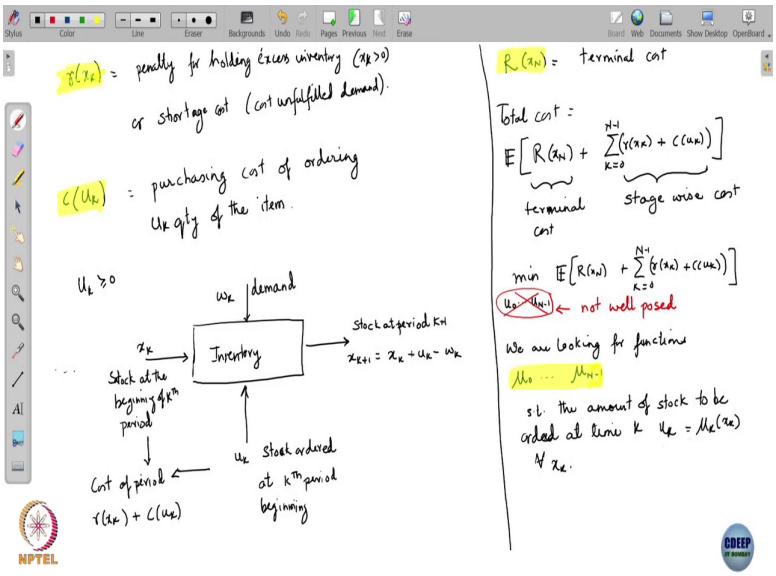
Optimization from Fundamentals
Prof. Ankur Kulkarni
Department of Systems and Control Engineering
Indian Institute of Technology, Bombay

Lecture - 23B
Policy and value function

(Refer Slide Time: 00:20)



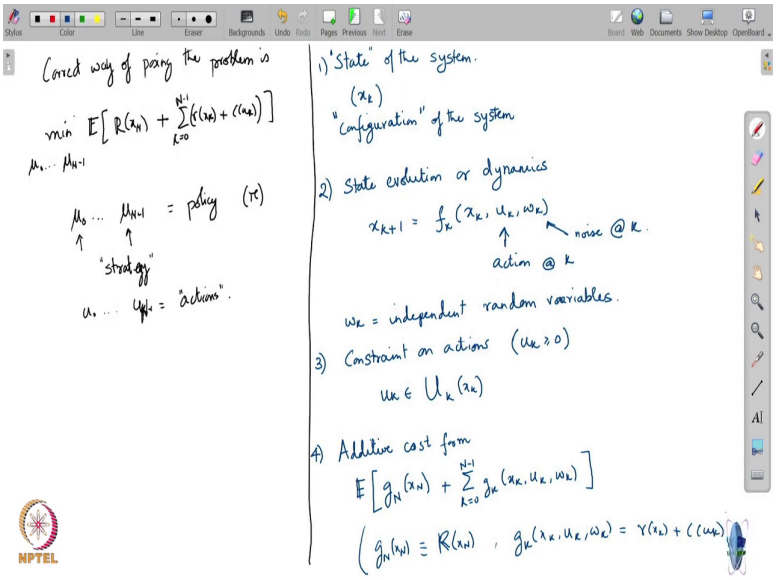
So, this way of writing the problem as I said is not, this is not well posed here, this is not well posed. So, because this is not well posed the right way of posing the problem is that we are not looking for actions u 0 to u n minus 1, but rather functions. We are looking for functions and now let me denote these functions in the following way 0 mu 0 to mu n minus 1 right.

So, mu 0 to mu n minus 1 such that, the amount of stock to be ordered at to be ordered at time k, which is u k is equal to then mu k of x k, for all values of x k. So, whatever be the value of

whatever be the value of the stock available at the beginning of the time period, you will be able to find u k as a function of that, once you know mu k.

So, what we are looking for are these functions ok. These functions mu 0 to mu n minus 1 and as and these functions would then once these functions are fully specified, what they let you do is pick your action, which is the amount of stock to be ordered as the function of the information that would be best.

