Refer Slide Time: 00:19)



$M_0, M_1, \ldots - M_k$

$M_i = \{\check{S}_i, \check{A}_i, \underline{\bar{R}_i}\}$ pseudo reward

$T_i \cup S_i = S$

Collection of SMDPs

M_i

$< S_i, A_i, \quad >$

$P_i(s', \tau | s, a) \sim M_a$

$R_i(s, a, s', \tau) = E\{r_t + \gamma r_{t+1} + \ldots | s_t = s, a_t = a, \pi\}$

So you can think of there being a Ti, okay so I will specified the states where it will be active in so whatever is there in the state space which is not there in Sa it is basically Ti and T or the states it will terminated, okay in fact they have made a specific dichotomy like this so they essentially means that if you are in a terminal state and call an action call a subtask it will not move basically it will just return right.

So because it is not in the only inactive states do I specify actions if I mean one of the terminal States it would not have one specific actions right on the contrary in options right the same state can be in I in the beta right options the same state can be an eye in beta so I can start this action option in a state that might be also some probability of ending the option in the state right but if I start the option I will take at least one action.

So that is the thing so if I start from here I will not end here I can take an action go somewhere else right and then but if I come to the state while executing the option I might also terminate here does that make sense right, I have some state if I start an option here then I will certainly move out of the state okay if I start an optional certainly mode of the state but while executing the option if I land here I will stop okay.

So there is a little conflicted a shin definition that allows you to do that look, okay any questions what so now let us take a look at value functions that how do value functions work here, so you recall what I say raise this is essentially collection of SMDP's it is a collection of SMDP so let me take one of those right let me take MI right so we have SI AI RI^ right so for the MI SMDP if the SMDP corresponding to MI so I have the states right so I have the actions right do I have the rewards RI6 + RA RR, right the original reward the two both together give me the reward function for the SMDP.

So RI ^ + RI bar + R gives me the rewards for the SMDP so the SMDP corresponding to MI so I have SI, AI it what about the p, p is a little tricky so now p I have to define this joint distribution right, so if I only take the okay I apologize yeah so we will come to that also what is that where are you getting that a powerful that is the task specification right when I specify this M is those are the things I have to specify I have to specify SI, AI, and RI bar.

It is just like options you have to specify even option policy in ham's you have to specify the structure of the machine so in maths Q you not only specify this graph I mean this graph is implicitly specified there right the graph is drawn from this is right, so this basically have to specify SAA RI bar for every task okay, plus 5 wrong what has to be part of our A bar right, only the minus 1 then you get to the answer is part of the both X.

Maybe only a drop of the passenger imitator the word the task is for you to drop off the passenger at the gold ring so for dropping off the passenger at the goal you might get a code MDP reward or doing a wrong pick up you know we try to pick up a passenger when there is no passenger or in trying to pick up a passenger when there is holiday passenger in the car so these are does not matter which subtask you are doing this in these are all wrong.

As far as the co-MVP so it is regardless of this stuff sub-task you are doing this in it will be wrong so that is what so picking up at the wrong location dropping at the wrong location all of this or bad and like you are saying if you want to make sure you are getting a smooth right you can also penalize it for running into obstacles that will be in the code MDP, what could possibly mean that is what I am saying if your goal is to have a smooth array.

Otherwise what will happen if you run into obstacle you will get a minus one anyway so you will try to avoid running into obstacles because every time step you get a minus one right whether you move or not if you run an obstacle you do not move so you get a minus one so if you want to really avoid going anywhere near an obstacle then you can give it a high- reward so that you well basically you cannot reach any of the goals.

Because you have to go very close to an obstacle to reach the goal, so we had to think about those thing t hat is fine okay so what about the transitions and the rewards I have erase the rewards also because those incorrect what I wrote was not correct, if A that a pick happens to be primitive action then all of it is fine so essentially it is a core MDP transition right and then the core MDP reward function for taking direction.

But if I happen to pick another sub task let a non primitive action as my not a is the bigger thing so the A that I pick at this time instance a non primitive action so what will be my transition so now I will have to define a probability over right so n is the number of time steps is going to take s prime is the final state you are going to land up ins comma a right so I will have to do this for task I, what is that n is the number of steps it will take to complete yeah.

If you want to reduce talk the same it is random variable right so if it is like down the yeah if I want I will just stop this is exactly like top in fact now it is stopped so how will you determine this will be determined by how many time steps the subtasks corresponding to a will take to complete okay, so this is determined by MA where MA is the eighth machine here a th machine here there is some task, correct.

People on board okay now what about RI, you know all about RL in the summation of the rewards that you are going to get well the actions corresponding to a are being executed there will be the MA will run right so the policy in MA will now run when I call A I am going to execute a policy so for every step along the way I am going to get some reward right I am just going to accumulate that.

That is essentially what my RI will be there will be discounting factor depending on how we are doing it if you want to use discount or what is the use of discontentment so the tricky thing is what should RI be, think of the MDP definitions what is RI and said what expectation so what I just described to you as a sample you have to convert that in expectation you have to write out the proper transition probabilities and other things.

In order for you to define that right. So it turns out conditioning on s Prime and τ is a little tricky I 20 right this reward functions so I can write it without conditioning on experimental so that is expected reward for taking action A in state s right so this is the expectation of this summation given that I started in state s and I did a as my first action and I followed my policy point so now it is little tricky.

So this expected reward that I plug in to solving MI already has the policy Π there because I need to know what is the policy I am following below MI for me to compute this reward right because I have call A here but then a might call some other action a prime a prime might call some other action a double prime rate so for example route I might have called pick up my call navigate and navigate my calls south.

So that might be the one action that gets executed maybe it finishes navigate then it will come back to pick up and then it will come back to route right so I basically need to know what was the entire policy that was executed for me to compute this expectation right so I will condition it on the whole Π this is reward for I sorry these are the rewards for the core MVP you are right now I define it as it works for the core MDP.

In fact I can the RI ^ RI bar so that is one tricky part because RI bar will be defined on the states of this MDP right this SMDP well these are these small transitions that I see here might be transitions in the smaller in the SMDP below me right so they might not correspond to the states here I mean they might correspond to different set of states correct so they might correspond different set of states is not entirely clear to me that.

This thing can I, I can assign a pseudo reward to every step along the way so what I should possibly do is add a separate term for our pseudo reward here and so that I construct my entire reward function, so this will be the core MDP rewards because these are coming from the subtasks below me right so during the execution of subtasks below me then I could potentially add a pseudo reward right.

So now I am going to remove the pseudo reward because it adds a lot of complications so most of the development that I will show you now is without the pseudo reward and how do you do the corresponding things along with the pseudo reward I am going to ask you to read the paper essentially allowed to maintain a second value function throughout so one value function which actually uses the only the core MDP values the core MDP rewards.

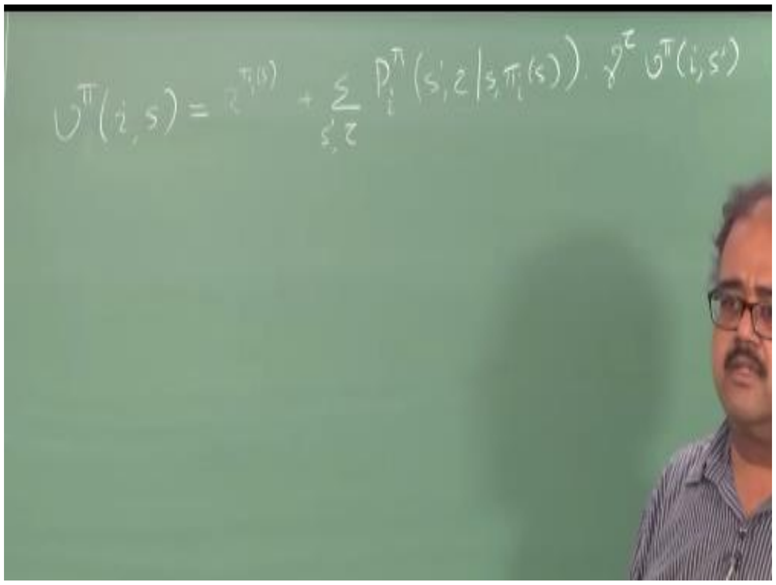
A another value function which is used only internally to the sub task which uses a pseudo rewards as well, sorry function chooses which one of that navigate refuse, no that is a core MDP reward sooner what tells you well if you are in pick out it right so to successfully complete the pickup task you use both the core MDP in the pseudo reward right but finally when you want to solve the problem.

But you do not use the pseudo reward of a navigate while you are trying to solve the pickup task that is the difference when you are trying to solve route so what will be the pseudo reward of route, I think because the route is trying to solve the original problem so the pseudo the reward for route should be the core reward function right, so the reward for route is the core reward function and it will never use the pickup pseudo reward while solving this right.

Likewise pickup will never use navigate pseudo reward navigate will never use pickup pseudo reward right the pseudo reward is limited only to that sub task for which you are defined the pseudo reward okay so just it ignore the pseudo reward from the time we will just come back to that later, right. So what I am going to do now that we have some kind of definition for the SMDP okay.