(Refer Slide Time: 02:31)



So, therefore, the problem the correct way of posing the problem, the correct way of posing the problem is that you want to minimize the expected cost; includes the terminal cost, plus all these running cost, starting from 0 to n minus 1 r of x k plus c of u k and you want to minimize this by choosing mu 0 to mu n minus 1, these functions.

So, mathematically this is what is being done, you are getting, you are trying to optimize this particular cost function, now by choosing a sequence of functions ok. Now, the reason we needed to do this was, because demand itself is something that was going to be realized in the future alright. And, it was not known it is value was not known to us.

So, we had to essentially plan for every possible value and which and when you are planning for every possible value, what we are effectively doing is finding this these the sequence of such functions right. Now, there would be another case, suppose in which someone told you what the exact demand is going to be ok. Hypothetically say, suppose someone gave you the exact value of the demand.

In which case this the expectation here would be moved, the expectation in this that I have written here, would have would be moved. And, then what you would know the exact value of the information or the exact value of the amount of stock that would be that would be present at the beginning of each time period.

In that case, in that case you can actually plan to for a, for taking a particular action, because you know the exact state right. So, there are no you do not need to plan for every possible state, you can simply try to find the value of the action for the particular state that is going to be realized alright. For the, or the particular stock level that is going to be realized. That is that, kind of case where the demand is known would be the case where the noise is deterministic.

So, it is really not noise any more the demand is deterministic it is known in it is value is known in advance and all you are doing is planning for specific events, that are going to happen in the future, specific known events that are going to happen in the future. That is a much easier version of this particular problem, but the more general version of the problem is involves taking these a taking these decisions without knowing, what the information is going to be alright.

So, this mu 0 to mu n minus 1 that, I have written here this is what is called a policy ok. And, it is often denoted by pi ok. So, the distinction here that I have let me mention 1 more a point here, the distinction between what is called, what I have been saying as taking a particular action, which is the amount of stock to be ordered versus planning for every possible, level of stock. The distinction between these two is the distinction between what we call in stochastic control and in games, as the distinction between actions and strategy.

What we have effectively done is because we do not know the information that is going to be realized in the future, we are not commit committing or deciding on what particular action is to be taken. Rather what we are doing is we are coming up with a strategy. A strategy which says that, you know if I had this information this is what I would do, if I have that information that is what I would do, alright.

So, this here these functions mu 0 to mu n minus 1, they to or your policy effectively constitutes decides doing what is called strategy. It involves coming up with these plans, whereas, u 0 to u k minus 1, these which are the actual decision they are going to take these constitute actions alright.

So, our problem therefore, is to come up with is to come up with these strategies or these policies right. Now, the every time a problem involves noise it whether you whether; however, simple or complex the noise may be the problem shifts from the space of actions to the space of strategies. Because, we do not know the value of the noise in advance, it means that the, you cannot plan for any specific actions.

So, problem in of choosing actions is simply a problem of choosing these vectors right. So, sorry this is not k minus 1 this should be N minus 1, the problem of choosing these action is simply a problem of choosing these N vectors u 0 to u N minus 1 alright. And, I can stack them up and essentially think of this as one composite decision problem involving one long vector right, so, u 0 to u N minus 1.

However, this problem here the problem above is not a problem of choosing vectors, it is a problem of choosing functions. So, the space of the problem itself has changed. The, the problem of choosing actions which is the problem that you would have if say the w s were deterministic, that problem is the problem of simply the problem of vector optimization whereas, this problem here is the problem of optimization over functions right.

It is a problem of finding the right sequence of functions, not just as sequence of vectors ok. So, this distinction is a, is actually what makes this dynamic programming or dynamic optimization, significantly harder than static optimization alright. Now, the r as I go a little further I will explain to you how we can actually reduce, somehow this problem which is involves trying to decide these N functions to a setting to somehow I setting where we have to decide only actions ok.