Let us start writing down the value functions okay then we can worry about how will we learn those value functions right.

(Refer Slide Time: 15:59)



$$V^{\pi}(i,s) = R^{\pi_i(s)} + \sum_{s',\tau} P_i^{\pi}(s',\tau \mid s,\pi_i(s))\, \gamma^{\tau} V^{\pi}(i,s')$$

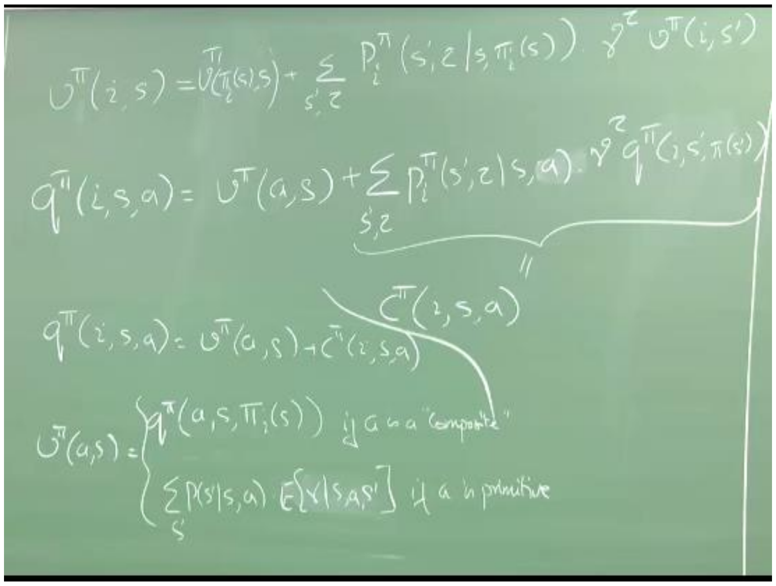
So that is essentially what we are looking at so I am going to use the following notation so I am going to say vΠ I, S is the value function of state s in task I right I mean I could have done I hear right across there is a value function corresponding to task I mean I would have said that is a more natural notation but I am NOT putting I there I am putting I here because that is why we defined it.

So it is easier for you to follow when you look at the paper later, right. So what do you think this will be you want to do bellman style writing a break how will I do this so I will say first say pick the cost corresponding to the first action you are doing right which will be let us say Π right so that will be the first action I am doing wait and then right so that is like the bellman equation. So what is missing here, the summation over Π.

Why is it missing already chosen apology Π from below I heard something some other murmur also let us see if someone gave a right answer why is the summation of Π missing, Oh come on guys wake up we did a version of this already because I am assuming Π is deterministic that is why I wrote Π as, heavy lunch today, noticing too many people with half glazed eyes summation over A that is missing yeah fine one of a summation over Π is A yeah.

How do you say summation of a is missing we are trying to figure out what x is it just say summation of EA's missing okay that is because you are assuming itself the domestic policy wait what is next why would it go to the front assembly but I am just looking at the value of state s and particular solution I see the particular stage cause well there is a medium is right, so this is too much notation in max.

(Refer Slide Time: 20:00)



$$V^{\pi}(i,s) = V^{\pi}(\pi_i(s),s) + \sum_{s',z} P_i^{\pi}(s',z \mid s,\pi_i(s))\,\gamma^{z}\,V^{\pi}(i,s')$$

$$q^{\pi}(i,s,a) = V^{\pi}(a,s) + \underbrace{\sum_{s',z} P_i^{\pi}(s',z \mid s,a)\,\gamma^{z}\,q^{\pi}(i,s',\pi(s'))}_{=\,C^{\pi}(i,s,a)}$$

$$q^{\pi}(i,s,a) = V^{\pi}(a,s) + C^{\pi}(i,s,a)$$

$$V^{\pi}(a,s) = \begin{cases} q^{\pi}(a,s,\pi_i(s)) & \text{if } a \text{ is a "composite"} \\ \sum_{s'} P(s' \mid s,a)\,E[r \mid s,a,s'] & \text{if } a \text{ is primitive} \end{cases}$$

Because one of the things which I should have until used earlier so we have value functions corresponding to the actual policy Π today and that is going to look at the entire execution of the policy right so I am going to call L define something called the projected value function which looks at the value function only for till the completion of task I, right so I am only worried about task I finishing I am not worried about what happens after task I.

So this is the projected value function so what we are writing here is the bellman equation for the projected value function okay infact in a later paper somebody pointed out that there is a problems with the max Q architecture the way the max Q value function decomposition the way Dietrich defined it precisely because he operates with the projected value functions because he operates with the projected value functions.

Because he operates with the projected value function there is no chance of you learning a hierarchically optimal policy so they came up with the different kinds of defiling function decomposition that allows you to choose to learn the recursive optimal policy or a hierarchical optimal policy but you need to have a different value function decomposition that did not just operate with the projected value function.

But it had something that came after the projected value function also I will explain what in a minute but what we are talking about is a projected value function and you are right that I am only worried about till the task I completes I do not care about what happens after I sorry right now given that caveat is it all clear what is happening here right but what is this guy now sooner the work think about it yeah Oh which task I itself does know if π I a subtask π I of s right I had taken some action here right.

So the entire reward I accumulate over the execution of that action should go in here and then what is that quantity they do not look here look at the notation we have introduced already here right, so it is a value function in that sub projected value function in that subtask so what is the predicted value function it is a total reward I accumulate till the end of that option the end of that sub task right so now what happens when I call π I (s) it will run till that completes and then it will come back.

So what is the projected value function in π I (s) it is essentially the road I accumulated till the end of π I (s) right so similarly I can write in sub task I or AC value function s, a this is even clearer to write this is essentially I do not get confused now I have flipped a and s this is why I said I do not like this notation if I had written A here would have been very clear right so here you have SA I here it is AS the decently means it is a value of State S in subtask A just like this is value of status in subtask I this is value of status in subtask K right and then what else you have essentially have the same thing.

You want to write that last term in terms of Q what will I do this Q π so I have no bellman equation for Vπ and a bellman equation for Qπ, so sometimes not sometimes so what we do this we define I am really see the off-color tray so we call this thing the completion function C very confusing here so Cπ (isa) is the cost or pay off or reward that they are going to accumulate in sub task I after you complete action a starting from status right.

So look at the way this work right I have completed action a once a complete action a will be in some s prime so I am looking at how much reward I am going to accumulate from s prime till the end of sub task I, so the completion function exactly says that so I started a in state s after I finish that how much more reward will I get till the completion of sub task I, so it is called the completion function so I can write my Q function as so my Q is essentially v + C right.

So C is equal to this so V π A, S = qπ right, so q π is this v π is okay, you need okay fine you did not sleep and fall over and so v π is q π of as π s if it is a composite action basically like one of the subtasks right under this primitive action then it is just the expected value of the one just because I made this into a s right so this is essentially in what is the value of state s if under primitive action a which is essentially this I marginalizing over s prime right so this will be what is a value of state s in subtask a right.

Which is essentially the Q function so now you can see kind of it goes recursively right so I want to compute the Q so I need the sea right but for computing the V if it is a composite action I go over go down and ask for the q of the lower action right if it is a primitive action I return this so it is some kind of a hierarchical thing so if I want to so I can make you really sweat by saying okay, here is the value of state X in 0 x cos x I of u see somebody is awake not me right so how will you write V π o (s).

First let me define some notation for you to write this out so I am going to say π is composed of right so any policy pie that I define is a hierarchical policy and it composed of π 0 π 1 π 2 π k which are essentially the policies I will be following in each one of the case of tasks that I have m0 to MK so π 0 is the policy I will follow in m0 π 1 is the policy I will follow in m1 and so on so forth so π is written out like this right now I want you to write out the value function for state s in subtasks 0 right.

So how will it look like so what will happen is π naught will call some other action below it first action it will call this something below it right so let us say it the first action called by π naught is a one correct so what I will essentially do is I will get I am going to write is slightly outlaw so that is easier for me to keep adding terms to the end so what should I have v π 0 would be q π 0 s, a1 because I say the a1 will be the first action I take right.

So π a (z) will by 0(s) will be a 1 let us say right so q π 0, s, a 1 right and then what do I do I add the right so I will essentially for that I will have to for computing that Q function what should I do 0, s, a1, so that will be v π of a 1, s + c π (0, s, a1) + (a, s). Now let us say that I go into a1 okay and let us say for sake of ease of writing the first action I take in a1 is a2, so now what will I do it is a composite action it is a v π a 1, s will be q π a1, s.

Which is a to write so q π a, s, a2 write and then I will go back to q π a, s, a2, I mean a one sorry a 1, s, a2 which will be v π a 2, s + C π + a1, s, a2, all the places we π right these are all conditioned on the whole policy the time executing right so π naught of s is a 1 (s) is a to right π1 (s) a2 right π2(s) is a3 that is what we will say within the subclass calculation then what will come here unless essentially π (a2, s), so like that I will keep going until I hit a primitive action right.