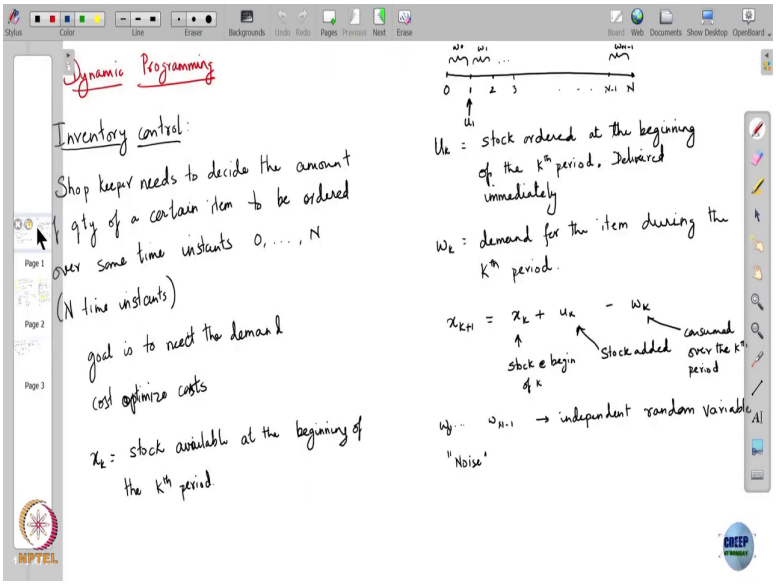
So, actions, but actions and from there from those actions somehow deduce what the function should be? So, all of that will happen in a subsequently as we go further down in this course ok. So, now, to summarize what, we said in this example. Let me, write for you a general dynamic optimization problem or a general dynamic programming problem and it is main constituents ok.

So, the main constituents of a dynamic programming problem involve, first a state the first component is state of the system ok. So, the state of the system simply is your x k in this case, which was the inventory level. So, this ok so, it is it will be denoted by x k it was in the previous problem it was the inventory level. It captures, it is whatever is needed, it is whatever is needed to capture the configuration of the system.

It is some description that we have of the configuration of the system that we are that we are dealing with. In this case since the, system we were in the example we saw the system we were dealing with was the inventory in a shop. So, it was enough for us to keep track of just simply the amount of inventory present at the beginning of the time period right.

So, in so, as a result of this the, we took that as the state of the system. Now, the state of the system is a part of problem modeling, what exactly want to define as the state is a it is a bit of a there are usually more than one ways of defining the state, but try to always keep the state as simple as possible.

(Refer Slide Time: 12:37)



For instance in the previous problem we could have, because we were talking of inventory at I am trying to decide the stock over these n time periods. We could have taken the state as the entire history of you know the stock levels up till time k.

Now, that entire history could be taken, but it has no bearing on trying to decide it does not help in you know in trying to decide what we are, what we are actually looking for which is the amount of stock that is to be ordered.

Because, that the amount of for in order to decide, what the stock that we have to order, it is enough to simply know what the level of the stock is at the latest time period, not and the entire history is actually not relevant for that.

So, as a result we have picked a parsimonious definition of the state, we have taken this state as simply the current the level of stock at the current time period ok. So, now, the state itself, state evolution or dynamics. Now the state itself evolves based on the action that you would take and the noise that, comes up in the system.

So, the way we express this is that we write x k plus 1 as some f k of x k comma, u k comma, w k right. So, here this is the action at time period k, this here is the noise, this is the noise at time period k alright. And, so, as a function of the action that you plan to take at time period k and noise that evolves at time period k, and the previous, and the state at time period k, you get the next state ok, there is state at the next time period right.

Now, because this w k is random this sequence will be random alright. So, these sequence of states will be random it is not something whose value you would know at the beginning of the time, at the beginning of the problem ok. And, so, as a result the actions that you would end up taking would also be random, because they would be as would be chosen as a function of the information that you have.

We so, this here is these are this w k is noise; usually we take w k as independent random variables. In many cases also identically distributed or 0 mean and so on, but that depends on the problem. Usually, we take these as independent random variables alright. And, they are independent of each other and they their distribution cannot be decided you know as a function of as a function of using through your actions ok.

Now, there would usually be a control constraint on actions. So, that constraint on actions is in this in our case was, in our case was say u k greater than equal to 0 ok. So, for example, that was our constraint, but more generally it could be any constraint and moreover that

constraint could depend also on the state that you are in. Say for example, it could be something as something like this u k belongs to capital U k of x k ok.

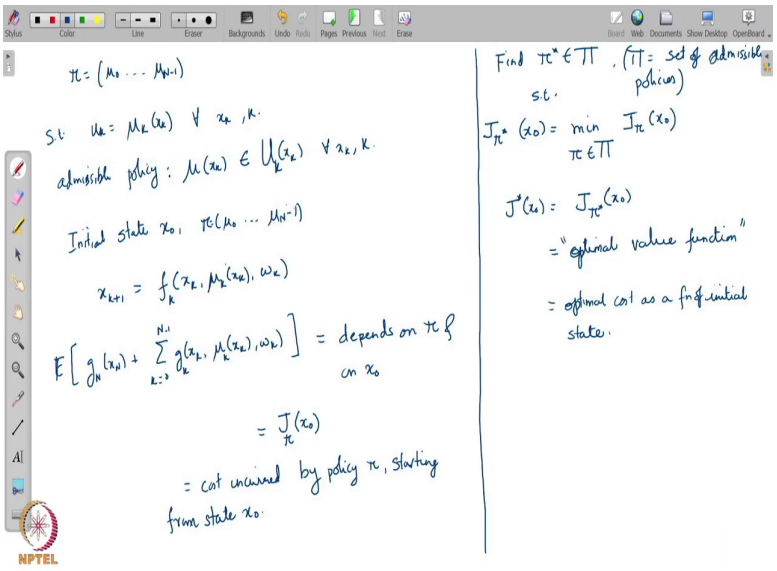
This would be the most general way of writing the constraint. So, as a function of the state x k that you are in the kind of the actions that you can potentially take have to be in this set capital U k of x k ok, alright. And, the additive cost form. The cost we would the cost form is denoted in this sort of way, here is your terminal cost g N of x N remember we are taking actions at starting at time 0 till time N minus 1.

The problem ends at time N. So, at time N whatever is the state based on that you incur a terminal cause g N of x N, we are not taking any further actions at time N. So, although I am saying I have said that we take actions at N time instance, the N time instance themselves are denoted 0 to N minus 1 alright.

So, this is so, the entire cost is therefore, g N of x N, which is your terminal cost plus g k of x k, u k sometimes we can we also include w k here, but it does not matter d k you it can this inclusion here is optional it is you can simply write g k comma u k, that is also without loss of generality alright.

So, as you can see we are in our in the earlier in the problem above g N of x N was this was simply R of x N and g k of x k, u k, w k this was r of x k plus c of u k ok alright. Now, ok. So, having formulated this the basic problem, then are the, are goal becomes the following, that we would like to decide, we would like to decide a policy.

(Refer Slide Time: 19:40)



So, we would like to decide a pi, which is denoted by mu 0 to mu N minus 1, what we would a policy mu 0 to mu N minus 1 such that, it would map the state to control actions or 2 actions that you want to take ok. So, u k would be equal to mu k of x k, for all values of x k for all values of k right. And, now moreover this now what kind of policies are admissible?

Now, remember since, we mentioned, we have control; we have constraints here on control actions, that control actions needed have to satisfy this particular constraint. The kind of policies that are admissible are an admissible policy is only one, where an admissible policy is one, which is in which u k or in which as mu k of x k always belongs to u of u capital U k of x k. So, this is to for all comma and all k.