So if I want to compute V π 0 of s so essentially I will make I look at what the action I pick is right I will make a query a look at the completion function for the reaction then I will go down look at the completion function of the choice made in the subtask then if I am if I reach the end and then I will just use the Vπ reached answer if I reach the end I will just use this expression right otherwise I will use the q and I keep going and so this is essentially what the value function decomposes what is the nice thing about it this.

If I take a2 to write for some other subtask also right here I am looking at v π 0, s comma is that suppose I look at v π 0 , s prime right, so that also I would have learnt this c π mean this is the same c π that I will be using regardless of how I come to there and I start from sand go to some state if I might start from s prime so I have to learn different this thing but the point is if you go further down the hierarchy it makes more sense not in the rule.

So further down the hierarchy there could be multiple ways in which you could have started the subtask right I have learnt the same completion function I can just reuse the completion function everywhere likewise I can reuse the yeah so if I am cashing the Q function I can reuse the Q function also everywhere right so essentially what this allows me to do is break down the value function right into chunks of rewards right.

Which I am likely to see again and again d chunk corresponds to executing a subtask so that is essentially what value function decomposition does and yeah so we are already getting into confusion say this really all is a pretty complex set up already without bringing into the pseudo works there are no pseudo rewards all right already the completion function and this becomes a pain so if you look at the root task right everything is good in the root task what did we have so we had the completion task a completion cost or complete the completion function after you finish a1 starting from s for completing the root task.

What is a reward so that is all fine right but then if you go to a sub level so what it what is the completion function here say only for completing a one I do not care what happens after a one right but in reality if I want to learn something hierarchically optimal and I need to know what is going to happen after a1 if you think about it right so it is not enough for me to get out of the door I need to know what happens after I get out of the door so that I can decide which note to get out of it is I need to know what happens after the completion of a1also.

So what was proposed as a modification was a second completion function which was for the whole problem right but then it becomes tricky how do you maintain this across the entire structure right how do you maintain this consistently it is like passing on information from some sub task very far down the line all the way up to the first sub task right suppose I am executing subtask one and my task ends after subtask hundred then whatever happens till 100 all that information has to be what back to sub task one for me to define.

The second completion function.Right so how do you do a fish in bookkeeping so that is essentially the problem that they have to solve for learning hierarchical optimal policies with the max Q so right now you can define these kinds of completion function and now you can define a Q learning SMD PQ learning variation right were at every sub task you update the Q value for the subtask as well as the completion value right.

So you do not update the q value of the subtask you just keep calling down till you reach a primitive action so you basically have this right as we saw here right so I can keep calling down until I reach a primitive action so I am not at reach the primitive action here but if I reach a primitive action then basically all I have is this right and this I do not know because this assumes that I have knowledge of p and the knowledge of this expectation so what should I do for this maintain some kind of a running average or something right.

So essentially that is what you will be doing so if you hit the primitive action you return that running average so otherwise you keep querying so you really need to store is that running average which is essentially the Q function the Q value right if you think about it the running average is essentially the Q value for taking action yen status and what is this the Q value for taking action in status at this expression I have written here the original q value no SMD PQ nothing the original core mdp Q value for taking action a in status.

So it is QSA is a trend yes or no yeah q value have the future expectations right so it is just the expected reward for taking action aid status right so that is all this is so you basically compute the expected reward for taking action named status and that is one thing that you have to remember and apart from that what else you have to remember for every action you have to remember the completion cost. So what you now you basically define an SMD PQ learning variation where you update the completion cost at every state right.

Whenever you take an action and whenever you come to a primitive action you update the expected reward whenever you take a composite action you update the completion function never take a primitive action you update the so essentially that is roughly what it will reduce to so there is something called max q0 learning so max q0 learning is for the cases where the pseudo reward is0 right uniformly 0 for all sub tasks in all states the pseudo what is 0 then you have max q0 learning which is essentially what I described to you update the completion function and the expected reward and that is basically it.

Then you have max QQ learning where it is defined for arbitrary pseudo reward functions where you maintain two completion functions ne for solving the subtask 11 completion function which you use for picking actions in the sub task and one completion function which you use for answering queries from above right, so when pickup asks navigate give me your completion function navigate will send the function that was learnt only with the core rewards.

But when navigate wants to know the Q values to pick an action it will use the completion function blunt with the pseudo watts okay make sense, so you have to but then the book keeping becomes more or more painful so you have to make sure you are updating the right Q functions right completion functions throughout so that is essentially what max QQ learning is all about so yeah we are done with Max Q today.

IIT Madras Production

Funded by

Department of Higher Education

Ministry of Human Resource Development

Government of India

www.nptel.ac.in