So, an admissible policy is one which satisfies the constraints the control constraints that we have. So, now, given the given so, how does the problem begin? The problem begins with an

initial state. So, you are given an initial state ok. The it is say the inventory level at the start, at the start of the time horizons, say an initial state.

You given that initial state, you would then and given the initial given the, a choice of a policy, given an initial state and a policy mu 0 to mu N minus 1, given the initial state and a policy, what how does the how does this system behave? Well, it takes the initial state uses the policy at times 0 to decide the action to be taken at time 0 right?

So, it takes the policy at time 0 to decide the action to be taken at time 0, which is that would be u 0, u 0, then feeds into the state dynamics and what we get from there is that you get this state at time 1. So, you get more generally x k plus 1 at time k a time x k plus 1 emerges as a function f k of x k and u k, where u k itself is just simply mu k of x k alright and the noise w k right.

So, because you are choosing now the action as a function of the state ok, because u k itself is being chosen as a function as a function of the state, you can now substitute this here. And, what you find is that the next state comes up is defined through the dynamics and the policy that you have chosen, which is mu k right.

So, after the substitution essentially, what I what one can do is you can do this substitution everywhere, in fact, you can do this substitution even in the cost function, if your cost function was g N of; g N of x N plus. So, g N of x N plus g k of u k w k. So, I have replaced the u k there by simply the by simply mu k of x k right. So, as you can see the cost function now depends on two things.

So, this depends on pi which is your policy and depends also on the initial state ok. It depends on where you are starting, because after all the state evolution that you that your system would see, would now get determined by the dynamics and the policy that has been stated.

So, once I fixed my initial state which is the state where I am starting from, and the kind of policy that I have chosen the noise distribution is what will determine the revolution of your of the state sequence ok. And, so, what we have therefore, is the expected cost of the

associate of evaluated over this particular noise sequence, and that depends as you can see on pi a as well as x 0.

So, we denote this we denote this therefore, by J subscript pi of x 0. So, this is therefore, this, what is this quantity this is the cost incurred by policy pi starting form state, starting from state x 0, this that is what this would be. So, what is therefore, the problem? The problem than 1 the problem that we have that we want to solve then is to find policy pi star in say a space capital pi. Now, what this capital pi here is the set of admissible policies, admissible set of admissible policies, sorry J pi star.

Such so, find a pi star, which is in the set of admissible policies such that the cost of the cost that you incurred by pi star is the least among the cost of all policies, in this set of admissible policies ok. Now, the cost the notice that, this is you it appears like this particular problem is actually a problem that we would now need to solve for each value of x 0, because we have not a gotten rid of the initial state as yet.

So, we would have a cost for each initial state. And, it appears a prime of see that the pi star which is the optimal policy would vary could vary with x 0. Because, I have just fixed 1 x 0 and found a pi star; however, the nature of the problem is such that it turns out at you know this it is typically possible to find a pi star, that does not depend on x 0 itself.

So, you can choose any token x 0 to start with and the policy that you would find you end up finding a policy that would work for any initial state ok. So, that is how that is how it often happens is it is because the policy itself is something that is a plan for every possible state. So, it does not it is not it is not merely for this any the specific state that you are looking for it you usually get the policy for every possible initial every possible initial state.

So, this particular quantity which is the J pi star this is also denoted by J star of x 0, this sometimes denoted by J star of x 0 that is simply another way of writing the cost of the optimal policy. This has a name it is what is called the optimal value function, optimal value

function. So, the optimal value function is simply a function that tells you what the optimal value of the dynamic optimization is going to be as a function of the initial state ok.

This is called the optimal value function or optimal cost function ok, this optimal cost as a function of initial state ok. So, this is so, what we will now what we will do in the next class is find ways of computing this particular the optimal value the or the optimal value function ok.

So, I will pause here and we will resume again in the next class.